Reliable Backup Routing in Fault Tolerant Real-Time Networks

V Rajeev
Software Engineer
Hughes Software Systems
Gurgaon
India 122015
rvadali@hss.hns.com

C. R. Muthukrishnan
Department of Computer Science and Engineering
Indian Institute of Technology
Madras
India 600036
crm@iitm.ernet.in

Abstract

Broadband integrated services digital networks (B-ISDN) are aimed to transport both real-time traffic and non real-time traffic. Many of these applications require Quality of Service (QoS) guarantees. In the literature, not much work is found to provide QoS guarantees based on fault tolerance. In this work, reliability of network links is considered as one of the parameters when providing QoS guarantees to applications. Considering reliability of network links as a parameter for QoS guarantees gives the applications more flexibility in choosing the network resources. A new terminology for dispersity routing is presented which will be useful in providing QoS guarantees based on reliability. Dispersity routing transmits the traffic along multiple paths. Also, a reliable backup resource allocation method is presented that can be used in the context of dispersity routing for fault tolerant real-time networks. Here, an assumption is made that higher capacity is assigned to the links which are more reliable. This will help in availability of resources for a longer period of time. Also, reliability of links is considered to compute multiple paths along with shortest path metric.

Keywords: Dispersity routing, Quality of Service (QoS), Fault tolerance, Reliability.

1. Introduction

Emerging distributed real-time applications require guaranteed end-to-end quality of service (QoS) and have stringent constraints on delay and cost. In [1], dispersity routing at physical layer is introduced. Here, unlike conventional routing procedure, which routes a message along a single path between source and destination, this routing technique, sub-divides the message and disperses it through multiple paths in the network. In [2], the dispersity routing is proposed at application layer considering the advances in workstation speeds. Moving dispersity routing to the high level layer provides increased flexibility and lower cost, as the application can choose the level of reliability required. Dispersity routing is characterized by the triple (N, K, S). The variable "N" stands for the number of paths between source and destination. Real-time traffic is passed on "K" paths. The rest of the paths "N - K" are used to pass redundant information to reconstruct the message in the event of loss in the network. The variable "S" limits the maximum number of paths over which a link can be shared. The various dispersity routing schemes are simulated in [3]. It is shown that fragmenting the traffic among multiple paths in the network utilizes the network capacity to a greater degree and provides fault tolerance at the same time. In [4, 5], a set of algorithms are discussed that route and reserve resources along multiple paths. It is shown that when there are multiple paths in a network, resource reservation along multiple paths is advantageous over single path reservation. In [6], a study was made in the areas of network architecture and protocols for supporting real-time services in packet switched networks. In [7, 8, 9], it was made possible to support real-time communication in packet-switched networks. This can be interpreted that packet-switched networks can provide guaranteed QoS to distributed real-time applications. Recently in [10, 11], a study was made about QoS in packet-switched networks and the issues in it. In [12, 13], it is shown that large amount of spare resources may seriously degrade the attractiveness of the backup channel scheme. The idea of backup multiplexing is proposed in [12, 13]. The idea is to use the probability of link failure to reserve a small fraction of spare resources at each link. Dynamics of client requirements and performance in real-time systems are dealt in [14]. It presents Dynamic Connection Management (DCM) scheme, which addresses issues such as network availability and flexibility.

1531-2216/01 $10.00 © 2001 IEEE

2 Capacity assignment in links of a network

Assigning capacity to the links of a network is an important issue. In earlier works [2, 3, 12, 13], it is assumed that the capacity is assigned to a link of a network depending on the incoming traffic. A link which is expecting higher traffic is assigned higher capacity. In our work, we assume that capacity is assigned to a link based on the incoming traffic and reliability of that link. A link with more incoming traffic or with higher reliability is assigned higher capacity. Hence, categorizing links based on reliability will help in maximizing the availability of network resources for longer amount of time.

3 Computing paths based on reliability

It is essential to provide fault tolerance to applications and also use the available resources efficiently. In earlier works [2, 4, 5], paths are computed depending on shortest path and delay constraint metrics. In our work, we suggest that there is a need to compute paths based on reliability, with shortest path and delay constraint metrics. This way applications can choose stable paths which could remain active most of the time. It should be noted here that, there is a tradeoff between various metrics when multiple metrics are considered for computing paths. Hence, in our work we limit the number of reliable paths used so that we could guarantee delay constraints. Reliable paths are computed using probability of link failures. It is assumed in our work that the probabilities of link failures are known a priori from previous studies on the network. Reliability of a path is calculated as shown in Equation 1, where p1, p2..., pn are the probability of link failures of each link in the path.

$$Reliability\_path = (1 - p1) * (1 - p2)... * (1 - pn) \quad (1)$$

In our work, multiple paths are computed in two phases. In first phase, the required number of reliable paths are computed using reliability of links keeping in mind the delay constraints of the application. In second phase, the rest of the multiple paths are computed based on minhop metric and also delay constraints of the application. The paths that are computed in the second phase are link disjoint with respect to the reliable paths computed in first phase.

4 Proposed Dispersity routing with reliable backup

The resources used for backup are randomly chosen in earlier dispersity routing schemes mentioned in [2, 3]. To make effective resource allocation for backup paths, the reliability of a path is considered in selecting backup paths. Reliability of a path depends on the probability of link failures of the links in that path. We assume that probability of each link failure in the network is known.

4.1 New terminology for dispersity routing

In our work, a new dispersity routing terminology is defined. Here, any dispersity routing request is characterized by (N, K, S, R, B). The variables "N" and "K" represent the usual meaning defined in earlier work. The variable "S" is relevant to computed paths based on shortest path metric and does not refer to reliable paths. The new term "R" represents number of reliable paths among "N" available paths. The term "B" is an array of "R" elements and each value of "B" corresponds to the number of backup channels allocated in each of the "R" reliable paths. The terms "R" and "B" gives flexibility to the applications to choose the QoS guarantees depending on the requirements. We use all the "N" paths (N=K) to transmit the real-time traffic. We make sure of the fact that, we can establish more than one channel on one path. For backup resources, we establish additional channels from source to destination in the reliable path where we already have one channel for real-time traffic. Hence, QoS guarantees based on fault tolerance is provided by varying the number of reliable backup paths and also varying number of backup channels in each reliable path.

4.2 Our Dispersity routing schemes with reliable backup

Dispersity routing with reliable backup can be classified based on the fault tolerance required for an application. To tolerate single faults in the network, dispersity routing scheme $(N, N, 1, 1, \{1\})$ is used and to tolerate double faults when there exists only one reliable path, dispersity routing scheme $(N, N, 1, 1, \{2\})$ is used.

The proposed dispersity routing request $(4, 4, 1, 2, \{1, 1\})$ to tolerate double faults is shown in Figure 1. Paths P1 and P2 are reliable paths (R1,R2) and backup channel ch2 is allocated in each reliable path to tolerate faults that may occur in paths P3 and P4. Channel ch1 of paths P1, P2, P3, and P4 is used to transmit real-time traffic. The corresponding illustration for earlier work which can tolerate double faults is shown in Figure 2 where paths P3 and P4 serve the purpose of backup paths. Here, paths P1 and P2 are used for transmitting real-time traffic.

It is observed from Figure 1 and Figure 2 that proposed work utilizes the network connectivity in the maximum way possible for real-time traffic where as earlier work do not utilize two paths for real-time traffic as they are reserved to tolerate double faults. When the network connectivity is low, keeping some paths separately as backup paths is an

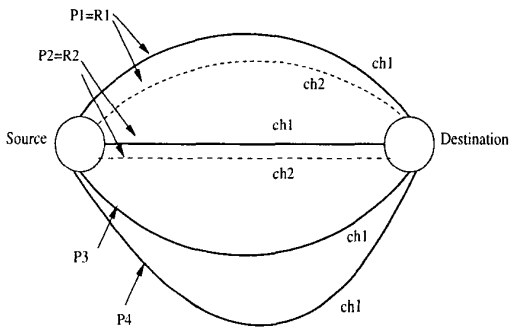

Figure 1. *Illustration for proposed* $(N = 4, K = 4, S = 1, R = 2, B = \{1, 1\})$ *dispersity routing to tolerate double faults*

overhead if the faults in a network occur less frequently. Also, the advantages of dispersity routing are lost when all the multiple paths are not used for real-time traffic in a network of low connectivity. In our work, the backup channels are established in a reliable path. As reliable path is also being used for real-time traffic, there will be no overhead in allocating backup channels as was the case in earlier work.

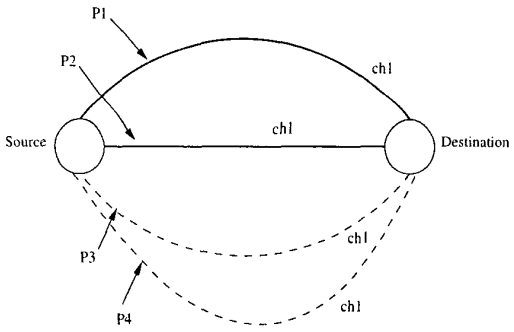

Figure 2. *Illustration for earlier* $(N = 4, K = 2, S = 1)$ *dispersity routing to tolerate double faults*

5 Alternate approaches when reliable link fails

It is still the case (with low probability) that fault may occur in the reliable path. This will result in slow degradation of the system. To avoid this slow degradation of the system, one of the following three approaches can be considered.

1. *First approach* is to use the reserved bandwidth of another application which is not being used at that moment and make the dispersity routing scheme to tolerate without much degradation. The real-time traffic on the failed path is passed through the underutilized link. This is feasible as some applications reserve the peak bandwidth and may not use it at all the time. Hence, considering the inherent dynamics of client requirements we can tolerate faults.

2. *Second approach* is to block a channel of an application and give priority to the failed link real-time traffic. The application that has to be blocked will be the one with no requirements of time guarantees like file transfer application or an e-mail. We have to make sure that, blocking of a non real-time application is done for a short time.

3. *Third approach* is to route the failed links real-time traffic along with the remaining paths, which are active in the same application. This will surely degrade the QoS guarantees of an application and this approach is used as the last resort if the above two approaches fail to allocate a channel.

6 Performance Study and Results

The experiments are carried out on a network with 17 nodes and 59 bidirectional links as shown in Figure 3. The same network is also referred in earlier work [14]. The advantage of proposed work over earlier work is shown using blocking probability. Blocking probability is calculated as the number of requests rejected divided by the number of requests arrived. The simulation was performed for both earlier work and proposed work on a wide range of request arrival rates. The network load is generated at each node with independent arrival of real-time requests according to Poisson distribution and exponentially distributed holding time. The simulation is implemented assuming a single fault model. Fault events and restoration events are generated according to Poisson distribution. The request arrival rate is varied from 0.1 requests per millisecond to 4.0 requests per millisecond. The number of requests ranges from 1712 to 66650 in the set of experiments. The number of multiple paths used were restricted to make sure that all accepted requests satisfy delay constraints. In our work, 33% of the links are assumed to be reliable. A number of experiments were conducted to show the advantages of proposed work over earlier works [2, 3].

Earlier work was simulated with applications requesting for a dispersity route (3, 2, 1). In the earlier work, it has been observed that all single faults were tolerated. Here, first two paths are used for transmitting real-time data and third path is used as backup path. In the proposed

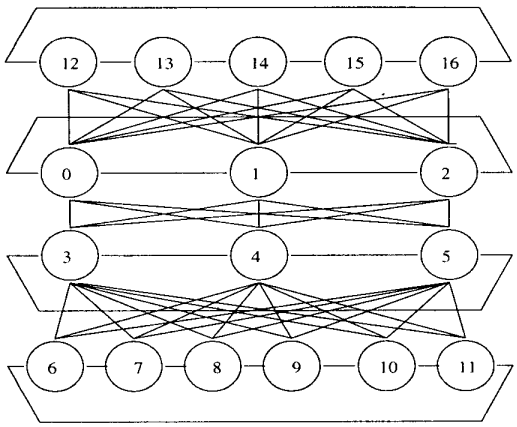

Figure 3. *Network (with 17 nodes and 59 links) used for experiments*

work, an application will request for a dispersity route with (2, 2, 1, 1, {1}). In dispersity routing (2, 2, 1, 1, {1}), first path is a reliable path and second path is a minhop path. In our work, it is known which links are reliable and which are not. Hence, reliable links are assumed to have higher link capacity. Also, when computing the routes for multiple paths, we take reliability of links and shortest path metric as the criteria. It can be noticed from Figure 4 that the blocking probability of earlier work is higher than the proposed work at all instants of request arrival rate. In the earlier work, all resources are assigned randomly and it does not give more preference to reliable links in assigning link capacity. The proposed work uses the reliable links in its favor by assuming that more link capacity is assigned to reliable links.

In the next experiment, it is assumed that both earlier work and proposed work use the same network where reliable links have higher capacity. It can be noticed from Figure 5 that the blocking probability of earlier work is higher than proposed work. It is observed that earlier work does not use efficiently the resources assigned to reliable links. Hence, it can be concluded that assigning high link capacity to reliable links cannot bring down the blocking probability of earlier work to the extent of proposed work. Computing multiple paths using reliability of links is also a factor in increasing the number of requests that are being serviced.

We also studied a case when a request comes for (3, 2, 1) dispersity routing and there is a possibility that (3, 2, 1) request may get rejected due to lack of three link disjoint paths or lack of link capacity. In the earlier work, when a (3, 2, 1) request is rejected it may opt for (3, 2, 2) dispersity routing. The shared link in earlier work is chosen without considering reliability of the link. In the proposed work, when a (3, 2, 1) request gets rejected, it will opt for (2, 2, 1, 1, {1}) dispersity routing. Here, one among the two paths is a reliable path. In the reliable path, two logical channels are established so that one channel can be used to tolerate faults. It has been observed from Figure 6 that the blocking probability for earlier work increases at a faster rate when compared to proposed work. Also, it is observed as shown in Figure 7, in earlier work as the request arrival rate increases the number of dispersity applications failing to tolerate single faults increases at a faster rate. In the proposed work, all the dispersity applications could tolerate single faults. Earlier work has higher blocking probability as (3, 2, 2) request is not using efficiently the higher capacity of the reliable link as in (2, 2, 1, 1, {1}) request of proposed work. In the earlier work, when computing the shared path it does not consider the reliability of the shared link(s). In the proposed work, since all shared links are reliable it could tolerate single faults.

In our work, when request (3, 2, 1) gets rejected, we are opting for (2, 2, 1, 1, {1}) dispersity routing. It was assumed that reliable links are close to each other which will result in complete reliable path between source and destination. When reliable links in a network are distributed evenly, then we may not always get a complete reliable path from source to destination. Hence, in our next experiment when request (3, 2, 1) gets rejected, we will opt for (3, 2, 2) dispersity routing and make sure that only one reliable link is shared. It is observed from Figure 8 that the blocking probability of proposed work is slightly higher than the earlier work. The slight increase in blocking probability is due to non availability of a shared path with one shared reliable link. It is also observed that all dispersity applications could tolerate single faults in proposed work unlike earlier work where there are some failures of dispersity applications as shown in Figure 9. Hence, there is a tradeoff between faults being tolerated and the number of requests being serviced.

7 Conclusions

We proposed a reliable backup approach for efficient resource reservation in dispersity routing for fault tolerant real-time networks. It is shown that identifying reliable links and using them efficiently will help in more requests being serviced. We have coined a new dispersity routing scheme (N, K, S, R, B) which gives more flexibility for an application to choose the required fault tolerance based on QoS guarantees.

References

[1] N. F. Maxemchuk, "Dispersity Routing", *Proceedings of ICC*, June 1975, pp. 41.10-41.13.

[2] A. Banerjea, "A Taxonomy of Dispersity Routing Schemes for Fault Tolerant Real-Time Channels", *Proceedings of ECMAST*, May 1996, pp. 129-148.

[3] A. Banerjea, "Simulation Study of the Capacity Effects of Dispersity Routing for Fault Tolerant Real-Time Channels", *Proceedings of ACM SIGCOMM*, August 1996, pp. 194-205.

[4] I. Cidon and R. Rom, "Multi-path Routing Combined with Resource Reservation", *Proceedings of IEEE INFOCOM*, April 1997, pp. 92-100.

[5] I. Cidon, R. Rom, and Y. Shavitt, "Analysis of Multi-path Routing", *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, December 1999, pp. 885-896.

[6] C. M. Aras, J. F. Kurose, D. S. Reeves, and H. Schulzrinne, "Real-Time Communication in Packet-Switched Networks", *Proceedings of the IEEE*, vol. 82, no. 1, January 1994, pp. 122-139.

[7] D. Ferrari, A. Banerjea, and H. Zhang, "Network Support for Multimedia: A Discussion of the Tenet Approach", *Computer Networks and ISDN Systems, Special Issue on Multimedia Networking*, July 1994, pp. 1267-1280.

[8] A. Banerjea, D. Ferrari, B. A. Mah, D. Verma, and M. Moran, "The Tenet Real-Time Protocol Suite: Design, Implementation, and Experiences", *IEEE/ACM Transactions on Networking*, vol. 4, no. 1, February 1996, pp. 1-10.

[9] A. Banerjea, E. Knightly, F. Templin, and H. Zhang, "Experiments with the Tenet Real-Time Protocol Suite on the Sequoia 2000 Wide Area Network", *Proceedings of ACM Multimedia*, October 1994, pp. 183-191.

[10] S. Chen and K. Nahrstedt, "An Overview of Quality-of-Service Routing for the Next Generation High-speed Networks: Problems and Solutions", *IEEE Network, Special Issue on Transmission and Distribution of Digital Video*, vol. 12, no. 6, November 1998, pp. 64-79.

[11] X. Xiao and L. M. Ni, "Internet QoS: A Big Picture", *IEEE Network Magazine*, vol. 13, no. 2, March 1999, pp. 8-18.

[12] S. Han and K. G. Shin, "Fast Restoration of Real-Time Communication Service from Component Failures in Multi-hop Networks", *Proceedings of ACM SIGCOMM*, September 1997, pp. 77-88.

[13] K. G. Shin and S. Han, "Fast Low-Cost Failure Recovery for Reliable Real-Time Multimedia Communication", *IEEE Network, Special Issue on Transmission and Distribution of Digital Video*, vol. 12, no. 6, November 1998, pp. 56-63.

[14] C. Parris, H. Zhang, and D. Ferrari, "A Dynamic Management Scheme for Real-Time Communications", *Proceedings of IEEE INFOCOM*, June 1994, pp. 698-707.

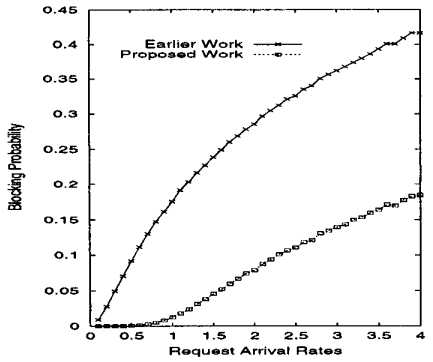

Figure 4. *Blocking Probability of proposed work (with higher link capacity in reliable links) in comparison with earlier work (without higher link capacity in reliable links)*

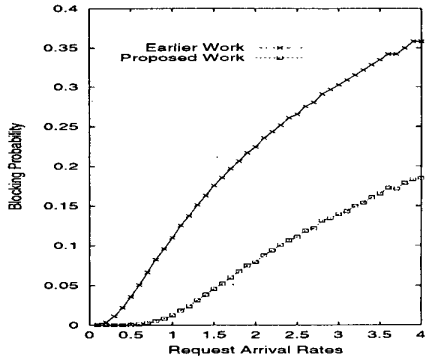

Figure 5. *Blocking Probability of proposed work (with higher link capacity in reliable links) in comparison with earlier work (with higher link capacity in reliable links)*

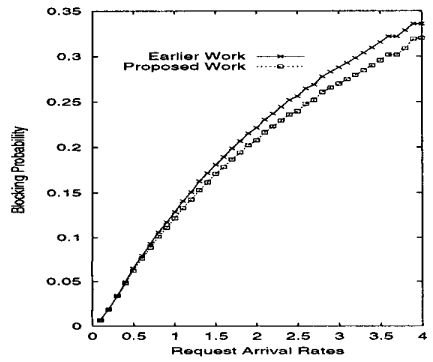

Figure 6. *Blocking Probability of proposed work (using reliable backup path) in comparison with earlier work for rejected (3, 2, 1) requests*

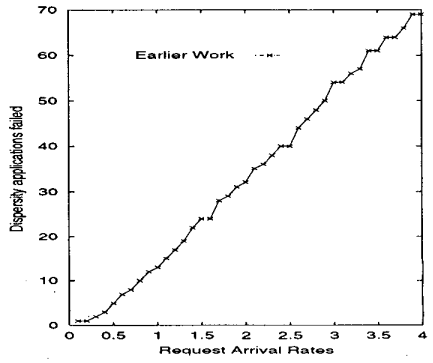

Figure 7. *Number of dispersity applications failed for earlier work with (3, 2, 2) dispersity routing*

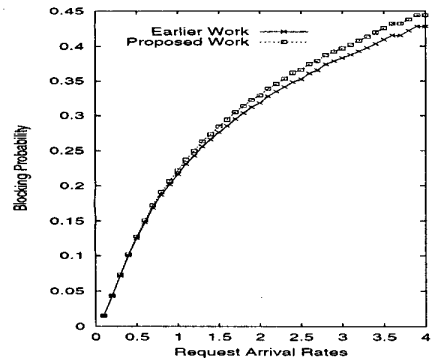

Figure 8. *Blocking Probability of proposed work (sharing of one reliable link) in comparison with earlier work for rejected (3, 2, 1) requests*

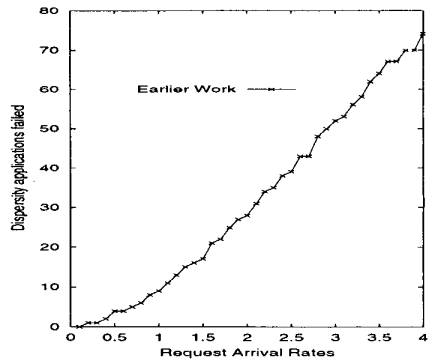

Figure 9. *Number of dispersity applications failed for earlier work (reliable links are distributed evenly) with (3, 2, 2) dispersity routing*