Dynamic Routing of Real-Time Virtual Circuits*

Ibrahim Matta
College of Computer Science
Northeastern University
Boston, MA 02115
matta@ccs.neu.edu

A. Udaya Shankar
Department of Computer Science
University of Maryland
College Park, MD 20742
shankar@cs.umd.edu

Abstract

Future integrated services networks, such as ATM networks, will support diverse services, including guaranteed real-time service required by many applications such as voice and video. To support such service, virtual circuit (VC) routing algorithms are often proposed. Typically, the source maintains a view of the network, and uses this view to select a path to the destination. A request is then made to setup a real-time VC over this path through resource reservations. The request is blocked if the requested resources are not available. These VC routing algorithms are usually evaluated individually in terms of steady-state performance measures. In this paper, we compare several VC routing schemes in terms of instantaneous *measures using a recently developed time-dependent evaluation method. Our results show that a routing scheme which defines the cost of a path as the sum of measured link utilizations yields more stable behavior and lower VC blocking probability over a wide range of workload parameters and network configurations than other traditional schemes.*

1. Introduction

Future integrated services networks, such as Asynchronous Transfer Mode (ATM) networks,will carry a wide variety of applications such as video-on-demand, video-conferencing etc. A key characteristic of some of these applications is that they require quality-of-service (QoS) *guarantees.* To support such service, routing protocols based on the Virtual Circuit (VC) model are often proposed. In particular, a VC is setup on some path from the source to the destination; resources are allocated to the VC on each of the links such that the QoS demand is guaranteed without violating QoS demands of existing VCs. Since network resources are limited, some requests for VC setup are denied (blocked) by the admission control algorithm. The objective of the routing algorithm is to choose routes that result in high successful VC setup rate (or equivalently low VC blocking probability).

Typically [3, 1, 6], topology and QoS information (e.g. measured link delay, utilization) is distributed *regularly* to all nodes. The proposed routing schemes differ in how they use this QoS information when they select a route for a VC. Generally, a cost function is defined in terms of the QoS information; for example, the cost of a path could be defined as the sum of measured link delays. The route selection algorithm then favors short paths with minimum cost.

Clearly, the choice of this cost function is crucial to the overall performance of the network. An overly sensitive function may result in all sources being overly aggressive, which may decrease the chances of successful VC setup. In particular, it may lead to *oscillatory behavior* with all sources selecting lightly loaded paths at the same time. These paths then become heavily loaded and thus completely avoided at the next routing update. Such route oscillations result in network under-utilization and higher VC blocking probability. On the other hand, a slowly changing function may result in slow adaptivity to load changes and thus higher VC blocking probability. Thus the chosen cost function should achieve a good balance between adaptiveness and stability.

Proposed routing schemes are usually evaluated individually in terms of steady-state measures. They are also sometimes evaluated using steady-state approximations that do not expose oscillatory behavior [4]. In this paper, we compare various routing schemes in terms of *transient* measures. Our results show that a routing scheme which defines the cost of a path as the sum of measured link utilizations yields more sta-

---

*I. Matta was partially supported by NU-RSDF grant 377090.

ble behavior and lower VC blocking probability over a wide range of workload parameters and network configurations than other schemes, including a traditional scheme which defines the cost of a path as the maximum of measured link utilizations (e.g., [2, 3]).

Our evaluations were carried out using a numerical-analytical method introduced and validated in [12, 11]. This method allows the *time-dependent* evaluation of a general multiple-class multiple-resource system. It has the advantage of allowing us to model an integrated services network in an accurate and flexible way, and with much less computational cost compared to discrete-event simulation which requires the averaging of a large number of independent replications to obtain reliable performance measures.

Organization of the paper. In Section 2, we formulate our time-dependent queueing model of a network with *arbitrary* topology offering *heterogeneous* real-time services. The network uses weighted fair-queueing link scheduling [14] and "effective bandwidth" admission control [7]. We review the evaluation method in Section 3. Section 4 discusses routing. We consider several VC routing schemes, including ones that adapt to *delayed* state information expressed in terms of link utilizations and delays. Section 5 contains results for two networks. In the first network, routing schemes consider one-link and two-link paths when routing incoming connections. Such schemes are often proposed for virtual path based ATM networks (e.g., [8]). The second network has the NSFNET-backbone topology and routing schemes consider shortest paths of arbitrary length. Such schemes are often proposed for the Internet (e.g., [3]). Section 6 concludes the paper.

2. Network Model

We consider networks of arbitrary topology supporting real-time communication using a connection-oriented reservation scheme. That is, before a real-time application (e.g., voice) can start transmitting its packets at the requested end-to-end QoS (e.g., delay), a connection has to be first established along a fixed physical route from the source node to the destination node. For this, the source node uses its routing information to choose a potential route to the destination node.

A local QoS is then requested from each of the links of this route such that the aggregate of these local QoS satisfies the connection's end-to-end QoS. If the request fails at any link due to lack of resources, the connection is blocked and lost; it is assumed that it is not attempted on another (alternate) route. Otherwise, the connection is established and resources are allocated to it. At the end of transmission, this connection is torn down and resources are released.

Routing model. Routing can be static or dynamic. For dynamic routing, we assume routing information is updated by periodic broadcasts by nodes of the status of their outgoing links during the last period. This periodic collection of status information is often used in routing algorithms proposed for integrated services networks (e.g., [1, 2]). We assume that broadcasts of all nodes are synchronized; we can easily model unsynchronized broadcasts. We also assume that these broadcasts reach other nodes instantaneously; this is justifiable because the time to propagate routing information is small compared to the routing update period.

After each update, a node uses its new routing information to compute new routes to be used for incoming connections until the next broadcast. The routes are thus updated at discrete time instants $nT, n = 1, 2, \cdots$, where T is the routing update period.

We assume a source node uses probabilistic routing, a type of routing proposed in many studies (e.g., [2, 3]). Here, a probability $\alpha_p(t)$ is assigned to every candidate path p and arriving connections are routed independently according to these path probabilities. With dynamic routing, the probabilities vary with time and are periodically updated according to dynamic status information (e.g. measured load). With static routing, the probabilities are constant over time. Deterministic routing, whether static or dynamic, is a special case of probabilistic routing where the $\alpha_p(t)$'s are 0 or 1.

The set of candidate paths that a source node chooses from is restricted to the set of minimum-hop and minimum-hop + 1 paths. This is desirable because using a longer path for a connection ties up resources at more intermediate nodes, thereby decreasing network throughput. Furthermore, it also ties up more resources at each intermediate node because satisfying the end-to-end QoS requirement would require more stringent local QoS requirements. Section 4 addresses routing in more detail.

Traffic model. We think of the network as providing real-time services. A *service* represents connections with the same source-destination node pair and the same traffic and QoS parameters. The parameters of a service s include the following:

- Time-dependent arrival rate of requests for a connection setup, $\lambda_s(t)$. Requests arrive according to Poisson processes.

- Time-dependent average lifetime of a connection from the time it is successfully established until it ends, $1/\mu_s(t)$. Connection lifetimes are exponentially distributed.

- QoS requirements of a connection of service s defined by an end-to-end statistical delay bound $(D_s,\ \varepsilon_s)$ denoting that the Probability[end-to-end packet delay $> D_s$] $< \varepsilon_s$. The delay does not include the propagation delay. This QoS requirement is also referred to as packet jitter [5].

  For a connection setup request on a multi-link route, the requested end-to-end QoS is divided equally among the links. This is the so-called "equal allocation" policy [13].

- Packet (or cell) generation characteristics of a connection. We assume a connection generates packets according to a two-state Markov Modulated Poisson Process model where it is either in a busy state sending packets back-to-back at peak rate or in an idle state sending no packets at all. A connection of service s is characterized by: M_s, the peak packet transmission rate; m_s, the mean transmission rate; and b_s, the average duration of the busy period.

  We assume the packet generation characteristics of a connection established on a multi-link route do not change from link to link, i.e. remain the same as the given external characteristics. This is valid in practice if the network admission control makes the same assumption [7] or if the network uses a non-work-conserving link scheduling discipline to reconstruct the traffic pattern at each link [15].

A connection of a service can potentially be established along any of the candidate routes between the service's source node and the service's destination node. The *class* of a connection is defined by its service and the route it takes.

Figure 1 shows a network offering two services: service s1 from node 0 to node 3, and service s2 from node 1 to node 3. Each service has two candidate routes for connection setup. Hence the network has four classes: classes c1 and c2 for s1 connections using route ⟨ 0, 4, 3 ⟩ and ⟨ 0, 1, 2, 3 ⟩ respectively, and classes c3 and c4 for s2 connections using ⟨ 1, 2, 3 ⟩ and ⟨ 1, 0, 4, 3 ⟩ respectively.

Thus, each link in the network is used by a subset of the classes. For example, in Figure 1, link ⟨4, 3⟩ is used by two classes, namely c1 and c4. Given the above assumptions, it is straightforward to obtain the parameters of a class at a link from the parameters of its service. Consider, for example, the parameters of class c1 at link $j \in \mathcal{R}_{c1} = \{\langle 0,\ 4\rangle,\ \langle 4,\ 3\rangle\}$. Connection setup requests arrive according to a Poisson process with rate $\lambda_{c1}(t) = \alpha_{s1,c1}(t)\ \lambda_{s1}(t)$, where $\alpha_{s1,c1}(t)$ is the (possibly dynamic) probability of a connection of service s1 being routed on class-c1 route. The average lifetime of a connection $\frac{1}{\mu_{c1}(t)} = \frac{1}{\mu_{s1}(t)}$. For an end-to-end QoS $(D_{s1},\ \varepsilon_{s1})$, the local QoS requirement $(D_{c1}^j,\ \varepsilon_{c1}^j) = (\frac{D_{s1}}{2},\ \frac{\varepsilon_{s1}}{2})$, because the route of class c1 is two-hop long. The packet generation characteristics $(M_{c1},\ m_{c1},\ b_{c1}) = (M_{s1},\ m_{s1},\ b_{s1})$.

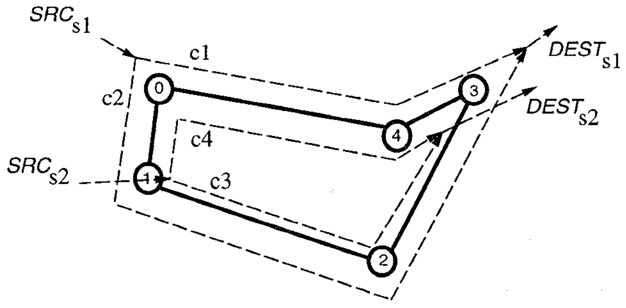

Figure 1. A network example.

Scheduling and admission control model. We assume each link in the network uses a "per-connection" scheduling algorithm of the weighted round-robin type. An example of this type of scheduling algorithms is weighted fair-queueing [14]. Here, each class-c connection is allocated (and guaranteed) a certain amount of bandwidth on link $j \in \mathcal{R}_c$ that is enough to satisfy its local QoS requirement. This required bandwidth[1], denoted by R_c^j, depends of course on the local QoS $(D_c^j,\ \varepsilon_c^j)$ and the packet generation characteristics $(M_c,\ m_c,\ b_c)$ of the connection.

R_c^j can be obtained from the following approximation [7]:

$$R_c^j = M_c \frac{\beta_c^j - X_c^j + \sqrt{[\beta_c^j - X_c^j]^2 + 4\ X_c^j\ \rho_c\ \beta_c^j}}{2\ \beta_c^j} \qquad (1)$$

where $\beta_c^j = ln(\frac{1}{\varepsilon_c^j})\ b_c\ (1 - \rho_c)\ M_c$; $\rho_c = \frac{m_c}{M_c}$ is the probability that the connection is active (in busy state); and $X_c^j = D_c^j \times R_c^j$ is the buffer space required by the connection.

R_c^j can be computed from equation (1) iteratively. For each class-c connection, we can then determine its requirements R_c^j and X_c^j on link $j \in \mathcal{R}_c$. From this, we can determine whether it is feasible for link j to accept the connection; R_c^j must be no greater than the

---

[1] Often referred to as effective or equivalent capacity [7].

current available (idle) capacity of link j, and X_c^j must be no greater than the current available buffer space of the link. We assume that there is adequate link buffer space. Then it is feasible to accept a class-c connection on link j if j can satisfy the R_c^j bandwidth requirement.

3. Solution Method

The above model can be solved to obtain various instantaneous performance measures. We are mainly interested in calculating the end-to-end measures of each service. An intermediate step in this calculation is to compute the end-to-end measures of each of the service's classes. Among the main measures of class c is:

- $B_c(t)$, instantaneous blocking probability of class-c connections.

These measures depend on the performance seen by class-c connections at each link $j \in \mathcal{R}_c$, where $\mathcal{R}_c$ denotes the route of a class-c connection. In particular, we define the following:

- $B_c^j(t)$, instantaneous blocking probability of class-c connections at link $j \in \mathcal{R}_c$.
- $N_c^j(t)$, instantaneous average number of class-c connections established on link $j \in \mathcal{R}_c$.

Then, assuming link independence, we have

$$B_c(t) = 1 - \prod_{j \in \mathcal{R}_c} [1 - B_c^j(t)] \tag{2}$$

Let $\mathcal{C}^j$ be the set of all classes of connections using link j. To calculate the time behavior of $\{B_c^j(t) : c \in \mathcal{C}^j\}$, we write the following difference equations for $c \in \mathcal{C}^j$, for time step $\delta \ll T$:

$$\begin{aligned} N_c^j(t+\delta) \;&=\; [1 - \mu_c(t)\,\delta]\; N_c^j(t) \;+ \\ &\quad\; \delta\, \lambda_c(t) \prod_{j' \in \mathcal{R}_c} [1 - B_c^{j'}(t)] \end{aligned} \tag{3}$$

The first term in the right-hand side of equation (3) represents the average number of class-c connections established on link j which remain on link j (i.e. do not terminate). The second term represents the average number of new class-c connections established on link j during $[t, t+\delta)$.

Observe that if we could express $B_c^j(t)$ in terms of $\{N_{c'}^j(t) : c' \in \mathcal{C}^j\}$, then we could solve equations (3) and hence (2) inexpensively for the time behavior of the performance measures. Obtaining such an expression is intractable. *However the instantaneous relationship between the $B_c^j(t)$ and the $N_c^j(t)$ is very well approximated by their relationship at steady-state* [12, 10], i.e., by the relationship between the B_c^j and the N_c^j assuming that the $\lambda_c(t)$ and $\mu_c(t)$ are constants. The steady-state relationship is relatively easy to obtain. We obtain it implicitly as a fixed point of two steady-state expressions, one defining B_c^j in terms of $\{\frac{\lambda_{c'}}{\mu_{c'}} : c' \in \mathcal{C}^j\}$, and one defining $\frac{\lambda_c}{\mu_c}$ in terms of N_c^j and B_c^j. These two expressions are obtained next.

Denoting the first expression by $\mathcal{S}_c^j$, we have for $c \in \mathcal{C}^j$:

$$B_c^j = \mathcal{S}_c^j(\{\frac{\lambda_{c'}}{\mu_{c'}} : c' \in \mathcal{C}^j\}) \tag{4}$$

$\mathcal{S}_c^j$ can be obtained as follows. Define a *schedulable state* of link j to be a $|\mathcal{C}^j|$-dimensional vector representing the number of connections of each class $c \in \mathcal{C}^j$ that can be established simultaneously on link j, i.e. for which the local QoS is satisfied for every connection [9]. Denote the set of schedulable states by $\mathcal{F}^j$. $\mathcal{F}^j$ can be determined knowing the effective bandwidth of each class at link j, i.e., the R_c^j obtained in Section 2.

We obtain $\mathcal{S}_c^j$ by solving the Markov chain over $\mathcal{F}^j$. In particular, denoting by $P(\sigma)$ the probability of being in a state $\sigma = (\sigma_1, \sigma_2, \cdots, \sigma_{|\mathcal{C}^j|}) \in \mathcal{F}^j$, we have

$$P(\sigma) = P(0) \prod_{c'=1}^{|\mathcal{C}^j|} \frac{(\lambda_{c'}/\mu_{c'})^{\sigma_{c'}}}{\sigma_{c'}!} \tag{5}$$

where $P(0)$ is the normalization constant. Then we have

$$B_c^j = \sum_{\sigma \in \mathcal{F}^j} \mathbf{I}\{(\sigma_1, \cdots, \sigma_c + 1, \cdots, \sigma_{|\mathcal{C}^j|}) \notin \mathcal{F}^j\}\; P(\sigma) \tag{6}$$

Equating the rates of departure and admission of class-c connections at link j, we have $\mu_c\, N_c^j = \lambda_c\, [1 - B_c^j]$. From this we have for $c \in \mathcal{C}^j$:

$$\frac{\lambda_c}{\mu_c} = \frac{N_c^j}{[1 - B_c^j]} \tag{7}$$

From equations (4) and (7), we can express $B_c^j(t)$ approximately in terms of $\{N_{c'}^j(t) : c' \in \mathcal{C}^j\}$ by replacing the steady-state measures B_c^j and N_c^j by their instantaneous counterparts $B_c^j(t)$ and $N_c^j(t)$, and replacing $\frac{\lambda_c}{\mu_c}$ by an instantaneous quantity $z_c^j(t)$ that we introduce. Doing this yields the following instantaneous equations for $c \in \mathcal{C}^j$:

$$B_c^j(t) \;=\; \mathcal{S}_c^j(\{z_{c'}^j(t) : c' \in \mathcal{C}^j\}) \tag{8}$$

$$z_c^j(t) \;=\; \frac{N_c^j(t)}{[1 - B_c^j(t)]} \tag{9}$$

For fixed $\{N_c^j(t) : c \in \mathcal{C}^j\}$, we can solve equations (8) and (9) iteratively for $\{B_c^j(t) : c \in \mathcal{C}^j\}$. In particular, we can start with initial estimates $\{\hat{z}_c^j(t) : c \in \mathcal{C}^j\}$ and obtain $\{B_c^j(t) : c \in \mathcal{C}^j\}$ using equations (8). Then we use equations (9) to obtain new values for $\{z_c^j(t) : c \in \mathcal{C}^j\}$. We repeat this process until the values for $\{z_c^j(t) : c \in \mathcal{C}^j\}$ stabilize.

Given the $N_c^j(t)$ and $B_c^j(t)$, we can then solve for the $N_c^j(t+\delta)$ using equations (3), and we repeat the process to obtain the time evolution of the performance measures for time instants 0, δ, 2δ, $\cdots$. Every T time units ($\gg \delta$), we also update the route selection probabilities $\alpha_{s,c}(t)$ based on information collected during the past time period, which gives rise to new values for the $\lambda_c(t)$.

4. Routing Schemes

Our model can capture several design choices when developing a route selection algorithm. An important design choice is related to how a source node determines which path to use for routing a new incoming connection. Recall that we assume a source node considers the set of minimum-hop and minimum-hop + 1 paths for connection routing. From this set, a path p is selected probabilistically using path weights W_p where

$$W_p \propto \frac{F_p}{H_p \times L_p} \tag{10}$$

Here, H_p is the number of hops of path p (this gives preference to shortest paths). L_p is a measure of the load on path p averaged over the last update period (discussed below). F_p is either 1 or 0 depending on whether the path p is feasible or not. A path p is said to be feasible if the source "expects" a successful setup on p [1]. For this, the source would take into account the bandwidth requirement of the new connection in addition to the current reserved capacity[2] on the path (it assumes it is accurate) to test the feasibility of the path. This is in fact an admission control function. The route selection probabilities $\alpha_{s,c}(t)$ are then computed according to (10).

An important issue here is how L_p is defined. L_p could be defined as:

(i) The sum of the utilizations of the links on path p, where the utilization of a link is the fraction of the link capacity reserved.

(ii) The maximum link utilization of the links on path p. This is a traditional definition of L_p (e.g., [2, 3]).

(iii) The sum of the delays of the links on path p, where the delay of a link j can be estimated as $\frac{1}{Cap^j - CapRes^j}$ where Cap^j is the total link capacity and $CapRes^j$ is the average reserved link capacity. Similar definitions of L_p were proposed in [1, 6], where the cost of a link is a highly nonlinear (e.g. exponential) function of its current utilization.

In the following section, we examine the transient behavior of several route selection schemes, which differ in how they assign the weights W_p. We mainly consider three schemes. The first selection scheme, referred to as **sumUTIL+HOP**, defines L_p as in (i) above and W_p as

$$W_p = \begin{cases} 100/H_p & \text{if } L_p \le 0.01 \\ 1/(L_p \times H_p) & \text{otherwise} \end{cases}$$

The second selection scheme, referred to as **maxUTIL+HOP**, defines L_p as in (ii) above and W_p as

$$W_p = (1 - L_p)/H_p$$

The third selection scheme, referred to as **sumDLY+HOP**, defines L_p as in (iii) above and W_p as

$$W_p = \begin{cases} 0 & \text{if } L_p = \infty \\ 1/(L_p \times H_p) & \text{otherwise} \end{cases}$$

We also considered variants of these schemes that do not give priority to shortest paths, i.e., H_p is set to 1. In this case, we omit the "+HOP" suffix from the name. We also consider **HOP**, a static scheme that does not consider load, i.e., $W_p = 1/H_p$. For a scheme that considers F_p when computing W_p, its name starts with "FEAS+". For example, **FEAS+maxUTIL** defines L_p as in (ii) and W_p as

$$W_p = \begin{cases} 0 & \text{if path is not feasible} \\ 1 - L_p & \text{otherwise} \end{cases}$$

5. Numerical Results

We compare the time behavior of the routing schemes described in Section 4. We use the network model and evaluation method described in Sections 2 and 3, respectively. We obtain instantaneous performance measures through equations (3), (8) and (9). We take the discrete-time step δ to be 0.1. The required bandwidth of a connection is computed using equation (1).[3]

We first present general observations about the results, and then the details of our evaluations on two

---

[2] In our model, we directly obtain the average reserved link capacity from the average number of established connections and the effective capacity of each of the link's classes.

[3] To simplify and speed up the computations, if the result is not integer, it is rounded to the next smallest integer. See [10].

networks. We assume all links have the same capacity *Cap*. We compare routing schemes on lightly, moderately and heavily loaded configurations by varying *Cap*.

General observations

(A) By giving priority to shortest paths, a routing scheme always performs better. This is mainly because the use of long paths for connections is undesirable since it ties up resources at more intermediate nodes, which can be used to admit many shorter length connections.

(B) sumUTIL+HOP performs better or as well as all other schemes over a wide range of workload parameters and network configurations. This is because its path cost function adapts more slowly than other adaptive schemes, striking a good balance between adaptiveness and over-sensitivity. In particular, the path cost functions of maxUTIL+HOP and sumDLY+HOP are more sensitive to load changes. The former takes the maximum, instead of the sum, over link utilizations, so whenever the link that is the bottleneck changes or its utilization changes the cost of the (possibly long) path changes. The latter uses delay, instead of utilization, to measure congestion. From the classical delay-utilization curve, around saturation, a small increase in utilization corresponds to a large increase in link delay. This dramatic change can result in the link becoming unattractive and thus completed avoided. Consequently, at the next routing update the link reports a very low cost and becomes attractive again. Such over-sensitivity leads to oscillatory behavior, which in turn degrades performance.

(C) By using F_p when computing W_p, a scheme performs as well. This indicates that although feedback information is delayed and hence inaccurate, admission control at the source does not cause excessive unnecessary blocking and hence it is effective in reducing the overhead of admission control inside the network.

Network 1. We consider the network topology shown in Figure 2. We consider 4 services using the network, with parameters as shown in Table 1. The routing update period equals 2. Figure 3 shows that maxUTIL+HOP has on average around 43% lower blocking probability than maxUTIL and around 71% lower blocking probability than HOP. It illustrates that giving higher priority to shortest paths results in better and more stable performance. It also illustrates that when the network is lightly to moderately loaded, all the adaptive routing schemes sumDLY+HOP, maxUTIL+HOP and sumUTIL+HOP have similar performance. They all outperform the static scheme HOP.

Figure 4 shows that when the network is heavily loaded, sumUTIL+HOP outperforms all other schemes as it strikes a good balance between adaptiveness and over-sensitivity. maxUTIL+HOP and sumDLY+HOP exhibit oscillatory behavior that results in higher blocking probability. sumUTIL+HOP outperforms maxUTIL+HOP by around 26% lower blocking probability, and outperforms sumDLY+HOP by around 20% lower blocking probability over $t \in (35 : 50]$.

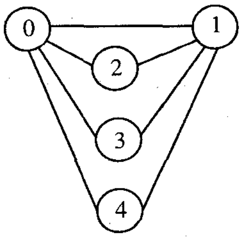

Figure 2. Network 1: 5 nodes, 7 bidirectional links.

$(SRC_s, DEST_s)$	$(M_s,\ m_s,\ b_s,\ D_s,\ \varepsilon_s)$	$(\lambda_s,\ \mu_s)$
(0, 1)	(30, 20, 0.1, 0.05, 10^{-4})	(20, 1)
(2, 1)	(30, 20, 0.1, 0.05, 10^{-4})	(20, 1)
(3, 1)	(30, 20, 0.1, 0.05, 10^{-4})	(20, 1)
(4, 1)	(30, 20, 0.1, 0.05, 10^{-4})	(20, 1)

Table 1. Parameters of the 4 services using network 1.

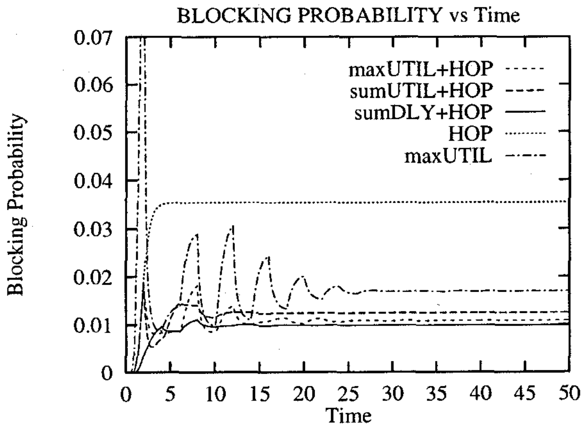

Figure 3. Blocking probability versus time. Network 1 with $Cap = 900$.

Network 2. We consider the NSFNET-backbone topology shown in Figure 5. We consider 100 ser-

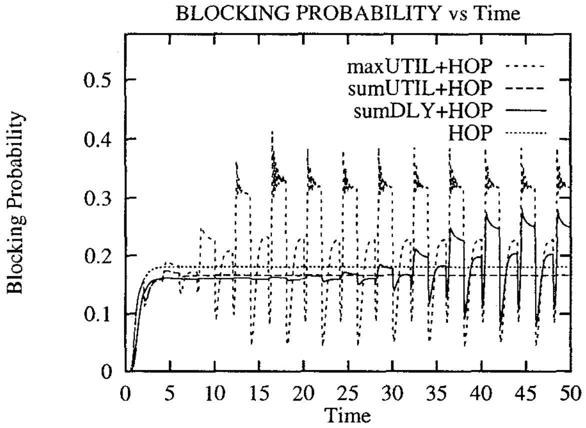

Figure 4. Blocking probability versus time. Network 1 with $Cap = 600$.

vices using the NSFNET-backbone, with parameters as shown in Table 2. Services with the same traffic and end-to-end QoS parameters, but with different source/destination pairs, are grouped in the same row. n $(SRC_s, DEST_s)$ means n services of this type s between nodes SRC_s and $DEST_s$ are specified. The routing update period equals 5. Figure 6 shows when the network is lightly loaded, all routing schemes have similar performance. maxUTIL+HOP exhibits some oscillations. HOP has a slightly higher blocking probability than other schemes.

Figure 7 illustrates that when the network is moderately or heavily loaded, maxUTIL+HOP performs the worst. It exhibits oscillatory behavior that results in higher blocking probability. All other schemes have similar performance, and outperforms maxUTIL+HOP by around 38% lower blocking probability over $t \in (30:50]$ when $Cap = 900$.

Figure 8 illustrates that with maxUTIL, admission control at the source is effective. Similar results were obtained for other schemes.

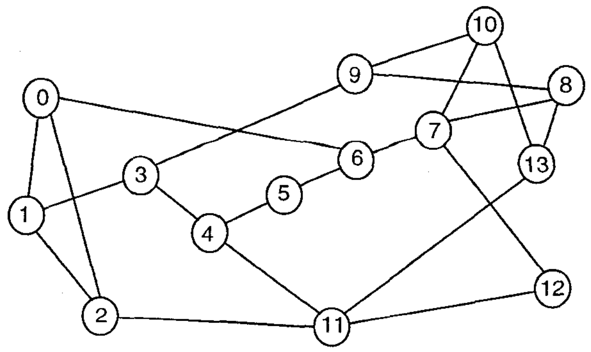

Figure 5. Network 2: 14 nodes, 21 bidirectional links, average degree 3.

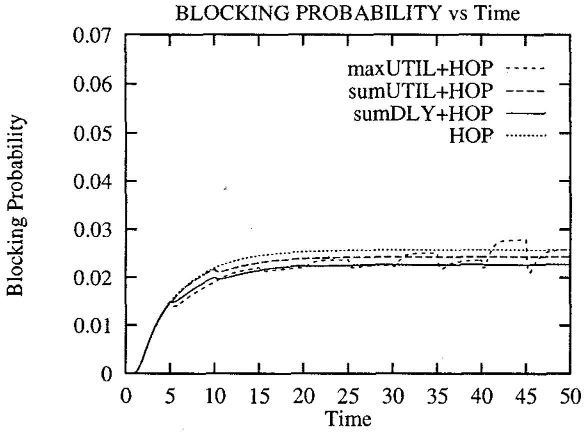

Figure 6. Blocking probability versus time. Network 2 with $Cap = 1000$.

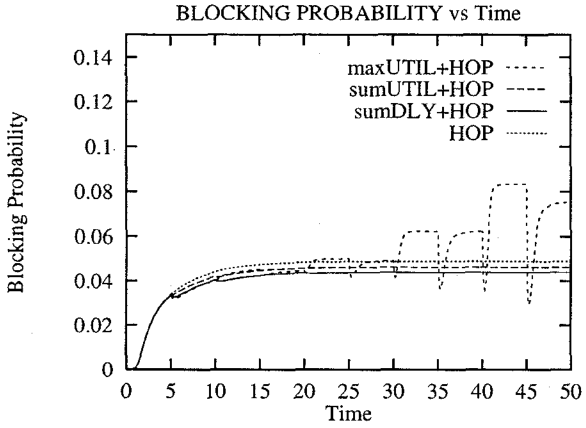

Figure 7. Blocking probability versus time. Network 2 with $Cap = 900$.

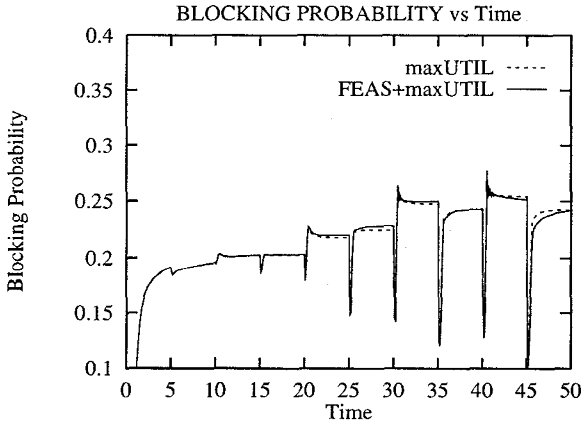

Figure 8. Blocking probability versus time. Network 2 with $Cap = 600$.

$(SRC_s, DEST_s)$	$(M_s,\ m_s,\ b_s,\ D_s,\ \varepsilon_s)$	$(\lambda_s,\ \mu_s)$
2(0, 13),2(1, 13),2(2, 13),2(3, 13),2(4, 13),2(5, 13)	(30, 20, 0.1, 0.05, 10^{-4})	(2, 1)
1(13, 0),1(13, 1),1(13, 2),1(13, 3),1(13, 4),1(13, 5)	(30, 20, 0.1, 0.05, 10^{-4})	(2, 1)
2(6, 13),1(13, 6)	(30, 10, 0.1, 0.05, 10^{-4})	(2, 2)
2(7, 13),2(8, 13),2(9, 13),2(10, 13),2(11, 13)	(30, 10, 0.1, 0.05, 10^{-4})	(1.8, 2)
1(13, 7),1(13, 8),1(13, 9),1(13, 10),1(13, 11)	(30, 10, 0.1, 0.05, 10^{-4})	(1.8, 2)
2(12, 13),4(13, 1)	(60, 20, 0.1, 0.05, 10^{-4})	(0.3, 0.2)
2(1, 13)	(60, 20, 0.1, 0.05, 10^{-4})	(0.3, 0.2)
1(13, 12)	(60, 20, 0.1, 0.05, 10^{-4})	(0.3, 0.2)
2(0, 1)	(30, 20, 0.1, 0.05, 10^{-4})	(2, 1)
2(2, 1),2(3, 1),4(4, 1),4(5, 1),4(6, 1)	(30, 20, 0.1, 0.05, 10^{-4})	(2, 1)
4(7, 1),4(8, 1),4(9, 1),4(10, 1),4(11, 1),4(12, 1)	(30, 10, 0.1, 0.05, 10^{-4})	(1.8, 2)
1(1, 4),1(1, 5),1(1, 6)	(30, 20, 0.1, 0.05, 10^{-4})	(2, 1)
1(1, 7),1(1, 8),2(1, 9),2(1, 10),2(1, 11),2(1, 12)	(30, 10, 0.1, 0.05, 10^{-4})	(1.8, 2)

Table 2. Parameters of the 100 services using Network 2.

6. Conclusions

In this paper, we compared several dynamic routing schemes for real-time VCs in terms of transient measures. Our results show that a routing scheme which defines the cost of a path as the sum of link utilizations yields more stable behavior and lower VC blocking probability over a wide range of workload parameters and network configurations than other traditional schemes. We evaluated the routing schemes using a recently developed time-dependent evaluation method [12, 11]. Future work include further evaluation of the dynamic behavior of routing schemes under various operating conditions.

References

[1] H. Ahmadi, J. Chen, and R. Guerin. Dynamic routing and call control in high-speed integrated networks. In *Workshop on Systems Engineering and Traffic Engineering, ITC'13*, pages 19–26, Copenhagen, Denmark, June 1991.

[2] S. Bahk and M. E. Zarki. Dynamic multi-path routing and how it compares with other dynamic routing algorithms for high speed wide area networks. In *SIGCOMM '92*, pages 53–64, Baltimore, Maryland, August 1992.

[3] L. Breslau, D. Estrin, and L. Zhang. A simulation study of adaptive source routing in integrated services networks. Available by anonymous ftp at catarina.usc.edu:pub/breslau, September 1993.

[4] S.-P. Chung and K. Ross. Reduced load approximations for multirate loss networks. *IEEE Transactions on Communications*, 41(8):1222–1231, August 1993.

[5] D. Ferrari. Client requirements for real-time communication services. *IEEE Communications Magazine*, 28(11), November 1990.

[6] R. Gawlick, A. Kamath, S. Plotkin, and K. Ramakrishnan. Routing and admission control in general topology networks. Technical Report STAN-CS-TR-95-1548, Department of Computer Science, Stanford University, Stanford, CA 94305, 1995.

[7] R. Guerin, H. Ahmadi, and M. Naghshineh. Equivalent capacity and its application to bandwidth allocation in high-speed networks. *IEEE J. Select. Areas Commun.*, SAC-9(7):968–981, September 1991.

[8] R.-H. Hwang, J. Kurose, and D. Towsley. MDP routing in ATM networks using virtual path concept. In *IEEE INFOCOM*, pages 1509–1517, Toronto, Ontario, Canada, June 1994.

[9] J. Hyman, A. Lazar, and G. Pacifici. A separation principle between scheduling and admission control for broadband switching. *IEEE J. Select. Areas Commun.*, SAC-11(4):605–616, May 1993.

[10] I. Matta. *Fast Evaluation and Dynamic Control of Integrated Services Networks.* PhD thesis, University of Maryland at College Park, Department of Computer Science, August 1995.

[11] I. Matta and A. Shankar. An iterative approach to comprehensive performance evaluation of integrated services networks. In *IEEE International Conference on Network Protocols '94*, Boston, Massachusetts, October 1994.

[12] I. Matta and A. Shankar. *Z*-iteration: A simple method for throughput estimation in time-dependent multi-class systems. In *ACM SIGMETRICS/PERFORMANCE '95*, pages 126–135, Ottawa, Canada, May 1995.

[13] R. Nagarajan, J. Kurose, and D. Towsley. Local allocation of end-to-end quality-of-service in high-speed networks. In *IFIP TC6 Workshop on Modelling and Performance Evaluation of ATM Technology*, page 2.2, January 1993.

[14] A. Parekh. A generalized processor sharing approach to flow control in integrated services networks. Technical Report LIDS-TR-2089, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, 1992.

[15] H. Zhang and D. Ferrari. Rate-controlled static-priority queueing. Technical Report TR-92-003, International Computer Science Institute, Berkeley, California, February 1992.