An Iterative Approach to Comprehensive Performance Evaluation of Integrated Services Networks*

Ibrahim Matta
imm@cs.umd.edu

A. Udaya Shankar
shankar@cs.umd.edu

Department of Computer Science
University of Maryland
College Park, MD 20742

Abstract

Future networks are expected to integrate diverse services. For this purpose, new algorithms and protocols have been proposed for link scheduling, admission control, and routing. The interaction between these three components is crucial to the performance of the network. However, this interaction is difficult to model realistically using available techniques. In this paper, we present an iterative discrete-time approach *that yields a realistic model which takes into account this interaction. The model applies to connection-oriented networks with different types of real-time connections. It allows the investigation of various control schemes for both transient and steady-state performances. Preliminary results indicate that our approach is computationally much cheaper than discrete-event simulation, and yields sufficiently accurate performance measures.*

1 Introduction

Integrated services networks have new characteristics not present in traditional networks (e.g., telephone or data networks). These characteristics include increased transmission speeds, diversity of applications (e.g., multi-media, voice, mail), and support of heterogeneous quality of service (QoS) requirements. This has given rise to new algorithms in all aspects of network resource allocation, including link scheduling, admission control, and routing. However, the interaction between these three components, although crucial to the overall performance, is not well-understood. Traditional approaches to evaluation appear ineffective in closing this gap.

In this paper, we present an *iterative discrete-time approach* that addresses this gap, making use of approximations to instantaneous blocking probabilities. Our approach integrates techniques from several areas, namely: (1) standard queueing theory techniques [7]; (2) the link decomposition technique widely used for packet delay analysis in packet-switched data networks [7] as well as for call blocking probability analysis in circuit-switched telephone networks [3]; (3) the dynamic flow technique used for approximating system dynamics and nonlinearity [1]; and (4) the technique of repeated substitutions used in numerical analysis to solve nonlinear equations [6].

Our approach yields a model that takes into account the interaction between link scheduling, admission control, and routing. In particular, our model permits the evaluation of a connection-oriented packet-switched network (e.g. ATM) that provides different types of real-time *services* (e.g. voice, video) by making use of various link scheduling, admission control and routing policies. Thus this model can be applied to achieve more comprehensive evaluation of existing strategies and to propose more effective network control schemes.

Outline of the model and solution method

We consider a network that provides real-time services between pairs of source and destination nodes. The connections of a service i arrive at the service's source node according to a Poisson process of rate λ_i, and requiring an *end-to-end* QoS requirement D_i (e.g. delay). Each connection, once it is successfully set up, has an exponential lifetime with mean $\frac{1}{\mu_i}$. The source node uses its routing information to choose for the ar-

*This work is supported in part by ONR and DARPA under contract N00014-91-C-0195 to Honeywell and Computer Science Department at the University of Maryland, and by National Science Foundation Grant No. NCR 89-04590. The views, opinions, and/or findings contained in this paper are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency, ONR, the U.S. Government or Honeywell.

riving connection a potential route[1] to the service's destination node. The route and service together define the *class* of the connection. We use the index k to range over classes. Let R_k denote the route of a class-k connection. The arrival rate of class-k connections, denoted by λ_k, is a function of λ_i and the routing algorithm.

A class-k connection of service i requests a *local* QoS requirement D_k^j from each link $j \in R_k$, such that the aggregate of D_k^j, $j \in R_k$, satisfies D_i, its end-to-end QoS requirement. An arriving class-k connection that finds insufficient resources on any link $j \in R_k$ to satisfy D_k^j is blocked and lost. Otherwise, the connection is accepted and resources are allocated to it on each link $j \in R_k$ for an exponential duration with mean $\frac{1}{\mu_k} = \frac{1}{\mu_i}$.

We are mainly interested in calculating the *end-to-end connection blocking probability* of each service (or equivalently the service's throughput). An intermediate step in this calculation is to compute the end-to-end blocking probability of each of the service's classes. For this, we compute $\tilde{B}_k^j(t)$, an approximation to the instantaneous probability that a class-k connection is blocked on link $j \in R_k$ (this computation is discussed below). Invoking the link independence assumption and using the fact that a class-k connection can be successfully setup on R_k iff it is not blocked on all links $j \in R_k$, we compute the class-k end-to-end instantaneous blocking probability as $1 - \prod_{j \in R_k}(1 - \tilde{B}_k^j(t))$.

Computing $\tilde{B}_k^j(t)$ is the crux of the problem. We proceed as follows. Let Γ^j be the set of all classes of connections using link j. Define the state of link j by the number of connections in each class $k \in \Gamma^j$. For a given link scheduling algorithm, we find the set S^j of link states at which D_k^j is satisfied for every $k \in \Gamma^j$. S^j is called the *schedulability region* of link j.

Next we compute $\tilde{B}_k^j(t)$ for any fixed t by iterating on "instantaneous" versions of two steady-state results, where the actual traffic intensities $\frac{\lambda_k}{\mu_k}$ are replaced by *fictitious* traffic intensities $z_k^j(t)$ that we introduce. The first steady-state result is that the steady-state blocking probabilities $\tilde{B}_k^j$ can be obtained in terms of $\frac{\lambda_k}{\mu_k}$ by solving the $|\Gamma^j|$-dimension Markov chain over S^j. The instantaneous version of this result yields $\tilde{B}_k^j(t)$ in terms of $z_k^j(t)$. The second steady-state result is that x_k^j, the steady-state average number of connections in class k, satisfies $x_k^j = \frac{\lambda_k}{\mu_k}[1 - \tilde{B}_k^j]$. The instantaneous version of this result is $x_k^j(t) = z_k^j(t)[1 - \tilde{B}_k^j(t)]$. Thus for any fixed t, we can use an iterative procedure on the two instantaneous results to obtain $\tilde{B}_k^j(t)$ and $z_k^j(t)$ in terms of $x_k^j(t)$.

Finally, to compute $\tilde{B}_k^j(t)$ for all t, we introduce difference equations which describe the dynamic behavior of the $x_k^j(.)$'s. Each equation expresses $x_k^j(t+\tau)$ as a function of $x_k^j(t)$ and $\tilde{B}_k^j(t)$ for $j \in R_k$. These can be solved iteratively in conjunction with the previous solution (of $\tilde{B}_k^j(t)$ in terms of $x_k^j(t)$).

We can thus obtain the dynamic behavior of the end-to-end connection blocking probabilities. This allows the investigation of *both* transient and steady-state performances of various control schemes. Our preliminary results indicate that our approach is computationally much cheaper than discrete-event simulation, which requires the averaging of a large number of independent simulation runs. Furthermore, the performance measures it yields are very close to the exact values obtained by simulation.

We have used our model to compare the performance of different routing schemes on the NSFNET backbone topology with a weighted fair queueing link scheduling discipline [13]. In summary, our results (presented in [9]) indicate that a routing scheme that selects paths which are both under-utilized and short (in number of hops) for routing new incoming connections gives the highest network throughput.

Organization of the paper

The rest of the paper is organized as follows. In Section 2, we formulate our discrete-time model and solution procedure. Section 3 discusses scheduling and admission control in more detail. Specifically, we adopt a particular link scheduling algorithm to illustrate how its effect is incorporated in our model. Section 4 discusses routing. Numerical results to validate our model are presented in Sections 5 and 6; Section 5 contains results for a single link, and Section 6 for a tandem network. Section 7 concludes and identifies future work.

2 Model and Solution Procedure

We consider networks of arbitrary topology offering heterogeneous real-time services using a *connection-oriented reservation* scheme. Figure 1 shows an example network offering three services: one voice service from node 1 to node 2, one video service from node 3 to node 2, and one voice service from node 3 to node 2.

A **service** is characterized by its source node, destination node, and traffic parameters and requirements. The traffic parameters and requirements of a service i include the following:

[1] We use the terms route and path interchangeably.

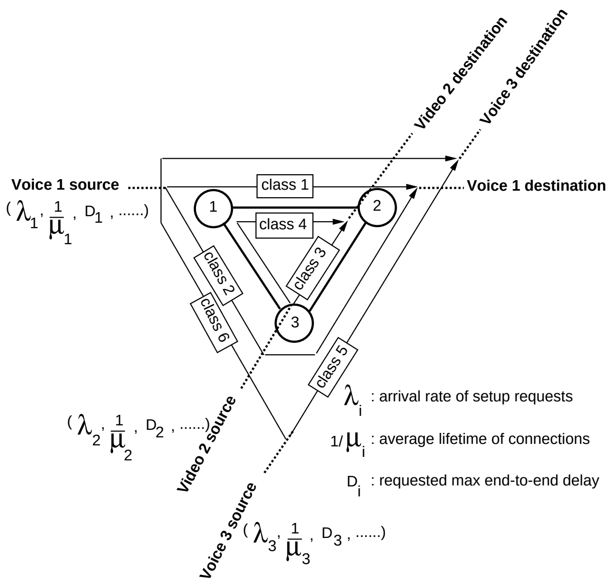

Figure 1: A network example.

- Arrival rate of requests for a connection setup (λ_i).
- Average lifetime of a connection from the time it is successfully setup until it ends ($\frac{1}{\mu_i}$).
- QoS requirements of a connection such as an upper bound on the end-to-end packet delay (D_i).
- Packet (or cell) generation characteristics of a connection such as its mean and peak packet rate.

Throughout the paper, we assume Poissonian arrivals of setup requests and exponential connection lifetimes. Also, for illustrative purposes, we only consider packet delay as the QoS requirement, and we assume that this delay does not include the propagation delay.

The connections of a service can potentially be setup along any of the possible routes between the service's source node and the service's destination node. For a service, the connections setup on different routes belong to different **classes**. For the network in Figure 1, we have six classes. Each one of the three services has two routes for connection setup: a one-link route and a two-link route. The voice 1 service has routes $(1,2)$ and $(1,3,2)$; hence for this service we define class 1 for route $(1,2)$ and class 2 for route $(1,3,2)$. Similarly, we define classes 3 and 4 for the video 2 service, and classes 5 and 6 for the voice 3 service. We note that for a connection to be successfully set up on a multi-link route, the sum of the maximum (local) delays guaranteed by the links constituting the route should satisfy the end-to-end delay requirement of the connection.

Each link in the network is used by a different set of classes. In Figure 1, link $(3,2)$, for example, is used by three classes, namely classes 2, 3 and 5. We define the **state** of a link by the number of connections in each class using the link. Every link is *constrained* to be only in a state where the QoS requirements of all connections setup on the link are satisfied. We say that this set of states satisfies the "QoS constraints". This set of states is referred to as the **schedulability region** of the link [5].

For a connection setup request on a multi-link route, we divide the requested end-to-end delay *equally* among the links; each link should thus guarantee the same maximum delay value for the connection. This is called "equal allocation" policy [12]. For example, in Figure 1, link $(3,2)$ should guarantee D_3 for every connection in class 5, D_2 for every connection in class 3, and $\frac{D_1}{2}$ for every connection in class 2. With this policy, given any network and services, it is then easy to determine for each link the set of classes using it along with their local QoS requirements.

Because a class is defined by the pair (service, route), it appears that we can have a large number of classes. This may cause a computational bottleneck. To avoid this, we can restrict the set of routes from which the source node can choose to only shortest (in number of hops) and close to shortest paths.[2] This is acceptable 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 now requires more stringent local QoS requirements. Section 4 and reference [9] address the selection of routes in more detail.

We assume that the packet generation characteristics of a connection setup on a multi-link route do not change from link to link, i.e. remain the same as the given external characteristics. (See [4, 8] for the validity of this assumption.) Given the link scheduling algorithm and knowing the packet generation characteristics of connections, we should then be able to determine, subject to the local QoS constraints, the number of connections from each class that can be accepted (setup) on a link using some packet level analysis (e.g., [4]). The result is thus a set of link states satisfying the local QoS constraints, i.e. the link's schedulability region. Note that each link will typically have a different schedulability region because links have different capacities, are used by different sets of classes,

[2] Experiences with circuit-switched networks show that this restriction results in simple and efficient routing schemes [3].

etc. Section 3 explains this in more detail.

We assume that the network uses a link-state routing algorithm, where each node maintains a view of the whole network. This view is updated by periodic broadcasts by nodes of the status of their outgoing links. For ease of presentation, we assume that broadcasts of all nodes are synchronized and that they reach other nodes instantaneously (it will be apparent later that we can easily model unsynchronized broadcasts). After each update, a node uses its new view to compute new routes to be used for incoming connections until the next broadcast. Routes are thus updated at discrete time instants $nT, n = 1, 2, \cdots$, where T is the time interval between two successive broadcasts. Without loss of generality, we assume $T = 1$.

We assume a request for a connection setup on a multi-link route is sent to all links of the route simultaneously. For example, if during some period $[n, n+1)$ only route $(1, 3, 2)$ is used for incoming connections of voice 1 service, then the arrival rate of connection setup requests for class 2 on *both* links $(1, 3)$ and $(3, 2)$ during $[n, n+1)$ equals λ_1. Knowing the routes used in a period $[n, n+1)$, it is then easy to determine, for each link, the arrival rate of setup requests during $[n, n+1)$ for the different classes using the link.

2.1 Calculating Blocking Probabilities

Considering a Markov chain over the states of the link belonging to its schedulability region, we can estimate (as we explain below) the *instantaneous* blocking probability during $[n, n+1)$ for each class using the link, i.e., the instantaneous probability that a connection for that class can not be admitted on the link since otherwise the local QoS constraints would be violated and hence the end-to-end QoS requirements would not be guaranteed.

From the instantaneous blocking probabilities on all links along the route of a class, we can determine the end-to-end instantaneous blocking probability of the class (or equivalently its instantaneous throughput) during $[n, n+1)$, assuming link independence and using the fact that a connection can be setup on the route if and only if it is not blocked on all links of the route. In particular, if $B_k^l(t)$ is the class-k instantaneous blocking probability on link l where link l lies on route R_k of class k, then the class-k end-to-end instantaneous blocking probability equals $1 - \prod_{l \in R_k}(1 - B_k^l(t))$.

We assume the following: A connection setup request is tried on the route selected for the connection by the source node. If the setup fails at any node along the route, the request is considered lost (rejected) and the connection blocked. The setup request is *not* attempted on another (alternate) route.

We assume that the information a node broadcasts to other nodes at time $n+1$ consists of the instantaneous blocking probabilities on the outgoing links averaged over $[n, n+1)$ (and possibly other measures as will be discussed in Section 4). It is based on this information that new routes are computed for the next interval $[n+1, n+2)$.

To summarize, our solution to evaluating the performance of the network is iterative: (i) Given the routes to be used in one routing update period, compute blocking probabilities (and throughputs); (ii) Given the blocking probabilities, compute the routes to be used in the next routing update period as defined by the routing algorithm.[3]

Now, we explain how we estimate the instantaneous link blocking probabilities.

The case of a single link with single class. We make use of a technique that was previously used in [2] in the context of single-rate circuit-switched networks. In particular, [2] uses the following differential equation for $x(t)$, the average number of calls carried on a single link:

$$\dot{x}(t) = -\mu x(t) + \lambda[1 - B(t)] \tag{1}$$

where λ is the arrival rate of calls to the link, $\frac{1}{\mu}$ is the average lifetime of a call, and $B(t)$ is the instantaneous call blocking probability. Equation (1) can be viewed as a fluid equation where $\mu x(t)$ is the rate of output flow and $\lambda[1 - B(t)]$ is the rate of input flow. We don't know an expression for $B(t)$. We know that at steady-state, it can be obtained from the Erlang-B formula [7], i.e., $B = E(\frac{\lambda}{\mu}, C)$ where $\frac{\lambda}{\mu}$ is the offered traffic intensity, C is the maximum number of calls the link can support (i.e. total number of channels), and

$$E(\frac{\lambda}{\mu}, C) = \frac{(\frac{\lambda}{\mu})^C / C!}{\sum_{n=0}^{C} (\frac{\lambda}{\mu})^n / n!}$$

We also know that, at steady-state (when $\dot{x} = 0$), $x = \frac{\lambda}{\mu}[1 - B]$ where x is the steady-state value of $x(t)$.

An approximation of $B(t)$, denoted by $\tilde{B}(t)$, can be obtained as follows:

$$\begin{aligned} x(t) &= z(t)[1 - E(z(t), C)] \\ \tilde{B}(t) &= E(z(t), C) \end{aligned} \tag{2}$$

[3] Recently, we have used a similar iterative discrete-time approach to study the interaction between link scheduling and routing in dynamic connectionless (datagram) networks [10, 11].

The above approximation introduces $z(t)$ as a fictitious instantaneous traffic intensity, so that the Erlang-B formula which is known to hold in the steady-state case is used as well in the transient case.

Knowing $x(t)$ at some fixed t we can solve equation (2) for $\tilde{B}(t)$ using the iteration:

$$z^{(q+1)}(t) = \frac{x(t)}{1 - E(z^{(q)}(t), C)}, \quad q = 0, 1, \cdots \tag{3}$$

We start with some $z^{(0)}(t)$ until $z^{(q)}(t)$ stabilizes and converges to $z(t)$. Then, we have $\tilde{B}(t) = E(z(t), C)$. Note that $\tilde{B}(t)$ is a function of $x(t)$. By numerically integrating equation (1), we can now obtain the time behavior of $x(.)$, and hence $\tilde{B}(.)$.

The case of a single link with multiple classes. We can apply the same technique described above to our problem in order to obtain the instantaneous link blocking probabilities. First, consider the simple situation of a network consisting of only one link. Let Γ be the set of classes using the link. Let $x_k(t)$ be the average number of connections in a class $k \in \Gamma$ at time t. Let λ_k be the rate of setup requests for class k, and μ_k be the average lifetime of a class-k connection. Then we can write the following differential equation (similar to the differential equation (1)), for every $k \in \Gamma$:

$$\dot{x}_k(t) = -\mu_k x_k(t) + \lambda_k[1 - \tilde{B}_k(t)] \tag{4}$$

$\tilde{B}_k(t)$ represents the instantaneous blocking probability seen by class-k connections. It is estimated using the following equations, where the $z_k(t)$'s are fictitious traffic intensities:

$$x_k(t) = z_k(t)[1 - \tilde{B}_k(t)] \qquad \text{for all } k \in \Gamma \tag{5}$$

We can solve iteratively for the $\tilde{B}_k(t)$'s and the $z_k(t)$'s as in the single class case. In a manner similar to the use of the Erlang-B formula in equation (2), we can compute $\tilde{B}_k(t)$, for every $k \in \Gamma$, in terms of the $z_k(t)$'s, by solving (for the steady-state blocking probability for each class k) the Markov chain over the link's schedulability region S. In particular, the probability $P(\sigma)$ of being in a state $\sigma = (\sigma_1, \sigma_2, \cdots, \sigma_{|\Gamma|}) \in S$ is equal to $P(0)\prod_{k=1}^{|\Gamma|} \frac{z_k^{\sigma_k}(t)}{\sigma_k!}$ where $P(0) = \{\sum_{\sigma \in S} \prod_{k=1}^{|\Gamma|} \frac{z_k^{\sigma_k}(t)}{\sigma_k!}\}^{-1}$ is the normalization constant [7]. Then $\tilde{B}_k(t) = \sum_{\sigma \in S} 1(\sigma_k + 1 \notin S)P(\sigma)$. This result can be used in equations (5) to iteratively solve for the $z_k(t)$'s as in equation (3).

The case of multiple links, multiple classes. Now consider the situation of a general network with multiple links. We divide the interval $[n, n+1)$ between two successive routing updates into subintervals of length τ each. We write for every link j a *difference equation* (similar to the differential equation (4)) using τ as the discrete-time step. Let L be the set of network links, i.e., $j \in L$. Let Γ^j denote the set of classes using link j. Let $x_k^j(.)$ be the average number of connections in a class $k \in \Gamma^j$. Then we can write for every $k \in \Gamma^j$:

$$x_k^j(n + (i+1)\tau) = (1 - \mu_k\,\tau)x_k^j(n + i\tau) + \lambda_k\,\tau \prod_{link\ l\ \in R_k} [1 - \tilde{B}_k^l(n + i\tau)], \quad i = 0, 1, \cdots, \frac{1}{\tau} - 1 \tag{6}$$

The first term in the right-hand side of equation (6) represents the average number of class-k connections setup on link j which remain on link j (i.e. do not terminate). The second term represents the average number of new class-k connections setup on link j during $[n + i\tau, n + (i+1)\tau)$. λ_k is as defined before, i.e., it denotes the rate of setup requests for class k. In a general network, λ_k depends on the routing algorithm, and here we use its value for the interval $[n, n+1)$. $\tilde{B}_k^l(.)$ represents the blocking probability seen by class-k connections on link l where link l lies on route R_k of class k. (Obviously link j also lies on R_k.) Note that the product term $\prod$ reflects the fact that a connection can be setup on the route if and only if it is not blocked on every link of the route (this invokes the link independence assumption).

For every link $l \in L$, $\tilde{B}_c^l(.)$ can be estimated from the following equations:

$$x_c^l(n + i\tau) = z_c^l(n + i\tau)[1 - \tilde{B}_c^l(n + i\tau)], \quad \text{for all } c \in \Gamma^l \tag{7}$$

Again, equations (7) are solved iteratively similar to equations (5), i.e. involving a multi-dimensional Markov chain over l's schedulability region.

Figure 2 summarizes our evaluation method as described to this point. We denote by $R(n)$ the routing pattern during $[n, n+1)$, and by $\bar{x}^l(t)$ the vector representing the average number of connections in each class using link l at time instant t.

3 Scheduling and Admission Control

In Section 2, we assumed that a connection has an end-to-end delay requirement D, where D is the maximum packet delay. This is called a *deterministic* delay bound in the literature. Henceforth, instead of the

```
begin
  1.  Given the network topology and services, determine for each link $l$ the set of classes
      using it along with their local QoS requirements.
  2.  For each link $l$, determine its schedulability region $S^l$ (* as described later in Section 3 *).
  3.  Initialize $\bar{x}^l(0)$ for each link $l$ (* $\bar{x}^l(0) = 0$ for empty initial network *).
  4.  Initialize $z_c^l(0)$ arbitrarily for every link $l$ and $c \in \Gamma^l$.
  5.  For $n = 0, 1, 2, \cdots$
          begin
            6.  Using $\bar{x}^l(n)$ for all $l$ and possibly other information, compute the routing pattern
                $R(n)$ as defined by the routing algorithm (* more details in Section 4 and [9] *).
            7.  From $R(n)$, determine arrival rates of setup requests to every link $l$.
            8.  For $i = 0, 1, \cdots, \frac{1}{\tau} - 1$
                    begin
                      9.  For each link $l$
                          begin
                            10.  Solve the multi-dimension Markov chain on $S^l$ using $z_c^l(n + i\tau)$
                                 for all $c \in \Gamma^l$ to obtain $\tilde{B}_c^l(n + i\tau)$.
                            11.  Compute for every $c \in \Gamma^l$, new $z_c^l(n + i\tau) \leftarrow \frac{x_c^l(n+i\tau)}{1-\tilde{B}_c^l(n+i\tau)}$.
                            12.  If new $z_c^l(n + i\tau)$ and $z_c^l(n + i\tau)$ not close for all $c \in \Gamma^l$
                                     begin
                                       13.  $z_c^l(n + i\tau) \leftarrow$ new $z_c^l(n + i\tau)$, for every $c \in \Gamma^l$.
                                       14.  Go to step 10.
                                     end
                          end
                      15. For each link $l$, compute $\bar{x}^l(n + (i + 1)\tau)$ using equation (6).
                    end
          end
end
```

Figure 2: Proposed evaluation method.

deterministic delay bound, we assume a connection requests an end-to-end *statistical* delay bound (D, ϵ), i.e., Prob [packet delay $> D$] $< \epsilon$. This is typically required by applications such as voice since they can tolerate some packet loss (a packet is considered lost if its delay exceeds D). If a connection with an end-to-end delay requirement (D, ϵ) is to be established on an n-link route, then we require that each link on the route guarantees $(\frac{D}{n}, \frac{\epsilon}{n})$ [12], which now becomes the local QoS requirement.

We now illustrate how to find the schedulability region S of a link under a scheduling algorithm we adopt, i.e. the set of all link states that satisfy the local QoS constraints, where a link state denotes the number of connections in each class using the link.

We assume a "per-connection" link scheduling algorithm of the weighted round-robin type[4], where each connection is allocated (and guaranteed) a certain amount of link bandwidth required for its local delay requirement to be satisfied. This required bandwidth depends of course on the local delay requirement (D', ϵ') of the connection. It also depends on the packet generation characteristics of the connection.

For a connection described by a two-state model where the connection is either in a busy state sending packets back-to-back at peak rate or in an idle state sending no packets at all, the required bandwidth[5], denoted by $\hat{C}$, can be obtained from the following approximation derived in [4]:

$$\hat{C} = R\frac{\alpha - X + \sqrt{[\alpha - X]^2 + 4X\rho\alpha}}{2\alpha} \tag{8}$$

where

- R is the peak rate of the connection.
- m is the mean rate of the connection.
- b is the average duration of the busy period.
- $\alpha = ln(\frac{1}{\epsilon'})b(1 - \rho)R$.
- $\rho = \frac{m}{R}$ is the probability that the connection is active (in busy state).
- $X = D' \times \hat{C}$ is the buffer space required by the connection.

Note that for a given connection, $\hat{C}$ can be computed from equation (8) iteratively.

[4] An example of this type of scheduling algorithms is weighted fair queueing [13].

[5] Often referred to as effective or equivalent capacity [4].

For each class using the link, we can then determine $\hat{C}$ and X for a connection in this class. Knowing these $\hat{C}$'s and X's, we can determine whether a state belongs to the link's schedulability region; it must satisfy the following two conditions:

- The sum of the $\hat{C}$'s at this state is no greater than the link capacity C.
- The sum of the X's at this state is no greater than the total available buffer space of the link.

We refer the reader to [9] for a simplification that allows us to use a *one-dimensional* link model rather than the above multi-dimensional model, thus reducing computational complexity.

4 Routing

Recall that we assume each node periodically computes new routes based on its current view of the whole network. This view consists of the network topology and load during the last period. The load information consists of link/path measurements, which may include quantities such as the average reserved link capacity. These quantities should obviously be measurable in practice; indeed a node can measure the average reserved capacity for each of its outgoing links from the connection setup/teardown procedure. We should also be able to obtain these quantities from our model. Indeed we can obtain the average reserved link capacity from the link state and the effective capacity for each of the link's classes, which we compute in our model.

We are interested in route selection algorithms for networks of *arbitrary* topologies and offering *heterogeneous* services. We want algorithms that result in low blocking probabilities (a high successful setup rate) and hence high network throughput. In [9], we list some design choices when developing such algorithm. We also use our model to compare different route selection policies assuming the link scheduling algorithm and admission control described in Section 3.

5 Numerical Results for Single Link

We consider the case of a single link used by multiple service classes. We consider 10 classes whose parameters are shown in Table 1. We assume every connection in a class requires a fixed amount of bandwidth. We obtain instantaneous performance measures through equations (6) and (7).

We compare the results obtained using our approach with those obtained using discrete-event simulation. Figure 3 shows the time behavior of the total throughput for the parameters given in Table 1. In our approach, we take the discrete-time step τ to be 0.1. In the discrete-event simulation, the total throughput at time instant t ($t = 1, 2, \cdots$) is defined to be the total number of connections admitted on the link in the interval $[t-1, t)$. Our approach yields results very close to the exact values. In addition, we found our approach much less time-consuming than simulation. This is especially because the latter requires the averaging of a large number of independent simulation runs.

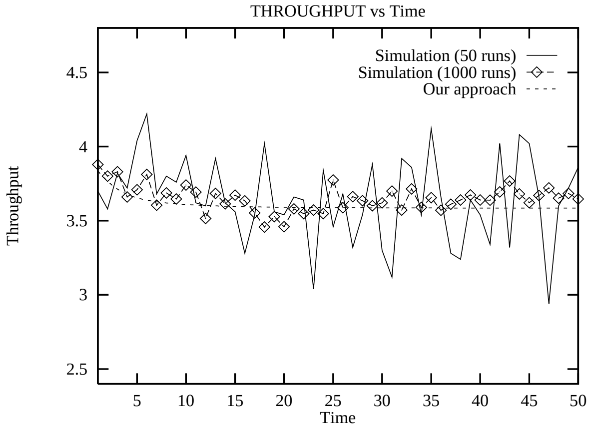

Figure 3: Throughput versus time for single link.

6 Numerical Results for a Tandem Network

Here, we validate our link independence assumption manifested in equation (6) by the product term. We consider a simple 3-node tandem network. Each link has a total bandwidth of 200. We assume that the 10 service classes, whose parameters are listed in Table 1, arrive at the top node. The last node is their destination node.

Figure 4 shows the instantaneous network throughput obtained using our approach and using discrete-event simulation. Although, in this situation, the link dependency is strong, our approach only slightly underestimates the throughput. More accurate results are expected for general networks where the link independence assumption is reasonable. We will report more validation results in a future paper.

7 Conclusions and Future Work

We presented an iterative discrete-time formulation and numerical solution procedure that gives approximate, yet sufficiently accurate, performance measures for reservation-oriented networks. Our results indicate the computational advantages of our approach over discrete-event simulation. We have also applied our approach to compare different routing algorithms.

There are several areas for future work in using our approach to investigate various routing, schedul-

Class	Required bandwidth	Arrival rate	Average lifetime
1	30	0.125	5
2	15	0.5	1
3	50	0.2	2
4	10	0.1	2
5	40	0.125	1
6	25	0.5	0.5
7	30	1.0	0.5
8	10	0.0625	10
9	5	1.0	0.2
10	50	0.25	2

Table 1: Parameters of the 10 service classes.

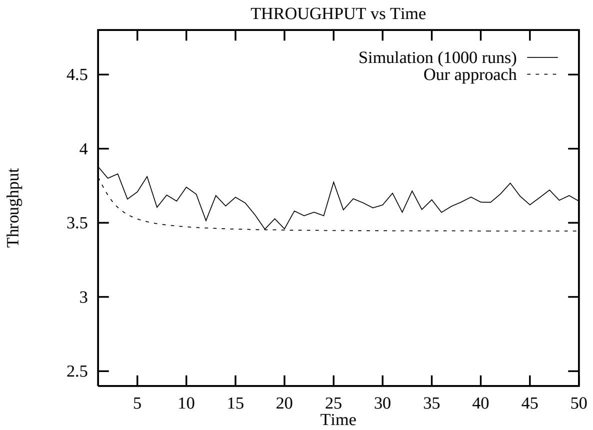

Figure 4: Throughput versus time for 3-node tandem.

ing, and admission control schemes on large networks. One area is to examine admission controls that block setup requests of connections of some types, even if their admission is feasible, possibly in order to reduce the chance of future blocking of connections of other types. Another area is to investigate policies for dividing the end-to-end QoS requirement among the links of a route other than the equal allocation policy we considered in this paper. Other QoS requirements such as packet loss will be considered.

References

[1] J. Filipiak. *Modeling and Control of Dynamic Flows in Communication Networks*. New York: Springer-Verlag, 1988.

[2] F. Le Gall and J. Bernussou. An Analytical Formulation for Grade of Service Determination in Telephone Networks. *IEEE Transactions on Communications*, COM-31(3):420–424, March 1983.

[3] A. Girard. *Routing and Dimensioning in Circuit-Switched Networks*. Addison-Wesley Publishing Company, 1990.

[4] 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.

[5] 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.

[6] D. Kincaid and W. Cheney. *Numerical Analysis: Mathematics of Scientific Computing*. Brooks/Cole Publishing Company, 1991.

[7] L. Kleinrock. *Queueing Systems*, volume I and II. New York: Wiley, 1976.

[8] W-C. Lau and S-Q. Li. Traffic Analysis in Large-Scale High-Speed Integrated Networks: Validation of Nodal Decomposition Approach. In Proc. *IEEE INFOCOM '93*, San Francisco, California, 1993.

[9] I. Matta and A.U. Shankar. An Iterative Approach to Comprehensive Performance Evaluation of Integrated Services Networks. Technical Report CS-TR-3235, Department of Computer Science, University of Maryland, College Park, MD 20742, March 1994. Available by anonymous ftp at ftp.cs.umd.edu:pub/MaRS/Papers.

[10] I. Matta and A.U. Shankar. On the Interaction between Gateway Scheduling and Routing. In Proc. *IEEE MASCOTS*, pages 84–88, Durham, North Carolina, January 1994. Available by anonymous ftp at ftp.cs.umd.edu:pub/MaRS/Papers.

[11] I. Matta and A.U. Shankar. Type-of-Service Routing in Dynamic Datagram Networks. In Proc. *IEEE INFOCOM*, pages 992–999, Toronto, Ontario, Canada, June 1994. Available by anonymous ftp at ftp.cs.umd.edu:pub/MaRS/Papers.

[12] R. Nagarajan, J. Kurose, and D. Towsley. Local Allocation of End-to-End Quality-of-Service in High-Speed Networks. In Proc. *IFIP TC6 Workshop on Modelling and Performance Evaluation of ATM Technology*, page 2.2, January 1993.

[13] 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.