Performance Evaluation of Dependable Real-Time Communication with Elastic QoS†

Jong Kim
High Performance Comp. Lab.
Dept. of CSE
Pohang Univ. of Sci. & Tech.
Pohang, 790-784, KOREA
Email: jkim@postech.ac.kr

Kang G. Shin
Real-Time Comp. Lab.
Dept. of EECS
The University of Michigan
Ann Arbor, MI 48109-2122
Email: kgshin@eecs.umich.edu

Abstract

When a client requests a real-time connection that requires an excessive amount of resources and/or a very high level of QoS (Quality-of-Service), the network service provider may have to reject the request, and in such a case, only a small number of connections could be accepted. On the other hand, if the client, out of fear of the service provider's rejection, requests only the minimum level of QoS, s/he may receive only bare-bone service even when there are plenty of resources available. One way of utilizing resources efficiently is to specify flexible (elastic) QoS requirements that can be adapted to the availability of network resources.

Recently, Han and Shin [1] proposed to allocate one primary channel and one or more backup channels to each *dependable real-time (DR-) connection. One drawback of this scheme is the severe reduction in number of DR-connections that can be accommodated, due mainly to the need for reserving resources for backups. This is equivalent to wasting precious resources in the absence of faults as far as the system's ability of accepting DR-connections is concerned. Using elastic QoS for this dependable real-time communication service, one can accept substantially more DR-connections and improve the utilization of resources efficiently and significantly.*

In this paper, we analyze the dependable real-time communication service with elastic QoS. Fault-tolerance is achieved by allocating one backup channel to each DR-connection. A Markov model is developed and used to analyze the average QoS level allotted to the primary channel of each DR-connection. Our evaluation results show that the proposed Markov model accurately represents the behavior of DR-connections with elastic QoS.

Keywords — Elastic Quality-of-Service (QoS), fault-tolerant real-time communication, Markov model.

1 Introduction

Advances in network technology have significantly improved the connectivity and the link bandwidth of point-to-point networks like the Internet. The application domain of the Internet has also been expanded to include time-critical/sensitive applications, such as remote medical services, multimedia, computer-supported collaborative work, and electronic commerce. Real-time communication has now become an essential service for these and many other applications. The real-time communication service provides a guaranteed "Quality-of-Service"(QoS), such as bounded message delay, delay jitter, and error rate. Considerable efforts have been made to provide timeliness guarantees necessary for the above-mentioned applications. See the survey paper by Aras *et al.* [2] for a detailed account of many of the existing real-time communication schemes.

Most of the real-time communication schemes known to date share three common properties [1]: QoS-contracted, connection-oriented, and reservation-based. Before actually transferring any message, a contract must be established between a client and the network. In the contract, the client specifies his input traffic-generation behavior and required QoS. The network then computes the resource needs from this information, finds a path between the client and his server/receiver, and reserves the necessary resources along the path. Messages will be transported only along

† The research reported in this paper was supported in part by the Office of Naval Research under Grant N00014-99-1-0465 and the Institute of Information Technology Assessment of Korea. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the ONR nor the IITA.

the selected path with the necessary resources reserved, and this unidirectional virtual circuit is called a *real-time channel* [3].

Besides the performance QoS mentioned above, there are growing needs for communication services with a guaranteed level of fault-tolerance for applications like remote medical services and command & control systems, which are not only time-critical but also reliability-critical. Failure of communication service jeopardizes the timeliness guarantees of these critical applications, and persistent faults like power outage and cable disconnection — most common failures reported by the Internet service providers — usually last for a long period of time. Restoration of real-time communication service after occurrence of a persistent network component failure is time-consuming, and to make things worse, there is no guarantee that such a restoration attempt will always succeed, especially when the network is congested, or tries to recover from multiple near-simultaneous component failures. Only recently, researchers have begun investigation of the fault-tolerance issues of real-time communication [1,4].

Fault-tolerance of real-time communication service can be achieved by allocating one *primary* channel and one or more *redundant/backup* channels to each real-time connection [1,4]. A redundant channel may be passive [1] or active [4]. In the passive approach, a backup channel that satisfies the dependability QoS requirement, which may be totally link-disjoint or maximally link-disjoint from its corresponding primary channel,[1] is allocated to each real-time connection after allocating a primary channel. No messages are transferred along a backup channel until it is activated upon occurrence of a fault to the corresponding primary channel. Multiple backup channels that traverse a link and collectively require more bandwidth than the link capacity, can share the same resources as long as they are not activated at the same time (i.e., overbooking link bandwidth for backups). In the active approach, multiple primary channels are created for each real-time connection and messages with redundant information (for the purpose of fault-tolerance) are transported along the multiple primary channels of the connection. The active approach requires more resources than the passive approach, since in the passive approach, the redundant resources allocated for fault-tolerance are not used until the backups are activated. In this paper, we will focus on the passive approach.

Our proposed approach (1) reserves only the amount of resources to provide the minimum required performance QoS for each real-time connection at the time of its establishment, and (2) allocates, at run-time, more resources, if available, to the connection. In the latter case, the connection will receive better performance QoS than its bare minimum required, yielding more "utility" for the client/application and hence contributing more revenue to the network service provider. Note that the performance QoS requirement of a connection is usually specified as a *single* number, and this value is often the minimum QoS requirement in order to accept as many real-time connections as possible, while (minimally) satisfying the clients' QoS requirements. This single-value QoS model has commonly been used in the QoS negotiation between clients and the network service provider. If the specification of performance QoS is given as a *range* instead, then it would be possible to improve the performance QoS at run-time using un-allocated resources or inactive backup resources. This type of flexible QoS is called *elastic QoS* [5,6].

The performance evaluation of dependable real-time communication is essential for the analysis of network service behavior and the future planning of the network. It also enables prediction of the behavior of an application on a given network. To the best of our knowledge, there does not exist any model that can analyze dependable real-time (DR-) connections with elastic QoS. Moreover, most performance QoS evaluation studies are based on simulations. By contrast, we develop an analytic performance evaluation model of DR-connections with elastic QoS.

The remainder of this paper is organized as follows. Section 2 presents the details of dependable real-time communication and the various elastic QoS models. Section 3 describes the performance evaluation model of dependable real-time communication with elastic QoS. Section 4 presents the performance evaluation results and discusses the modeling accuracy against the detailed simulation results. Finally, the paper concludes with Section 5.

2 Preliminaries

2.1 Fault-Tolerant Real-Time Communication

2.1.1 Real-time channel

The realization of real-time channel generally consists of two phases [3, 7]: off-line channel establishment and run-time message scheduling. In the channel-establishment phase, the network manager computes the resource need from the client's traffic specification and finds a path that has the resources necessary to meet his QoS requirement. In the run-time message scheduling phase, each link resource manager schedules messages belonging to different real-time channels to satisfy their respective timeliness requirements. The channel-establishment phase is of prime importance to the realization of a real-time channel, and thus, the focus of this paper.

A real-time channel is established as follows. First, a client specifies his traffic-generation behavior and required QoS from which the network manager then computes the resource need from this information. The network manager selects a route between the source and destination of the channel along which sufficient resources can be reserved to meet the client-specified delay and buffer requirements. The number of possible routes between two communicating

[1] if there does not exist any link-disjoint backup path between the source and destination

peers could be very large, so that selecting a route for each real-time channel could be a time-consuming task.

There are two approaches to route selection: centralized and distributed [7, 8]. Most existing schemes are based on the centralized approach [8]. In the centralized approach, there is a global network manager which maintains the information about all the established real-time channels, the topology, the resource availability and commitment of the network. Hence, the network manager can select an "optimal" route for each requested real-time channel. All clients that want to establish real-time channels send their requests to the central network manager, and the manager accept the requests if there are enough resources. The network manager also informs all intermediate nodes on the selected route of the establishment of each new channel, and provides the information necessary for the run-time scheduling of this channel's messages.

In the distributed approach, each node maintains the information of the entire network by exchanging link-state information with all other nodes, so that the source or destination can determine a route for a newly-requested channel solely based on the information kept in each node. The link-state information exchanged to get the information of the entire network is minimal for best-effort communication. However, to set up a real-time channel, the detailed information of each link this channel traverses must be exchanged between nodes, inducing substantially more overhead than the minimal information that needs to be exchanged in the original best-effort protocol. The distributed route-selection approach is more attractive than the centralized approach that suffers the performance and reliability bottleneck problem.

Route selection in the distributed approach requires to search routes and perform admission tests on them [7]. Generally, the search and test is done at the same time. There are two approaches to finding candidate routes: sequential and parallel. In the sequential approach, all possible routes are checked one by one until a qualified one is found or all possibilities are exhausted. Shortest routes are picked and checked first, sequentially one by one. Complete search of all possible routes could be very time-consuming. In the parallel search, all possible routes are searched concurrently [7, 9]. The flooding scheme [9] falls into the category of parallel approach. Although this scheme is fast in finding a route, it induces a large traffic overhead. A bounded flooding algorithm [7] is proposed to reduce this traffic overhead.

2.1.2 Fault-tolerant communication

Connection dependability is of great importance to many applications where service disruption has a serious negative impact. When a real-time channel is disabled by a component failure, a new channel that does not run through the failed component should be established before resuming the data transmission. However, such channel re-establishment attempts can fail because of resource shortage at that time (as a result of network congestion or contention among multiple simultaneous recovery attempts). Even when such a channel is re-established successfully, it may make a seriously detrimental impact on the underlying application if it takes a long time. To handle network component failures, several fault-tolerant communication schemes have been proposed [1, 4, 10]. The common characteristic of these approaches is to use redundant resources.

Fault-tolerant communication schemes can be classified as *active* or *passive*. In the active approach, the redundant resources are always used for actual data transfer. Multiple-copy [10] and disperse [4] routing schemes belong to this category. In the multiple-copy scheme, more than one copy of each message are transmitted through link-disjoint routes. The more message copies are transmitted, the more resources are wasted in the absence of failure. In the disperse-routing scheme [4], a single message with error recovery information is divided into multiple small messages which are then transmitted through link-disjoint routes. This scheme reduces the waste of resources, but still the redundant resources are actively used.

In the passive approach, the redundant resources reserved for recovery from component failures stay inactive during the absence of failure, or the normal operation. In the backup-channel approach [1], the network first establishes a primary channel, then sets up a link-disjoint backup channel for quick recovery from a component failure. Although the backup channels require reservation of resources, they don't "consume" the resources until they are activated as a result of component failures. The amount of resources to be reserved for backup channels can be reduced by multiplexing multiple backups, or overbooking resources. The backup channel multiplexing will not degrade the communication dependability as long as not all of the backup channels are activated at the same time (thus not exceeding the resource capacity). The resources reserved for backup channels can, in the absence of component failures, be used for transporting non-real-time traffic.

2.2 Elastic QoS

As mentioned earlier, a connection's performance QoS requirement is usually specified as a single value that represents the client's bare minimum QoS requirement. Specification of higher performance QoS by a client may result in the network's rejection of the requested channel, or the blocking of future real-time channel requests. It is therefore desirable to specify each client's QoS requirement with multiple values or a range, so that a channel, after its acceptance based on its minimum requirement, can (1) receive more resources and hence higher QoS, if more resources are available, and (2) release resources beyond its minimum required if there is resource shortage. We call this type of QoS "elastic QoS."

There are two different models for elastic QoS: range QoS [5, 11] and interval QoS [12, 13]. In the former, the QoS requirement is specified as a range that covers the value guaranteeing the minimum performance to that guaranteeing the best performance. In the interval model, QoS is expressed in the form of k-out-of-M within a fixed time interval, meaning that at least k but less than or equal to M packets should arrive within a fixed time interval. The link manager can selectively ignore a packet as long as it can satisfy the minimum k-out-of-M requirement. The range QoS is applied to offline channel establishment and the interval QoS is applied to run-time channel management.

The range QoS model requires an adaptation policy to adjust the QoS of existing channels when there is a change in the amount of available resources of a link, as a result of admitting a new channel or terminating an existing channel. There are two adaptation schemes: one is the max-utility scheme [11] and the other is the coefficient scheme [5]. In the max-utility scheme [11], each client specifies the utility value of his channel. When more resources become available, the system allocates these extra resources so as to maximize the system's total "utility" or reward. This scheme allows a real-time channel to monopolize all the extra resources even when its utility is slightly higher than the others. In the coefficient scheme [5], each client specifies the coefficient value of his real-time channel. When more resources become available, the extra resources are allocated proportionally to the coefficient value of each channel.

The range QoS model is very useful for fault-tolerant real-time communication. The resources reserved for backup channels cannot be used in the single-value QoS model, although the amount of required resources can be reduced using the backup-channel multiplexing. However, in the range QoS model, these reserved resources can be utilized as extra resources to enhance performance-QoS. When a backup channel is activated due to the occurrence of a failure, then all channels using the extra resources retreat their QoS to their minimum required. This way, all channels can safely continue their service while at least meeting their minimum QoS requirements. When there are extra resources available after activating a backup channel, the extra resources are re-allocated to the existing channels.

In this paper, we will focus on the performance evaluation of fault-tolerant real-time channels, or DR-connections, with the range QoS model.

3 The Performance Evaluation Model

We first overview the overall operation of DR-connections with elastic QoS. We then discuss the network performance modeling and propose an evaluation model. Finally, we discuss how to obtain the parameters needed to analyze the model.

3.1 Network Operation

A client requests a DR-connection with the specification of his traffic-generation characteristics and QoS requirements. The QoS requirements consist of two parts: performance and dependability QoS. The performance QoS can be presented in various forms such as packet-delivery deadline, maximum network delay, or bandwidth. We assume that one form of performance QoS can be transformed into another, and vice versa. To guarantee a given delivery deadline, the maximum network delay should be less than the difference between the issuance time and deadline of each packet. Also, to limit the network delay below a certain value, one must reserve enough network resources in advance. We assume that the performance-QoS requirement is given in the form of bandwidth. (This is not a restriction because, as mentioned above, it can be converted to other performance-QoS parameters.)

The elastic QoS model adopted here is the min-max range QoS model. The client specifies the minimum bandwidth required to satisfy the minimum performance QoS requirement, the maximum bandwidth used to achieve the best performance QoS, and the utility/reward he will achieve when extra resources are allocated.

The dependability QoS is presented as single-value QoS and used to guarantee that each DR-connection is assigned to have one backup channel even if component failures occur in the network.

When a client requests a new DR-connection, the network floods, within a bounded region around the client, the request to find routes from the source to the destination that have enough resources to meet the connection's QoS requirement. Any node that received this request tries to forward it with its bandwidth allowance to all of its neighbors except the node which the request came from. However, if there is not enough bandwidth to be allocated to the newly-requested connection, or a request copy received earlier[2] has a better bandwidth allowance, the new request copy will be discarded. Those request copies that exceed the specified flooding bound will also be discarded. This flooding continues until there are no more copies of the request forwarded, or a certain time limit is reached.

As soon as the destination node receives a request,[3] it sends a confirmation message with a resource reservation request back to the source along the route in the direction opposite to that the request had taken to reach the destination. Each intermediate node on the route reserves the required resources as the confirmation message travels toward the source node. This first route is used as the *primary channel* of the requested DR-connection. The destination waits until it receives more copies of the same request traversed different routes. If the route of the primary channel and the route determined by a request copy arrived later have better dependability QoS than the required value, the destination sends a *backup channel confirmation* to the source along the second route in the reverse direction. This second route is used as the *backup channel* of the requested DR-connection.

Resources, if available, necessary to provide the minimum acceptable level of the requested connection's QoS

[2] A node may receive more than one copy of the same request.

[3] The request arrived first at the destination is likely to have traversed the shortest path between the source and the destination.

will be reserved along the primary-channel route. If not enough resources are available, then each link on the primary-channel path reclaims the extra resources allocated to the existing primary channels of other DR-connections. After allocation of the minimum amount of resources on each link to the new primary channel, the remaining available resources will be distributed to all of the primary channels on the link according to their utility values.

The resource reservation for a backup channel is slightly different from that for a primary channel. The link controller/manager first tries to multiplex the backup of the newly-requested connection with the existing backups on the link, i.e., sharing the resources already reserved for the other existing backup channels. This multiplexing is possible only when no other backup channels need to be activated simultaneously with the new backup due to a single component failure that might occur somewhere else in the network. If this is not possible, then one must reserve additional resources for the new backup by using the same procedure of reserving resources for primary channels.

When a component failure occurs in the network, all backup channels whose primaries traverse the failed component must be activated. At this time, all of the existing primary channels that share links with the activated backup channels should release their extra resources allocated beyond their minimum required, since some of the extra resources are reserved for the backups to be activated (but temporarily borrowed by primary channels to enhance their QoS). After the activation of backup channels, the extra resources that still remain available are distributed to the existing primary channels according to their utility values.

There are two cases in which the primary channels can increase their reservation of resources: (1) when a DR-connection is released/ terminated, and (2) when a new "indirectly-chained" DR-connection arrives. When a DR-connection has completed its service, the resources reserved for that connection are released, and the primary channels that have shared links with this terminating connection can now reserve more resources. In the second case, two channels that do not share any link are said to be *indirectly-chained* when there is a third channel that traverses at least one link of both channels' paths.

3.2 The Analysis Model

To analyze the performance of DR-connections with elastic QoS, one can consider two plausible approaches. The first approach is to analyze the network behavior, such as the amount of message traffic increased and the number of DR-connections that can be accommodated. This analysis approach reflects a *network-centric* view. The second approach is to analyze the channel behavior, such as the average bandwidth reserved. This approach reflects a *channel-centric* view. In this paper, we take the second approach or the channel-centric view, since our main concern lies in the analysis of channel behavior.

The performance metric considered here is the average bandwidth reserved for each primary channel.[4] The average bandwidth reserved is important in predicting the behavior of a primary channel before actually reserving resources for the primary channel.

Prior to the analysis, one must have knowledge of the network behavior, most of which was described in the previous subsection. One thing which was not discussed there is the amount of change in the bandwidth reserved. If channels are allowed to have *any* bandwidth between the minimum and the maximum specified in the elastic QoS requirement, the resource management would become unmanageably complex. A slight change in the available resources for a channel will trigger re-adjustment of the resources reserved for the channel, and this could occur very frequently, thus overloading the network. In practice, it is desirable to change resource reservation only when there are noticeable changes in the available resources. Also, it would be better to change resource reservation in multiples of the minimum amount of resources, known as the *increment size*. This way, the network can handle the available resources easily and efficiently. We assume that the increment size is given and the interval between the minimum and the maximum resources is an integral multiple of the increment size.

We use the following notation/symbols in describing the analysis model.

λ : DR-connection request arrival rate. We consider that the request inter-arrival time is exponentially distributed with rate λ.

μ : DR-connection termination rate. We also consider that the interval between two successive DR-connection terminations is exponentially distributed with rate μ.

γ : Link failure rate. The inter-arrival time of failures is exponentially distributed with rate γ.

B_{min} : The minimum bandwidth that can be reserved for a DR-connection. If the network cannot even reserve this minimum amount of bandwidth, the request is rejected.

B_{max} : The maximum bandwidth that can be reserved for a DR-connection.

Δ : The increment size of bandwidth.

N : The number of different bandwidths (measured in increment sizes) that a DR-connection can have. $N = 1 + \lceil \frac{B_{max}-B_{min}}{\Delta} \rceil$.

S_i : The state representing the bandwidth reservation for a DR-connection, where $i = 0, \ldots, N-1$. S_0 represents that the reserved bandwidth is B_{min}, S_{N-1} represents that for B_{max}, and S_i for $B_{min} + i \times \Delta$.

P_f : The probability that a channel shares at least one link with the newly-arrived channel.

[4]Note that only minimum required, or less resources are reserved and remain unchanged for backup channels.

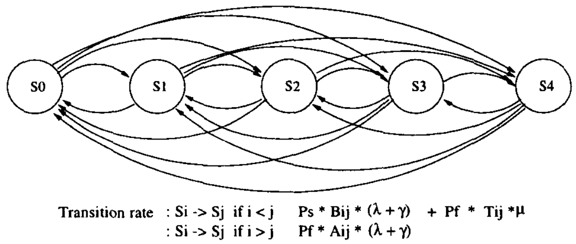

Figure 1. A Markov chain with 5 states and transitions.

P_s : The probability that a channel is indirectly-chained with the newly-arrived channel.

$A_{i,j}$: The transition probability from state S_i to state S_j, $i > j$, caused by the arrival of a new channel. Directly-chained channels take this transition.

$B_{i,j}$: The transition probability from state S_i to state S_j, $i < j$, caused by the arrival of a new channel. Indirectly-chained channels take this transition.

$T_{i,j}$: The transition probability from state S_i to state S_j triggered by the termination of an existing channel.

The arrival rate λ and the termination rate μ are assumed the same since we only analyze the steady-state behavior. If one is greater than the other, the network will have a continuously increasing or decreasing number of connections, which is not stable.

We analyze the system using a Markov chain with N states. The Markov-chain model is constructed from the viewpoint of a single primary channel, but this is representative of all channels. A Markov chain with 5 states is shown in Figure 1.

Each state S_i in the figure represents a primary channel with $B_{min} + i \times \Delta$ bandwidth. The transition from S_i to S_j indicates that there is a change in the bandwidth reservation. There are three cases in which a primary channel experiences a change in its bandwidth reservation: arrival of a primary channel, termination of an existing primary channel, and activation of a backup channel. When a new primary channel arrives, all the existing primary channels that share at least one link with the new channel should release their extra resources (beyond their required minimum), which are then re-allocated to the primary channels according to their utility values. This will give two consecutive state transitions from S_i to S_0 and S_0 to S_j. Since the release and re-allocation of the extra resources occur instantly, a direct transition from S_i to S_j is specified in the figure. Note that there are only downward state transitions, i.e., from S_i to S_j where $i > j$. The transition rate is $P_f \times A_{i,j} \times \lambda$. P_f represents the probability that a channel has at least one link shared with the newly-arrived channel. $A_{i,j}$ represents the probability that there is a transition from S_i to S_j and λ represents the arrival rate of new primary channels.

The resources released by the existing channels, due to the arrival of a new channel or the activation of backup channels, remain un-allocated at a link on which the newly-arrived channel does not pass through. These resources are considered extra resources and are allocated to the existing channels on that link, which do not have any link overlapping with the newly-arrived channel. These existing channels are "indirectly-chained" channels. The transition rate is $P_s \times B_{i,j} \times \lambda$. P_s represents the probability that a channel is indirectly chained with the new channel. Note that there are only upward transitions, i.e., from S_i to S_j where $i < j$.

Another case in which an existing channel can increase resource reservation is the termination of an existing channel. Only those channels that share at least one link with the terminating channel can enjoy this benefit. The transition rate corresponding to this case is $P_f \times T_{i,j} \times \mu$ where μ represents the channel termination rate.

The last case is the occurrence of a fault. Fault occurrence to a link triggers activation of the backup channels running through the failed link. The activated backup channels reclaim their resources which were given as extra resources to other co-existing primary channels to temporarily enhance their QoS. This will cause the primary channels to return their extra resources. If there still remain extra resources after the activation of backup channels, they will be re-distributed to the existing primary channels. This transition rate is $P_f \times A_{i,j} \times \gamma$ where γ represents the link failure rate.

3.3 On obtaining parameters

In the previous subsection, we presented a Markov model to analyze DR-connections with elastic QoS. So far, we described all the parameters associated with state transitions. Next, we discuss how to determine these parameters.

The link failure rate is a network-dependent parameter and can be obtained from network service providers. Parameters such as DR-connection arrival and termination rates are application-dependent and can be obtained from application service providers. Parameters such as the probabilities of overlapping with the new channel and being indirectly-chained are network-dependent parameters, and are also partly related to applications. When the underlying network is a regular-topology network, these probabilities depend solely on the network topology and the average number of hops of channels. However, since the network we consider is a public network such as the Internet, it is almost impossible to parameterize these probabilities analytically. Moreover, since a channel may not take a shortest route due to the insufficient amount of available resources, the analysis based on network topology will differ in a real situation. A better approach would be to obtain them from real networks with running applications. This is possible and simple for network service providers to achieve by measuring and analyzing network activities. In this paper, we obtained these parameters through detailed simulations. Likewise, the probabilities of transitioning from one state to another when a new DR-connection request arrives or an existing connection terminates, are obtained through simulations.

4 Numerical Results and Discussion

This section presents and discusses our evaluation results of the proposed Markov model. The performance metric used is the average bandwidth reserved for primary channels.

The environments considered for obtaining the numerical results are as follows. The bandwidth of each link is 10Mbps. Although the bandwidth of each link in the Internet is different, we assume that the bandwidth is the same for all links in a given network. This kind of environment can be easily found in intranets, and it is not difficult to relax the assumption. The minimum bandwidth required by a DR-connection is 100Kbps and the maximum bandwidth required is 500Kbps. For example, a video service requires at least 100Kbps for recognizable continuous images and 500Kbps for a high-quality image. The size of bandwidth increment or decrement is 50Kbps or 100Kbps. The only difference between the two increment sizes is the number of states in the Markov chains.

The probabilities P_f, P_s, $A_{i,j}$, $B_{i,j}$, and $T_{i,j}$ are obtained using simulations. The simulation environments are as follows. A random network is generated using the GT Internetworking Topology Models (GT-ITM) package [14]. The generated network is a random or transit-stub network with 100 nodes [14]. The parameters used to generate networks are described in each figure showing the results if necessary. We measured the probabilities P_f and P_s after setting up a certain number of DR-connections. Also, we generated and terminated randomly a certain number of DR-connections while maintaining the number of DR-connections in the network close to the initial number of DR-connections to measure the transition probabilities $A_{i,j}$, $B_{i,j}$ and $T_{i,j}$. The Markov models are solved using the SHARPE package [15].

Figure 2 shows the average bandwidth of a DR-connection as the number of DR-connections in the network increases. The network is generated using the Waxman distribution [16] with parameters $\alpha = 0.33$ and $\beta = 0$ where α and β are the parameters of Waxman distribution. The number of nodes in the generated network is 100 and the number of edges is 354. The average degree of connection is 3.48 and average diameter is 8. The DR-connection arrival rate (also termination rate) is given 0.001 and the link failure rate is given 0 to see only the effect of the new DR-connection arrival and termination. The utilities of all connections are the same for fair distribution of resources. The solid line represents the simulation results and the dashed line with the x marks represent the analytical results using a 9-state Markov chain (Δ = 50Kbps). The top dotted line represents the ideal average bandwidth of the network when all the network resources are utilized and equally distributed

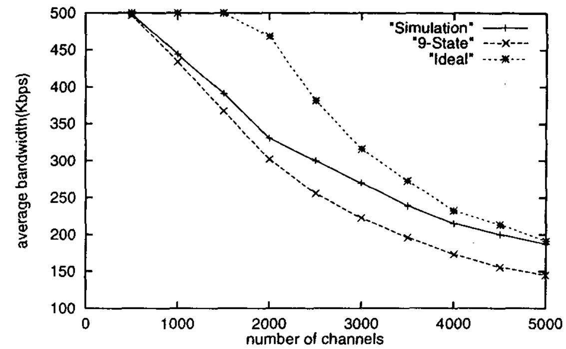

Figure 2. An average bandwidth when the number of DR-connections increases.

to DR-connections in the network. This ideal average bandwidth is computed by the following formula:

$$\frac{bandwidth\ of\ one\ link}{avg.\ no.\ of\ realtime\ channels\ on\ one\ link} = \frac{BW \times Edge}{NChan \times avghop}$$

where BW represents the bandwidth of one link, $Edge$ represents the number of edges in the network, $NChan$ represents the number of channels in the system, and $avghop$ represents average hops of channels. The small discrepancy between the simulation and analytical results is due to the difference between the assumption on the network topology and the reality of the generated network topology. We assumed that all the nodes in the network show same behaviors, but in reality, the leaf nodes have different behaviors as compared to the non-leaf nodes.

Table 1 shows the average bandwidth for different increment sizes of bandwidth. The label "Random" in the table represents a random network using the Waxman model and "Tier" represents a transit-stub network model [14]. Note that the actual number of DR-connections in the "tiered" network is much less than the number of connections shown in the left column. This is because most DR-connections are rejected due to the shortage of bandwidths in the transit-stub network. The number of connections shown in the left column represents the number of connections which have been tried to be set up. The table shows no difference in the average bandwidth even though they have a different number of states. The two schemes show a similar average behavior, but the scheme with a smaller increment size provides bandwidth close to the average bandwidth. However, the scheme with a smaller increment size changes its bandwidth more frequently than the scheme with a larger increment size.

Figure 3 shows the average bandwidth when the number of nodes in the network varies. The number of nodes is varied from 100 to 500. Networks are randomly generated using the Waxman distribution with $\alpha = 0.33$ and

Table 1. Comparison of average bandwidth of the Markov chains with different numbers of states.

No. of channels	Random		Tier	
	5	9	5	9
1000	432.49	434.48	270.12	271.64
2000	298.48	301.98	238.43	240.52
3000	220.80	222.69	252.05	255.96
4000	173.50	173.15	259.19	264.63
5000	146.97	145.29	241.02	244.64

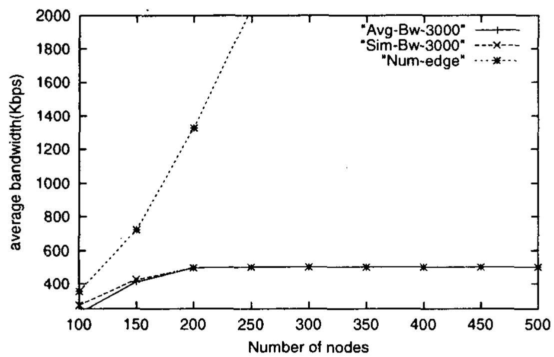

Figure 3. Average bandwidth when the number of nodes is varied.

$\beta = 0$. The number of DR-connections loaded is 3000. The solid line represents the analytical results and the dashed line represents the simulation results. The upper dotted line represents the number of edges in the randomly-generated network. The number of edges increases rapidly with the number of nodes when the parameters of Waxman distribution remains unchanged.

Figure 4 shows the effect of link failures in the network. The network is randomly generated with 100 nodes and 354 edges. A Markov chain with 9 states is used to evaluate the effect. The failure rate is varied from 0.001 to 0.0000001. The DR-connection request arrival and termination rates are set to 0.001. The solid (labeled "Avg2000ft") and dotted (labeled "Avg3000ft") lines represent the average bandwidth of real-time channels when there are 2000 and 3000 real-time channels in the network, respectively. The result shows no effect of link failures on the average bandwidth since the link failure rate is too small compared to the DR-connection request arrival and termination rates.

5 Conclusion

In this paper, we have modeled and evaluated the performance of dependable real-time connections with elastic

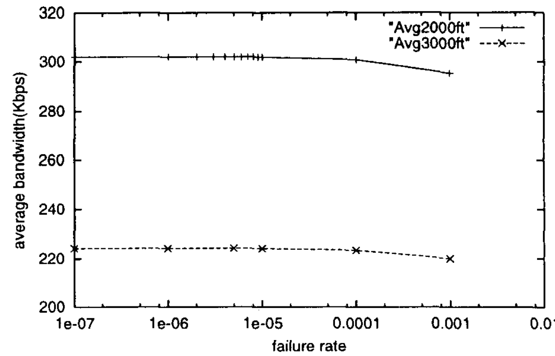

Figure 4. Average bandwidth when the link failure rate is varied.

QoS. Fault-tolerance is achieved with the backup-channel reservation scheme and the elastic QoS described by a min-max model. Our analysis is based on the development of a Markov model for the dynamics of DR-connections. Three parameters characterizing transitions between states are arrival of a new DR-connection, termination of an existing DR-connection, and activation of backup channels to recover from a component failure. The probabilities of transitioning to different states are functions of network topology and network congestion. Since the network considered here is a random point-to-point network like the Internet, it is almost impossible to find closed-form expressions for these transition probabilities. Since these probabilities must be obtained from real networks, we derived them using realistic simulations. Using the Markov model, we have analyzed the average bandwidth reserved for each DR-connection and shown the trend of average bandwidth change caused by the increase of network load. The proposed analysis model can be expanded to include other issues related to, for example, network traffic.

References

[1] S. J. Han and K. G. Shin, "Fast restoration of real-time communication service from component failures," in *Proc. of ACM SIGCOMM*, pp. 77–89, 1997.

[2] C. M. Aras, J. F. Kurose, D. S. Reeves, and H. Schulzrinne, "Real-time communication in packet switched networks," *Proceedings of IEEE*, vol. 82, pp. 122–139, Jan. 1994.

[3] D. D. Kandlur, K. G. Shin, and D. Ferrari, "Real-time communication in multi-hop networks," *IEEE Trans. on Parallel and Distributed Systems*, vol. 5, pp. 1044–1056, Oct. 1994.

[4] A. Banerjea, "Simulation study of the capacity effects of dispersity routing for fault-tolerant realtime

channels," in *Proc. of ACM SIGCOMM*, pp. 194–205, 1996.

[5] G. C. Buttazzo, G. Lipari, and L. Abeni, "Elastic task model for adaptive rate control," in *Proc. of RTSS*, pp. 286–295, 1998.

[6] C. Lee, D. Siewiorek, R. Rajikumar, and J. Hansen, "A scalable solution to the multi-resource qos problem," in *Proc. of RTSS*, pp. 315–326, 1999.

[7] S. K. Kweon and K. G. Shin, "Distributed qos routing using bounded flooding," Tech. Rep. CSE-TR-388-99, The University of Michigan, Ann Arbor, Michigan, 1999.

[8] K. G. Shin, C.-C. Chou, and S. K. Kweon, "Distributed route selection for establishing real-time channels," *IEEE Trans. on Parallel and Distributed Systems*, vol. 11, pp. 318–335, Mar. 2000.

[9] D. Ferrari and D. C. Verma, "A scheme for real-time channel establishment in wide-area networks," *Journal of Selected Areas in Communications*, vol. SAC-8, pp. 368–379, Apr. 1990.

[10] P. Ramanathan and K. G. Shin, "Delivery of time-critical messages using multiple copy approach," *ACM Trans. Computer Systems*, vol. 10, pp. 144–166, May 1992.

[11] S. J. Han, *Fast low-cost failure recovery for real-time communication in multi-hop networks*. PhD thesis, The University of Michigan, Ann Arbor, MI, 1998.

[12] G. Koren and D. Shasha, "Skip-over: Algorithms and complexity for overloaded systems that allow skips," in *Proc. of RTSS*, pp. 110–117, 1995.

[13] M. Caccamo and G. Buttazzo, "Exploiting skips in periodic tasks for enhancing aperiodic responsiveness," in *Proc. of RTSS*, pp. 330–339, 1997.

[14] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *Proc. of IEEE Infocom*, 1996.

[15] C. Hirel, R. A. Sahner, X. Zhang, and K. S. Trivedi, "Reliability and performance modeling using sharpe 2000," in *Computer Performance Evaluation / TOOLS 2000*, pp. 345–349, 2000.

[16] B. M. Waxman, "Routing of multipoint connections," *IEEE Journal of Selected Areas in Communication*, vol. 6, pp. 1617–1622, Dec. 1988.