DEPARTMENT OF COMPUTER SCIENCE
COLLEGE OF SCIENCES
OLD DOMINION UNIVERSITY
NORFOLK, VIRGINIA 23529

PHOTONICS AND OTHER APPROACHES TO HIGH SPEED COMMUNICATIONS

By

Kurt Maly, Principal Investigator

Progress Report
For the period ended February 29, 1992

Prepared for
National Aeronautics and Space Administration
Langley Research Center
Hampton, VA 23665

Under
Research Grant NAG-1-908
Nicholas D. Murray, Technical Monitor
ISD-Systems Architecture Branch

Submitted by the
Old Dominion University Research Foundation
P.O. Box 6369
Norfolk, Virginia 23508-0369

June 1992

DEPARTMENT OF COMPUTER SCIENCE
COLLEGE OF SCIENCES
OLD DOMINION UNIVERSITY
NORFOLK, VIRGINIA 23529

PHOTONICS AND OTHER APPROACHES TO HIGH SPEED COMMUNICATIONS

By

Kurt Maly, Principal Investigator

Progress Report
For the period ended February 29, 1992

Prepared for
National Aeronautics and Space Administration
Langley Research Center
Hampton, VA 23665

Under
Research Grant NAG-1-908
Nicholas D. Murray, Technical Monitor
ISD-Systems Architecture Branch

Submitted by the
Old Dominion University Research Foundation
P.O. Box 6369
Norfolk, Virginia 23508-0369

June 1992

Photonics and Other Approaches to High Speed Communications

omit primary

Semi-Annual Report

submitted by

Old Dominion University

to

National Aeronautical Space Agency

June 9, 1992

Technical Contact:

Kurt Maly
(804)683-4817
Fax:(804)683-4900
Computer Science Department
Old Dominion University
Norfolk, VA 23529-0162
Internet: malycs.odu.edu

This report consists of:

- Summary
- References
- Copies of relevant publications

Summary

Network Research at the Computer Science Department Old Dominion University

Historical Perspective

Our research group of 4 faculty and about 10 - 15 graduate students has been actively involved (as a group) in the development of computer communication networks for the last five years. Many of its individuals have been involved in related research for a much longer period. The overall research goal is to extend network performance to higher data rates, to improve protocol performance at most ISO layers and to improve network operational performance. In the next section, we briefly state our research goals, then discuss the research accomplishments and direct your attention to attached and/or published papers. (Section III), which provide an in-depth review of our work. This is followed by a listing of the research support from funding agencies and companies and then by a very brief curriculum vitae of the Network Research Group members.

Present Research

Multilevel Parallel Networks

A considerable portion of the research effort has been directed toward the problem of parallel networking. We define multilevel parallelism in communications as the representation of a single user pairs' data stream (sender plus receiver) as a set of multiply concurrent data streams where the number of streams may vary from level to level. This encompasses intra-level parallelism. Our initial and a continued strong effort has been directed toward single level parallelism. Our goal is to use network parallelism to provide improved performance, flexibility and incremental network growth as well as added reliability. We developed an efficient media access protocol system, DRAMA, for MAN networks based upon parallel channel CSMA/CD [1, 2, 3, 4]. We were able to provide an extremely effective, dynamic load balancing scheme within and between node groups. A second research effort investigated media access/physical level parallelism for token ring networks. Particularly, we look for policies which assign incoming messages to a particular ring. Here, a minimum queue size policy is found to be effective in achieving higher throughputs while minimizing the resequencing problem for both homogeneous and heterogeneous loads [5, 6].

The main emphasis of our group, for the last two years, has been to investigate network multiple level parallelism. The work is a joint effort with the Fiber & Electro-optics Research Center at Virginia Tech University. In this context, our effort concentrated on a

general framework for parallelism in different layers in an OSI stack [7, 8, 9, 10, 32, 33]. A logical framework (see Figure 2 [32]) illustrates parallel operations at the application, transport, network and media access layers. Since each layer's parallelism differs, interfacing, with scheduling capability, couples each layers. Figure 2 also shows that significant layer interaction can occur between nodes. In this study a number of aspects of parallelism were investigated. Three architectures were compared to support parallel TCP/IP at the upper level. One data transfer request from a user could be served by one or more TCP processes, simultaneously. The presence of multiple FDDI channels, provides parallelism at the media-access/physical level. The results obtained through these investigations are available in [7, 8, 9, 10, 32, 33]. We feel that it is feasible to construct a multilevel parallel prototype network to obtain hundreds of Mbps throughput using presently available hardware technology. In addition, the proposal further develops parallel network modelling and understanding to the point where its performance improvements, reliability and flexibility capabilities would be much better understood and its utility available to a wide range of users.

High Performance Media Access

Our goal is to develop improved gigabit media access techniques. Electro-optical circuitry for these systems will be expensive since it must performance at very high clock rates. Therefore, we are seeking simple systems.

- **FDDI and DQDB** One aspect of our work concentrates on the 100 Mbps range protocols such as FDDI and DQDB [12, 13, 14]. The FDDI research was directed toward improving throughput, increasing data rates and extending operational capabilities to larger distance networks, i.e., MANs. In the DQDB area, we researched controls which significantly improved high load fairness without deteriorating the overall network throughput as with other suggested fairness control systems.

- **High Data Rate Electro-optical Network Media Access** We have developed a new high-speed ring-based, media-access protocol, Carrier Sensed Multiple Access/Ring Network - Circulating Reservation Packet Protocol (CSMA/RN - CRP) [15, 16, 17]. The system supports integrated traffic by a novel, non-interferring, dual access protocol where each protocol support a different traffic type. The protocol is shown to be eminently fair for wide range of traffic load distributions including the complete range of integrated traffic and for file server type traffic with out adding any fairness control. In case of unfairness, the CRP mechanism is shown to provide effective relief for starving nodes. In the attached paper[17], we show that CSMA/RN - CRP is superior to other gigabit ring network protocols in a number of significant areas.

- **Photonic Node** The photonic node research seeks to design networks (protocols and topologies) which are appropriate for use in NASA's next generation space vehicle. These networks should be designed to take advantage of advances in photonics. Our current research activities have been

  - To identify photonic devices currently available or in development which seem appropriate for use in networks of this kind.

- To list detailed functionality requirementsfor appropriate protocols
- To identify those functionalities for which a photonic implementation appears viable.

Based both on likely capabilities of available photonic devices and functional requirements, we can then identify where the transition from photonic to electronic processing should occur with the goal of more fully exploiting the potential bandwidth (greater than 10 THz) of single-mode fibers for LANs.

Network Management

Another area of work is related to network management. Our goal is to develop tools and techniques for effective, adaptive, dynamic management of complex LAN network topologies. Based upon optimization of user response time, tools are developed to observe actual traffic and present the results in a visually effective fashion so that the network administrator can reconfigure information within the network storage elements [18]. Additional tools which will help the manager obtain faster reconfiguration and better performance evaluation and prediction are under development.

Heterogeneous Network Connectivity

Recent research has been directed toward the development of effective bridge/gateway systems for heterogeneous network environments [19, 20]. Heterogeneous networks are bound to exist; for example, if one tries to add a link with FDDI or ATM technology to an already existing Ethernet. Serious interfacing problems occur for this configuration. The gateway provides coupling between networks with widely different data rates and protocol formats. To support this feature within the gateway system, identification of packets (connection oriented protocol) is required. However, it is shown that the same connection oriented structure can be used effectively to support both lossless congestion control (a real problem in heterogeneous networks) and link error correction which are significantly superior to similar end-to-end controls. There features provide a physical/data link/network layer which is operationally equivalent to transport layer features in XTP but which provides significantly improved performance.

Related Areas

Designing a distributed database system is a complex process. It requires solutions for a number of computationally complex problems. Data partitioning, data allocation, and query optimization are some of the key issues in this process. Though several approaches have been suggested few integrate the approaches. Also, many approaches are only suitable for small-scale problems. As a solution to this problem [21, 22, 23, 24, 25, 26] a methodology that employs heuristic algorithms to solve the individual problems and integrates the solutions by employing an iterative approach has been used. Load balancing is another important aspect for efficient use of resources and efficient transaction executions in distributed systems. While, several load balancing schemes designed for fully replicated distributed systems exist, very few are for partially replicated systems. We designed a three-phase heuristic approach

for load balancing in partially replicated systems. Availability and consistency are two important issues in the context of replicated distributed database systems prone to network partitions. We developed a new replica control algorithm combining the advantages of two algorithms resulting in a higher availability and at the same time not assuming the availability of up-to-date information of the partitions.

Another related area is distributed simulation where the goal is to speed up long-running discrete events simulations by using several CPUs which may be, for exmaple, distributed on a network. The effectiveness of this approach depends properties of simulation model and communication latencies among participating CPUs [27, 28, 29, 30, 31]. Identification of model decompostions which permit a speeding up the simulation and addressing of effective techniques for managing interactions among model componennts are the problems which must be resolved in order for distritubed simulation to be effective.

References

1. Sharrock, S.; Maly, K.; Ghanta, S. Du, H.: "A Broadband Integrated Voice/Data/Video Network of Multiple LANs with Dynamic Bandwidth Partitioning," INFOCOM'87 March 1987.

2. Maly, K. Overstreet, C. Xiao-ping, Q.; Tang, D.: "Dynamic Resource Allocation in a Metropolitan Network," SIGCOMM'88 pp. 13-24; Aug. 1988.

3. Maly, K., Foudriat, E.; Game, D.; Mukkamala, R.; Overstreet, C.: "Traffic Placement Policies for a Multi-Band Network," SIGCOMM'89 pp.

4. Maly, K., Foudriat, E.; Game, D.; Mukkamala, R.; Overstreet, C.: "Dynamic Allocation of Bandwidth in Multichannel Metropolitan Area Networks," Computer Networks & ISDN; 1992.

5. Mukkamala, R.; Foudriat, E.; Maly, K.; Kale, V.: "Modeling and Analysis of High Speed Parallel Token Ring Networks," ODU Tech. Report; 1991.

6. Mukkamala, R.; Foudriat, E.; Maly, K.; Kale, B.; Sekhar, Y.; Yerraballi, R.: "Message Assignment Policies for High Speed Parallel Networks," TBP 1st Int. Confer. on Computers and Communications; 1992.

7. Maly, K.; Khanna, S.; Overstreet, C.; Mukkamala, R.; Zubair, M.; Sekhar, Y.: "Multiprocessor Architectures for High Speed Networks: A Performance Study," TBP Proc. of the 12th World IFIP'92; Sept 1992.

8. Maly, K.; Overstreet, C.; Mukkamala, R.; Zubair, M.; Paterra, F.; Khanna, S.; Sekhar, Y.; Yerraballi, R.: "Parallelism for High Speed Networks," 6pp; Proc. of IEEE Workshop on Architectures & Implementation of High Performance Communication Subsystems; Feb 1992.

9. Maly, K.; Paterra, F.; Overstreet, C.; Mukkamala, R.; Khanna, S.: "Concurrent Use of Parallel Communication to Enable Remote Visualization," TBP Proc. of 4th Inter. Conference on Computing and Information, May 1992.

10. Maly,K; Paterra, F. Overstreet, C.; Mukkamala, R.; Zubair, M.; Khanna, S.; Weincko. J.; Srivatsan, H.: "Parallel Communications: A Performance Evaluation," ODU Tech. Report 91-25; 1991.

11. Maly, K.; Overstreet, C.; Mukkamala, R.: "Multilevel Parallelism for High Data Rate Networks," A Proposal Submitted to NSF; May 1992 .

12. Maly, K.; Game, D.: "Improvements for the Next Generation of FDDI," Inter. Phoenix Conference on Computers and Communication, pp. 710-716; March 1991.

13. Maly, K. Rao, N. Olariu, S.; Zhang, L.; Game, D.: "Average Waiting Time: Profiles of DQDB," 1991 Inter. Conference On Parallel Processing; pp 268-279; Aug. 1991.

14. Game; D.; Maly, K.: "Performance Analysis of High Speed Network Protocols: FDDI at Gigabit Speeds," 21st Pittsburgh Conf. on Modeling and Simulations; pp 1543-1547; May 1990.

15. Foudriat, E.; Maly, K.; Overstreet, C.; Khanna, S.; Paterra, F.: "A Carrier Sensed Multiple Access Protocol for High Data Rate Ring Networks," SIGCOMM Computer Communications Review, pp 59-70; April 1991.

16. Foudriat, E.; Maly, K.; Overstreet, C.; Khanna, S.; Zhang, L.; Sun, W.: "Combining Two Media Access Protocols to Support Integrated Traffic on High Data Rate Networks," Proc. of 16th LCN; Oct. 1991.

17. Foudriat, E.; Maly, K.; Overstreet, C.; Khanna, S.; Zhang, L.: "A Simple, Effective Media Access Protocol System for Integrated High Data Rate Networks," Submitted to IEEE Jour. on Selected Areas for Communication.

18. Maly, K.; Overstreet, C.; Pate, L.; Radhakrishnan, R.; Khanna, S.: "User Response Time Optimization in a Metropolitan Area Network: A Concept." IWACA'92: pp 199-210.

19. Foudriat, E.; Maly, K.; Overstreet, C.; Zhang, L.; Sun, W.: "High Performance Interconnection Between High Data Rate Networks," Proc. of Inter. Workshop on Advanced Comm. and Applications for High Speed Networks (IWACA'92); pp 441-450.

20. Foudriat, E.; Maly, K.; Overstreet, C.; Zhang, L.; Sun, W.: "Support for PC/Workstation Interconnection to High Speed Data Networks," TBP IFIP'92; Sept. 1992.

21. R. Mukkamala, S. C. Bruell, and R. K. Shultz: "Design of Partially Replicated Distributed Database Systems: An Integrated Methodology", Proc of ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems: 1988.

22. R. Mukkamala, S. C. Bruell, and R. K. Shultz: "A Heuristic Algorithm for Determining the Minimum Number of Nodes to Access in a Partially Replicated Distributed Database System," 1988 IEEE Data Engineering Conference, pp. 330-337.

23. R. Mukkamala and S.C. Bruell: "Load Balancing in Partially Replicated Distributed Database Systems," Under Revision for a Journal publication.

24. S. Jajodia, R. Mukkamala, and K.V.S. Ramarao: "A View-based Dynamic Replication Control Algorithm," To be submitted for Publication 1992.

25. S. Jajodia and R. Mukkamala: "Measuring the Effect of Commutative Transactions on Distributed Database Performance," To appear in Information Sciences Journal 1992.

26. R. Mukkamala: "Measuring the Effect of Data Distribution and Replication Models on Performance Evaluation of Distributed Database Systems," IEEE Trans. on Knowledge and Data Engineering December 1989.

27. M. Overstreet, F. Paterra, K. Maly, R. Mukkamala, and C. Hajzak, "Utilization of Special Architectues in Support of Parallel Discrate Event Simuation," Proceedings IEEE SouthEastCon 1991, April 1991.

28. M. Overstreet, F. Paterra, K. Maly, "Distrituted Simulation, No Special Tools Required," Proceedings Winter Simuation Conference '90, pp. 423-427.

29. M. Overstreet, K. Maly, and Frank Paterra, "Distritubed Simulation of Network Protocols," Twenty-First Annual Pittsburgh Conference on Modeling and Simulation, pp. 1525-1530, May 1990.

30. K. Maly, M. Overstreet, E. Foudriat, S. Kahnna, and F. Paterra, "Modelling High Data Rate Communicaiton Network Access Protocol, Twenty-First Annual Pittsburgh Conference on Modeling and Simulation, pp. 1543-1547, May 1990.

31. M. Overstreet, "Model Analysis and Simplifications Using Graph-Based Representation, invited paper, Proceedings 12th World Congress on Scientific Computation, pp. 467-469, July 1988.

32. K. Maly, S. Khanna, C. M. Overstreet, R. Mukkamala, M. Zubair, Y. S. Sekhar, E. C. Foudriat, "Scalable Parallel Communications", Submitted to IEEE Journal on Special Areas in Communication, 1992.

33. K. Maly and F. Paterra and C. M. Overstreet and R. Mukkamala and S. Khanna, "Remote Visualization using Parallelism in the context of Existing Networks", To appear in Proceedings of International Phoenix Conference on Computers and communications. April 1992.

Publications

Scalable Parallel Communications*

K. Maly S. Khanna C. M. Overstreet R. Mukkamala M. Zubair
Y. S. Sekhar E. C. Foudriat
Computer Science Department
Old Dominion University
Norfolk VA 23529

February 28, 1992

Abstract

Coarse-grain parallelism in networking (that is, the use of multiple protocol processors running replicated software sending over several physical channels) can be used to provide gigabit communications for a single application. Since parallel network performance is highly dependent on real issues such as hardware properties (e.g., memory speeds and cache hit rates), operating system overhead (e.g., interrupt handling), and protocol performance (e.g. effect of timeouts) we have performed detailed simulation studies of both a bus-based multiprocessor workstation node (based on the Sun Galaxy MP multiprocessor) and a distributed-memory parallel computer node (based on the Touchstone DELTA) to evaluate the behavior of coarse-grain parallelism. Our results indicate: (1) Coarse-grain parallelism can deliver multiple 100Mbps with currently available hardware platforms and existing networking protocols (such as TCP/IP and parallel FDDI rings); (2) Scale-up is near linear in n, the number of protocol processors and channels (for small n and up to a few hundred Mbps); (3) Since these results are based on existing hardware without specialized devices (except perhaps for some simple modifications of the FDDI boards), this is a low cost solution to providing multiple 100Mbps on current machines. In addition, from both the performance analysis and the properties of these architectures, we conclude: 1) Multiple processors providing identical services and the use of space division multiplexing for the physical channels can provide better reliability than monolithic approaches. It also provides graceful degradation and low-cost load balancing; 2) Coarse-grain parallelism supports running several transport protocols in parallel to provide different types of service. For example, one TCP handles small messages for many users, other TCPs running in parallel provide high bandwidth service to a single application; (3) Coarse grain parallelism will be able to incorporate many future improvements from related work (e.g., reduced data movement, fast TCP, fine-grain parallelism) also with a near linear speed-ups.

1 Introduction

Motivation

Past research directed towards increasing the speed of communication has focused mainly on the physical media used to link computers together and the media access methods for such media[2, 6, 9, 19, 10] but limited bandwidth on the physical transmission media is no longer the performance bottleneck in communication systems. The development of high speed data networks in the range of Gbps imposes new requirements on the processing of communication protocols at the network node. Our objective in this paper is to report on a study of communication protocol processing that can handle throughput in excess of gigabit per second for a single user application.

The need for such a system can be justified by examining several distributed collaborative computer applications which require gigabit per second services made to be available to an end user. *Full motion video*

*This work is sponsored by CIT (596045), DARPA (N00174-C-91-0119), NASA (NAG187263), and SUN (596044) grants.

for use in video conferencing which needs high bandwidth and puts strict limits on average and variance of the network delay introduced as data are delivered to end application. *Computer imaging* - medical, seismic, and weather - demands low latency for data collection and high throughput for image transfers. *Visual techniques* are also becoming increasingly important as a means of understanding the results of advanced computer models. Breaking down a problem among networked computing resources - *computer steering* - is used for visually oriented modeling. Such needs result in large bandwidth requirements. For example, to drive a 1280x1024 24-bit color display updated 15 frames per second requires 472 Mbps data streams.

Background

Made possible by progress in fiber-optic and VLSI technologies, networks offering increased transmission capacity at decreased error rates are now becoming available. In many applications, the performance bottleneck hence has shifted from the network to the processing required to execute communication protocols in workstations and servers[7, 13, 27, 8, 11, 1]. Owing to this, a single application running on many "standard systems" cannot utilize a reasonable fraction of the bandwidth even on communication networks with data rates in the 10Mbps range. This restriction becomes more acute as networks offering larger bandwidths, e.g. FDDI (100Mbps), are becoming widespread. Thus the design decisions used in developing many existing protocols and computer architectures are inappropriate for the evolving high-speed networks. The three main decisions in question are the use of processing power to reduce transmission costs, addition of processing cost to recover from errors, and the use of relatively simpler flow-control mechanisms [11, 3, 4, 8, 17, 29, 30, 12].

Zitterbart [32] classifies the approaches to speeding up protocol processing into those which modify the seven layer protocol stack and those which provide high speed implementations of standard protocol suites. An example of the first category is discussed by Haas in [14]. In this work, the seven layers of the traditional OSI stack are grouped into three new layers; the lower three layers are combined into the network access layer, the transport, session, and presentation layers are combined into a communication interface layer, and the application layer becomes the new third layer. The tasks in the communication interface layer are decomposed into functions that can be executed in parallel to provide improved throughput. An example of the second category is explored by Van Jacobson in [17] with an optimized implementation of TCP/IP. XTP [5] is a variation of this approach that combines the transport and network layers into a VLSI implementation.

Approach

The central idea or theme of our approach is to use scalable parallelism at multiple levels as a basis for a network architecture. In this context parallelism is defined as the representation of a single user's data as a set of concurrent data streams which can be moved in parallel. This is not to be confused with multi-user concurrency where data streams from several users may be moved concurrently. The number of data streams may vary as they are being processed at various levels of the protocol stack. The degree of parallelism at a given level is determined by the number of processors needed to operate on the streams in parallel without introducing a bottleneck in the overall flow. Scalability is defined both in terms of a system scaling with the number of processors and physical channels as well as the ability of different nodes to use different amounts of bandwidth in the network.

We classify parallelism in protocol processing as seen in the literature at two levels[32]: *(i) fine-grain*, and *(ii) coarse-grain*. Fine-grain parallelism[33, 21] is the decomposition of a protocol task at a given level into several tightly-coupled, parallel, smaller subtasks. In the coarse-grain approach[15, 23], separate identical protocol processing tasks run independently on several processors, each processing packets from one or more data streams.This is illustrated in Figure 2. The key difference between the two approaches is that a process is decomposed in the fine-grain approach and replicated in the coarse-grain one. The two

approaches are complementary rather than mutually exclusive since, for example, in Figure 2(b), any of the TCP processes could use fine-grained parallelism.

Note that the fine-grain approach results in parallelism limited by the number of distinct subtasks in protocol processing, the dependency among subtasks, and the dissimilar computation times of individual subtasks. In the coarse-grain approach, different degrees of parallelism can be applied at different layers to provide services sufficient for that level. Because processors are operating on separate packets, they are largely independent and synchronization overhead is small. The approach is scalable with physical processors providing network services when needed and available to other tasks at other times. We note that fine-grain parallelism alone cannot provide any improvement in reliability.

Coarse-grain parallelism also provides the flexibility of different classes of service to communicating applications based on the individual application's needs and priority. This gives the flexibility of running, for example, several copies of TCP on different processors and copies of UDP on other processors. Further, for TCP support, one processor might handle all of the small packets for several light load user applications and others share the process load for a single "high-bandwidth" application. Thus, even in shared memory machines, this approach could avoid frequent context switches and cache misses for those TCP processors handling the high bandwidth application. In addition, previous work indicates performance improvement from placing traffic with similar attributes on separate channels[22].

The niche for which we see parallel networks most appropriate is the interoperable distribution of gigabit NREN traffic[25]. In Figure 1, the dark line represents a set of concurrent channels (for example, wavelength division or space division); each node is configured to attach to a subset (possibly all) of those channels to access whatever bandwidth they need. Inside a node protocol processing is done in parallel using multiple physical processors if available and when beneficial.

Parallel networks are important for many reasons. They complement other work on high data rate networks; by building on existing parallelism in computer and data storage systems, e.g. RAID[26, 24], and providing a natural mechanism for connecting them to a parallel network; by utilizing and improving transport level parallelism[18] in networks; by improving data-recovery codes[31]; by applying media level parallelism in addition to media-level concurrency. Parallel networks provide an upward migration path from existing networks. They allow scalability and fault tolerance capabilities not achievable with serial networks and they can provide a favorable cost/benefit ratio in certain environments.

Objectives

The long term objective of our project (jointly conducted at Old Dominion University and Virginia Polytechnic Institute and State University) is to demonstrate the feasibility of a gigabit node both from a technological and an economical point of view. We have developed a general model, we have done a performance study involving at least two different classes of nodes, namely, workstations and massively parallel computers, and we have designed a 3-node, 4-channel prototype. We next plan to realize the prototype, use the experimental results of the prototype to validate the predictive power of our model, and complete the design of a gigabit node.

As outlined below, this paper is a performance study of a workstation node and parallel computer node having parallel interfaces to parallel networks. This study shows that parallelism increases performance nearly linearly for small values of n (under 10), where n is the number of protocol processors or media channels. Given the low cost of FDDI cards (currently about $2,000) and the availability of multiprocessor workstations, we can bridge the gap to gigabit networks now using moderately priced components. Similarly, as industry is developing "gigabit workstations", i.e. workstations more suited to interact with gigabit networks, we will be able to multiply the effect of emerging protocol technologies at a future time. This study also shows that the inherent advantages of parallelism, fault tolerance and graceful degradation, can indeed be achieved with feasible designs. Node, processor, or link failure in a parallel system simply means that traffic is shifted to a different processor or link, only reducing available "bandwidth" but not terminating a connection. Parallelism is particularly suited to forward error correction to solve the latency problem in long networks. Coding across parallel channels combines the advantages of serial

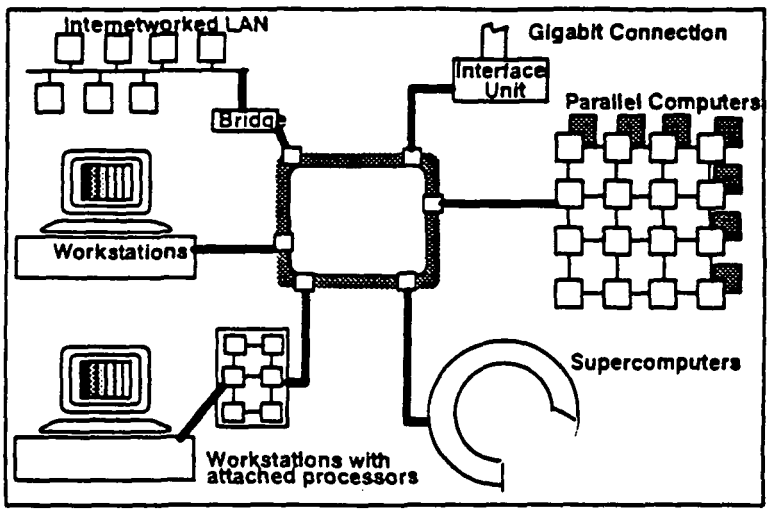


Figure 1: Need for local distribution

forward error corrections and the reliability of a parallel network providing better (latency and reliability) overall performance.

In this paper we study scalability from a performance point of view to see if the approach scales with processors as well as links. We do not address the architectural and routing issues of interconnecting nodes of different throughput capacities to different subsets of available parallel channels; this will be a matter of future papers.

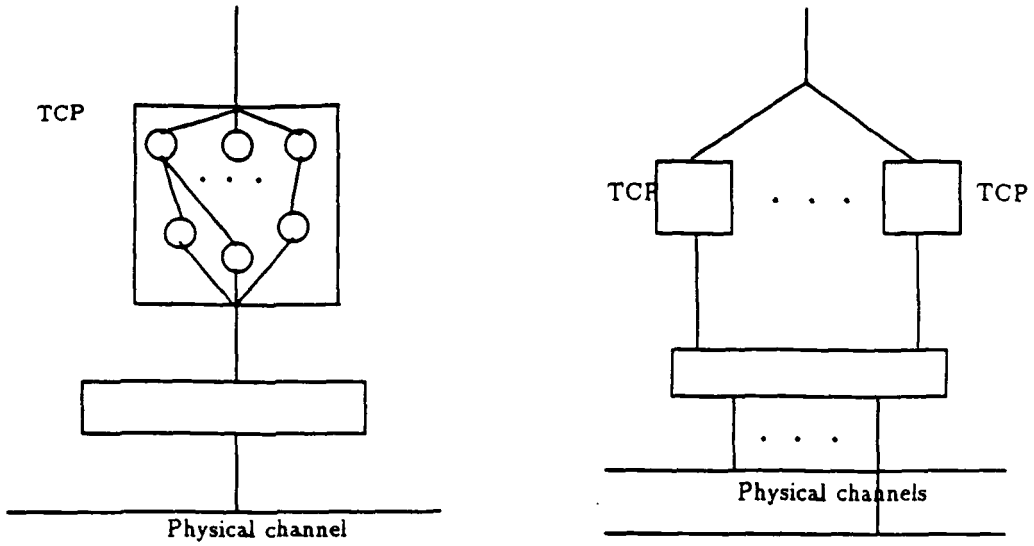
Organization

In Section 2 of this paper we develop a general framework for parallel networks by describing a model and discussing the issues and options posed. Since validation of this approach to high speed low-cost networking must be assessed based on the constraints imposed by real hardware (e.g., memory, bus, and processor speeds), operating system effects and protocol overheads, Sections 3 and 4 instantiate the general model for workstation and parallel computer nodes akin to actual existing machines, and present an analysis of the performance on these architectures. Section 5 summarizes the results of the study.

2 A General Framework For Coarse-Grain Parallel Networks

Communication systems are typically implemented as a hierarchy of protocol layers. Thus processing of a stream of data from a single sender to a single receiver can be done in parallel at different levels both at sender and receiver sites. In addition, if data from a single application are split into multiple streams at some level in the protocol stack, processing can then be performed in parallel on these separate streams. These streams can then proceed separately through additional layers and be rejoined later as appropriate (depending on what works best). If streams are split, this is an ideal application for parallel processing since interactions among the processes working on the separate streams can be very limited, even though they are all working at the same level of the stack. Thus parallelism at a given level is based on replication: separate identical processes (potentially on different physical processors) working on individual data streams.

In addition, this approach is realizable on several existing hardware architectures and provides a general framework for presenting the work that follows. The ideal degree of parallelism both within and among levels depends on physical properties of a particular hardware architecture and which types of network services are most important. The generic model which describes multilevel and replicated parallelism



a) Fine-grain parallelism

b) Coarse-grain parallelism

Figure 2: Different types of parallelism in communications

is intended to provide a framework which allows us to study both these issues which will be generally applicable to the use of parallelism in networking and to study the interaction between a particular hardware structure and the use of parallelism.

2.1 Generic Model Description

Figure 3 is a representation of our model for parallel communication in its most general form. The central idea is to examine the use of parallelism wherever it may be effective; the model presents those places where we expect parallel processing may be useful. We do not expect any effective concrete realization to be this complex but as the model is mapped onto different real architectures, what can effectively be run in parallel will change.

Since we provide for different degrees of parallelism at adjacent levels (and can demonstrate advantages with this approach), we now have the opportunity to schedule work for individual processors as data are moved from one layer to the next.

Figure 3 has 6 levelss, from parallel physical channels at the bottom to the application level (which also may include the presentation and session layers) at the top. Between adjacent layers are one or more schedulers. The model makes no assumption about the assignment of tasks to processes or processes to processors. It includes the idea that several tasks may be performed by a single process and several processes may be assigned to a single processor; effective assignments depends on such items as the degree of interaction among tasks and processes, underlying hardware, the operating system overhead, and processor loads from other tasks. In addition, in a particular hardware architecture, some tasks may be realized in hardware. For example, the scheduling scheme between two levels may be determined by the bus arbitration scheme in the bus hardware.

At the application level, a single user application, operating as a set of processes, A_i, transfers large amounts of data to another user at the remote end. The data units processed and exchanged at application level are called chunks. A chunk may be divided into one or more segments and these segments are exchanged between application A_i and transport T_i layer processes. There may be "n" such transport processes. Since segments must be moved from an application process to many transport processes, scheduler processes S_a are placed logically between application and transport layers to manage this segment

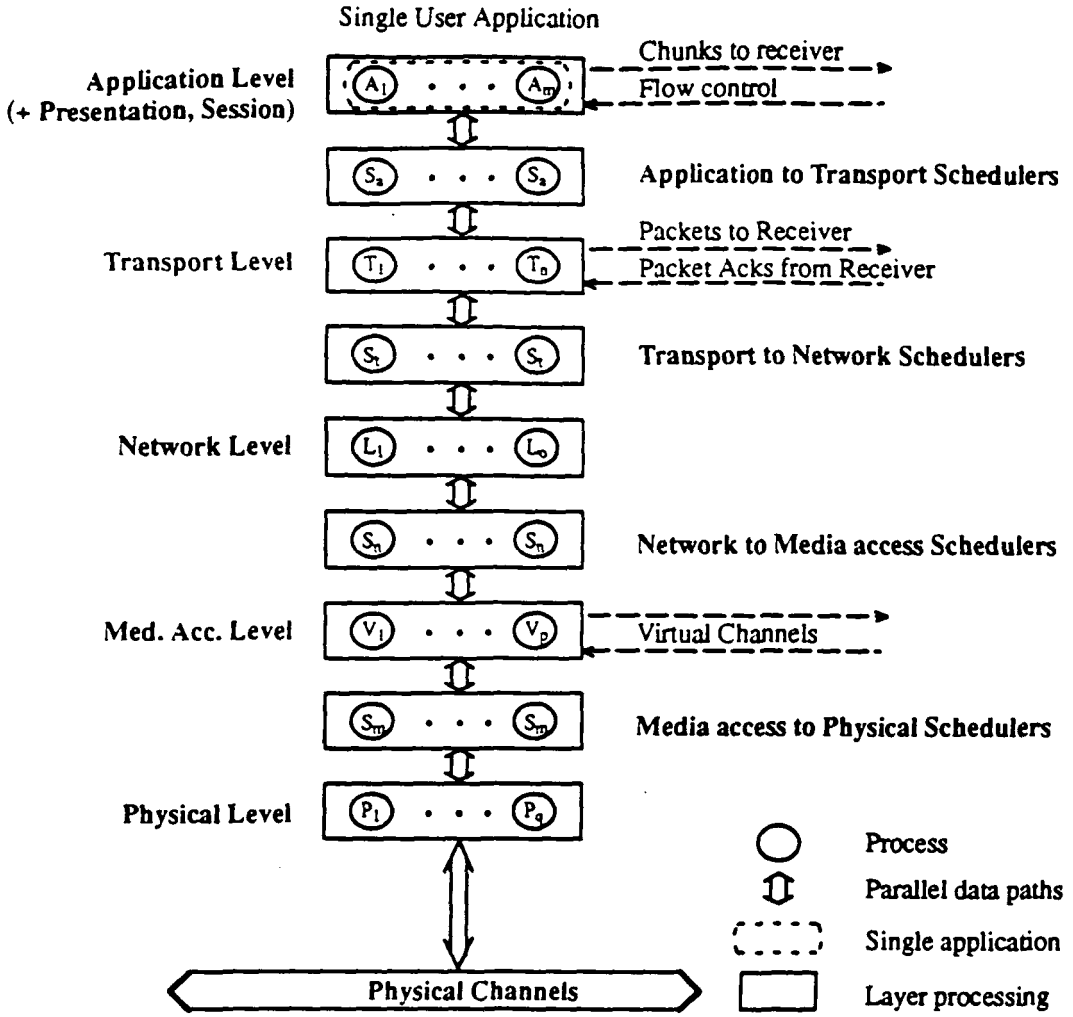


Figure 3: Parallel Communication Logical Framework: A General Concept

allocation. Similarly, scheduler processes S_t are located between transport and network level processes. A scheduler process, S_t, accepts a packet from a transport process and schedules this packet to one of "m" network level processes, L_i. Likewise, a network process, L_i, must select one of the virtual channels. This is achieved through scheduler processes, S_n, located in between the network and media access layers. Lastly, these virtual channels are then mapped onto physical channels by the scheduler processes, S_m. We assume that either many parallel physical channels exist or that one or more high bandwidth physical channels will be multiplexed to provide the appearance of several channels. A S_m scheduler may select the physical channel based on destination and the route selected by network process L_i to reach that destination. A sender and a receiver may have different degrees of parallelism at different layers.

This model assumes the existence of several physical processors. It can be mapped into a variety of realizations. For example, we include the very simplistic approach consisting of single application processor preparing data which are split into two streams. Each data stream is processed independently by a separate TCP/IP processor and sent to a separate FDDI board (for the media access level) and onto physically separate FDDI fibers. These data streams are then received by two separate FDDI boards located in a single node which are connected to two TCP/IP processors, and rejoined to present a single stream to a receiving application process. Thus, once the data streams are separated, they do not interact until rejoined at the receiver. Many variations are possible. For example, the two TCP/IP streams could be

rejoined and transmitted by a single FDDI board.

Because the identified bottlenecks in the protocol processing and absence of standard media access protocols designed for gigabit speeds, we chose to use parallelism at the transport and media access layers. We have one or more fast application processors (APs) connected by some communication fabric (dependent on the particular hardware present in a node) with m protocol processors (PPs) running in parallel. These PPs send and receive data using n physical channels with only one scheduler to load balance the physical channels. Usually the number of PPs and the number of channels differ. Variations on this basic structure are studied in sections 4 and 5.

2.2 Issues and Options

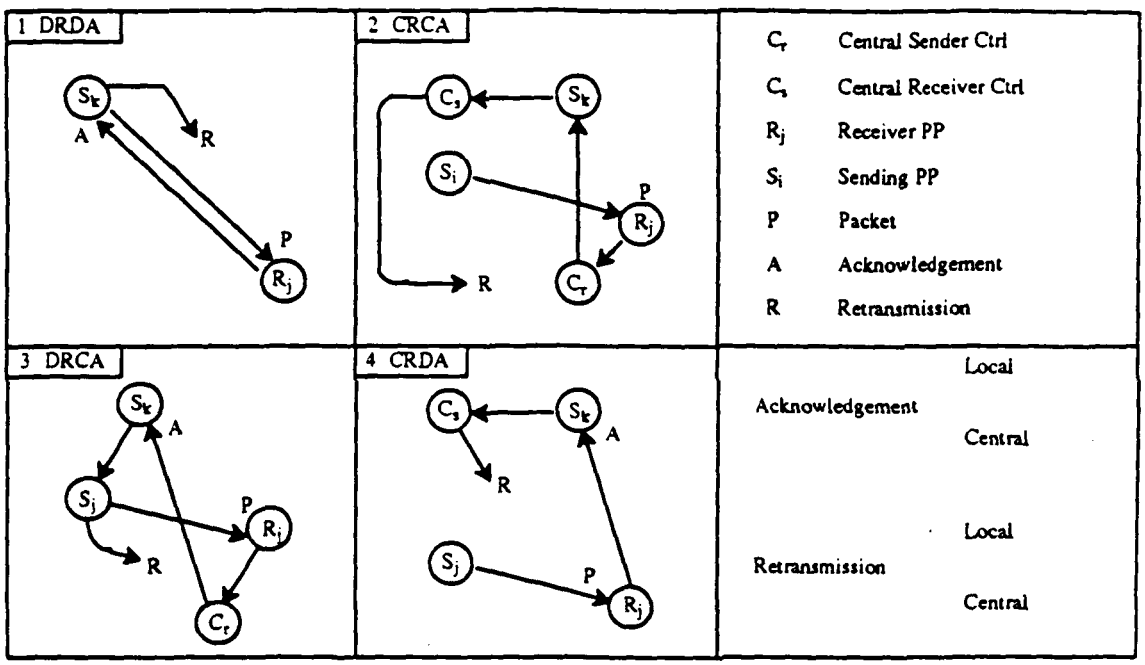


Figure 4: Acknowledgments and retransmissions

In communication systems that use multiple processes at several protocol layers, many issues which effect performance are dependent on properties of hardware present in the system and will be unique to that hardware. However several important issues which would not exist in a serial implementation will be present in almost any parallel solution. This are discussed here.

1. **Scheduling policy.** A major issue is finding an optimal data allocation strategy from an application process to many independent protocol processes and a protocol process to many network interfaces for transmission. The scheduling policy will affect the load balancing potentials among hardware components (processors, buses, memory, and channels, for example) and can have dramatic effects on performance. The data scheduling task can either be adaptive based on information from lower and upper layers or it can be simple such as first-come-first-served or round-robin. An adaptive scheduler may result in better performance utilization at an additional cost of collecting state information: it can make decisions based on information about what is going on below it (queue length at each

processor or expected time of token arrival, for example) or what is going on above it (for example, when the next data will arrive). The cost of doing adaptive scheduling and obsolescence of some types of potentially useful state information are issues which should be used to analyze adaptive scheduling schemes.

2. **Scheduler location.** A scheduling process can run either independently on each protocol (PP) processor (distributed scheduling) per layer or on a single processor (central scheduling) in between two layers. For example the transport to network level scheduler can run locally on each PP or in a central device located conceptually between network layer and MAC layer. This location of a scheduler process will impact the performance and fault-tolerance of the system.

3. **Window management, time-outs and acknowledgments.** One can assume that timers and acknowledgments are processed by the PPs. In this solution, all PPs operate independently and still contribute to the task of transmitting a block of data. This is obviously not the only way to perform these functions and we have identified four basic ways in which they can be done. Each of these are illustrated in Figure 4 and are discussed more below.

4. **Error detection and correction strategies.** Error detection and correction strategies across multiple parallel physical channels can be employed to recover from lost/corrupted data and channel loss efficiently. These strategies can save retransmissions at a cost of additional code bits per packet. If a forward error correction mechanism is employed, all bits transmitted in parallel must be present at the receiver at about the same time so that errors can be corrected quickly. On the other hand not all packets need to be present to compute all correct data packets.

5. **Retransmission timer value.** Retransmission timer value in existing nonparallel transport protocol implementations is computed based on the smoothed round trip time of packets over a single channel. This computation may not be valid when a single transport connection spans over parallel channels which behave independently. Hence, retransmission timer management is an issue in parallel implementations of communication protocols.

6. **Shared memory.** In multiprocessor architectures, distributed shared memory for processors versus local memory per processor is a major performance issue in achieving the higher throughput. Although shared memory makes processor coordination simpler, but it comes at the cost of lower performance. Local memory reduces the load on global memory access, but may require additional data copy operations from one processor's local memory to another processor's local memory.

7. **Memory access, data copying and operating system interrupts.** These issues are equally important as in serial implementations. Their impact though is different because some of the operations can be overlapped and pipelined.

Acknowledgements and Retransmissions Management

Figure 4 illustrates the four choices in handling acknowledgments and retransmissions. The first solution, called DRDA for Distributed Retransmissions Distributed Acknowledgments, assigns the task of timer maintenance to the PP that generated a packet. The receiving PP generates an acknowledgment and sends it to the transmitting processor directly. This is a distributed solution and essentially uses separate transport connections between the sending and receiving PPs.

The second solution, CRCA - Centralized Retransmissions, Centralized Acknowledgments - uses separate PPs on both sides to maintain timers and to generate acknowledgments. In this solution, the PP labeled C_s assigns a packet for transmission to PP labeled S_i and S_i sends it to any receiving PP labeled R_j. On receipt of a packet, R_j notifies the PP labeled C_r which determines if an acknowledgment is required. If one is needed, it is built and sent to any PP on the transmitting side. Received acknowledgments are

forwarded to C_s and the corresponding timer is stopped. This solution is similar to that proposed by Jain et al. in [18].

The third configuration, DRCA - Distributed Retransmissions, Centralized Acknowledgments - uses a central PP on the receiving end, C_r, but each of the sending PPs maintains timers for its packets. When a packet is received, C_r is notified as in CRCA, and it sends an acknowledgment to any PP on the sending side. As before acknowledgments are forwarded to the sending PP and the timer is stopped.

The final solution, CRDA - Centralized Retransmissions, Distributed Acknowledgments, uses a central PP for timer maintenance and packet retransmission as with CRCA, but each of the receiving PPs generates acknowledgments locally and sends them to any PP on the sending side.

Providing a completely decentralized solution (DRDA), like the first one shown in Figure 4, reduces interprocessor communication requirements on the sending side and we can expect throughput to increase. The cost associated with this solution, however, becomes evident when resequencing is performed at the receiving end. The host or receiving application must have enough storage space for many segments of data, rather than packets, because while packets within a segment are sequenced by the sending and receiving processors, the segments must be resequenced by the receiving application host.

A central processor performing acknowledgment generation makes packet resequencing easier and may reduce system latency because only complete packets must be received, rather than segments. It also implies that a single window is maintained for data coming from all processors, so more memory is required. Additionally, increased interaction among processors is required and may reduce overall system throughput.

3 Bus-based Multiprocessor Workstation Architectures

Previous work in parallel computing shows that inappropriate decomposition of a problem results in poor performance rather than naively predicted speed-ups[20]. We feel that the potential for significant improvements both in speed and reliability clearly exists with this parallel approach, but must be demonstrated. Proper evaluation must take real operating system, protocol, and hardware effects into account. In the next two sections, we describe our analysis of parallel network performance on both a Sun Galaxy Multiprocessor and a Touchstone machine. These machines have very different architectures. Before beginning detailed analysis, we felt that the likely performance bottlenecks would be related to processing speeds, memory contention, and bus contention. These machines are of interest because they present different communication fabrics: Sun is bus-based with a single global memory, Touchstone has a mesh interconnection fabric with local memory for each processor. Concerns about memory access and bus contention also led us to consider a variation of the Sun architecture with two buses and local memory for each processor. Results from analysis of these architectures are presented in this and the next section. The work reported here presents positive support of the benefits of coarse-grain parallelism.

Our conceptual model for parallel protocol processing represents processing functions as cooperating processes. To instantiate this model on a given parallel hardware architecture, we need to describe the facilities for interprocess communication, memory architecture and the location, number and types of schedulers and map the application and the protocol processes onto different processors. As a first instantiation of our model we have selected a multiprocessor workstation. We believe for gigabit networks to succeed they cannot rely on a few supercomputers to operate in a distributed fashion but rather gigabit communication must be made available to the large number of workstation users who constitute the vast majority in our computing community. Although current workstations are not designed to handle gigabit traffic, many now have multiprocessors. We will show that these workstations can handle multiple 100 Mbps traffic when conjoined with a parallel network. Within the current technology framework we are restricted to bus-based architectures when instantiating the model for workstations.

Here, we describe two bus-based instantiations to realize the proposed conceptual model for parallel processing. The instantiations have the following common features.

- Each application process is located in an independent applications processor (AP).

- Each protocol process is located in an independent protocol processor (PP).
- Each protocol process does TCP and IP processing. We assume that the application requires reliable stream oriented transmission of data and postpone for later studies the impact of using such protocols such as UDP for high bandwidth applications.
- To allocate the segments generated by the application processor to the protocol processors, we employ a scheduler (called the application scheduler). The scheduler is located in the application processor itself.
- The scheduler to allocate packets from the protocol process to the network interface (called the network scheduler) is located either in each protocol processor or in a special mux/demux device.
- The bus (or buses) acts as the basic communication fabric for interprocess communication.
- If shared memory is present, it can also be used as a medium for interprocess communication.

With this concept of parallel processing in mind, the challenge is to design MIMD (multiple instructions multiple data) like architectures in which processors independently execute TCP/IP on multiple data streams originating from a common application source stream and push processed packets on multiple channels to provide higher end-to-end throughput. Since an application generates data and data are distributed to protocol processors, the data should be available to all processors at a common place which can be a global memory, to prevent multiple copying of the data. This flow across AP and PPs must be controlled. One possible and readily available solution being providing a bus between these two components to serialize the access to the bus. Keeping this as motivation the following feasible architectures are studied for such applications.

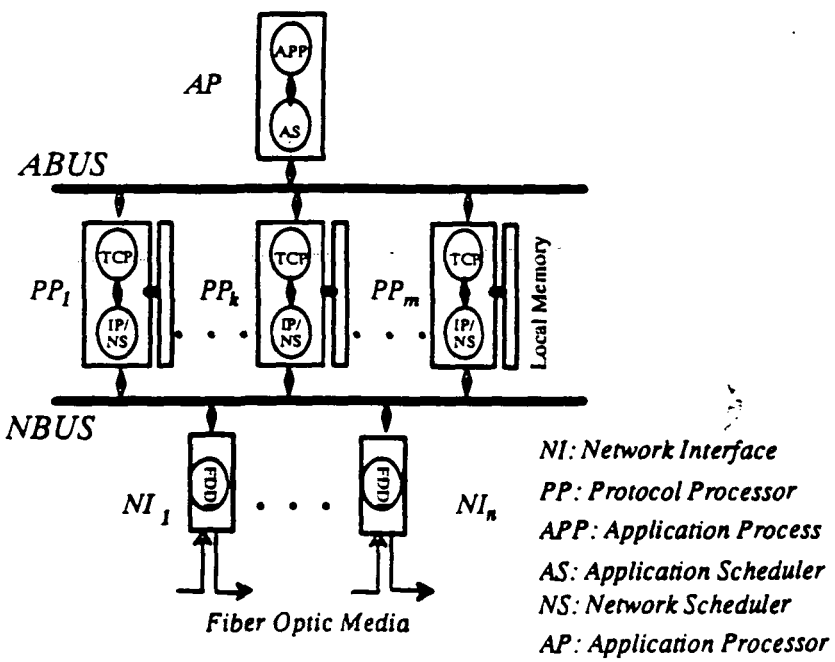


Figure 5: 2-Bus Distributed Scheduler architecture(2-Bus/DS)

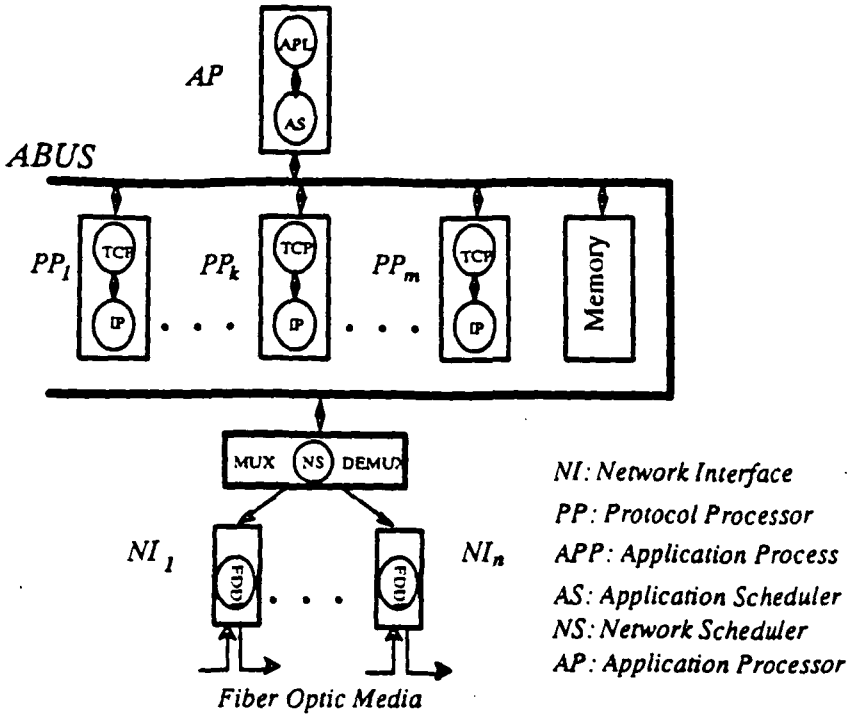


Figure 6: 1-Bus centralized scheduler architecture(1-Bus/CS)

3.1 A Two-Bus Instantiation

Figure 5 shows a two-bus instantiation of the general parallel architecture. The main objective in selecting this architecture is to determine the impact of distributed scheduling and local memory on the overall performance of the system. Here a single high performance processor is running the application and is capable of generating (at sender's end) or consuming (at receiver's end) data at the rate of multiple 100 Mbps. The two buses are referred to as ABUS (or the application bus) and the NBUS (or the network bus). Since a single processor cannot efficiently utilize the available channel capacity, multiple independent protocol processors (PPs) are connected between an ABUS and an NBUS where each PP executes an independent copy of TCP/IP. Each processor is assumed to have sufficient local memory to do processing as well as data storage functions. The application generated data, in the form of segments, are copied from the AP's local memory to the local memory of a PP over the ABUS. The PP is chosen based on a scheduling policy implemented by application scheduler process running on application processor. When a PP gets a segment, it does TCP/IP processing for it and schedules packets for the network interfaces according to a policy implemented by network scheduler. This architecture is a distributed scheduler architecture because network scheduler task is distributed among the PPs.

3.2 A Single-Bus Instantiation

Figure 6 presents an instantiation based on Sun Microsystem's Galaxy Multiprocessor system. The main objective of selecting this architecture is to determine the effect of global memory and central scheduling on the performance of the overall system. Since distributed scheduling is assumed to be more expensive in terms of propagation of information and the use of probably obsolete load information, the centralized scheduler is expected to result in improved performance. The increased performance may be achieved at the expense of reduced fault-tolerance. This architecture, referred to as 1-bus, has a very high speed bus which can support up to sixteen Sparc processors and I/O devices. (The current release, however, supports four processors, three I/O devices, and the memory, and is adequate for our experimentation.) Each processor gets its instructions and data from the shared memory. Since many independent processors get their data

and text from one shared memory, the memory contention problem can be solved temporarily with faster memories and using a "Split Protocol" on the bus[16]. This protocol guarantees that any module requesting memory does not block the bus during the time of service of that request. The network scheduler is located in the Mux/Demux device. The custom-made device Mux/Dmux is definitely a performance bottleneck because it serializes the parallel effort achieved in the transport layer. However, it enables us to analyze the tradeoffs in the use of forward error-correction techniques and a centralized scheduler. With cross channel coding, a form of forward error-correction technique, it is possible to recreate the data stream at the receiver even when l out of m channels are faulty reducing the need for retransmissions[31]. For efficient coding of data on all channels the data stream has to be serialized at some point before it is put on the physical channels.

Various issues which may effect the performance of these multiprocessor architectures when parallel protocol processing concept is implemented on them are detailed in the next section and options for efficiency and higher performance are also proposed.

3.3 Model and Simulation

Here we describe the simulation models developed to evaluate the proposed 2-bus and 1-bus architectures. The major common components in all the architectures are: application process, application scheduler, protocol process, network scheduler and FDDI interface. But the specifications of these common components may vary between the architectures. The discrete event simulator was written in C with every process and hardware component modeled as an object. The model was validated in several ways. Early on, detailed traces were generated to confirm that the proper events where occurring as they should. In addition, the model was run with parameter set to values which should produce results with well know performance features to confirm that the program would replicate those featues. We also determined the run-lengths sufficient to ensure that output statistics are stable.

- **Application process.** An application process runs on the application processor (AP) generating segments of data at uniform rates. It requires an end-to-end transfer of its data in excess of 300 Mbps (arbitrarily assumed in our experiments). The application data are available as a sequence of segments and such segments are allocated to the protocol processors (PPs) by the application scheduler process.

- **Scheduling policies.** The application scheduler process implements a first-come-first-served scheduling policy (FCFS) in allocating segments generated by the application process to the protocol processes. Here, as soon as a protocol process is found to be capable of handling an additional segment (e.g. due to availability of free buffers), that PP is allocated the next segment in line. In our model, we assumed PPs never had to wait for segments from APs since we were concerned with PP processing speeds and not with processing speed at the application level.

  The network scheduler process implements two types of scheduling policies in allocating packets from the protocol process to the network interface. The round-robin or RR policy is the simplest one, where the network Scheduler allocates packets to NIs in a round-robin fashion. The second policy, the adaptive scheduling policy, allocates packets to the NIs based on the minimum value of a function which depends on the queue size at a NI and the last cycle token rotation time of the NI. Clearly, a NI with smaller queue size is preferred to the one with a larger queue size; a NI with smaller last token rotation is preferred to one with a larger token rotation time.

- **TCP/IP connection.** Each sender PP works on an independent TCP connection with a receiver PP. The data stream is handled by a PP according to the RFC 793 [28]. The processor speed is assumed to vary from 1 MIPS (typical of a Sun3/50) to 25 MIPS (typical of a Sparc). Instructions for TCP/IP are assumed to be in cache with a 100% hit rate, but data to be sent has a 0% hit rate.

- **Network interface.** The network interface (NI) is modeled as an FDDI interface connected to a fiber-optic ring with 20 nodes evenly distributed along a length of 20 kms. Among these nodes, two are designated as the sender and the receiver for our experiment. Since we assume the high-bandwidth requiring nodes to coexist with other low-bandwidth nodes in the network, we simulated the latter traffic by introducing the background traffic. Hence, the background traffic is an integral part of the total network traffic and is aggregated as a single entity in our simulations. The background traffic on one ring is independent from the traffic on other rings. We consider both uniform and nonuniform cases of background traffic. Also the background traffic on a ring is not delivered to eiher the parallel sender and parallel receiver; they only process data belonging to the application. If we assume, for example, 30% of total capacity of six FDDI channels as background traffic, then the available channel capacity will be approximately 6*(100-30-token rotation loss) or approximately 300-390 Mbps.

- **Memory.** When local memory is present in a processor board, data are exchanged between processors via the bus (i.e. a memory to memory copy is performed). In case of shared memory, each processor's transmit/receive buffers are managed as separate entities in the memory and hence no explicit data transfer is necessary. To resolve bus contention in the case of local memory and memory contention in the case of shared memory, we use the FCFS policy.

- **Acknowledgments and retransmissions.** Acknowledgments and retransmissions are handled in a completely distributed manner (DRDA). Experiments with other policies are currently underway. Errors on channels are simulated as negative acknowledgments and sender PP starts retransmission of all the packets starting from the error packet in its window till the last packet sent.

- **Mux/Demux.** The multiplexor/demultiplexor device in the 1-bus architecture schedules packets from PPs to NIs in both round robin and first-come-first-served fashion, based on the option selected. It may also encode data packets and pad them with some extra bits to enable complete reconstruction of data at receiver when employing the cross channel coding.

Since our approach to protocol processing is radically different from the traditional serial approach, we have considered the serial architecture for baseline comparisons. For this purpose, a serial architecture is represented as consisting of one application processor, a single protocol processor, communicating on a single bus. We have not made any attempts to optimize the parameters of this architecture to function efficiently at high speeds. Instead, when the serial system is to be compared to a parallel system with four protocol processors, the speed of the serial processor is increased four times. Similarly, other serial system parameters are also set on the same basis. In comparing the performance of the proposed parallel architectures with the baseline serial architectures, our intention is to determine whether parallelism has an advantage over traditional architecture in reliability and cost while not loosing in performance. It may also be true that we may be trading one for the other. We wish to determine the facts with our experiments.

Table 1 illustrates various hardware and software parameters important for simulation and their corresponding values adopted in our experiments. The parameters in Table 1 are classified into three categories: hardware-related, software-related, and control parameters. While the hardware and software related parameter values chosen here represent a sample of reasonable values in the current technology, the values in the control section represent optimal values obtained through experimentation.

3.4 Performance Results

The main objective of our experiments is to determine the feasibility of the proposed architectures to achieve the desired scalability in performance. Especially, we are interested in determining the reliability-roundtrip delay tradeoffs within the architectures. In addition, we are also interested in determining the efficacy of each of the options proposed to solve issues in these architectures. This section attempts to summarize our results and conclusions in this context, without strongly supporting one architecture over the other to build a system. The performance of an architecture is measured in terms of throughput and

	2-Bus/DS	*1-Bus/CS*	*Serial*
Selected Hardware Parameters	Local Memory	Shared Memory	Shared Memory
No of APs	1	1	1
No. of PPs	4	4	1
No. of NIs	6	6	1
MIPS	25	25	200
ABUS speed (Mbps)	800	2400	2400
NBUS speed (Mbps)	800	-	-
Error rate	10e-9	10e-9	10e-9
FDDI speed (Mbps)	100	100	600
Memory Access (nsecs/word)	80	20	5
Selected Software Parameters			
Application Scheduler policy	FCFS	FCFS	-
Network Scheduler policy	RR/Adaptive	RR/Adaptive	-
Selected Control Parameters			
Segment size (Kbits)	36	36	144
Window size (kbits)	216	72	864
Packet size (kbits)	36	36	36

Table 1: Table of simulated hardware and software parameters

round trip delay per segment (i.e., the time a segment has been generated by the application to the time it has been fully acknowledged at the sender application scheduler).

Experiments were done with the simulator to find the optimal values of (i) segment size, (ii) TCP window size, (iii) processor speed, (iv) bus speed, and (v) memory access time for all architectures operating under variable environment parameters such as: (a) background traffic on channels, (b) background traffic on the bus, (c) scheduler policy, and (d) error rate. Our aim is to provide end-to-end throughput around 300 Mbps at minimal round trip delay per segment. We identified 300 Mbps as a throughput level because it is a significant improvement over what is currently available and is also sufficient to raise major issues in use of parallelism.

To make maximum use of a FDDI frame, the TCP packet size was fixed at 4500 bytes. Through successive experimentation, it was observed that at least four PPs and six FDDIs (NIs) are needed to achieve around 300 Mbps end-to-end throughput in all the two architectures. Subsequent subsections present a brief analysis and interpretation of the experiment results.

- **Bus Speeds, Processor speeds and Memory Access Time.** Through experimentation with processor speeds and bus speeds, we conclude the following.

  - For the 2-bus architecture, the processor speed can be anywhere between 1 MIPS (86% utilization) to 25 MIPS (4% utilization) and bus speeds should be 700 Mbps or greater. To avoid overloading of the components, 2 MIPS processor (with 45% utilization) and 800 Mbps bus speed were selected for subsequent experiments.
  - For the 1-bus architecture, bus speed (2400 Mbps) and processor MIPS (25 MIPS) were chosen to resemble the SUN Galaxy MP system. Experiments, where memory access time was varied from 20ηsecs/word to 80ηsecs/word, showed that throughput and round trip delay would deteriorate by 90% for the slower memory. Table 2 shows this behavior. For subsequent experiments, we chose 25 MIPS protocol processor speeds for a fair comparison of the architectures but the bus speed for 2-bus architecture was kept at 800 Mbps.

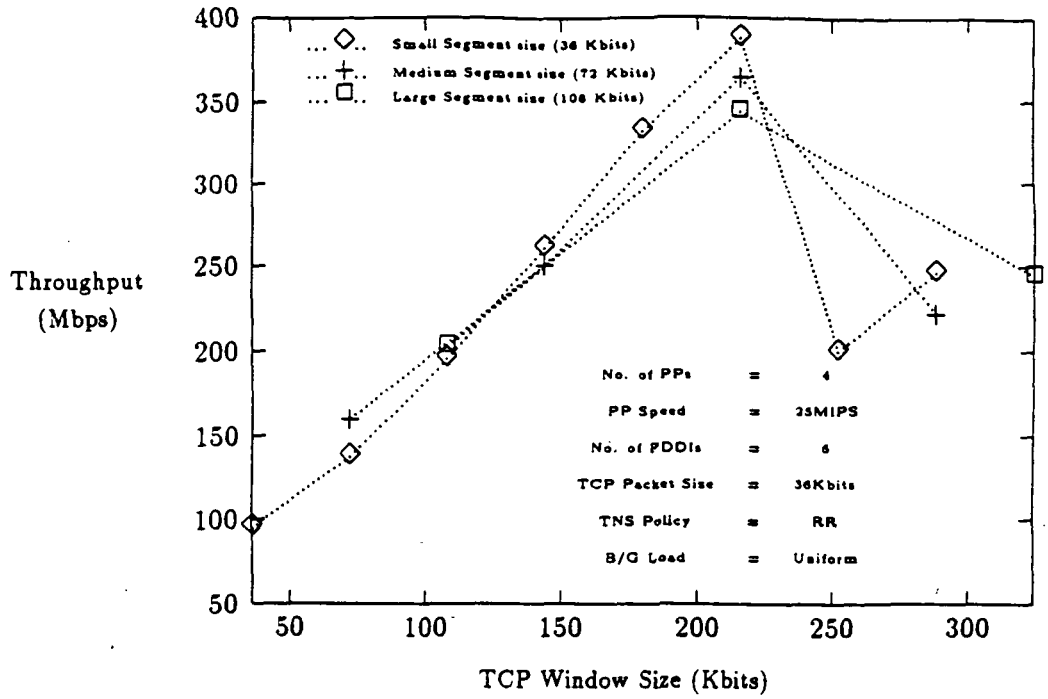


Figure 7: 2-bus architecture: End-to-end throughput vs window size

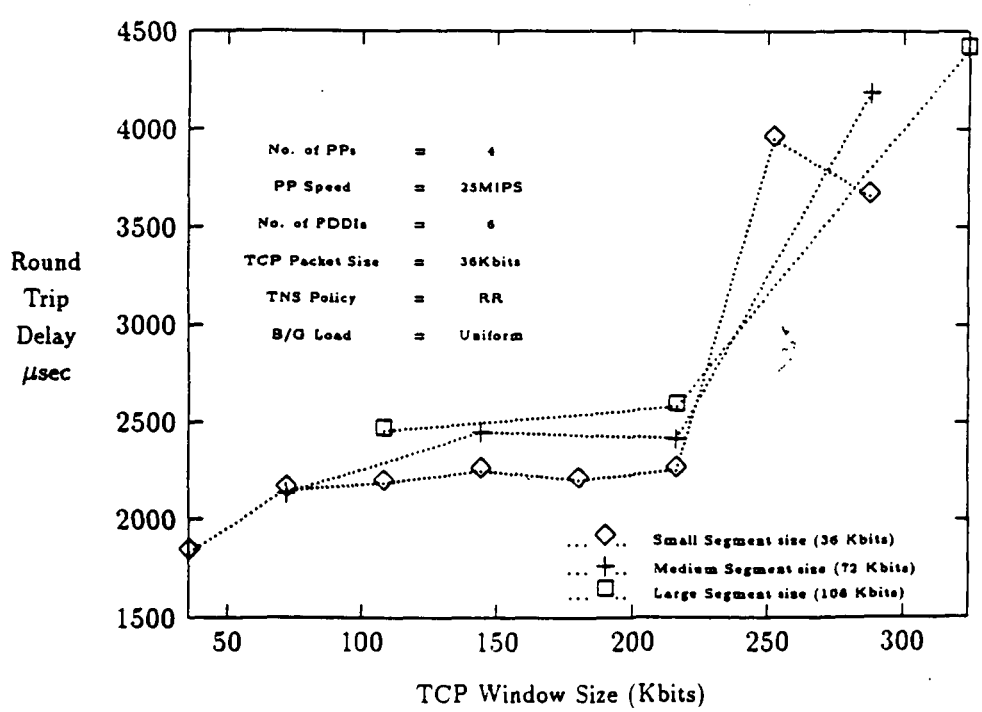


Figure 8: 2-bus architecture: Round trip delay per segment vs window size

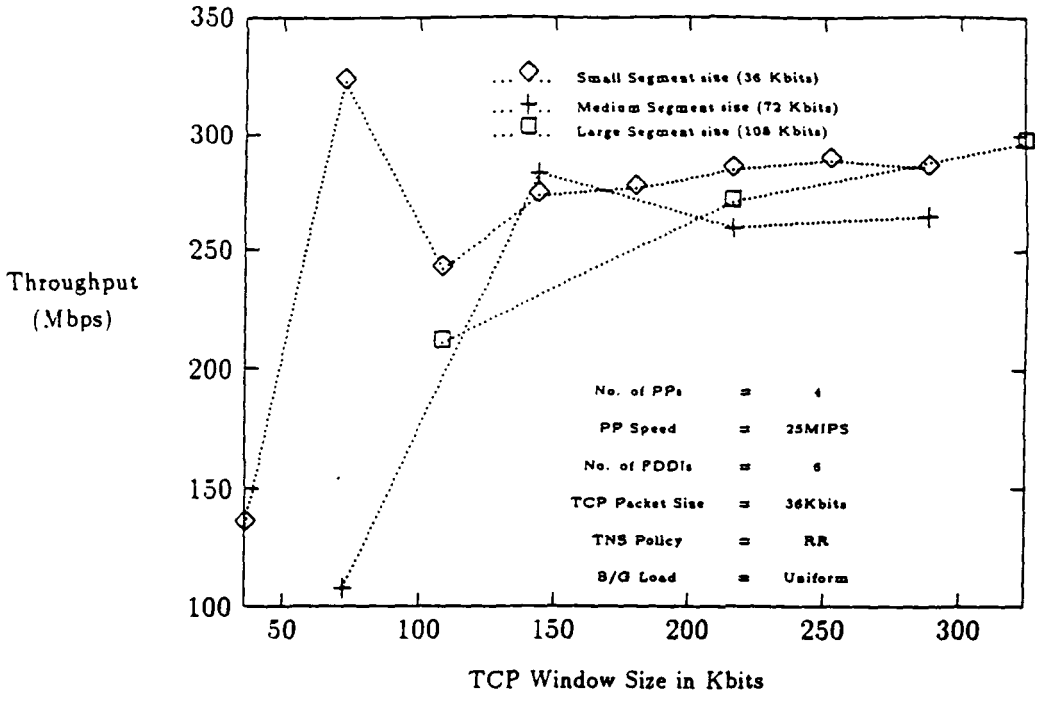


Figure 9: 1-Bus: End-to-end throughput vs window size

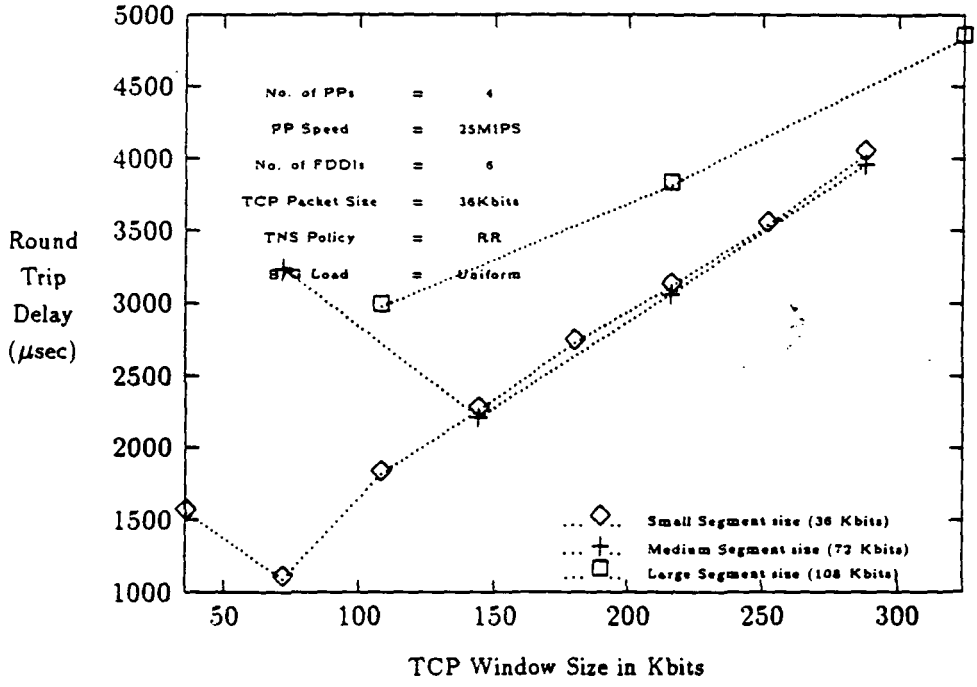


Figure 10: 1-Bus architecture: Round trip delay per segment vs window size

Access time per word ηsecs	Delay per read/write μsecs	System Throughput Mbps	Round Trip time per segment μsecs
20	7.1	323	1092
40	17.46	284	1235
80	54.87	170	2085

Table 2: Comparative performance of 1-bus architecture under varying memory speed

- **TCP Window Size.** To determine the optimal TCP window size which delivers maximum throughput with minimal round-trip delays, three experiments were conducted corresponding to small (36 Kbits), medium (72 Kbits) and large (108 Kbits) segment sizes. Figures 7 through 10 illustrate the impact of window size for the 36 Kbit, 72 Kbits and 108 Kbits segment sizes. From the plots, the optimal window size for the 2-bus architectures is found to be 216 Kbits and for the 1-bus architecture it is 72 Kbits for segment size of 36 Kbits. These results indicate that the window size selection is such that -
  - No protocol processor starves for data to be sent while waiting for the acknowledgments (This helps in achieving higher throughput by proper scheduling.);
  - No packet waits unnecessarily in sender's window. (This helps in achieving lower round-trip delays.)
- **Segment Size.** To avoid introducing additional latency, the segment size for any architecture should be bigger than the TCP packet size and smaller than the TCP window size. Experiments suggest a minimal segment size of 36 Kbits for both the architectures which is the lower bound on the segment size. A larger segment size for any architecture provides similar throughput but at around 7% to 23% higher round trip delay. A larger segment size results in higher round trip delay because the segment has to contend multiple times for bus and memory (for every TCP packet a segment makes). Also every TCP packet of the segment suffers protocol processing delay.
- **Round Trip Times per TCP packet.** Figures 11 and 12 illustrate the round-trip timing distribution for TCP packets sent by a PP under nonuniform channel loads for both the architectures under study. Figure 13 shows the round trip time distribution for TCP packets for a conventional (single processor) architecture. While the average round-trip delay in Figures 11 and 12 should not be directly compared due to architectural differences, the large variations in TCP packets' round trip time in the parallel architectures considered here as compared to the conventional case is apparent. This variation may cause serious concerns in managing TCP timeout timers in multiple channel implementations. If a timeout timer is managed as specified in current TCP recommendations, the TCP connections will timeout for many packets for this large variation of round trip delay. Hence many retransmissions will occur. (To get round trip time distributions, these simulation experiments were conducted with a very high timeout timer values.) To confirm this belief, the distributions were processed according to suggested smoothed round trip time ($srtt = (1 - \alpha) * rtt + \alpha * srtt$) and retransmission timeout timer ($rto = \beta * srtt$) calculations according to the equations given below from [28]. The values of α and β chosen were 0.2 and 2.0 respectively. The number of computed timeouts which could have occurred in each case are depicted in each figure. These variations in round trip times are observed to occur only under nonuniform channel load conditions.
- **Comparison of Architectures.** Table 3 presents a comparative estimate of the performance of two architectures for various policies implemented in network scheduler and various background loads on

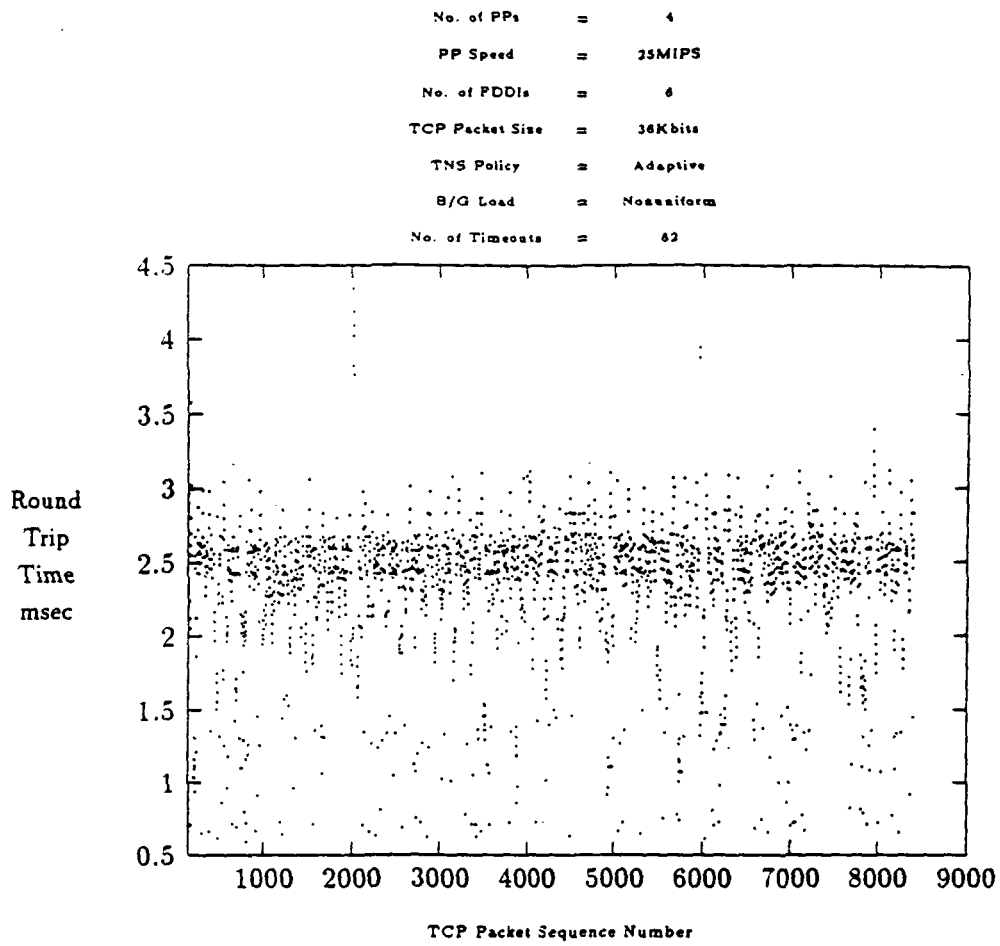


Figure 11: 2-Bus architecture: RTT Distribution for TCP Packets

FDDI channels. In nonuniform case, background load on each channel was varied in steps of $\pm 10\%$ (lower and upper bounds being 10% and 50% respectively) with an initial and final loads of 30%. Hence, nonuniformity starts and ends with a uniform channel loads on all channels. Half of the channels take positive steps and remaining half take negative steps during six such variations evenly distributed over a simulation run. Under all channel conditions (refer Table 3), the 2-Bus architecture performs better than 1-Bus architecture in terms of smaller processor MIPS and slower bus speeds for similar throughput values.

- **Conventional vs. Bus-based Multiprocessor Architecture Performance.** Tables 1 and 3 present a comparison of performance of the two proposed instantiations with the serial architecture. (Recall from Section 3 that a serial architecture is a single processor, single bus system.) As observed from the experiments, the window size needed for TCP is 864 Kbits which is three times TCP window size (4 x 72 Kbits) in the 1-bus architecture. The protocol processor MIPS and memory speed are eight and four times morer respectively. The main reason the performance of the serial architecture is actually less than the "equivalent" parallel system is twofold: one, "equivalent" is not really accurate because we only adjusted the processor's speed, and two, more importantly, in our simulation the serial implementation did not pipeline its memory access and computation cycles. Performance of the serial architecture is still in the same range as the parallel architecture, but requires a hypothetical network interface of the order of 600 Mbps is needed which would be more expensive using current technology. In terms of reliability and fault-tolerance, the serial architecture has no fault-tolerance

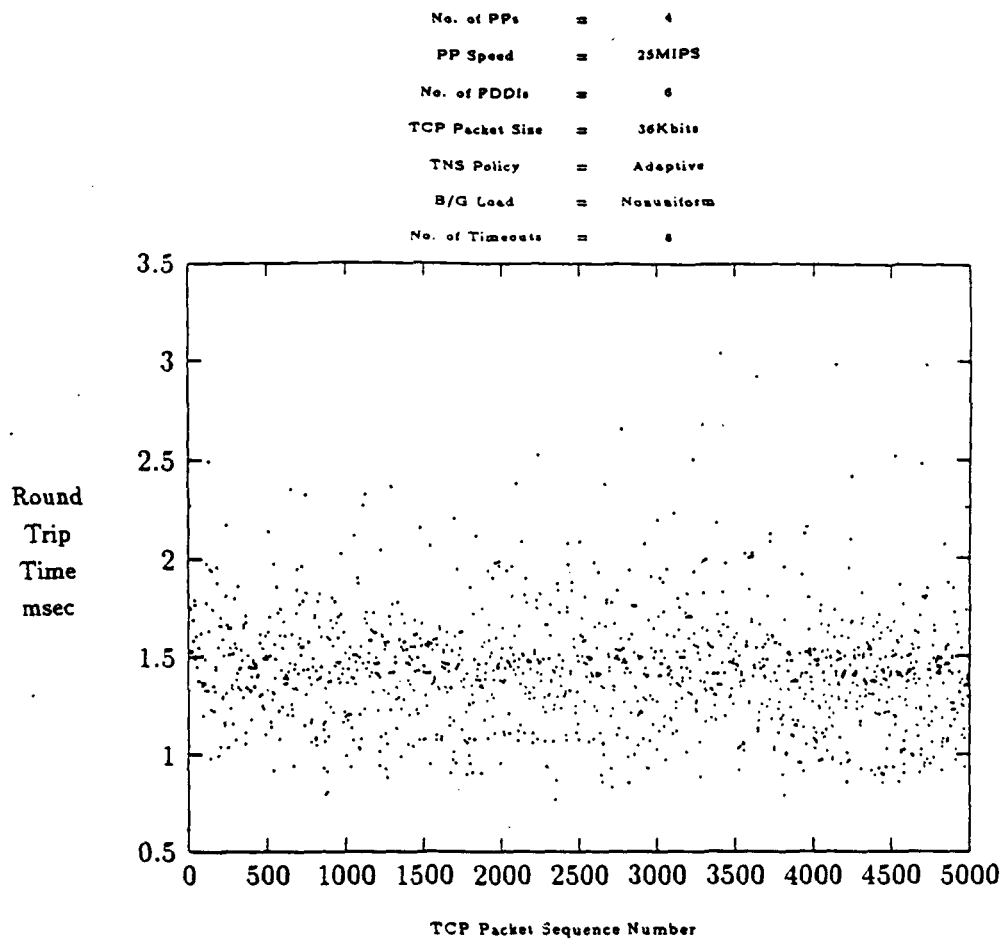


Figure 12: 1-Bus architecture: RTT Distribution for TCP Packets

while the 1-bus architecture can tolerate 3 protocol processor failures, and 5 channel failures. We can conclude that "comparable performance" is juxtaposed to a serial architecture with more cost and less reliability.

3.5 Memory Access and Operating System Interrupts

As mentioned in the introduction, a main objective of our study is to determine the effectiveness of coarse grain parallelism processing in building high-speed networks using the existing technology. It is widely believed that processing speed is the singlemost bottleneck in achieving higher throughput between single application process pairs. While this statement appears to be valid in a nonparallel system, its validity in the parallel case is not obvious. For example, if we consider the TCP/IP protocol processing, some components of the processing need to be done in a strictly serial fashion. Updating windows, for example, is a function that is to be executed serially for a given pair of communicating application processes. Other functions such as computing checksum may be executed in parallel for different packets within the same window. For this reason, we have analyzed the fast-path TCP/IP and categorized its processing requirements in terms of serial and parallel. The time estimates based on the 1-bus shared memory instantiation are summarized in Table 4. For brevity, only the results of our analysis for the *send* call are presented. As shown in Table 4, when the packet size is 500 bytes, for a memory speed of 40 ns, processor speed of 15 MIPS, and the checksum computed in software, then the parallel portion of TCP/IP consumes

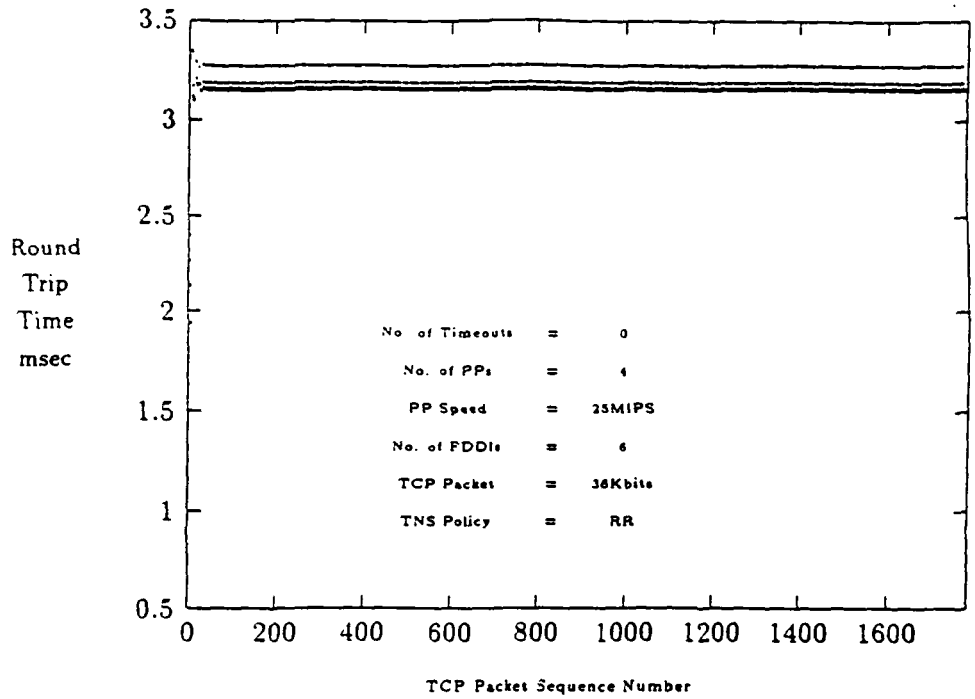


Figure 13: Serial architecture: RTT Distribution for TCP Packets

58.2 μs and the serial portion consumes 12.1 μsec. This indicates that while a serial protocol processor of 15 MIPs would take 70.3 μs to send a single 500-byte packet, a two-processor system could send two packets in (2*12.1+58.2) or 112.4 μs, a speedup of 1.25. Continuing this analogy, the maximum achievable speedup of 5.7 ($\approx$ 70.3/12.1) is possible. Similarly, when the packet size of 4500 bytes is considered, a speedup of up to 3.5 can be obtained for a memory speed of 40 ns.

From this analysis as well as the simulation results discussed earlier, we conclude that parallelism at the protocol processing level can result in a 3-7 fold increase in the network throughput. This gain may be obtained in spite of the inherent limitations such as the existing TCP/IP protocols and the bus speed in bus architectures. Further, if the current efforts of other researchers to develop more efficient protocols succeed (thereby reducing the sequential portion), then we can obtain even higher throughput by employing the proposed parallel architectures.

4 Distributed Memory-Based Parallel Computer Architectures

A second application of parallel networking we believe to be particularly fruitful is where parallel physical networks are connected to a device which already uses parallelism in some fashion, for example massively parallel computers. In this section we study the instantiation of the generic model in which a single application is running as a set of cooperating processes on a large set of individual processors, each with its own memory and connected through a complex communications fabric. The selection of this model was motivated by the structure of the Touchstone DELTA machine. The DELTA is a MIMD machine and can have up to 512 processors and has six I/O nodes (gateway processors). It is suitable for several computation intensive applications. For example, a user at a remote site is interested in the time evolution of, for example, the solution of a computational fluid dynamics problem. This application in addition to the need for intensive computation requires a large amount of data to be moved through the network at a very high speed (Gbps).

Background Traffic			Performance			
A-Bus Traffic	N-Bus Traffic	FDDI Channel	Architectures	Network Scheduler Policy	Throughput (Mbps)	Round Trip Delay (μsecs)
30% uniform	30% uniform	30% uniform	2-bus	RR	388	2253
			1-bus	RR	323	1092
			Serial	-	236	4311
30% uniform	30% uniform	30% non uniform	2-bus	RR	268	2601
				Adaptive	302	2246
			1-bus	RR	228	40997
				Adaptive	270	3449
			Serial	-	236	4442

Table 3: Summary of performance comparison

Packet Size (bytes)	Memory Speed (ns/word)	Processor Speed (MIPS)	Checksum Comput.	Time for Send() call Parallel Part (μs)	Time for Send() call Serial Part (μs)	Maximum Speedup
500	40	15	Software	58.2	12.1	5.8
	20	25		30.4	5.8	6.2
	40	15	Hardware	19.2	3.1	7.2
	20	25	(on the fly)	10.0	1.5	7.6
4500	40	15	Software	164.3	63.6	3.5
	20	25		98.6	39.6	3.4
	40	15	Hardware	14.3	3.6	4.9
	20	25	(on the fly)	8.8	2.1	5.1

Table 4: Potential speedup for send() system call

The instantiation of the model based on Touchstone Delta is shown in Figure 16 with the following features:

- a set of k application processors interconnected through a mesh type interconnection network,
- a set of n protocol (gateway) processors interconnected as a linear array,
- a set of m network units, one for each gateway processor,
- all types of processors are assumed to have local memory, and
- both the schedulers, application scheduler and network schedulers, are mapped onto the protocol processor.

4.1 Model and Simulation

This simulator builds on the simulator used for the workstation node and uses the same modules for the TCP/IP and FDDI simulations and the same DCRA method for handling acknowledgments and

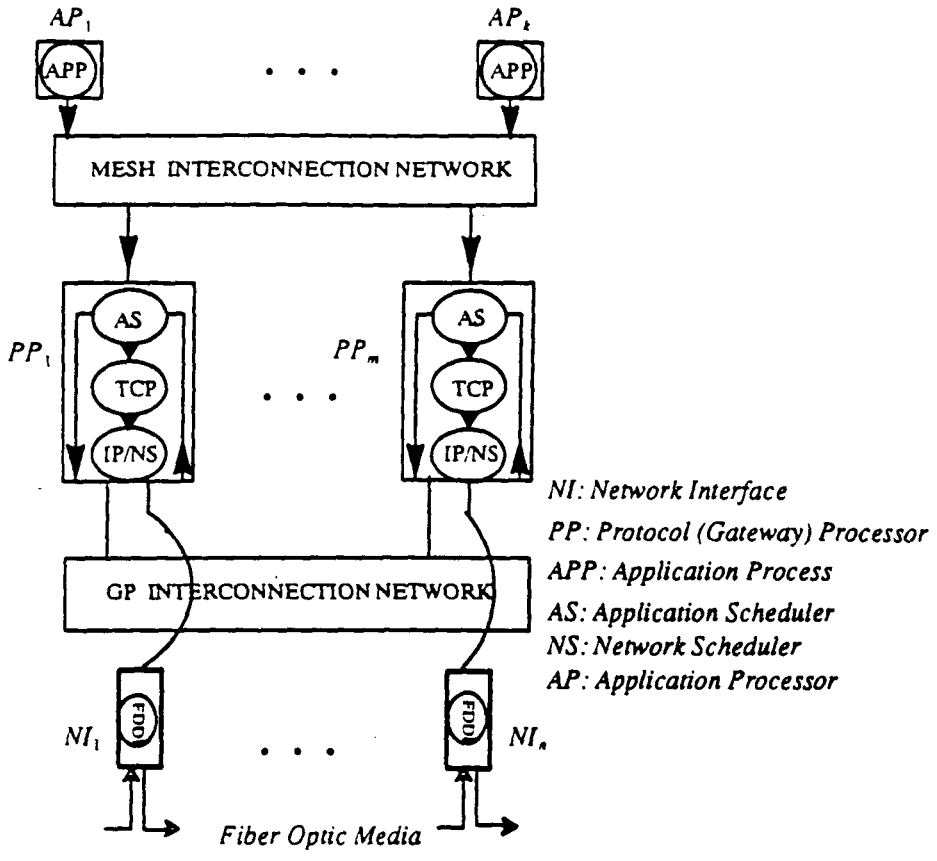


Figure 14: Distributed memory machine model

retransmissions. Some of the features of the model for which we did the simulation runs are summarized below.

- **Independent data generators.** The simulation model does not incorporate the modeling of the application data generation from the various processors. The protocol processors receive data from independent data generators.

- **Fixed mapping.** The number of protocol processors is much less than the number of application processors, thereby implying more than one application processor sends data to a single protocol processor. In general, this mapping of application processors to protocol processors may not be fixed. That is, an application processor with time can change its destination protocol processor. However, for simplicity we assume this mapping to be fixed. By changing the arrival data rate on a protocol processor we can have the same effect.

- **Scheduling strategy.** We simulate *fixed* and *adaptive* strategies for the application scheduler and network scheduler. In the fixed strategy the application scheduler sends data to its TCP/IP process. In the adaptive strategy the application scheduler, using the information from TCP/IP, can choose to send data to an application scheduler residing in the neighboring processor. An application scheduler accepts or rejects a segment depending upon the status of its input queue. If its input queue reaches the "high" threshold value, the application scheduler rejects the segment and passes it to the application scheduler on the right adjacent processor. This application scheduler behaves in

Sched. Policy	Throughput (Mbps)			
	No Background Load	Uniform Background Load (30%)	One Channel Overload	Two Channels Overload
Fixed/ Fixed	104.87	88.18	74.39	73.15
Adaptive/ Fixed	153.83	153.57	151.26	145.97
Adaptive/ Adaptive	154.54	153.86	151.81	146.23

Table 5: Impact of scheduling with data generation rate being 37% of the available capacity at a fixed window size of 64 Kbits and network length of 20 Km

Sched. Policy	Throughput (Mbps)			
	No Background Load	Uniform Background Load (30%)	One Channel Overload	Two Channels Overload
Fixed/ Fixed	224.47	125.18	100.54	88.15
Adaptive/ Fixed	374.29	270.88	266.53	258.04
Adaptive/ Adaptive	374.80	275.30	224.03	202.66

Table 6: Impact of scheduling with data generation rate being 87% of the available capacity at a fixed window size of 64 Kbits and network length of 20 Km

like fashion unless application scheduler queues on all the processors have reached the high threshold values. In this case the data segment will stay with the *last* processor. Here, the last processor is the left adjacent neighbor of the processor which initially had the data segment. To keep track of the last processor, a count byte is appended with the data segment. The count is incremented by a processor before it passes the data segment to the right neighbor. (We assume the linear array interconnection network has wrap around connection.) Similarly in the fixed strategy for network scheduler, data are sent to the network units residing in the same processor whereas in the adaptive strategy data can be send to the network units residing in the adjacent processor.

- **Control flow.** Once a data segment reaches the last processor, a control flow signal is broadcast to all data generators to stop sending data. The data generators are restarted once the application scheduler queue of any processor falls below the low threshold value. Note that we use two thresholds: "high" threshold and "low" threshold for the input queue at application scheduler. The "high" threshold is used for blocking the data generator, and the "low" threshold is used for starting the data generator.

- **Acknowledgments and retransmissions.** The acknowledgment for a packet after coming to a network interface is passed to the network scheduler of the connected processor. The network scheduler diverts the acknowledgment to the proper TCP where the packet was generated. A TCP,

Sched. Policy	Throughput (Mbps)					
	Network Length of 5 Km		Network Length of 20 Km		Network Length of 100 Km	
	Uniform Load	Two Channel Overload	Uniform Load	Two Channel Overload	Uniform Load	Two Channel Overload
Fixed / Fixed	88.65	73.15	88.18	72.15	75.2	60.34
Adaptive/ Fixed	153.82	147.43	153.56	145.97	125.2	120.20
Adaptive/ Adaptive	154.01	147.56	154.20	146.10	146.52	127.58

Table 7: Impact of scheduling on different network lengths with data generation rate being 37% of the available capacity at a fixed window size of 64 Kbits

after collecting all the packet acknowledgments of a segment (which it had broken into packets), sends the segment-ack to the application scheduler.

- **Hardware Characteristics.** In our simulation some of the parameter values such as time taken to move data from one processor to an adjacent processor do not correspond to the actual values of the Touchstone DELTA. But these timings do not affect our conclusions since they are based on relative performances. We also assume that in the future fast lower level protocols such as FDDI will be supported on the system.

4.2 Performance Results

Our approach to evaluate the system mirrors the one described in Section 3.2. The emphasis here is on the impact of scheduling, window size, and network length.

- **Scheduling impact.** The scheduling impact results are summarized in Tables 5 and 6. The first column of these tables give the scheduling strategies adopted for application scheduler and network scheduler.

  For example, an entry *Adaptive/Fixed* implies that we use adaptive strategy for application scheduler and fixed strategy for network scheduler. The second column lists the throughput under no background load conditions. The third column lists the throughput under uniform background load. The fourth and fifth columns list throughput when one and two channels are overloaded. Table 5 gives results when the application offers data at a rate well below the capacity of the parallel network and Table 6 gives the results when the application has to be throttled because the parallel net is saturated.

  Below the performance knee of the delay curve for overall throughput, the scheduling can maintain throughput even when individual channels are overloaded. At saturation level the scheduler helps improve throughput but cannot effectively shift the load. As currently simulated the network schedulers actually degrade performance when the channels are overloaded. The key reason for this behavior is the linear interconnection scheme of the Touchstone I/O nodes. We are developing alternative scheduling policies which, we hope, will solve these problems.

Window Size	Latency (μsec)			
	No Load	Uniform load (30%)	One Channel Overload	Two Channel Overload
4000	76	88	180	1150
8000	53	77	80	220
14000	70	70	70	90

Table 8: Impact of scheduling on window size with data generation rate being 37% of the available capacity at a fixed network length of 20 Km for adaptive/adaptive scheduling

- **Window size and network length.** In Table 7 we observe that throughput is maintained better under overload for longer networks when we use scheduling (although they are not yet optimal). In longer networks queues start building up at the network interface units which are redistributed by the schedulers.

  Table 8 shows the expected impact of latency by increased window size under no load and shows that the effect is exacerbated by overloading of channels. We observed the same randomization effect on round-trip delays over parallel channels which we saw in the workstation node and expect solutions for the window management and retransmissions used in the workstation node to work as well here.

In summary, we have shown that coarse grain parallelism is equally applicable to distributed memory parallel machines. The protocol issues and solutions carry forward from the bus-based machines but scheduling policies have to be adapted to the architectural features of parallel machines.

5 Conclusions

In this paper we have analyzed the use of coarse-grain parallelism (that is, the use of both multiple protocol processors running replicated software and several physical channels) to provide high speed communication services to a single application at a single network node. We developed a general structure for coarse-grain parallelism appropriate for use in a gigabit per second node. Since performance is highly dependent on real issues such as hardware properties (e.g., memory speeds and cache hit rates), operating system interference (e.g., interrupt handling), and protocol performance (e.g. effect of timeouts) we performed detailed simulation studies of both a bus-based multiprocessor workstation node (based on the Sun Galaxy MP multiprocessor) and a distributed-memory parallel computer node (based on the Touchstone DELTA). Mapping of the general model into concrete architectures requires selection of scheduling algorithms and assigning processes (protocol and scheduling) to physical processors. To operate near the potential speeds possible using the particular architecture, this mapping must reflect the communication fabric available in the underlying hardware.

We can draw some general conclusions about the use of coarse-grained parallelism for high performance networking. Some are based directly on the performance studies which we concluded of the two hardware architectures we studied:

- Coarse-grain parallelism can deliver multiple 100Mbps with currently available hardware platforms

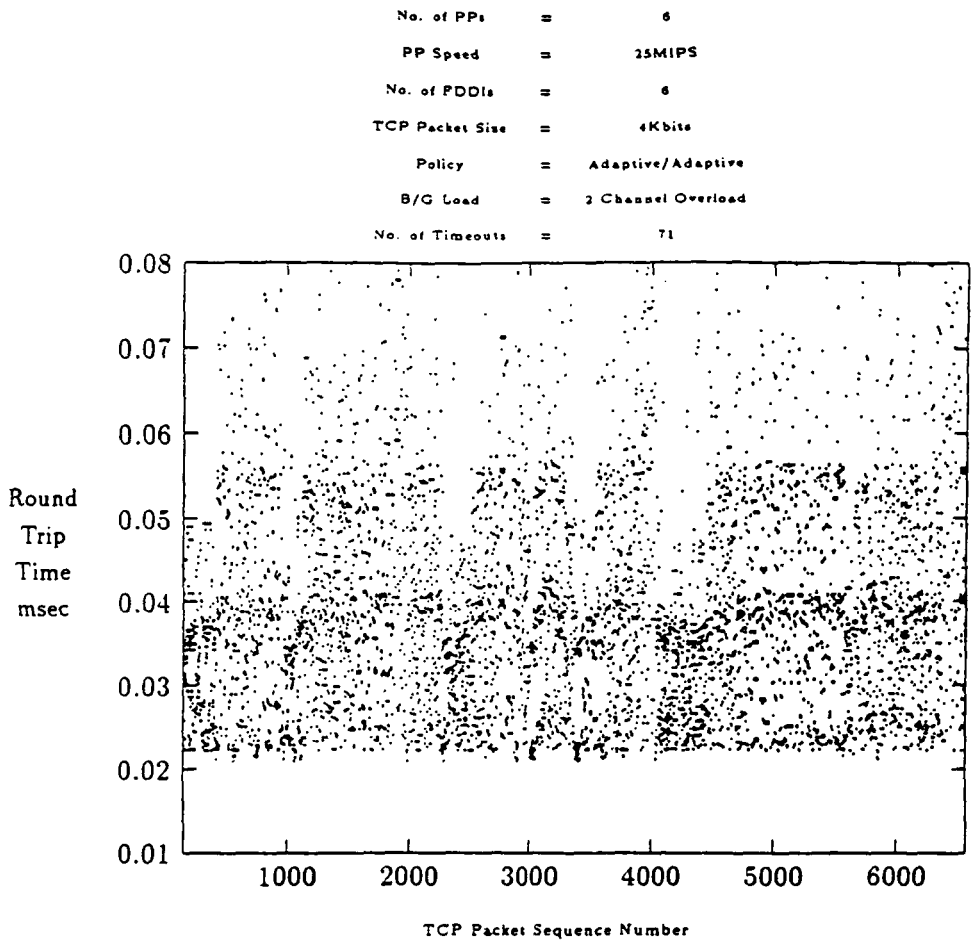


Figure 15: RTT Distribution for TCP Packets

(such as the Sun Galaxy) and with existing networking protocols (using TCP/IP) using parallel FDDI rings.

- Scale up is near linear in the number of protocol processors and channels (at least up to a few hundred Mbps). This is more significant since the analysis is for hardware architectures and existing protocols (TCP) and MAC layer components (FDDI) not designed with high speed network applications in mind.
- Since these results are based on existing hardware (both the Sun and the Touchstone) without specialized hardware (except perhaps for some simple modifications of the FDDI boards), we feel that this is a low cost solution to providing multiple 100Mbps on current machines.

In addition, from both our detailed performance analysis of the physical structures of these architectures based on coarse-grained parallelism, and the components assumed to be present in the architectures, we conclude:

- The use of multiple processors providing identical services and the use of space division multiplexing will provide better reliability than monolithic approaches if properly designed. It also provides graceful degradation and low-cost load balancing.
- This architecture supports running several different protocols (e.g., TCP and UDP) in parallel. This allows, for example, different TCPs to manage network connections with different service requirements

(many small messages for many users handled by one TCP with several other TCPs providing a high bandwidth network connection for a single application).

- The basic architecture will be able to incorporate many improvements from other work (e.g., reduced data movement, fast TCP, gigabit nodes, fine-grained parallelism) again with a near linear speed-ups (at least for a small n) as these improvements become available.

We can also make some detailed conclusions about the particular architectures we studied. These conclusions should also apply to other hardware platforms with similar features.

- The Sun MP Galaxy based parallel processing architecture is capable of delivering throughput in excess of 300Mbps provided sufficiently high speed memory is available.
- A Touchstone-like machine is capable of delivering 300 Mbps network service to a multiprocessor distributed application using FDDI (though as is typical with parallel algorithms, performance depends on how effectively the application uses the Touchstone processors).

Scheduling and scheduler placement has significant impact on performance. Related to scheduler placement and scheduling algorithms:

- The 2-bus distributed scheduler architecture outperforms other bus-based architectures in terms of higher throughput, lower delay, lower processor MIPS and bus speeds.
- The throughput capability of a 1-bus centralized scheduler architecture is limited by the shared memory bandwidth.
- For the bus-based architectures a simple scheduling policy of transport to network scheduler fails to push expected throughput at lower round trip delay under nonuniform channel load conditions.
- On the parallel computer node, the scheduling technique which we implemented (shifting packets among protocol processors) maintains throughput irrespective of load on the channels at lower data generation rates but does not work well in overload conditions.
- On the parallel computer node, the current TS scheduler is more effective for larger network lengths.

Related to properties of protocols when used in a parallel environment:

- The segment size of the data flowing from application to the transport layer must be the same size as MAC frame size to reduce round-trip delay time.
- TCP window size should be such that none of the protocol processors starve for data and no packet waits extra for available TCP send window.
- Efficient implementation of the TCP timeout timers mechanism in case of multiple parallel channels is nontrivial and requires further study.
- As window size increases, latency increases for each architecture which we examined.
- On large diameter networks (with large latencies) the parallel channels can be used with techniques such as cross channel coding to reduce latency (somewhat) and to significantly reduce the need for retransmission of data.

Our general conclusion is that coarse-grain parallelism is effective for increasing the networking capabilities of currently available hardware and is a promising approach for building a true gigabit node. It is compatible with, and complements, much of other work designing gigabit nodes.

We are continuing our analysis and are in the process of designing and implementing a functional hardware prototype capable of providing several 100Mbps to a single application. This will allow us to validate our concepts and to refine our analysis and modeling techniques so that we can design a gigabit node based on this approach.

Acknowledgments

We wish to thank R. Yerraballi, S. Kelkar and H. Srivatsan for their contributions to this work.

References

[1] Bharat Bhargava, Tom Mueller, and John Reidel. Experimental Analysis of Layered Ethernet Software. *IEEE Publication*, pages 559–568, 1987.

[2] W. E. Burr. The FDDI Optical Data Link. *IEEE Communications*, pages 18–23, May 1986.

[3] D. Cheriton and C. Williamson. VMTP as the Transport Layer for High-Perfromance Distributed Systems. *IEEE Communications Magazine*, 27:158–169, June 1989.

[4] G. Chesson. XTP/PE Design Considerations. *Processings of IFIP Workshop on Protocols for High-Speed Networks*, pages 27–33, May 1989.

[5] G. Chesson. XTP/PE Design Conserations. *Protocol Engines, Inc.*, 1990.

[6] I. Chlamtac and A. Ganz. A Multibus Train Communication (AMTRAC) Architecture for High-Speed Fiber Optic Networks. *IEEE Transactions on Communications*, pages 903–912, July 1988.

[7] D. D. Clark. Modularity and Efficiency in Portocol Implementation. *RFC 817, NIC*, July 1982.

[8] D. D. Clark, V. Jacobson, J. Romkey, and H. Salven. An Analyis of TCP Processing Overhead. *IEEE Communications Magazine*, pages 23–29, June 1989.

[9] P. Cochrane and M. Brain. Future Optical Fiber Transmission Technology and Networks. *IEEE Communications*, pages 45–60. November 1988.

[10] J. F. Ewen et al. Gb/s fiber optic link adapter chip set. *10th Annual IEEE Gallium Arsendie Integrated Cricuit Symposium*, pages 11–14. Nov. 6-9, 1988.

[11] William et. al. Survey of Light Weight Transport Protocols for High Speed networks. *IEEE Transactions on communications*, pages 2025–2039, November 1990.

[12] E.C. Foudriat, K.J. Maly, C. M Overstreet, S. Khanna, L. Zhang, and W. Sun. Combining two media access protocols to support integrated traffic on high data rate networks, 14pp. *Proceedings of the 16th Local Computer Network Conference, Minneapolis*, 1991.

[13] A. G. Fraser. Towards a Universal Data Transport System. *IEEE Journal on selected areas in communications*, SAC-1:803–816. November 1983.

[14] Z. Haas. A Communication Architecture for High-Speed Networking. *Proceedings of IEEE Infocom*, 1990.

[15] B. Hoffman, W. Effelsberg, T. Held, and H. Konig. On the parallel implementation of osi protocols. *Proceedings of IEEE workshop on the architecture and implementation of high performance communication subsystems*, February 1992.

[16] R.N. Ibbett and N.P. Topham. *Architectures of High Performance Computers - Volume II.* Springer-Verlag, NY., 1989.

[17] Van Jacobson. Congestion Avoidance and Control. *Processidngs of ACM SIGCOMM'88*, pages 314–329, 1988.

[18] N. Jain, M. Schwartz, and T.R. Bashkow. Transport Protocol Processing at GBPS Rates. *Proceedings of Sigcomm '90*, pages 188 –199, 1990.

[19] K. Kaede. Twelve-channel parallel optical fiber transmission using a low-drive-current 1.3 μm led array and a pin pd array. *OFC'89 Technical Digest*, page 15, Feb. 1989.

[20] S. R. Kunkel and J. E. Smith. Optimal pipelining in supercomputers. *Proceedings of 13th Symposium on Computer Architecture*, June 1986.

[21] T. LaPorta and M. Schwartz. Design, verification and analysis of a high speed protocol parallel implementation architecture. *Proceedings of IEEE workshop on the architecture and implementation of high performance communication subsystems*, February 1992.

[22] K. Maly, E. C. Foudriat, D. Game, R. Mukkamala, and C. M. Overstreet. Dynamic allocation of bandwidth in multichannel metropolitan area networks. *Computer Networks & ISDN*, 1992. To appear.

[23] K. Maly, C. M. Overstreet, R. Mukkamala, M. Zubair, F. Paterra, S. Khanna, Y. S. Sekhar, and R Yerraballi. Design, verification and analysis of a high speed protocol parallel implementation architecture. *Proceedings of IEEE workshop on the architecture and implementation of high performance communication subsystems*, February 1992.

[24] K. Maly, F. Paterra, C. M. Overstreet, R. Mukkamala, and S. Khanna. Remote visualization using parallelism in the context of existing networks. *To appear in Proceedings of International Phoenix Conference on Computers and communications*, April 1992.

[25] NRI. *Toward a National Research Network*. National Research Council, 1988. NRI Review Committee.

[26] David A. Patterson, Garth Gibson, and Randy H. Katz. A Case for Redundant Arrays of Inexpensive Disks (RAID). *ACM SIGMOD Conference*, pages 106–116, June 1988.

[27] Thomas F. La Porta and Mischa Schwartz. Architectures, Features, and Implementation of High-Speed Transport Protocols. *IEEE Network Magazine*, 5:14–22, May 1991.

[28] J. B. Postel. Transmission Control Protocol. *RFC 793*, September 1981.

[29] L. Svobodova. Implementing OSI Systems. *IEEE Journal on Selected Areas in Communications*, pages 1115–1130, September 1989.

[30] R. W. Watson and S. A. Mamrak. Gaining Efficiency in Transport Services by Appropiate Design and Implementation Choices. *Processings of IFIP Workshop on Protocols for High-Speed Networks*, 1987.

[31] J. A. Wiencko. Cross-channel coding implementation guide, Novemeber 1991.

[32] M. Zitterbart. High-Speed Transport Components. *IEEE Network Magazine*, pages 54–63, 1990.

[33] M. Zitterbart. Parallel protocol implementations on transputers - experiences with osi tp4, osi clnp, and xtp. *Proceedings of IEEE workshop on the architecture and implementation of high performance communication subsystems*, page 4, February 1992.

High Performance Interconnection Between High Data Rate Networks

by

E.C. Foudriat, K. Maly, C.M. Overstreet, L. Zhang, W. Sun
Computer Science Department, Old Dominion University
Norfolk, VA 23529

Presented at the International Workshop on Advanced Communications and Applications for High Speed Networks
March 16 - 19, 1992
Munich, Germany

Abstract

The paper discusses the bridge/gateway system needed to interconnect a wide range of computer networks to support a wide range of user quality-of-service requirements. The bridge/gateway must handle a wide range of message types including synchronous and asynchronous traffic, large, bursty messages, short, self-contained messages, time critical messages, etc. The paper shows that messages can be classified into three basic classes, synchronous and large and small asynchronous messages. The first two require call setup so that packet identification, buffer handling, etc. can be supported in the bridge/gateway. Identification enalbes resequencing of messages at the bridge/gateway which supports interconnection between networks having large differences in packet size. The third class is for messages which do not require call setup. Resequencing hardware is presented in the paper based to handle two types of resequencing problems. The first is for virtual parallel circuitwhich can scramble channel bytes. The secondsystem is effective in handling both synchronous and asynchronous traffic between networks with highly differing packet sizes and data rates. The two other major needs for the bridge/gateway are congestion and error control. The paper presents a new dynamic, lossless congestion control scheme which can easily support effective error correction. Results indicate that the congestion control scheme provide close to optimal capacity under congested conditions. Under conditions where error may develop due to intervening networks which are not lossless, intermediate error recovery and correction takes 1/3 less time than equivalent end-to-end error correction under similar conditions.

I. Introduction

Network systems require support for interconnection between networks and to users since they span widely different environments and must provide a variety of Quality-of-Service (QOS) features. This is especially true now that gigabit networks and multimedia environments [13] are becoming a reality. This paper investigates systems required to interconnecting high data rate networks. Special attention is directed toward providing support for a wide variety of applications which are implied by multimedia applications.

Present network interfacing supports service between connectionless networks usually via bridges or gateways [1]. Tthe major interconnection problems has been routing. Gateways, such as in Internet, provide a compatible protocol at the network layer which forwards packets to toward their destination on the selected "best route" and, where necessary, breaks up the packet to conform to any lower layer packet length restrictions [1, 2]. Thus, all stations must implement the same network layer protocol.

To over come the problem of common network layer, especially where not needed the transparent bridge approach is used. Bridges generally implement some form of spanning tree algorithm which eventually gets the packet to its destination but some times via a "non-shortest" route [1, 2, 3, 4]. Packet reformatting in bridges is minimal at best.

Typical interconnection between networks include those which support X.25, SNA, DECNet, XNS. etc., and span the data rate range from modems to LANs. DARPA's Internet [2] is a packet switched WAN network implemented over leased telephone and satellite links but its protocols have been used in LANs also.

Interconnecting systems for high data rate networks, that is, systems directed toward the implementation of Broadband ISDN (BISDN) also consider mainly connectionless protocol support. One system considers the use of DQDB for MAN interconnections [5]. Another supports the OSI connectionless network protocol (CLNP) in a ISDN environment [6]. Still another, frame-relay provides interconnection support for ISDN but does so by supporting critical functions at the end points only [7]. Other forms of network interconnections include the HIPPI-to-ATM HAS interconnection scheme for Nectar [8], and direct ATM-to-host for workstation coupling [9, 10] but,

[1]Research support has ben provided by Sun Microsystems, RF539044, NASA. Langley Research Center grant NAG-1-908. and Virginia Center for Innovative Technology grant INF-89-002-01

for the most part, these are point-to-point connections which do not span a range of different networks. Others have proposed transferring most of the protocol operations to the application level [11] but this means that the each host will have to implement the QOS needs for the user.

While there is no question that BISDN and its related systems provide capable, flexible network technology, they do not appear to be suitable for the wide range of network interconnections which are possible and which certainly might be considered desirable. First, they do not consider connection to vastly different formats or data rates, such as Ethernet and most certainly not to twisted pair Ethernet systems which are so common in personal computer network systems. A recent paper [12] does consider an ATM - FDDI gateway but provides only minimal user support. Second, while BISDN networks are suppose to support a wide range of services, it is not clear that all services will be provided by the interconnecting network itself as noted for frame relays [7] and discussions of ATM systems [14]. Third, BISDN packet and protocol header size lead to considerable bandwidth inefficiency for connecting networks which do not maintain its format but must incorporate its packets [12]. Further, many network problems such as routing are considerably different when "on-premise" networks are included. Finally, in some common carrier network situations, it may be reasonable to provide improved performance and/or lower cost by outright leasing of bandwidth and using network interface systems provided by the customer [15]. Many of the above problems and situations are best handled by employing bridge/gateway interfacing to common carrier high data rate networks and between private systems.

In this paper, we discuss new features which, when incorporated into bridge/gateway systems, provide significant improvement in overall network connectivity. In the next section, the requirements bridge/gateways are presented and QOS factors discussed. These lead to bridge/gateway features, presented in Section 3. Features include call handling, message classification and packet identification to support QOS requirements. Next, receiver handling which provides resequencing and restructuring systems, and sender handling which includes a new lossless congestion and error control system are discussed. Performance information about both handlers is presented. The last subsection presents information on intermediate error correction available in the receiver and sender handling systems. It substantially reduces error correction times.

II. Bridge/Gateway Requirements

Bridge/gateway requirements include interfacing to:

1. a wide range of data rates from submegabit/sec. to gigabit/sec.;
2. a wide range of protocol and packet structures;
3. a wide range of connectivity and topology structures; and
4. support for a wide range of quality of service requirements which can include, synchronous, asynchronous, and bursty traffic, low latency, acceptable error to error free and reliable service, etc.

Each of these factors influence the bridge/gateway structures which are needed. For example, a typical scenario might be:

> Three people cooperate to develop an architectural drawing including layouts, 3D moving scenes, correlated voice commentary, etc. Each person has a different graphics configuration and connectivity through a different network system. The information is developed by one person, shipped to the others and then they participate both on-line and off-line in changes. Finally, the finished product is shipped to a fourth party again through a different network and displayed as an advertisement but not copied. The original and transfer files are compressed resulting in traffic conditions which include bursty, error free, alternatively high and low data rates, multicast, long periods where no data is being exchanged, etc. When shipped to the fourth party, delay shift for the components is critical.

In this situation, a direct network to host connection would require each cooperating party to have multiple network interfaces and a considerable variety of software to handle the differing network interconnections efficiently. Would it not be better to connect each host to a flexible bridge/gateway system which could effectively support a large range of interfaces with tailored quality of service requirements for a large group of clients?

III. Bridge/gateway Features

To support the diverse bridge/gateway requirements, we need to isolate the bridge/gateway operational features. The critical feature are:

1. call handling - especially service requirements and resource allocation;
2. receiving - especially resequencing for the next network segment;
3. sending - formatting and controlling flow for minimal loss; and
4. error detection and correction which involves both sending and receiving.

Figure 1 illustrates the general logic structure for a bridge/gateway interconnection system. On the receiver

side, the first set of blocks are those needed to handle the access protocol, that is, decoding the bit pattern, maintaining clock synchronization and decoding header and trailer information. This latter decoding provides information including message type, destination, etc. The basic packet is stored in a shift register while decoding and checking are accomplished so that the packet may be forwarded in the event that its destination is for another node.

The shaded blocks in Figure 1 are those devoted to special bridge/gateway operations. Here the packet header information determines the message identity and places or links packets in memory in order to properly resequence packets. In a next section, operations needed for handling specific message classes will be discussed. New packets are then prepared for transmittal to the outgoing network.

A. Call Handling

Although this paper is mainly directed toward the hardware and lower layer software to control network interfacing, certain aspects of call handling need to be discussed. It is assumed that call handling will be supported mainly in software, so its features are reasonably flexible to suit specific needs.

Call handling for multimedia operations is considerably more complex than reserving channels in frames or routing packets at immediate nodes toward their destination. It is expected that call handling will require a dialogue between the caller and the network system where the user specifies and even may negotiate with the network over services which are available. The elements of the dialog will include the QOS requirements which the user deems to be critical and the ability of the intervening network to provide such service and the expected cost. To further complicate the call handling operations, the QOS requirements may be transient, i.e., different QOS requirements may exist over different portions of the total call. While it would be nice to assume that the differing QOS requirements would be known at call time further complexity may be required to alter call QOS requirements because of unanticipated situations which arise after the call has started. The user may wish to examine various network configurations to arrive at the one which best suits all his needs.

Once the call handling is completed, then the bridge/gateway operations which send, receive and service the packets and message can take place.

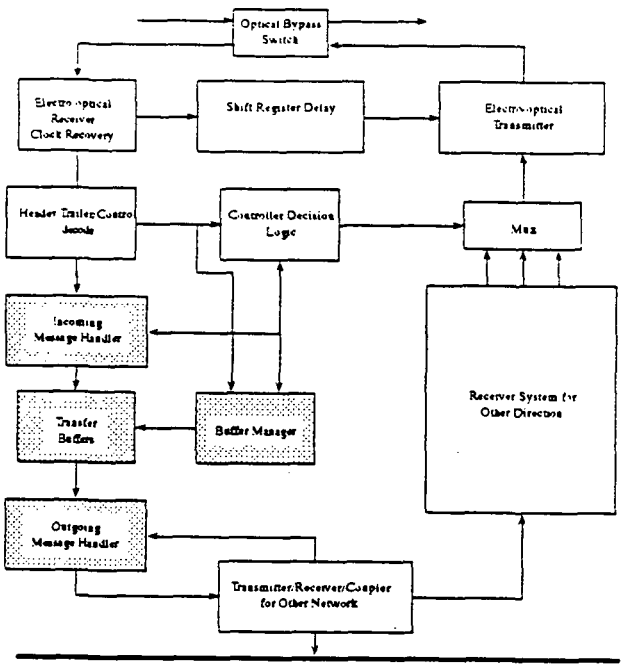


Figure 1. Bridge/Gateway Connection Diagram
Bridge/Gateway specific components shaded.

Message Classification

We do not treat many aspects of call handling in this paper. However, we identify three types of messages which are significant by the fact that they support a wide range of QOS requirements and should be identified individually for bridge/gateway operations. They are:

1. Synchronous messages which require call setup/termination at all B/G nodes between source and destination. Typically, these messages are voice or video type messages. Conditions for this class include known data rates, known route and known latency. This translates into operations where a fixed size buffer is allocated and headers for each link are built apriori. Synchronous messages are designated as class A
2. Asynchronous messages which choose to establish message control at B/Gs. Typically, large data blocks and file transfers use this class. Conditions for this class include unknown but source estimated data rates and known route. This translates into operations where fixed headers for each link are built apriori and where nominal buffer sizes are allocated. The operation requires a call setup/termination procedure. Asynchronous messages are designated as class B.
3. Asynchronous messages which are self-contained. A typical member of this class is e-mail. Data rates are not considered and routing is either self-contained or inserted by the bridge/gateway from a fixed route table. No call setup/termination operations are needed. Self-contained messages are designated as

class C.

The message classification scheme provides the minimum number of classes needed to satisfy the range of quality of service requirements. Class A, synchronous messages must have a unique classification since media access protocols support synchronous (isochronous) traffic differently from asynchronous traffic. Although voice and video have significantly different data rates, both are effectively handled with a single reservation type system. Class B and C traffic support typical asynchronous traffic situations. Most networks experience a bimodal traffic size distribution. Class B messages are identified through call setup and termination in order to preserve the efficiency over network links some of which require very small packet size, like ATM, and others accept large packet sizes, like FDDI. Class C traffic is for messages which need to get there but do not require a separate identity. Messages encompass those which are short and do not require call setup or negotiation for services which in itself may be time consuming. Latency is handled by establishing priorities within and between each traffic class and supporting it through queue control at the bridge/gateways. Note that reference [12] provides for two distinct types of service control, user and persistent connection/connectionless oriented data services but they are not oriented to the broad range of user needs.

B. Receiver Handling

A significant part of receiving packets and messages is the problem of resequencing. Based upon the nature of the interconnecting networks, two resequencing problems exist.

Virtual Circuit Resequencing

This resequencing problem arises when parallel virtual circuit channels are used to provide a high bandwidth connection. It occurs when individual blocks arriving at the receiver do not have message identity within their own right but whose ordering is provided by the location of the channels within the frames as they arrive. As in [16, 17], we assume that once the parallel channels to provide bandwidth have been reserved, that no further packet restructuring takes place dynamically. Hence, the resequencing between channels is formulated at call setup time and will remain fixed for the duration of the call.

The logic circuit, shown in Figure 2, consists of two parts. Based on present telephone virtual circuity, messages are separated into channels in frames. Each channel supports 64 Kbps, i.e., a byte of data each 12μsec. The FIFO buffers provides the necessary incremental delay to resequence channels in arriving frames assuming that each frame may arrive at the receiver via different routes including different intermediate switching nodes. At setup time, each buffer is loaded with the correct number of dummy bytes so that the byte at the head of each buffer represents the correct order for the bytes when sent. The stream address and transfer control portion of the circuit handles the reordering of channels when they arrive as a stream of bytes. Stream resequencing is required because bytes may be switched in a frame based upon random selection of channels circuits when the call is setup and the receiving circuity may arbitrarily order frame placement in the stream to the resequencer. The address stream is established at call setup by a special message where order is known and where the 125μsec. frame of bytes is deciphered to obtain the FIFO buffer address where each byte should be placed and the number of dummy bytes loaded into each FIFO buffer at initiation time.

Reference [17] discusses a similar resequencing system based upon transputers. In [16], we analyzed the use of parallel virtual circuit channels to support a high data rate ring network over existing telecommunications circuit. In that paper, we develop and discuss in greater detail the resequencing logic system shown in Figure 2. While the logic circuity is not able to handle dynamic bandwidth changes as easily as the transputer system [17], it reduces circuit latency from 5 msec. for the transputer system to a few μsec and should be significantly smaller and less costly to build.

Packet/Message Resequencing

The second resequencing situation occurs whenever packets are received and must be resequenced to be forwarded to another network where different operational conditions exist. In this situation, it is either impossible or impractical to maintain packets from messages in an identical format with only replacement of header information and operations such as checksum recalculation. Unlike virtual circuit resequencing, it is assumed that sufficient header information exists to correctly identify the packet and its placement in the message.

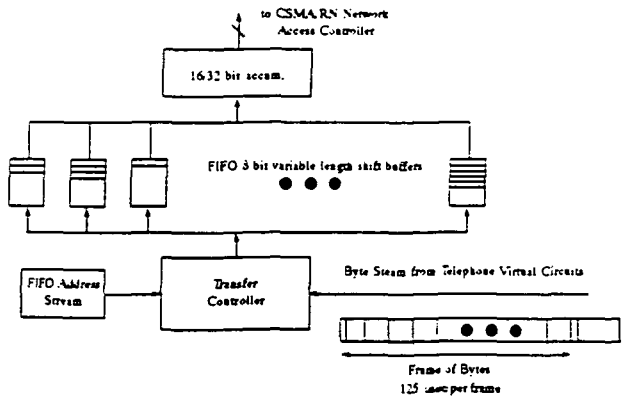


Figure 2 Virtual Circuit Resequencing Logic Diagram

Figure 3 shows the resequencing logic structure. As packets arrive, they are sent to the message handler which routes the message id and to the associative memory system and the packet length to the message update controller. A signal for additional space to place the packet is sent to the memory management unit. The message part of the packet is sent directly to free space in the memory buffer and is stored at the address provided by the MMU. The memory controller structure uses the concept of associative content addressable memory.

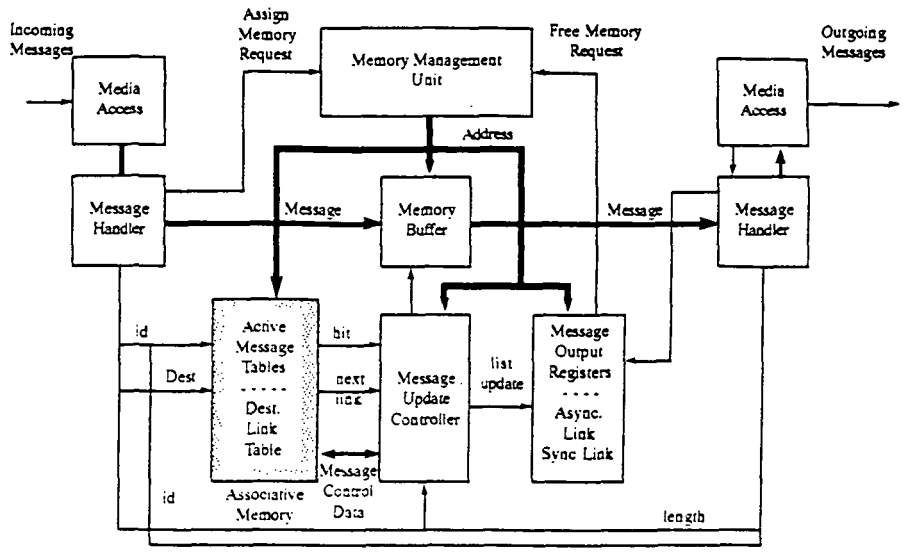


Figure 3 Bridge/Gateway Resequencing System

Outgoing messages are handled in a similar matter manner. After the controller selects the next message to be submitted to the outgoing channel, the control information is transferred to the message output registers. A number of message packets may be transferred since both synchronous and asynchronous messages ready to send may exit in the bridge/gateway memory buffer. Upon indication of the next media access, the information is transferred through the message handler and the information such as id and length which will change the content addressable memory data is feed back to the associate memory table to update the circuit information.

The system shown in Figure 3 is similar to systems which are being developed for ATM to host interfacing [9]. The major difference from the resequencing standpoint is that here a number of messages may exist simultaneously, that message classes require different handling than those to a host interface and that information obtained by the packet arrival is used to support both congestion and error control (see next two subsections).

Processing speeds for the resequencing system are estimated from the preliminary design. The resequencing system should be able to handle arrivals at high megabit data rates if nanosecond logic circuits are used [28]. Handling times, i.e., from the time the packet is received until it is placed in memory and linked properly, are estimated to be in the tens ofmsecs. While there is significant latency buildup while accumulating packets when the outgoing network has large block sizes, the actual added latency for the last message bit due to the resequencing is only the packet processing time. This is generally small in comparison to other delay such as end-to-end higher level protocol processing [18] and propagation delay in MAN and WAN systems.

C. Sender Handling

In past network systems, sender operations were generally straightforward. They consist of breaking messages into packet size blocks acceptable to the network, and, under the condition where the node has multiple links, routing packets. With the advent of ATM and the use of virtual paths which will be selected at call setup time [19, 20], the routing problem for networks using high data rate services over common carrier links will diminish. Routing for private "on-premise" systems is generally easily handled since the network configuration remains constant for large periods of time after installation or upgrade. However, with ATM and with the requirement to interconnect between a wide range of network data rates, the routing problem has been replaced by another equally important and difficult problem, that of congestion control

A number of congestion control methods have been proposed many based upon the concept of *statistical multiplexing* [21, 22] and implemented using the concept of a leaky bucket [21, 23]. The concept of statistical multiplexing implies that, should an overflow occur, the network can discard packets enroute to alleviate the congested condition. The argument in favor congestion alleviation by discarding packets is based upon the fact that both voice and video signals have a significant amount of redundancy so some loss should readily be tolerated. However, in multimedia environments, not all synchronous traffic will be voice/video and even then there is

[1]Many recent papers, conferences and journals have articles related to high data rate network flow control problems. The IEEE Communications Magazine, Vol. 29, No. 19, Oct 1991, devoted the entire issue to congestion control

strong evidence that compression techniques will be used since they can decrease bandwidth resource requirements by a factor of 50-100. Compressed data and many asynchronous messages have no tolerance for unreplaced lost packets. In the following subsection, we discuss a lossless congestion control scheme and follow that discussion an error control system with improved correction performance. Note that the FDDI-ATM interface [12] provides no flow or packet loss control services.

Lossless Congestion Control System

The lossless congestion control system operates similar to many end-to-end systems except that different feedback parameters are used and that control is exercised between bridge/gateway points. Here, the receiver periodically sends a control packet to its sender(s). This control packet contains the present free buffer space, the number of the last arriving packet accepted, and an error indication bit. When this information arrives at the sender, it calculates the remaining free buffer space at the receiver at the time the feedback packet was sent. It can send packets at the maximum rate until it has reduced the free buffer space to zero pending the arrival of a new feedback packet from the receiver. If the control packet indicates that an error has occurred then the number of the last arriving packet accepted is used by the sender as the starting point for resubmittal of all subsequent packets, i.e., a go-back-N scheme. In a direct replacement scheme, the feedback packet would contain the numbers of only the missing packets.

The concept of returning an acknowledge with the last accepted packet number is well known and has been used both for window congestion and error controls [24]. This information, in itself, is insufficient to avoid potential loss due to buffer overflow and packet discarding. However, adding free buffer space information allows the source to have sufficient knowledge to fill up but not overflow the destination buffer, regardless of the destination's ability to forward packets which frees additional space in its buffer. Thus, the system suffers no loss due to discarding at any node, intermediate or sender, participating in the control scheme.

Further, the lossless congestion control system is dynamic not only with respect to load but also resources. If the destination node has additional free buffers which it can commit to the message, when it sends its next control packet it adds these buffers to its free buffer count. The source does not know or care whether the amount of free buffer space is due to commitment of new buffers or the destination has been able to empty buffers by forwarding packets to subsequent destinations. Likewise, if the destination takes buffers from a link, it sends a control packet indicating fewer free buffers and as the buffers become empty, they can be assigned to another circuit. Again, the source is unaware of the cause. Additionally, the system is flexible with respect to potential loss. Although we have described the system as lossless, if the source wishes, it can send packets above those for which the destination indicates it has free buffers. It takes a chance that the buffer space will not be available upon their arrival at the destination and hence, may be lost. However, with the integrated error control, noted above and discussed more fully in the next subsection, the loss is easily and quickly replaced.

The lossless feedback scheme fulfills the requirement that the control information is readily and easily available. One needs only to keep a counter to maintain free buffer size and to register the last packet number when the incoming header and data are transferred to storage. No complex system is required to scan a frame or to do averaging to attain information such as present bandwidth use and no separate clocking is involved.

Figures 4a) - 4e) show typical results for the lossless congestion control system. Figure 4a) illustrates the tandem network test configuration and the conditions simulated. Simulator runs were takes for a steady state arrival and service rates. Data was started after the system had time to reach steady state, i.e., at least 10 times the maximum transfer delay between nodes, and data was taken so that 90% confidence interval results are expected. The lossless congestion control scheme and its performance are described more fully in reference [25].

Figure 4b) and 4c) show the mean packet delay time and mean packet service period at the source. As the congestion traffic is reduced, both the packet delay and service period decrease to their nominal conditions for an uncongested system. Figure 4d) shows that the feedback packet load on the network is less than 5% of the capable traffic and that it is not significantly influenced by the congestion condition. Hence, the control concept used to provide information to the sending node does not increase network use significantly. Figure 4e) illustrates that the control law is doing a good job, since under congested conditions, the percent of time that a node's queue is empty is less than 20%. This means that packets are available to be forwarded most of the time which is the best performance that can be obtained under congestion. Figure 5 shows the affect of buffer length during congestion. As buffer length increases, mean packet delay increases since the packets spend more time in intermediate node buffers but overall service rate decreases, since more total packets are transferred because the intermediate buffers are not empty as often. Thus, buffer assignment size is an important consideration is attaining overall performance.

[2]Note that packet identification for classification also supports *resequencing, congestion control and error correction.*

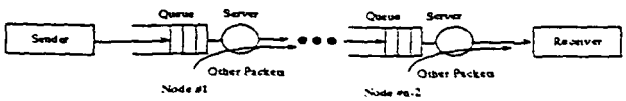


Figure 4a) Tandem Network Congestion Simulator

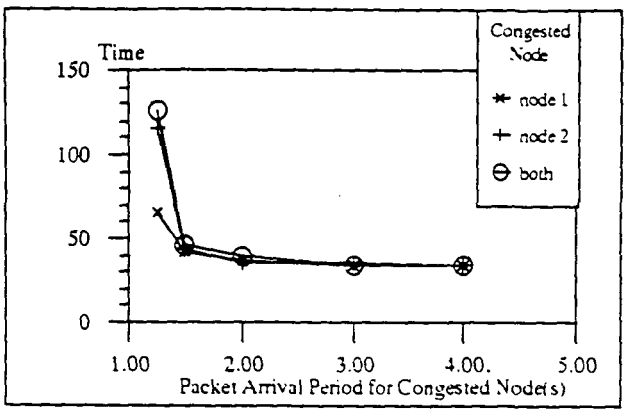


Figure 4b) Mean Transit Time

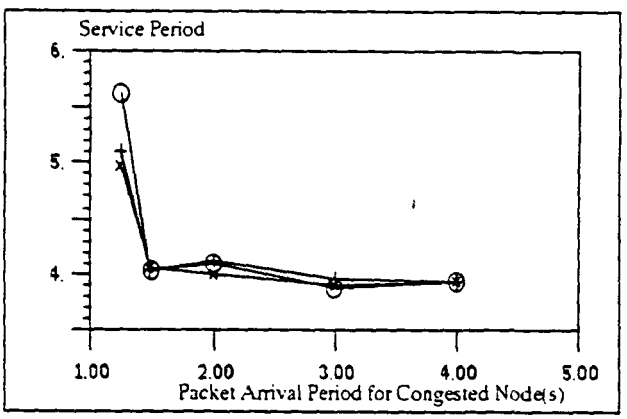


Figure 4c) Effective Source Service Period

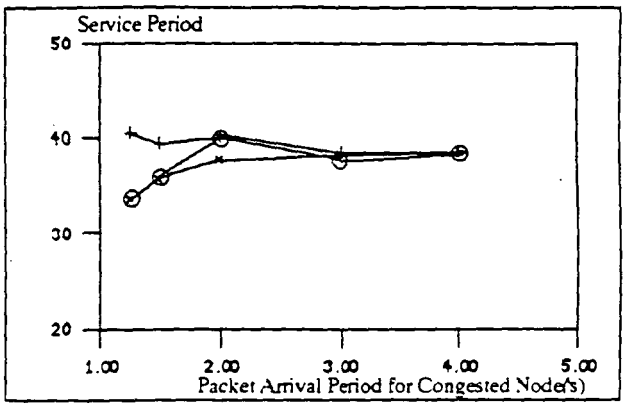


Figure 4d) Feedback Mean Service Period

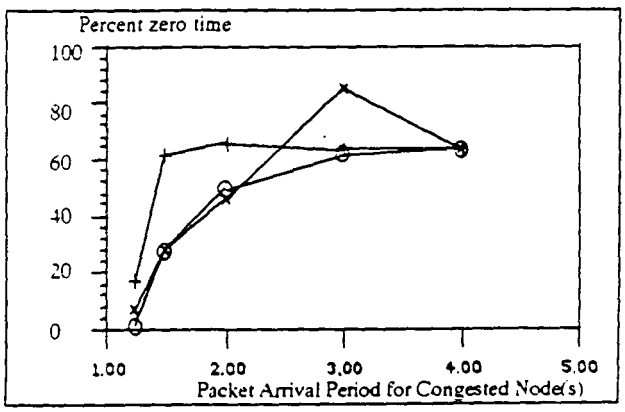


Figure 4e) Percent Time Node 1 Queue is Zero

Figure 4. Performance of Lossless Congestion Control System

Conditions - Nominal Service Period = 1.0
Other Packet Arrival Period = 1.25 - 4.0
Node Distance = 10.0
Sender Nominal Period = 4.0
(See Figure 4b) for Congestion Conditions)

D. Error Control

With the concept of congestion control relying heavily on the ability to drop packets, the question for error control becomes not whether but how. As with Internet, the mechanism suggested for ATM is based upon end-to-end error detection and correction at the transport layer [14, 26, 27]. While this mode is certainly feasible, it is highly questionable whether it is capable of enabling the wide QOS range that users have come to expect in LAN systems.

The error control technique as noted above can provide reliable datagrams between bridge/gateway systems handling messages. The congestion control scheme has the capability based upon the error indicator bit and last received packet number to alert the sender of loss regardless of cause. The error information is sufficient for the sender to replace the damage packet and reinitiate the remainder of the message without further coordination with the receiver. We have called this method of error correction between bridge/gateways intermediate error correction.

Performance for the intermediate error correction scheme is compared with the end-to-end error control implemented at transport level. The results are shown in Figure 6a) - 6b) for the conditions where intermediate links are 100 km and 1000 km long, respectively. It is assumed

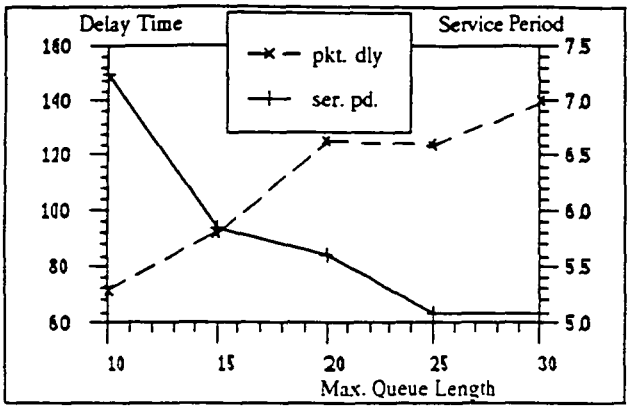


Figure 5 Effect of Queue Length on Congestion Control Performance

Conditions - Both node arrival periods - 1.25
Other conditions same as Figure 4

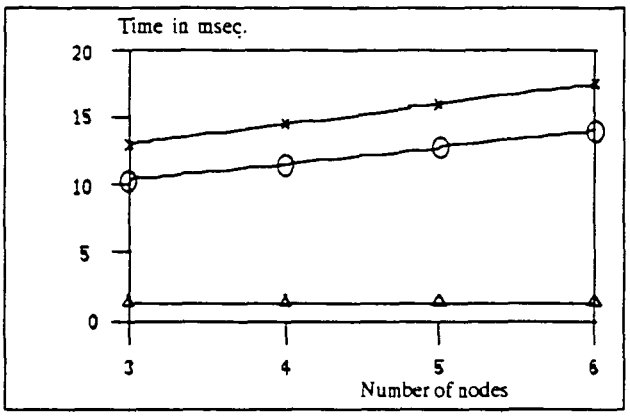


Figure 6a) Error Correction Time - 100km/link

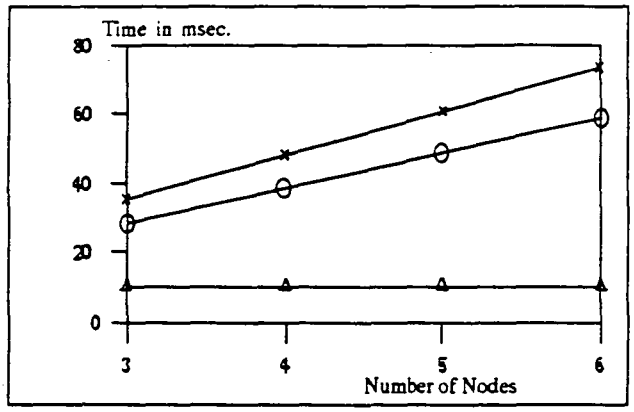


Figure 6b) Error Correction Time - 1000km/link

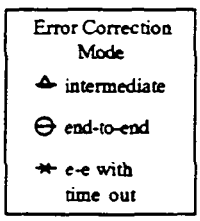


Figure 6 Intermediate Node Error Correction System

Conditions for Error Correction System
Node Processing
Upper Level - 2 msec.
Lower Level - 500μsec.

that bridge/gateway processing times are 100μsec and end-to-end transport processing for error control takes 2 msec. For transport layer correction, two mechanisms are modeled, one where the packet error is detected at the receiver and the error message immediately sent to the sender and the condition where the message times out. Time out is assumed as 1.5 times nominal end-to-end packet transfer time.

Link error correction shows significant improvement in time to correct an error. For example, correction time is always less than 1/3 of that required for end-to-end correction and as important, is dependent upon internode length instead of overall network length. This latter situation is especially valuable since constant QOS conditions can be supported independent of the total length of the communications link as long as maximum intermediate node distances are preserved.

V. Concluding Remarks

The bridge/gateway components required for interconnecting a wide range of high performance networks is developed in the paper. They exist because of the major requirement for bridge/gateway systems to support a wide range of network QOS requirements when hosts with differing capabilities are interconnected with through networks with equally wide range of capabilities.

Classification of messages into three classes where found to be the minimum by which bridge/gateways could support a wide range of traffic. Class A messages handle synchronous traffic; Class B and C for asynchronous traffic. Class B is for large messages which can tend to be bursty and requires a call setup and buffer allotment. Class C is for smaller "self-contained" messages which can be truly "packetized". To enable low latency message handling, the two asynchronous message classes, B and C, must have priority designations within the classes so that the bridge/gateway nodes can expedite critical traffic.

The paper presents effective resequencing hardware to support to conditions where arriving information may be disoriented. The first handles parallel virtual ciruits situations where messages may arrive in different channels. The combined switching buffering system provides rapid resequencing using hardware logic. with additional delays of only a few μsec.

The second resequencing systems is provided to handle messages where basic packet sizes are significantly different between the networks. If resequencing is not available, significant inefficiencies can occur. It is based upon identification of message packets as they arrive and to linking the packets in a buffering system. The structure of the buffering system is controlled by an associative memory system when packets are linked head-to-tail and where address pointers and lengths are associated with each

identifiable call. The resequencing system is designed to handle high data rates both incoming and outgoing and to provide delays ranging in the 10s to 100s of useconds.

Since message and packet identity are used for message classification and resequencing, the same information is availabe for congestion and error control. These features are extremely important in order to provide user QOS. A dynamic, lossless congestion control system is developed and its performance under typical operation is presented. The system, because it is based upon feedback of information from receiver to sender, completely avoids loss due to receiver buffer overflow. However, the system is very flexible so that operation where some loss may occur are easily implemented.

The parameters of the congestion control system directly enable intermediate node error recovery. Data is presented to demonstrate that intermediate node error control is able to correct errors usually within 1/3 the time as similar end-to-end error recovery systems. Thus, the combination of congestion and error control implemented at bridge/gateways provides significant performance improvement and hence, the ability to provide improved overall network QOS.

VI. References

1. Seifert, W.M.: "Bridges and Routers,"*IEEE Network;* Vol. 2; No. 1; Jan 1988.
2. Comer, D.:*Internetworking with TCP/IP*, Prentice Hall, NY. 1988.
3. Lin, Y.; Gerla, M.: "Brouter: The Transparent Bridge with Shortest Path in Interconnected LANs," Proc. of 16th LCN; Minneapolis, MN.; Oct. 14-17, 1991; pp. 175 - 183.
4. Hamner, M.C.; Samsen, G.R.: "Source Routing for Bridged Local Area Networks,"*IEEE Network* Vol 2; No. 1; Jan. 1988; pp. 33 - 36.
5. Perry, R.: "LAN/MAN Internetworking in the 802.6/SMDS Environment," Proc. of INFOCOM '90; pp. 639 - 648.
6. Braun, T.; Zittervart, M.: "A Transputer Based OSI-Gateway for LAN-Interconnection Across ISDN," Proc. of 16th LCN; Minneapolis, MN.; Oct. 14-17, 1991; pp. 158 - 165.
7. Chen, K.; Ho, K.; Saksena, V.R.: "Analysis and Design of a Highly Reliable Transport Architecture for ISDN Frame-Relay Networks,"*IEEE Jour. on SAC*, Vol. 7; No. 8; Oct. 1989; pp 1231 - 1242.
8. Cooper; E.C.; Steenkiste, P.A.; Samson, R.D.; Zill, B.D.: "Protocol Implementation on the Nectar Communication Processor," Proc. of SIGCOM '90; pp. 135 - 144.
9. Traw, C.B.; Smith, J.M.: "A High-Performance Host Interface for ATM Networks," Proc. of SIGCOM '91; pp. 317 - 325.
10. Davie, B.S.: "A Host-Network Interface Architecture for ATM," Proc. of SIGCOM '91; pp. 307 - 315.
11. Clark, D.D.; Tennenhouse, D.L.: "Architectural Considerations for a New Generation of Protocols," Proc. of SIGCOM '90; pp. 200 - 208.
12 Kapoor, S.; Parulkar, G.M.: "Design of An ATM-FDDI Gateway," Proc. of SIGCOM '91; pp. 173 - 183.
13. Little, T.D.C.; Ghafoor, A.: "Network Considerations for Distributed Multimedia Object Composition and Communication," *IEEE Network Magazine* Nov. 90; pp. 32 - 49.
14. Ohnishi, H.; Okada, T.: "Flow Control Schemes and Delay/Loss Trade-offs in ATM Networks,"*IEEE Jour. on SAC*; Vol. 6; No. 9; Dec. 1988; pp. 1609 - 1616.
15. Harita, B.R.; Leslie; I.M.: "Dynamic Bandwidth Management of Primary Rate ISDN to Support ATM Access," Proc. of SIGCOM '89; pp. 197 - 210.
16. Foudriat, E.C.; Maly, K; Zhang, L. Sun, W.: "Gateway Resequencing in Multinode Networks," ODU Tech Report; To Be Published.
17. Burren, J.W.: "Flexible Aggregation of Bandwidth fro Primary Rate ISDN," Proc. of SIGCOM '89; pp. 191 - 196.
18. Clark, D. Jacobson, V. Romkey, J. Salwen, H.: "Analysis of TCP Processing Overhead,"*IEEE Communications* Vol. 27; No. 6; June 1989; pp. 23 - 29.
19. Gerla, M.; et. al.: "Interconnecting LANs and MANs to ATM," Proc. of 16th LCN; Minneapolis, MN.; Oct. 14-17, 1991; pp. 259 - 270.
20 Sato, K; Ohta, S; Tokizawa, I.: "Broad-Band ATM Network Architecture Based on Virtual Paths,"*IEEE Trans. on Comm*; Vol 38; No. 8; Aug. 1990; pp. 1212 - 1222.
21. Dighe, R.; May, C.J.; Ramamurthy, G.: "Congestion Avoidance Strategies in Broadband Packet Networks," Proc. of INFOCOM '91; pp. 295 - 303.
22. Fendick, K.W.; Mitra, D.; et. al.: "An Approach to High-Performance, High-Speed Data Networks," IEEE Communications Magazine; Vol. 29; No. 10; Oct 1991; pp. 74 - 82.
23. Shoraby, K.; Sidi, M.: " On the Performance of Bursty and Correlated Sources Subject to Leaky Bucket Rate-Based Access Control Schemes," Proc. of INFOCOM '91; pp. 426 - 434.
24. Haas, Z.; Winters, J.H.: "Congestion Control by Adaptive Admission," Proc of INFOCOM '91; pp. 560 - 569.
25. Foudriat, E.C.; Maly, K; et. al.: "A Dynamic, Lossless Feedback System to Solve Network Congestion and Error Recovery," ODU Tech Report; To Be Published.
26. Mankin, A.: "Random Drop Congestion Control," Proc. of SIGCOM '90; pp. 1 - 7.
27. Jacobson, V.: " Congestion Control and Avoidance,*ACM Computer Comm*; April 1990; pp. 553 - 573.
28. Goksel, A.K.; et. al.: "A Content Addressable Memory Management Unit with On-Chip Data Cache,*IEEE Jour. of Solid-State Circuits*, Vol. 24; No. 3; June 1989; pp. 592 - 596.

A Simple, Effective Media Access Protocol System for Integrated, High Data Rate Networks

by

E.C. Foudriat, K. Maly, C.M. Overstreet, S. Khanna, L. Zhang
Department of Computer Science
Old Dominion University
Norfolk, VA 23529
foudr_e@cs.odu.edu

Submitted to the Journal on Selected Areas in Communications

Abstract

The paper describes the operation and performance of a dual media access protocol for integrated, gigabit networks. Unlike other dual protocols, each protocol supports a different class of traffic. The Carrier Sensed Multiple Access - Ring Network (CSMA/RN) protocol and the Circulating Reservation Packet (CRP) protocol support asynchronous and synchronous traffic, respectively. The two protocols operate with minimal impact upon each other. Performance information presented in the paper demonstrate that the they support a complete range of integrated traffic loads, do not require call setup/termination or a special node for synchronous traffic control, and provide effective pre-use and recovery, The CRP also provides guaranteed access and fairness control for the asynchronous system. The paper demonstrates that the CSMA-CRP system fulfills many of the requirements for gigabit LAN-MAN networks most effectively and simply. To accomplish this, CSMA-CRP features are compared against similar ring and bus systems, such as Cambridge Fast Ring, Metaring, Cyclic Reservation Multiple Access and DQDB.

I. Introduction

The direction for networking has been to increase basic network data rates toward the gigabit range. For gigabit networks, the most of the recent development is concentrated on circuit-based, packet networks like ATM which are applicable generally to WAN and MAN systems [1]. Since circuits and switching add to the complexity of ATM based systems, there is a place, especially in the LAN - MAN area for fixed connectivity networks based upon a ring-and bus architecture. Here, no connection needs to be established and information can flow from any-to-any/many simply by access to the network and insuring uncorrupted data to the receiver(s).

Gigabit ring and bus network media access protocols have been described in recent literature. These include Metaring [2-4]; the Cyclic-Reservation Multiple-Reservation system [5-8], most recently CRMA-II; the Cambridge Ring [9, 10]; DQDB [11]; and the Carrier Sensed Multiple Access - Circulating Reservation Packet system, CSMA/CRP, [12, 13]. The CSMA - CRP protocol is a dual, non-interfering protocol which supports both asynchronous and synchronous traffic. It is the intent of this paper to describe the CSMA-CRP system and its performance and then to compare its features with the other systems above based upon their published results.

There are a number of requirements for high data rate network systems. Since data is arriving at the node at 1000 bits/msec., logic decisions related with network access must be high speed. Thus, a major requirement for high data rate media access is that the protocol be simple, i.e., it leads to simple logic circuitry. If complex data formulation and logic operations are required they should be to a large extend pre-computable, and the event times to which the operations relate be highly predictable. This permits these complex operations to be executed quickly based upon boolean indications of status.

A second major requirement is for these high data rates systems to handle integrated traffic efficiently. In the future, high data rate networks will be the backbone upon which distributed multimedia operations are supported. At times, much of the network load may be voice/video traffic while at others almost all of the traffic may be asynchronous packets [16]. Also, load can vary from full load to no load over short periods of time. Therefore, the protocol must support highly dynamic load and load type ranges without renegotiation of its operational parameters.

A third requirement is for these systems to support maximum capacity since even for simple protocols, equipment will be expensive. For example, dual counter rotat-

[1]Research support has been provided by DARPA, N00174-C-91-0119. NASA. Langley Research Center grant NAG-1-87263 and Virginia Center for Innovative Technology grant 596045.

ing ring configurations can, in some cases, provide as much as 8X the capacity with the same equipment count a dual bus configuration [2]. Also, within the scope of effective equipment utilization, the network structure and operations should be easily reconfigured in case of node or link failure. This feature provides added reliability.

In this paper, we describe a system, CSMA - CRP, with most adequately supports these requirements. The next section describes the system operation. This is followed by development of an analytical model and then a description of system performance based upon both analysis and simulation modelling. In the remaining sections, the operational and performance features of CSMA-CRP are compared to the four other systems listed above based upon descriptions and performance information found in the open literature. To accomplish this task each system's operation is described briefly. This is followed by a comparison of capabilities related to the features discussed above.

II. The Carrier Sensed Multiple Access - Circulating Reservation Packets Media (CSMA-CRP) Access Protocol

The CSMA-CRP system is a dual media access protocol for high data rate networks. Unlike other dual systems which support asynchronous traffic over different load ranges, the CSMA and CRP protocols support asynchronous and synchronous traffic, respectively. The following sections describe each protocol.

A. Asynchronous Protocol System

The CSMA/RN system has been described in reference [12]. Figure 1 shows the basic logic operation which occurs. The physical level media access logic interface can be "almost" all optical since it conforms closely to the logic operations in reference [17]. The logic must make the decision as to whether the incoming data or data queued at the node should be transmitted on the outgoing line. Queued data may be transmitted under the condition that incoming block is idle or that the incoming block is destined for this node, i.e., it uses destination removal (spatial reuse). At low data rates, the idle condition is equivalent to the fact that the total network is free, as for example in the CSMA/CD Ethernet protocol. At high data rates, like 1 Gbps, the access conditions are local to a node, which allows simultaneous access by multiple, geographically separated nodes. Most high data rate protocols [2, 5, 9, 12, 14] including some of those listed above use simultaneous access and destination removal.

Destination removal has demonstrated that it can significantly enhance peak performance for high data rate networks. In a single ring under uniform load conditions, the nominal capacity of a network is twice the basic data rate and for dual counter rotating rings capacity can increase to 8 times. The CSMA-CRP system is designed so that it can use this feature. It will be shown later in the

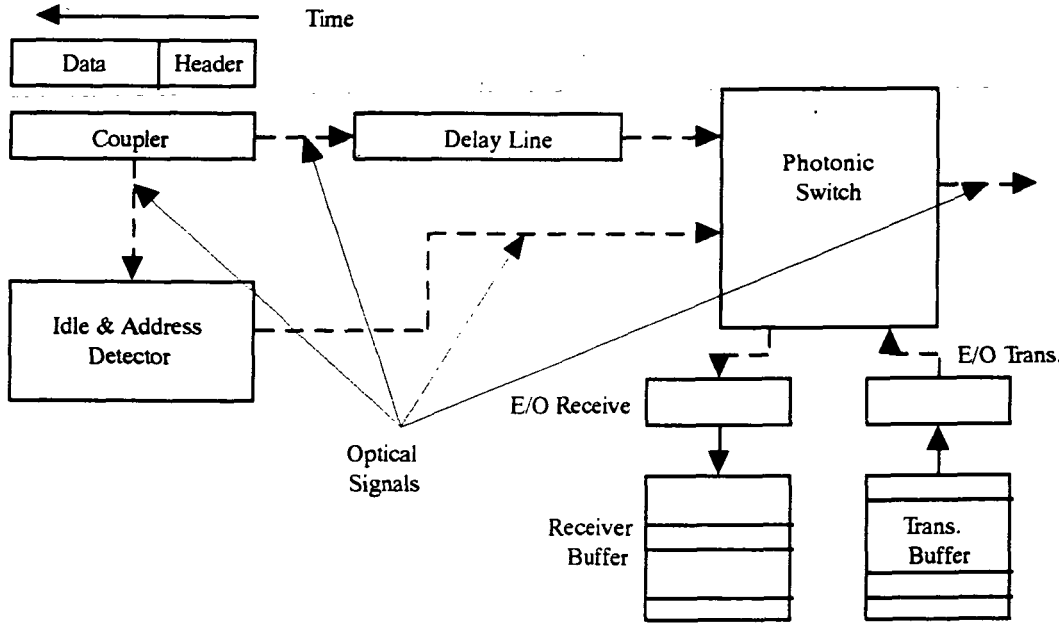


Figure 1 Node Inteface Logic for CSMA-CRP System

paper, that destination removal also influences fairness access performance at high load factors.

CSMA/RN can be viewed as a slotted system with a slot length of 1 bit. If a node is transmitting data during a period when the incoming traffic changes from idle to data, the conflict is alleviated by truncating the node's transmission of its data. Under this operation all packets on the network have preemptive rights. This decision logic requires that the delay line for incoming packets be long enough so that the truncation operation can take place, i.e., in the range of 100 bits. The delay line is not operated as an insertion buffer as in some access protocols but only so that the logic operations have sufficient time to complete. Hence, the delay serves only to increase the total ring length. The truncation operation does not require breaking messages into fixed slot lengths as in slotted systems, but at higher loads and for larger messages, fracture does occur.

Figure 2 shows the CSMA/RN packet structure. Information in the header provides for destination removal or source removal for multicast and broadcast messages. Control information is used to signal request for acknowledge, to indicate network control packets and to enable removal control for packets where address is corrupted. Trailer information assists a receiver in reassembly since it provides message initiation and termination information as well as providing for error detection using a check sum. A similar packet structure is used for the CRPs.

B. Synchronous Protocol System

Synchronous access uses the Circulating Reservation Packet (CRP) system. Its operation is described in reference [13]. It uses a special, small packet which circulates continuously around the ring. Synchronous traffic is attached behind this packet. On the cycle prior to a node's need to submit synchronous traffic, the node using the CRP, requests a block of space following the packet be freed upon its return. When the CRP returns, the node frees the packet and places the traffic in the free space behind the packet. Multiple CRP can be used on a net and they can be assigned in a round robin fashion or used in a free access mode. The separation of circulating reservation packets will limit the size of regular data packets as well as the size of synchronous message block a node can send at one time so the number and location of reservation packets on the system should be regulated. Since CRPs circulate without interference or delay, there occurrence at a node is strictly periodic and completely predictable.

As the CRP request circulates around the network, the space behind the CRP is freed as the packets arrive at their designated destination. *Pre-use*[13, i-] is available since the space being freed can be used by a node which has a message for another node between and up to the node that set the request. Recovery of synchronous traffic space occurs automatically since the node controlling the circulating packet does not request space to a greater extent than needed. For example, a telephone call which goes into a "non-talk" mode requires no or only a minimal reserved space during this phase. In addition, there is no need for a node having new synchronous traffic to establish a *call*, since when the node gets its CRP, it can request additional reserved space within the limit of the space available following the CRP.

In addition to handling synchronous traffic, CRPs support guaranteed access. Access is guaranteed one revolution after receipt of a free CRP. To guarantee that nodes have access to free packets requires that a node cannot use a packet on a number of successive revolutions and/or that nodes are assigned CRP use in a round robin fashion. Fairness control operates in a similar manner. Since CRP circulation is strictly periodic and CRP use for synchronous traffic highly regular, the node can easily predict when access will occur. Finally, CRPs can be used for guaranteed time to acknowledge. Since the node sending the acknowledge request can predict the arrival of the packet at the destination node, the time the next CRP arrives at the destination node and the arrival of the returned acknowledge packet, it knows how long it must store the data to replace a lost packet.

The CRP operate without interference or interfering with the CSMA/RN asynchronous access operations. To the CSMA/RN system, it is just another packet since all information on the ring has preemptive rights. When the CRP is set with a request, each node must monitor the block requested. With regular use, the node knows when a CRP is expected to arrive and the node using it. Hence, it can prepare its traffic to take advantage of *pre-use* effectively.

The CRP system causes minimal impact on asynchronous access. It should be noted because of this minimal

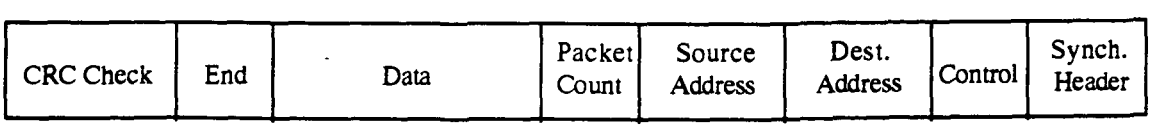


Figure 2 CSMA/RN Packet Structure

impact, the CRP protocol concept is applicable to other networks besides CSMA/RN. These include slotted rings and dual bus configurations if the end stations transfer CRPs from one unidirectional bus to the other.

III. CSMA - CRP Access Performance

A. Analytical Model

An analytical model for describing the CSMA/RN performance was developed in reference [12]. In that paper it was noted that a number of models including those involving contention over the total ring length were evaluated. Since the capability for node access is based upon local conditions, i.e., the condition of the immediate upstream arriving information, and since once placed, the message can not be preempted, a simple priority queuing model suffices.

The model developed has been extended to cover the conditions where load factors are greater than the service rate, i.e., the condition in reference [13, 15] where r > 1. The equation for arrival is:

$$Pr(x = \text{available} = p = \begin{cases} 1 - lf + lf/(n-1) & lf < 1 \\ 1/(n-1) & lf > 1 \end{cases} \quad (1)$$

where lf = load factor;
n = number of nodes; and

where an available packet is describe as one which is either empty or one whose destination is the node under consideration. Note that lf > 1 constitutes a network completely filled with packets.

The remainder of the analysis follows that of reference [12]. Figure 3 compares the calculated and simulator service times for the conditions in equation (1). Note that for lf > 1, the service time, the time it takes a message to be sent once reaches the head of the queue, approaches a constant. This occurs with uniform destination address and destination removal because each node receives 1/(n-1) of the packets on the network. Even though the service time is finite, the wait and response times for this condition are unstable, as is evident based upon the Pollaczek-Kintchine formula [15] and the equation for response time in reference [12].

B. A Question of Fairness

Under the conditions of multiple access and destination removal, if destination addresses are uniformly distributed then node starvation *does not* occur. Further, if each node generates messages in proportion to those it receives then the network is eminently fair, i.e., all nodes will have identical wait time conditions. These conditions exist even if the input message rate exceeds the network capacity. While this can be easily proved, a better demonstration was obtained from the simulator. A file server system condition was modelled. In the worst case, all other nodes (with identical arrival rates) send data to the file server and the file server send to all other nodes uniformly. The load conditions were increased to where the network data rates exceeded capacity. All nodes still had equal wait and service times.

This creates an interesting question with regard to high data rate network design using multiple access and destination removal. Node starvation can occur only when two conditions exist simultaneously. They are: (1) the network is heavily loaded, i.e., greater then 75% capacity and (2) one or more nodes are sent a significantly reduced message count. Since the simultaneous occurrence of this

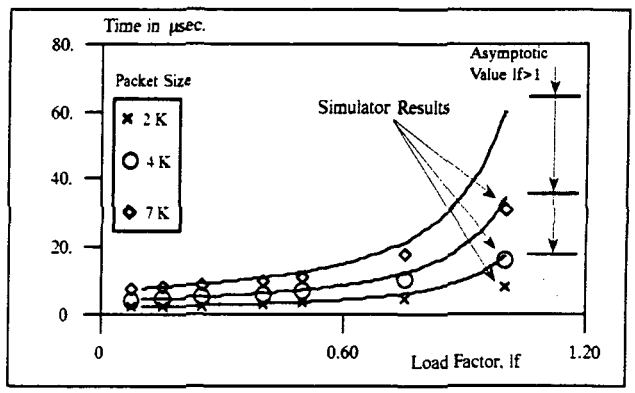


Figure 3 Service TimeComparison Between Analytical and Simulation Models
Conditions - Data Rate = 1 Gbps
Node Count = 10
Ring Length = 10 km

condition is probably rare, to what extent should high data rate networks go in order to provide access fairness? If, for example,the fairness system reduces capacity by 25%, it may be doing more harm than good. If the fairness system is complex, is it worth the cost since it is rarely needed?

C. CSMA/RN Performance

The simulation determined CSMA/RN performance under a range of system parameters [12]. Three conditions of major interest are message length, ring length and node count. Runs were made to examine the effects of these parameters. Figures 4a - 4d present data for a 1 Gbps, 10 km, 10 node ring for message lengths ranging from 2K to

20 Kbits. Note that load fraction is shown as a function of basic network bit rate and that because of destination removal the capacity of the network is actually 2 Gbps. We see from Figures 4 that average performance characteristics for CSMA/RN are not detrimentally altered by message length. Both wait and service time are similar to that predicted by the analysis. Mean response time is greater for the larger packets, primarily because service time is greater. Finally average message fracture ratio does not change significantly as packet length increases, indicating that message fracturing does not materially increase, at least, when all messages are of the same size and nodes are uniformly distributed around the ring.

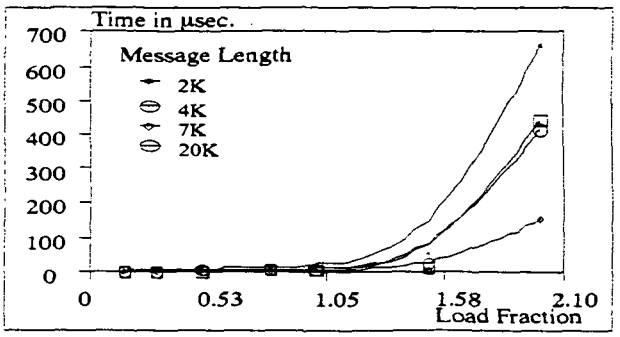
A similar set of curves is plotted in Figure 5 to show the effect of ring lengths from 2 km to 1000 km. Only response time is shown because the other performance measures are only minimally affected. It shows significant dependence upon ring length, mainly due to the travel time necessary from source to destination. For longer length rings, travel time dominates, so service and wait time become less significant. However, the latter two are the only controllable factors in the response performance. The CSMA/RN protocol provides excellent operations over a range of LAN and MAN conditions. Additional information provided in reference [12] demonstrates that the protocol is applicable to WANs as well.

Figures 6a - 6b show the simulation results when node count is varied from 10 to 200 nodes for a 50 km ring; node spacing range from 0.25 km to 5 km. Service time and packet fracture ratio are shown because at the higher node count they are the factors most affected. At the larger node count and high load factor, both service time and message fracture show a definite increase. Under these conditions, the CSMA/RN protocol would have it worst operational problems as the packets on the ring would have the greatest tendency to fracture. However, performance is completely acceptable for loads up to 150%.

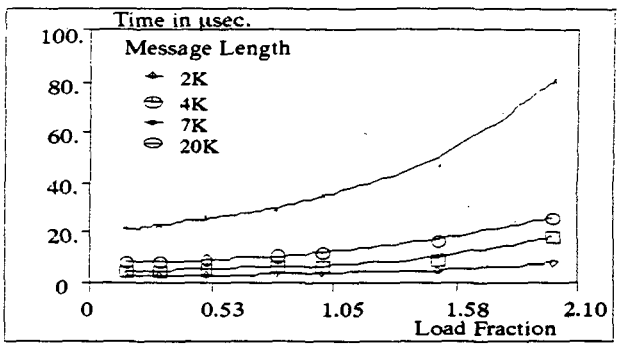
Reference [12] also included performance information for data rates from 100 Mbps to 1 Gbps. At the lowest speed, short length networks, for example, 2K, tend to suffer because multiple simultaneous access no longer exists. At data rates above 1 Gbps CSMA/RN performance only gets better. Also, it shows that CSMA/RN can handle overloads without significant capacity decrease, and recovery, after load is reduced, depends upon queue buildup during overload.

CSMA/RN provides a number of excellent performance features for supporting asynchronous network traffic including:

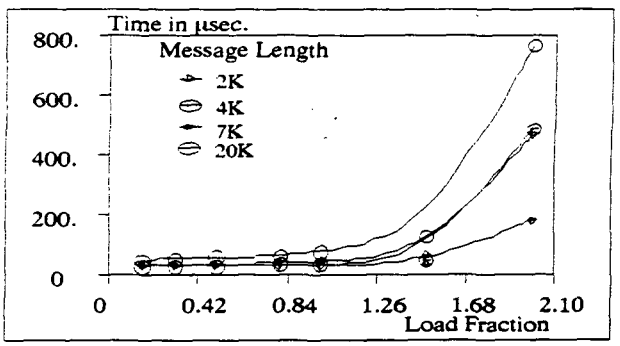
1. immediate access for an idle network since there is no waiting for a token or slot;
2. no automatic message breakup due to slotting;



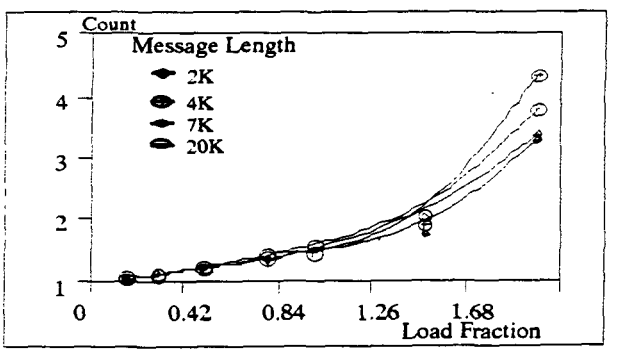
a) Wait Time



b) Service Time



c) Response Time



d) Message Fracture

Figure 4. CSMA/RN Performance for Various Message Length
Conditions 10 km, 10 nodes
1 Gbps

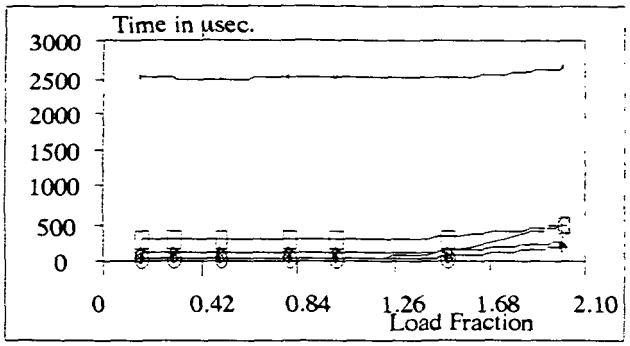
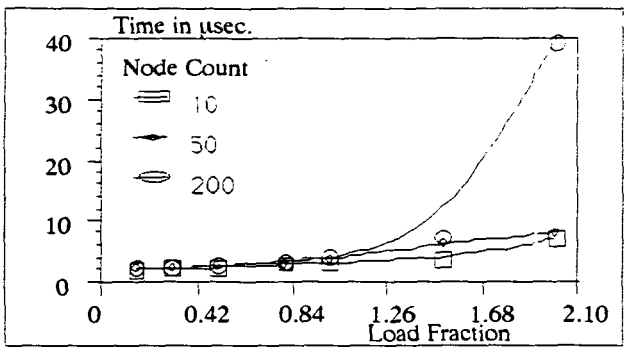
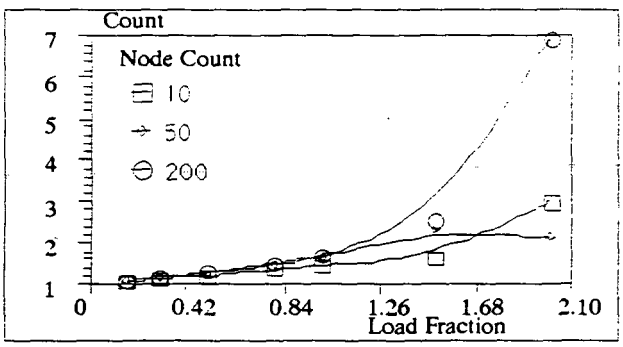


Figure 5 Response Time For Ring Lengths of 2, 10, 50, 200 & 1000 km



a) Service Time



b) Fracture Ratio

Figure 6 Network Performance for Node Counts
Conditions - Data Rate = 1 Gbps
Ring Length = 50 km
Packet Size = 2 Kbits

3. no wasted capacity at high loads due to collisions which abort transmissions;
4. applicable to a wide range of network conditions including:

lengths	10 - 5000 km;
nodes	10 - 1000;
data rates	>100 Mbps; and

5. simple node operations since nodes react to local traffic conditions only.

D. CSMA/RN Performance with CRP Supporting Integrated Traffic

1. CRP Implementation

CRP performance in its own right, is fully deterministic, since CRP travel is completely predictable. Here, synchronous traffic is served strictly periodically and both the CRP and its attached traffic are not affected in any way by the CSMA/RN operations. Any study of the combined protocols need only determine the effects of the CRP system and its synchronous traffic on CSMA/RN operations.

In the simulator, the CRP system and synchronous traffic conditions were established at initialization time. CRPs (one or more depending upon run conditions) were assigned in a simple round robin fashion. They were spaced at equal distances around the network and nodes were assigned a single CRP. Assignments were interlaced, e.g., for 3 CRPs, CRP #1 was assigned nodes 1, 4, •••; CRP #2 nodes 2, 5, •••, etc. No complex assignments were used to mitigate particular load assignments or alleviate starvation and no sharing of CRPs between adjacent nodes was implemented. When the CRP arrives at the node for which it is scheduled, synchronous traffic is checked, and the total space needed is calculated and loaded in to the CRP data structure. When the CRP arrives back at the starting node, the synchronous traffic is placed using a last-out, first-in structure, i.e., messages traveling farthest are placed first in the free space. This format is used because as messages are removed, the free space available may be contiguous and better suited for CSMA/RN operations.

Interference effects were studied for fixed synchronous traffic for each run. Except for fairness runs, source and destination for both synchronous and asynchronous traffic was assigned uniformly. Asynchronous traffic arrival was Poisson. For synchronous traffic, two message types, standard TV and digital telephone, were used; the former requiring a bandwidth per call of 10 Mbps and the latter 64 Kbps. TV blocks were sent at a minimum interval of 1 msec. (>10,000 bits per access); telephone at a 10 msec. (>64 bits per access). All bits accumulated since the last CRP arrival were sent. Synchronous traffic mix could be varied between all TV or all telephone, but for most runs TV and telephone traffic load was 90 % and 10%, respectively.

After fixing synchronous traffic load, asynchronous traffic was varied over the remaining range. Run combinations used fixed synchronous traffic load while asynchronous traffic varied over the remaining range. For example, if synchronous traffic was 25%, i.e., 250 Mbps, then

Table I. Total Load Bit Rate

in BPS x 10 -6

Async. Load Percent	Synchronous Load Percent				
	0%	10%	25%	50%	75%
0%	0	100	250	500	750
50%	500	550	625	750	875
100%	1000	1000	1000	1000	1000
150%	1500	1450	1375	1250	1125

asynchronous traffic loads could range from 0 to 1500 Mbps for a network with 1 Gbps data rate. Under these conditions and asynchronous load of 750 Mbps in conjunction with the 250 Mbps synchronous traffic is considered a 100% load. Table I shows bit loads and splits under various load conditions.

2. CSMA Performance Under CRP Operational Conditions

Table II illustrates the CRP access time capabilities for some typical LAN - MAN network configurations. By selecting the number of CRP based upon the ring length and the number of nodes, a maximum access time of 1 msec is feasible in cases where the nominal ring travel time is somewhat less than 1 msec. For WAN length rings, guaranteed access after arrival of the CRP below 1 msec. can not be obtained. However, with multiple CRPs, their arrival can be made to occurs at less than 1 msec. intervals.

Table II. Interarrival Time for Multiple Use CRPs

Time in sec.

CRP Count	Node/Lng(km)				
	4/10	10/10	20/50	40/100	50/200
1	2.50E-04	5.50E-04	5.25E-03	2.05E-02	5.10E-02
2	1.50E-04	3.00E-04	2.75E-03	1.05E-02	2.60E-02
5	N/A	1.50E-04	1.25E-03	4.50E-03	1.10E-02
10	N/A	5.00E-05	7.50E-04	2.50E-03	6.00E-03
20	N/A	N/A	2.50E-04	1.50E-03	3.50E-03
50	N/A	N/A	N/A	N/A	1.00E-03

Here, nodes would have to use substitution or reserve extra space apriori if critical message time constraints exist.

Figure 7a - 7d show the wait time, service time, response time and fracture ratio for a LAN configuration of 10 km - 10 nodes using 2 CRPs. It can be concluded that CRPs, by themselves, have very little affect on CSMA/RN operations with the exception that the fracture ratio at low asynchronous loads is raised from 1.1 to 1.4. Here, the curves to compare are those with No CRPs and 2 CRPs with 0% synchronous traffic conditions.

Examination shows that asynchronous and synchronous traffic are very similar in their affect on CSMA/RN operations. As synchronous load increases from 0 to 75%, both wait and service time for low asynchronous traffic loads are increased, from 0 to 6 msec. and from 7 to 16.5 msec., respectively. This increase is nearly identical to that which occurs when asynchronous traffic load increases from 0% to 75% with no synchronous load.

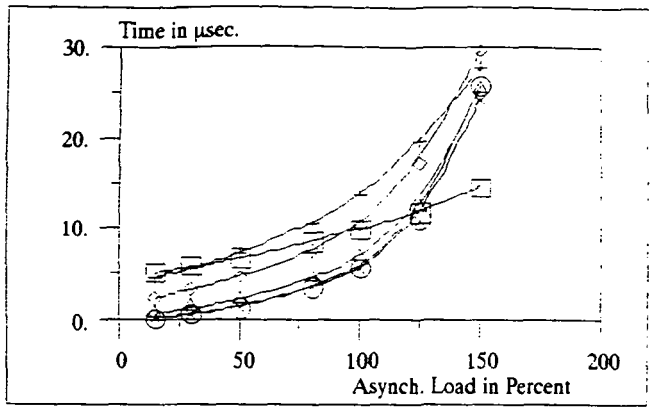
Reference [13] discusses the affect of CRP count. Larger CRP counts actually improve network operations especially at high traffic loads. The reasons are that the network load is more evenly distributed since the CRPs tend to be in different stages at any particular time and that no extremely large reserved block of space is being requested which can shut off many nodes simultaneously for a long period of time. Here with minimal CRP count, a node receives the CRP at longer intervals, as shown in Table II, so it may request an extremely large block of space, effectively shutting down other nodes for a long period as the space is first cleared, then filled and finally cleared during CRP use cycle.

Figures 8 shows the adaptability of CRPs to MAN network conditions for a ring of 200 km and 40 nodes. Note that for low loads, minimal response time. At higher synchronous and asynchronous loads on the MAN, travel time still dominates, while in a LAN, the response time, as illustrated in Figures 7, is more dependent on protocol access changes.

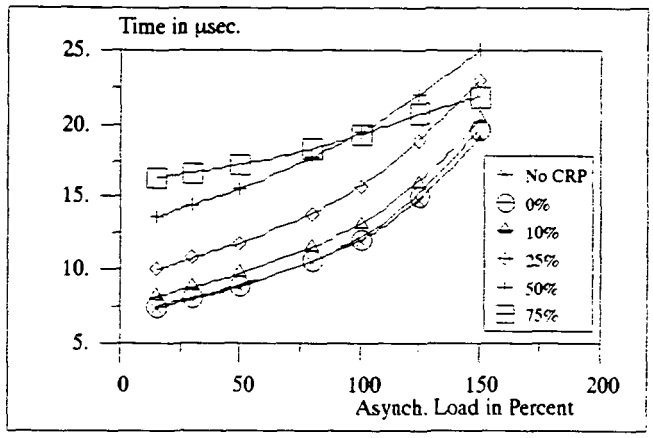
It can be concluded that CRPs, do indeed, operate with minimal impact on the underlying asynchronous CSMA/RN access protocol and that the combined protocols provide an extremely effective system for supporting integrated traffic over a wide range of network conditions. The results illustrate that CRPs are effective over a wide range of network conditions and configurations.

3. Pre-Use Effectiveness

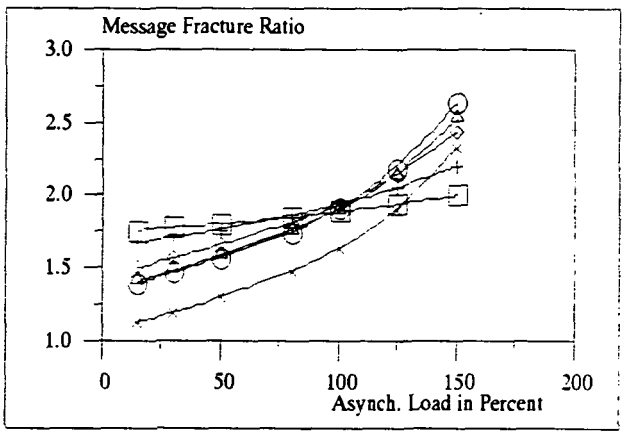
Pre-use is the capability for a node to use requested reserved space when it has traffic destined for nodes between itself and the reserving node. Pre-use was used for the runs presented in the previous figures. To illustrate



a) Wait Time



b) Service Time

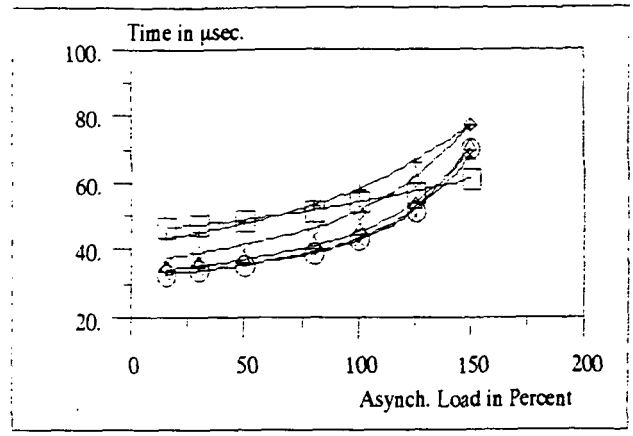


c) Message Fracture

Figure 7. Asynch. Message Performance for Dual Protocol System

Conditions: Ring Length - 10 Km; Node Count - 10; CRP Count - 2; Data Rate - 1 Gbps; Message Lngth - 7 K bits.

Note: Synchronous Load Key in Figure 7b).



d) Response Time

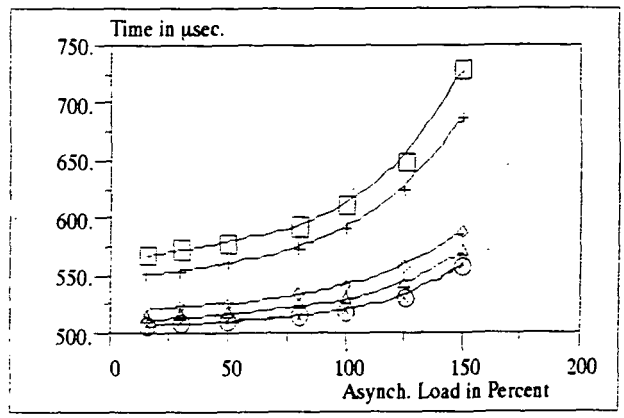


Figure 8. Dual Protocol Response Time for MAN Conditions

Conditions: Ring Length - 200 Km; Node Count - 40; CRP Count - 8; Data Rate - 1 Gbps.

Note: Synchronous Load Key in Figure 7b)

its effectiveness, runs were made for the LAN with a 50% synchronous load condition both with and without pre-use. The comparison, Figure 9, shows that pre-use significantly improves asynchronous access for operations especially under high load conditions. For larger synchronous loads conditions, pre-use is even more effective.

4. Fairness Using CRPs

As noted previously, CRPs can be used to support fairness and guaranteed access as well as synchronous traffic. Since unfairness situations in multiple access, destination removal, ring networks are undoubtedly complex, our approach in this paper is restricted to demonstrating the CRP's effectiveness, not developing a comprehensive fairness system. With this objective, the simulator

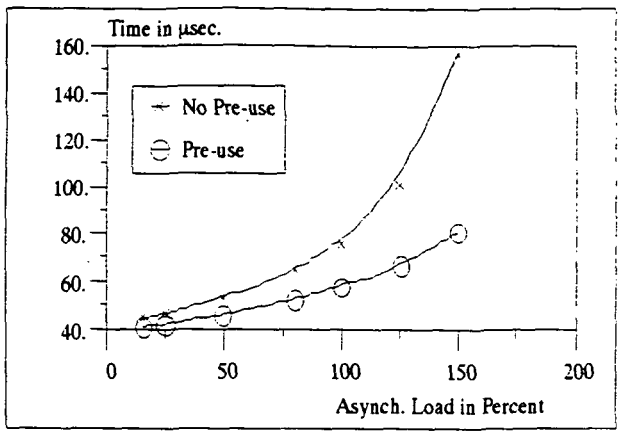


Figure 9. Wait Time Showing Pre-Use Effectiveness
Conditions: Ring Length - 50 Km; Node Count - 20; Data Rate - 1 Gbps; Synch Load - 50%; CRP Count - 5.

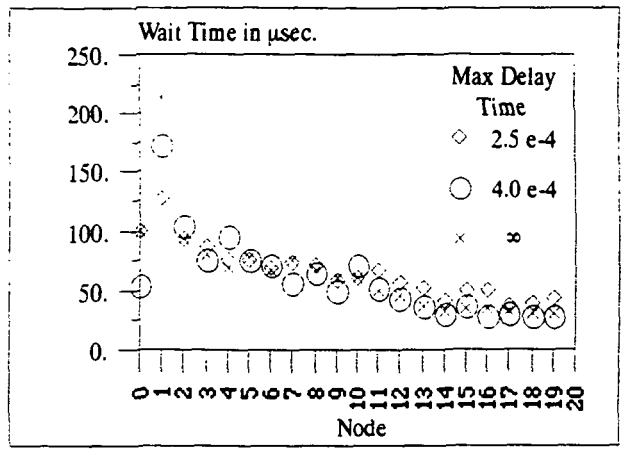


Figure 10. Mean Wait Time Reduction Using CRPs for Fairness Control
Conditions: Ring Length - 50 Km; Node Count - 20; Data Rate - 1 Gbps; CRP Count - 4 (fairness) - 4 (Synch Load); Synch. Load - 10%; Asynch. Load - 175%.

message generation system was modified to incorporate both non-uniform message arrival load and to effect node starvation through non-uniform destination selection. The first condition represents network loads typical of file server systems. As noted previously, no unfairness was observed. The second can create unfairness conditions because the model assumes uniform message arrival addresses but destination addressing reduces the reception of messages at designated nodes. For example, a starving node would receive only x% of the messages which it would normally receive under uniform loading. Messages not received by starving nodes were distributed to non-starving nodes uniformly. Starving nodes could be selected at any location, in any combination and at any percentage of starvation. However, for the runs used to show starvation and the CRP's effectiveness for fairness control, two adjacent nodes were selected to starve. Runs were made under varying load conditions and it was found that starvation was most severe for asynchronous loads above 150% when synchronous traffic load was small.

A simple fairness control scheme was used. Fairness CRPs where added to the synchronous traffic CRPs. They were used only to provide fairness access and were not used for synchronous traffic. Alternatively, synchronous traffic CRPs could not be used to alleviate starvation although in a realistic situation this is easily implemented. If a message waits for greater than Y time, the node uses the next free fairness CRP to request space to service this message. Since it takes the CRP one rotation before it can be used, there is the possibility that the delayed message will be serviced by the time the CRP returns with free space. When the CRP returned, the present message at the head of the queue is tested to see if its wait time interval is greater than Y, regardless of whether this is the message in the queue at the time that the CRP request was made. If so, this message is sent in the CRP space available.

Figure 10 shows node fairness control results for nodes 0 and 1 starving at 25% of their expected traffic. Node starvation affects not only the targeted nodes (in this case nodes 0 and 1), but also nodes downstream. Hence, the fairness control CRPs may be used by nodes other than those actually starving. Figure 10 shows that the fairness system using a delay time of Y = 250 μsec. significantly reduces the mean wait time for node 1 which is starving badly while the mean wait times for the non-starving nodes is not altered significantly. Runs at other network conditions and loads showed similar results using fairness CRPs.

IV. Comparison of CSMA-CRP with Similar Media Access Protocols

In this section we will briefly describe alternative media access protocols designed to operate in a manner similar to the CSMA-CRP system. After this description, we will compare the features of each.

A. Cambridge Fast Ring (CFR)

As originally conceived, the Cambridge Fast Ring (CFR) [9, 10] is a slotted ring with source removal. Access logic for asynchronous traffic uses a free/busy bit but since source removal is required logic decisions must be made on address also. Others have redesigned the ring to use destination removal [10]. Performance of asynchronous

slotted rings is well known [10].

Synchronous access is by one of two mechanisms - slot reservation or a cycle time scheme [10]. Slots in the former system are either normal or channel; channel slots are reused by the source but normal slots must be passed on. For the reservation system, nodes must capture sufficient channel slots for their synchronous traffic. In some configurations, slot rotation may not fit efficiently with synchronous traffic rates, so slots may not be efficiently used. Pre-use is possible but it is impossible to recover unused slots for messages in "silent" phase.

Cycle time, similar to an implicit token rotation time, establishes a time which is sufficient for nodes to send all their synchronous traffic plus some asynchronous traffic. Nodes cannot send newly arrived asynchronous traffic until a new round has started. Using cycle time leads to some difficult choices with regard to setting parameters. Poorly set cycle time parameters can lead to wasted capacity and changes in load may require renegotiation of parameters [10].

B. Metaring

The Metaring is being developed by IBM [2 - 4]. The ring is described is a dual, counter rotating, slotted ring with insertion buffers and destination removal. Nodes use the insertion buffers to avoid conflict between incoming and data being transmitted by the node.

Asynchronous traffic is controlled by a fairness algorithm using a SATisfied control packet. The SAT packet circulates in the opposite direction. Nodes between SAT arrivals can send up to a fixed amount of traffic. If a node, when it gets a SAT packet is not satisfied, it holds the packet until it is. SAT packets, which use uncoded data words, have preemptive priority so that a SAT packet must be serviced even within the middle of data packet operations. Results [2, 4] show that the SAT system can reduce peak capacity by about 12.5% but that significant fairness can be achieved for nodes which are starving. However, as noted previously true starvation can be a rare occurrence. While multiple SATs are mentioned it is not clear that multiple SATs can be used effectively in longer distance rings, without an additional mechanism for regulating SAT spacing.

Synchronous traffic access is supported by an ASYNC-EN packet with rotates in the opposite direction to the SAT packet. The ASYNC-EN system is equivalent to the timed token function in FDDI. The reservation protocol accompanying the system requires that a node send a request to all intervening nodes between source and destination indicating how many synchronous traffic data units it needs. If all nodes have less than a maximum synch traffic then the each intervening node adds that traffic to its reserve. Like the cycle time system above, synchronous traffic is transmitted before asynchronous traffic after receiving the ASYNC-EN packet. Normally, this packet circulate around the ring in at a fixed rate. Holding the ASYNC-EN packet, which is done by a node when its synchronous traffic queue builds up, signals nodes on the ring to stop sending their asynchronous traffic. No information is given showing either synchronous or integrated traffic performance in the open literature. To what extent delaying the ASYNC-EN packet throttles asynchronous packets is not clear but it would appear holding the token will reduce asynchronous access for at least that ring cycle. It is not clear that either SAT nor SYNC-EN operations which support Metaring are amenable to larger ring distance.

C. Cyclic Reservation Multiple Access-II

CRMA-II is also an IBM development [5 - 8]. It is a reservation based protocol where access to slots is either restricted by a scheduler or free gratis if the slot is not reserved and is free. The concept is applicable to either a ring or bus architecture but we will only consider its operation in a ring environment because, in a bus configuration, it is similar to DQDB.

Nodes may access gratis slots. These may occur either by the scheduler issuing free slots between reservation cycles or by destination removal. If a node has packets backed up when a reservation request arrives, it request slots, noting how many gratis slots it has had since the last reservation cycle and how many it needs. It gets a count of slots reserved by the scheduler on arrival of the confirmation packet and uses the subsequent reserved slots as they arrive. If it has been able to use gratis slots in the interim, it marks the reserved slots as free.

One node acts as scheduler. It issues a reservation packet which travels the network. Nodes, as noted above, place their request immediately behind the reservation packet. This requires a limited form of insertion buffer at each node. Upon return of the reservation packet, the scheduler calculates the number of reserved packets it will supply each node and sends this information to each node in a confirmation packet. It then marks sufficient slots to handle all confirmed requests. Note that a reservation access takes one round trip cycle plus the scheduler calculation time plus the slot times of all prior node reservations. Determination of slot status is by bits but since destination removal is used logic based upon address recognition is required.

No asynchronous message performance data is available for CRMA-II or for previous versions of CRMA. As noted previously, equally distributed send and receive load

conditions with destination removal is inherently fair. It is questionable whether the reservation scheme maintains this uniformity since now a large number of reserved slots exist for nodes immediately down stream of the scheduler and a large number of gratis slot immediately up stream. Thus, the cyclic reservation scheme could actually reduce network capacity significantly while enforcing its fairness scheme at high loads. Even Metaring's SAT system, which does not change the symmetrical operational characteristics of the ring had a tendency to reduce capacity although to a limited degree.

No discussion of synchronous traffic access is provided although reference [5] implies that buffer insertion bypass mechanisms are available to support isochronous traffic. With the schedular it would appear that a framing technique similar to FDDI-II [18] and DQDB [14] solutions would be applicable. Here, call setup is an additional task for the scheduler and each node must to keep track of slots reserved for its synchronous use. The framing operation requires frames at fixed periods so that scheduler reserve and confirm operations would have to be integrated it the framing cycle.

D. DQDB

DQDB is being promoted by AT&T [11, 14, 19, 20, 21]. In a unidirectional bus configuration each node must provide two complete sets of hardware and two scheduler nodes are required at each end. With free slot bit access only, hardware is simpler than if destination removal is used but the advantages of additional capacity are not available. In addition, bus configurations are less amenable to multicast and broadcast since now two messages must be sent. Bus configurations also require two scheduler nodes at the bus ends which can be replaced only by immediately adjacent nodes if failure should occur.

Considerable information on DQDB asynchronous performance is available [11, 14, 19-21]. Unfairness is the major issue but to a large extent this has been solved [21, 22]. However, fairness protocol additions tend to increase access time to some extent.

Synchronous traffic is supported in DQDB [14] by a framing system as noted above in the CRMA-II discussion. Call control by the each scheduler and knowledge of nodes reserved for synchronous traffic must be kept by each node. In a WAN version of DQDB [14], both destination removal and pre-use were employed but this adds to the complexity of each access port. Destination removal requires address recognition and for preuse, each node must keep track of the source for each reserved slot. In addition, using framing for synchronous traffic access makes recovery and support for dynamic traffic variation difficult.

E. Comparison of the Media Access Protocols

A comparison of system features is provided in Table 3. The categories include equipment complexity and flexibility, operational and performance for under both asynchronous and synchronous traffic loads, reliability considerations and the expandability to MAN and WAN conditions.

The five systems discussed above are compared. Actually, two Cambridge Fast Ring systems, one based upon slot access and one based upon destination removal, are included.

It can be seen from Table 3 that the CFR systems are the simplest, but they do not support the fairness and guaranteed access requirements for high data rate networks. In addition, the support for synchronous traffic is not effective. Metaring provides excellent asynchronous performance. However, it requires twice as much equipment at each node; synchronous traffic operations are complex with call setup and termination at each node and integrated traffic performance is questionable; and it is not certain whether Metaring can be extended to longer networks. Also, it can not be reconfigured in case of node or link failure. The Cyclic Reservation Multiple Access-II system is most complex. The addition of the scheduler causes this complexity and since the system is no longer symmetrical, it is not capable of supporting increased capacity by using dual counter rotating rings like CFR, Metaring and CSMA-CRP. Also, it is questionable where the reservation capability provided will pay off in fairness performance at high loads. Integrated traffic performance is unknown and since dual ring operation is not supported, reconfiguration is not available. DQDB, like Metaring requires twice as much equipment, but unlike Metaring is not able to use this equipment effectively to increase capacity. Complexity must be added to DQDB to enable fairness through additional reservation control, since, as a bus based system, it does not contain the inherent fairness available in ring based systems. Further complexity is required to support synchronous traffic. Finally, DQDB can not use the additional equipment to enhance reliability.

The CSMA-CRP supports all of the requirements effectively. It can be operated as either a single or dual ring with 2X and 8X capacity, respectively. Complexity is added because of message truncation instead of slots. In most cases packet fracture is not large, and resequencing help is provided in the packet header. However, it is the CRP portion of the dual protocol which really enhances the system. It not only provides an extremely effective mechanism for synchronous traffic with properties such as strictly periodic access with both pre-use and recovery, but also supports fairness for the asynchronous system. It has

Table 3. Operational and Performance Comparison Between High Data Rate Access Protocols.

Rating
+ - good
++ - better
+++ - superior

Network	Equipment Configuration & Flexibility.	Asynchronous Traffic Access Complexity.	Synchronous Traffic Access Complexity	Async. Performance & Fairness	Synch. Performance	Reliability (use secor ring to reconfigure)	Expandable to MAN-WAN
Cambridge Fast Ring (simple)	Connectivity - 1 T/R per node Capacity - 1X data rate. Dual rings - 2 T/R per node. Capacity - 2X data rate	Slot bit access. Message partitioned into slots. On ring preemption. No fairness control. +	Slot Reservation System. +	Access Time - -> 1/2 slot time. Guaranteed Access - none. Fairness - none. Single message multicast. +	Slot Reservation System. Strictly periodic access. No pre-use or recovery. Start - Capture slots. Reduced capacity unknown. Term. - none. ++	Reconfiguration - yes. No special node. ++	Yes with no changes. ++
Cambridge Fast Ring (Destination Removal)	Connectivity - 1 T/R per node. Capacity - 2X data rate. Dual rings - 2 T/R per node. Capacity - 8X data rate +++	Spatial reuse complexity. Message partitioned into slots. On ring preemption. No fairness control. +	Cycle Time. Timers. Renegotiation for parameter setting. +	Access Time - -> 1/2 slot time. Guaranteed Access - none. Fairness - none Single message multicast. +	Cycle Time. Aperiodic access Capacity reduction due to unused slots at end of cycle. Renegotiation time - unknown. ++	Reconfiguration - yes. No special node. ++	Yes with no changes. ++
Metaring	Connectivity - 2 T/R per node. Capacity - ~8X data rate. ++	Spatial reuse complexity. Message partitioned into slots. Register insertion system. Preemption for control pkts. Timer & counter controls. +	Call setup & term. at all nodes. Renegotiation for ASYNC- EN parameter setting. Timers. Preempt operation for ASYNC-EN. +	Access Time - -> 1/2 slot time. Guaranteed Access - Unknown. Fairness - Excellent, ~ 10% capacity reduction. Single message multicast. ++	Aperiodic due to ASYNC-EN rotation time. Start - call set up req. time Term - require time. Capacity reduction - unknown. +	Reconfiguration - no. No special node. ++	Unknown - Fairness and sync. control may not be applicable.
CRMA-II	Connectivity - 1 T/R per node. Capacity - ~1X data rate.	Spatial reuse complexity. Message partitioned into slots. Register insertion system. Reserve/confirm pkt controls. Scheduler complexity. Handling reserve/confirm system with calculations.	Unknown. Same as DQDB?	Access Time - -> 1/2 slot time. Guaranteed Access - Cycle time+sched. calc.+ reserv. slot count. Fairness -Unknown.	Unknown. Same as DQDB?	Reconfiguration - no. Special node.	Undocumented in ring configuration.
CSMA-CRP	Connectivity - 1 T/R per node. Capacity - 2X data rate. Dual ring - 2 T/R per node. Capacity - 8X data rate. +++	Spatial reuse complexity. Message truncation circuitry. On ring preemption. ++	CRP control and access. Control based upon CRP length request. +++	Access Time - -> bit rate. Guaranteed Access - Next CRP + cycle time. Fairness - Excellent, 1% capacity reduction. Single message multicast. +++	Strictly periodic. No interference other than reduced capacity to async. pkts. Pre-use & recovery available. +++	Reconfiguration - yes. No special node. ++	Yes with no changes. ++
DQDB	Connectivity - 2 T/R per node. Capacity - 2X data rate.	Slot bit access. Message partitioned into slots. On bus preemption. Two special end nodes. Counters for reserve and access control. Additional fairness control. +	Call setup & term. at end nodes. Reserved slot access at each node.	Access Time - -> 1/2 slot time. Guaranteed Access - Unknown. Fairness - Excellent, capacity reduction depends upon method used. Two message multicast. ++	Strictly periodic. Start - call set up req. time Term - require time. Capacity reduction - unknown. Pre-use Recovery - none. +	Reconfiguration - no. Two special nodes must be at ends.	Yes with no changes. ++

minimal impact on the asynchronous traffic access. In doing so, it still provides enhanced traffic capacity with addition of a second ring and support added reliability through reconfiguration. Further, it is shown to be effective for LAN through WAN configurations.

V. Conclusions

It is shown that ring networks with destination removal require less equipment per node and can have significant capacity advantages and fairness advantages over bus architectures with identical basic data rates. However, to maintain these advantages for rings, it may be necessary that "operational symmetry" exist, i.e., there is not a special node that does scheduling or slot marking in such a manner that all nodes do not experience identical accessibility in a statistical sense.

Three of the ring networks described in this paper provide "symmetry" - CFR, Metaring and CSMA-CRP. Of these CFR is simplest but it does not provide good integrated traffic access and does not provide any mechanism for fairness or guaranteed access. Both Metaring and CSMA-CRP provide fairness. However, CSMA-CRP configuration and operation is simpler than Metaring and its performance capability for integrated traffic over a wide range of network parameters and conditions is demonstrated. Of the systems compared, CSMA-CRP supports integrated high data rate network operations in the most simple and effective manner.

VI. References

1. Minzer, S.E.: "Broadband ISDN and Asynchronous Transfer Mode (ATM)," *IEEE Commun. Magazine*; Sept 1989; pp. 16-25.
2. Cidon, Y.; Ofek, Y.: "Metaring - A Full-Duplex Ring with Fairness and Spatial Reuse," IEEE INFOCOM'90; San Francisco, CA.; June 1990; pp. 969-981.
3. Simha, R.; Ofek, Y.: "A Starvation-free Access Protocol for a Full-duplex Buffer Insertion Ring Local Area Network," IEEE Paper CH2799-5/90/0000/0531. May 1990.
4. Chen, J.; Ahmadi, H.; Ofek, Y.: "Performance Study of the MetaRing with Gb/s Links," Proc. of 16th LCN; Minn. MN.; Oct. 1991; pp. 137-147.
5. Van As, H.R.; Lemppenau, W.W.; Pitro; Z., Zurfluh, E.A.: "CRMA-II: A Gbit/s MAC Protocol for Ring and Bus Networks with Immediate Access Capability," EFOC/LAN '91; London, UK; June 1991.
6. Muller, H.R.; et. al.: "DQMA and CRMA: New Access Schemes for Gbit/s LANs and MANs," Proc of IEEE INFOCOM'90; San Francisco, CA.; June 1990; pp. 185-190.
7. Nassehi, M.M.: "CRMA: An Access Scheme for High-Speed LANs and MANs," Supercomm/ICC'90; Atlanta, GA; April 1990.
8. Heinzmann, P. Muller, H. Wilson, K.: "Configuration Control for Bus Networks," Proc. of 15th LCN; Minn. MN.; Oct. 1990.
9. Greaves, D.J.; Lioupis, D.; Hopper, A.: "The Cambridge Backbone Ring," Proc. of INFOCOM '90; pp 8-14.
10. Zafirovic-Vukotic, M; Niemegeers, I.G.; Valk, D.S.: "Performance Analysis of Slotted Ring Protocols in HSLAN's," Jour. on Selected Areas in Communications; Vol 6; No 6; July 1988; pp 1011-1023.
11. Newman, R.M.; Budrikis, Z.L.; Hullett; J.L.: "The QPSX Man," IEEE Communications Magazine; Vol 26, No 4; April 1988; pp 20-28.
12. Foudriat, E.C.; Maly, K.; Overstreet, C.M.; Khanna, S.; Paterra, F. : "A Carrier Sensed Multiple Access Protocol for High Data Rate Ring Networks," Comp. Comm. Review; Vol. 21; No. 2; April 1991; pp. 59-70.
13. Foudriat, E.C.; Maly, K. Overstreet, C.M.; Khanna; S.; Zhang, L.; Sun, W.: "Combining Two Media Access Protocols to Support Integrated Traffic on High Data Rate Networks," Proc. of 16th LCN; Minn. MN.; Oct. 1991.
14. Edmond, W.; Seo, K.; Leib, M.; Topolcic, C.: "The DARPA Wideband Network Dual Bus Protocol," Proc. of SIGCOMM'90; pp. 79-89.
15. Jaiswal, N.K.: Priority Queues; Academic Press; NY; 1968.
16. Little, T.D.C.; Ghafoor, A.: "Network Considerations for Distributed Multimedia Object Composition and Communications," IEEE Network Magazine; Nov. 1990; pp. 32-49.
17. Haas, Z.: "Optical Distribution Channel: An "Almost-all" Optical LAN based on the Field-coding Technique," Proc. of 16th LCN; Minn. MN.; Oct. 1991, pp. 128-136.
18. Boston, T.: "FDDI-II: An Overview of Call Setup for Isochronous Traffic," British Telecom - TA1131, Aug. 24, 1987.
19. Potter, P.G.; Zuckerman, M.: "Cyclic Request Control for Provision of Guaranteed Bandwidth within the DQDB Framework," Proc. ISS'90, Stockholm, 1990.
20. IEEE 802.6: "Distributed Queue, Dual Bus (DQDB) Subnetwork of a Metropolitan Area Network(MAN)," Draft 15, 1990.
21. Hahne, E.L.; Maxemchuk, N.F.: "Improving the Fairness of DQDB Networks," Proc. of INFOCOM'90, San Francisco, CA, 1990.
22. Maly, K.; Zhange, L.; Game, D.: "Fairness Problems at the Media Access Level for High-Speed Networks," ODU CS Tech. Rpt. TR #90-15; March 1990.