Old Dominion University Research Foundation

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

EXTREMELY HIGH DATA-RATE, RELIABLE NETWORK SYSTEMS RESEARCH

By

E. C. Foudriat
K. Maly
R. Mukkamala
N.D. Murray
C.M. Overstreet

Annual Report
For the period August 15, 1889 to August 15, 1990

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

May 1990

Old Dominion University Research Foundation is a not-for-profit corporation closely affiliated with Old Dominion University and serves as the University's fiscal and administrative agent for sponsored programs.

Any questions or comments concerning the material contained in this report should be addressed to:

Executive Director
Old Dominion University Research Foundation
P. O. Box 6369
Norfolk, Virginia 23508-0369

Telephone: (804) 683-4293
Fax Number: (804) 683-5290

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

EXTREMELY HIGH DATA-RATE, RELIABLE NETWORK SYSTEMS RESEARCH

By

E. C. Foudriat
K. Maly
R. Mukkamala
N.D. Murray
C.M. Overstreet

Annual Report
For the period August 15, 1889 to August 15, 1990

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

May 1990

Extremely High Data-Rate, Reliable Network Systems Research

Annual Report and Renewal Request

Submitted to the

NASA Langley Research Center

April 23, 1990

E. C. Foudriat †
K. Maly †
R. Mukkamala †
N. D. Murray ‡
C. M. Overstreet †

Computer Science Department
Old Dominion University
Norfolk, VA 23529-0162

---

† Department of Computer Science, Old Dominion University

‡ NASA Langley Research Center, Hampton, VA 23665

Abstract

Significant progress has been made over the past year in the four focus area of this research group: gigabit protocols, extensions of metropolitan protocols, parallel protocols, and distributed simulations. As evidence, we have presented one paper at SIGCOMM'89, have submitted three papers to SIGCOMM'90, four papers will be presented at the Pittsburgh Summer Simulation Conference in May, and one paper has been submitted to the Winter Simulation Conference. Two activities, a network managment tool and our CSMA/CD protocol, have developed to the point that we plan to apply for a patent during the next year; a toolset for distributed simulation using the language SIMSCRIPT also has commerical potential and will be further refined this year. We summarize this year's results for each of these areas and describe next year's activities.

1. Introduction

This report serves two purposes: (1) as a summary of the accomplishments of this research group over the past year (attached copies of submitted papers and technical reports provide a more detailed report) and (2) as a proposal for next year's activities for the research group.

Over the last year our research has focused on four areas. The first is an in-depth study of FDDI as it will perform in realistic situations, i.e., networks from a few nodes and less than a kilometer up to 1000 nodes over hundreds of kilometers. The study takes into account various internal parameters such as L-MAX (set up time for packed transmission), station latency time, and the relationship between manufacturer specifications and stated standard specifications. Additionally we have studied how the choice of external parameters such as the token-rotation-time influences performance. As part of this effort we are comparing FDDI's performance to that of DQDB's (formerly QPSX) which we are studying as well. This effort is described in more detail in Section 2 below.

We were invited by DARPA to develop a proposal based in part on this work; the proposal has now passed the first level of peer review. This has directed some of our efforts to the study of protocols for gigabit networks. In particular, we wanted to develop a protocol which combines the best of CSMA/CD and FDDI and is at the same time efficient at the gigabit, wide-area level. We have evolved a protocol which, at this stage of our study, looks very strong not only in the absolute sense (e.g., access delays on the order of 10-100 μs at loads ranging up to 200% of nominal network capacity) and much better when compared to a protocol like FDDI when it is extended to a gigabit rate. This is further described in Section 3.

A third effort is the study of using inexpensive hardware to obtain high bandwidth networks. The basic premise is that parallelization will help in

communications as well as it has done with computation. We believe this will become an extremely important research issue and have preliminary results which show promising improvements. It is described in Section 4.

The fourth effort of our research group—distributed simulation—has less successful in providing immediate improvement for the simulation studies which we have completed. This is an difficult problem and at this point we experienced some negative results. For example, one distributed technique we used took more time than running the simulation on a single processor. To meet immediate needs, we have resorted to distributed processing of simulation; that is we are distributing individual simulation runs to as many machines as we can find on the net which can handle them. Thus we achieve parallelization of runs but still have to wait sometimes a day before we can get the results and decide what to run next. On the positive side, we have developed a collection of support tools which can be used with SIMSCRIPT, a widely used simulation programming language, to support distributed simulation. These tools look promising and are still being refined and evaluated. This summary of this effort is presented in Section 5.

During the past year we have produced nine technical reports, one paper has been presented at SIGCOMM'89, one paper has been submitted to the journal of *Computer Networks and ISDN*, three papers have been submitted the SIGCOMM'90, four papers will be presented at the Pittsburgh Summer Simulation Conference in May and one paper has been submitted to the Winter Simulation Conference.

Product Development

While some aspects of this research require additional development before leading to a commercially viable product, other efforts are now much closer as a result of this year's effort; support and interest from Sun Microsystems for commercialization continues to be very strong. Two have developed to the point that we feel it is now appropriate to apply for a patent for each during the next year; we intend to do so. A third, a toolset for distributed simulation, also has commercial potential.

The first product is a tool to assist in station management for FDDI networks. When setting up an FDDI network, an individual charged with station manager responsibilities must select several parameters which have significant impact on network performance. For example, one key parameter is Token Rotation Time, and current FDDI literature provide little guidance in its selection. Our modeling of FDDI performance has led to development of procedures, based on attributes of a particular network, which can suggest plausible values for Token Rotation Time. The product will assist a station manager is parameter selection using both simulation and analytic tools to propose and assess adjustable network parameters. Because of Sun Microsystems' perception that FDDI will be the network of the 90's both for commercial and military applications, we anticipate that this tool, after refinement from a prototype to a commercial quality

product, will be carried Sun's software product catalog, Catalyst. Over the next year we will refine this into a commercial grade product for use by Sun and inclusion the Catalyst catalog.

Interest and response to our CSMA/CD protocol has been strong and encouraging. Initial performance evaluations based on analytic and simulation modeling are very promising. While some aspects for the protocol require additional refinement, (for example, the 'fairness' issue is discussed in [Maly90a] with some proposed solutions in [Foudr90a]); the next step in product development is the development of an engineering model as a proof of concept. Over the next year, we plan to continue to specify the protocol so that an engineering model can be built as a proof-of-concept. Building the engineering prototype will require an significant effort and external support. Support from CIT, Sun Microsystems and NASA over the next year will allow us to refine and analyze the protocol so that one year hence, development and commercialization can continue without support from CIT. To support this product development, Sun is purchasing three FDDI boards for our use and is providing an additional high performance work station.

Parallel networking is a promising alternative for high data rate networks because of cost, reliability and more promising performance improvements above those of just raw data rate increase. We need, over the next year, to solidly document these advantages so that we can identify and seek additional commercial support for for further development of this technology.

We continue to focus distributed simulation efforts towards effective utilization of environments consisting of networks of scientific work stations; we feel that over the next ten years, this type of computing environment will become even more commonplace. We have a prototype of tools which allow components of a simulation to run on different machines in a network. Initial timing studies are promising. However, this approach—as with all others proposed for distributed simulation—will work well in some circumstances and poorly in others. To assess the product viability of these tools, we have three tasks over the next year: to continue timing studies to measure speed-up effects; to refine the prototype based on both these timing studies and the tools use in several models; and to develop procedures, based on a particular's model properties, which can predict if significant speed-ups can be achieved with this approach.

2. MAN Protocols

Problems and Approaches

One of our primary goals is to develop protocols for gigabit speed networks. One of the means of accomplishing this is by employing parallelism of currently emerging MAN protocols such as FDDI or DQDB (formerly QPSX). Given that little performance analysis currently exists for these networks, an understanding

of how these networks perform under various parameters was lacking. Specifically, our interest was to understand how number of nodes, network length, packet length, data rates and traffic types affected these networks with the intention to use the results to determine strategies for extending these protocols (or developing new ones) to use at gigabit speeds.

Accomplishments 1989-90

Preliminary investigations of FDDI and DQDB have been ongoing in parallel and led to a number of interesting results [Game90a, Maly90b, Maly90a]. The major effort in DQDB has been in understanding how the dual-bus, reservation scheme impacts the availability of the network to various nodes depending on the node's physical position in the network. Simulation results have shown that during high loads, nodes near the ends of the two busses have a tendency to either starve or dominate the network depending on the traffic placement strategy. We are currently developing a strategy which will adapt the strategy to the load conditions and provide a more equitable distribution of network bandwidth among the nodes.

A significant body of results currently exists on performance of token rings. The primary difference between assumptions made in this research and FDDI is the mechanism used to determine how long a node can hold the token. FDDI employs a token-holding-timer algorithm, the impact of which we have investigated. Our current research of FDDI, pointing towards extension of the throughput to gigabit per second rates, has been to determine the effect of removal (and reuse) of a packet at its destination and employing multiple rings.

Research Efforts 1990-91

In addition to investigation of various topologies (ring of rings, mesh, and braided mesh) and their performance, we would like to develop a software package to aid in configuration of FDDI networks. Particularly, many systems managers do not understand the impact of the token rotation time in FDDI on performance. We would propose to develop a package which will allow a systems' manager to easily enter his configuration and traffic requirements. The package will predict network performance using both with static estimations and simulation. This will be eventually extended to include bridges to other types of networks (token ring, Ethernet, DQDB, etc.) and provide for a measure of overall system performance. We anticipate that support for this effort will come from Sun Microsystems and that the product will be marketed by them as an analysis tool for communications managers. Sun has an interest specifically in FDDI and is marketing FDDI as a backbone for its Ethernets.

3. Media Access Protocols

Problems and Approaches

Our major interest is in investigating network protocols for optical networking at gigabit rates. Many of the present protocols which operate in the kilobit and lower megabit per second range fail in one or more requirement areas at higher data rates and/or over the wider range of distances which will be used in gigabit networking. Several problems must be considered in developing EHDNS media access protocols. We have defined two areas critical to effective protocol operation: traffic placement and resource allocation [Maly88a, Maly89a] papers on resource allocation and traffic placement] Traffic placement policy describes the node's decision structure as it attempts to transmit messages across the network. Our studies have resulted in analyzing a number of traffic placement policies. Resource allocation generally divides the system resources—typically a multiplexing operation—into various units to support heterogeneous traffic. Resource allocation must operate dynamically so that nodes attain a fair share according to needs and resources can be easily reallocated as needs change. These areas encompass the media access protocol requirements of minimal access delay, throughput performance at all load levels, fairness(to nodes) and rapid recovery due to traffic perturbations [Maly90a].

Our basic research is directed toward understanding the performance features and limitations of present media access protocols and toward the development of revised or new approaches which can fulfill these stringent requirements. In so doing, we have examined several access protocols including extensions to token rings (FDDI) [Maly90b], slotted reservation systems (DQDB) [Newma88a], a protocol concept based upon local carrier sensing (CSMA/RN) and parallel networks (discussed in Section 4).

Accomplishments 1989-90

During the 1989-1990 grant period, the ODU network research group made significant progress in studying media access protocols. In last year's proposal, we mentioned briefly a new protocol, carrier-sensed multiple access for ring networks (CSMA/RN), which we had started researching. Our studies during 1989-90 have demonstrated CSMA/RN to be an extremely effective and a most promising protocol for gigabit networking. It provides many desirable features to support the requirements above including:

1. virtually immediate access if the network is free—no wait for a token or an empty new slot;
2. ability to handle widely varying message sizes;
3. up to 200% of rated network capacity without overload (2 Gbps traffic for a 1 Gbps network data rate);
4. excellent message performance (low access delay and minimal message fracture) up to 150% of rated capacity;

5. synchronous traffic with little overhead, less than 1%, with no global master controller and automatic recovery of unused synchronous traffic bandwidth;
6. guaranteed maximum access time;
7. capability to span distances from 2km—10,000km and wide range of node counts; and
8. feasible for physical or virtual rings, i.e., those constructed from fiber optic telephone trunk lines.

Hence, CSMA/RN approaches a universal media access protocol for gigabit networks.

During the 1989-1990 grant year, the basic operational and performance features of CSMA/RN were documented using analytical and simulation models. The analysis, based upon queuing theory demonstrated the access protocol capability. The simulation models were expressly designed to study the performance features relative to the system parameters and to gain insight into interactions between message on the ring and the nodes with messages in their queue ready to send.

We have submitted a CSMA/RN paper to the SIGCOMM'90 conference [Foudr90b], have briefly described its operational features in a White Paper[Foudr90a] and are preparing a journal paper for submission to the IEEE Transactions on Communications.

Our research on media access protocols has examined extensions of other concepts besides CSMA/RN. Token rings, for example, have difficulties with large token interarrival times for rings which span long distances; this can be alleviated to a significant degree by use of parallel rings. Slotted reservation systems do reasonably well; however, with a fixed slot size there are always messages which are heavily fractured or wasted capacity depending upon the slot size selected and the reservation scheme has significant fairness problems at high loads.

Research Efforts 1990-91

During the 1990-1991 grant period, we, in conjunction with our other grant support, will initiate a feasibility demonstration model of the CSMA/RN electro-optical controller which interfaces the physical fiber media with the receiving and transmitting of information at the node. This piece of the CSMA/RN system is critical to its successful operation. This model can use lower data rate equipment, i.e., 100—200 Mbps components in order to use cheaper, readily available parts and logic chips. The operational features and development requirements for the hardware model are discussed in detail in the attached white paper. In addition and in conjunction with Sun Microsystems and NASA, we will seek additional support so that the design and development of a laboratory prototype model can be initiated and the capabilities of CSMA/RN demonstrated.

In addition to the physical demonstration development, considerable research is required to more fully document the performance capability of CSMA/RN. In this direction and with support from our funding sources we plan to expand both the analytical and simulation modeling. First, the present analytical and operational simulation models will be improved to further document the capability of CSMA/RN to support the gigabit networking requirements. These models treat operations at the bit structure level so they can be used to study the detailed interaction each message with the nodes and each other. This limits the time frame over which the network can be modeled to the range of milliseconds to a few seconds. These simulations generally take between 2 and 24 hours to complete. While most asynchronous data operations occur within this range, most synchronous communications like telephone, video and control data consider the time frame of minutes to a few hours. In addition, the synchronous data can vary, for example, as in a silent period in a telephone call. Over that time, the network must periodically provide space for the transfer of new accumulated information and support the initiation, placement and termination synchronous traffic. With the present simulation models it is impossible to study this time frame of minutes to hours. Thus, we will investigate new modeling techniques which can adequately represent the interactions of synchronous and asynchronous data on the network and which can predict the performance under the wide variations in traffic patterns which can occur in network operations. We plan to examine our present and new models to determine to what extent they or the ideas with in can be incorporated into useful and commercial products.

4. Parallel Protocols

Problems and Approaches

In developing extremely high data rate networks, we have considered the concept of parallel communications as an alternative mechanism for accomplishing gigabit rates. In analogy with parallel computing, parallel communications can:

1. improve reliability by initiating identical or redundant information transfer which can compare results and correct any errors;
1. speedup operations by parallelizing normally serial operations; and
1. improve operations by new procedures which take advantages of the systems inherent parallelism.

The key to developing parallel communications is to identify the unique performance features it provides within the context of the high data rate networking problems and to incorporate those features into present or revised lower data rate access systems.

Accomplishments 1989-90

Our research on parallel communications has progressed significantly during the 1989-90 grant year. Our major accomplishment has been to develop an analytical technique for separating and identifying those factors in a parallel network that contribute to its performance improvement. They are improvements due to: 1) more rapid insertion speed, i.e., more bits can be placed into the network in a given time; 2) more rapid interarrival of the access mechanisms fundamental control, i.e., the token in a token system or an empty slot in a slotted system; and 3) the reduced variance in service time which occurs due to smoother operational access and which affects the expected access time in many queueing theory models.

During the grant period, we have developed a token ring simulation model which can identify separately each of the above factors and how they are influenced by single or multiple ring structures. As yet we have not had a chance to fully explore all aspects of multiple ring protocols but the results of our early studies are indeed promising. For example, we have found that in multiple token rings one can use a non-exhaustive message placement policy and avoid the inherent instability due to nonuniform traffic patterns at a node which prohibits a non-exhaustive policy in a single token ring. In additions, the non-exhaustive policy provides significantly improved performance in the multiple ring configuration, in that access delay and response time remain virtually constant up to 90% of network capacity whereas in the single ring the access delay and response time deteriorate significantly above 50% of capacity. We are expanding the analytical and simulation modeling to examine other parallel token ring network media access protocol properties.

Our results on modeling of a parallel token ring network and its performance under a variety of conditions will be presented at the Simulation and Modeling and Conference in Pittsburgh in May 1990 [Mukka90a]. We are presently preparing an article to describe the characteristics of parallel networking for a gigabit network workshop in November, 1990.

Research Efforts 1990-91

We plan to expand analytical and simulation modeling other network structures. First, we plan to model other access mechanisms, like slotted, train and reservation systems (Cambridge, Expressnet and DQDB are examples, respectively) to determine the performance advantages which can occur due to parallelism. Another important aspect of parallel networks is structure, that is connectivity, does not have to be identical in all units. This can lead to a fourth performance improvement, travel time between nodes. We plan to explore and exploit all of the factors in order to document the improvements that parallelism can provide in extended high data rate networking.

5. Distributed Simulation

Problems and Approaches

Simulations of high speed network protocols are often very CPU intensive operations requiring long run times. Very high speed network protocols (Gigabit/sec rates) require longer simulation runs in order to reach steady state, while at the same time requiring additional CPU processing for each unit of time because of the data rates for the traffic being simulated. As protocol development proceeds and simulations provide insights into problems associated with the protocol, the simulation model is often changed to generate additional or finer statistical performance information. This process is time consuming due to the required run times. Given the wide availability of networked high-performance scientific work stations, efforts continue to develop methods for performing these simulations in a distributed fashion, utilizing additional CPUs to reduce the time required to obtain results.

Accomplishments 89-90

Distributed simulation is a hard problem [Tinke89a]. Simulations of tightly coupled systems such as network protocols which share a common resource have proven to be difficult due to the amount of shared information that is required. Our initial efforts in developing decompositions proved ineffective. In order for a distributed simulation to provide reductions in run time, the modules must be designed so that they can perform compute intensive operations and require very little intermodule communication. Our efforts this year concentrated on developing decompositions that either closely parallel the physical network, called physical decompositions [Pater89a] or ones that distributed operations of the simulation program mechanics such as event list processing. Because of the amount of data sharing and processor synchronization required, both of these methods provided disappointing results. One observation made during our tests was that if data could be provided by one module to a second one without a time dependent cycle developing, the synchronization needs are much less. The major problem being addressed currently is the development of methods for model decomposition that reduce the amount of intermodule time-based dependencies.

A set of distributed simulation tools (software routines) which can be used directly with SIMSCRIPT II.5 have been developed and tested. SIMSCRIPT II.5 is a widely used simulation programming language. Since the routines can be used with standard SIMSCRIPT, they potentially support the development of distributed simulation with little retraining of programmers. Initial timing studies indicate that—if effective model decompositions can be found—these tools provide a modeler, particularly one proficient in SIMSCRIPT, the ability to easily distribute a a simulation across several work stations. Because of the general interest in distributed simulation and the availability of networked scientific work stations, these tools may be of significant commercial value. Studies during the next year will continue their development and evaluation.

Research Efforts 90-91

Using static code analysis tools previously developed at Old Dominion University, we will perform data flow analysis of existing simulation models, written in the SIMSCRIPT, C, and Pascal languages, to determine the prevalence of code sections which either supply or consume time independent data objects during the simulation run. Additionally, we believe that code containing time dependent data cycles can be distributed if there is sufficient computation time between data requests to allow for the synchronization to occur or that the dependencies are not tight, one generation per synchronization, so that values can be precomputed and the simulation can be made to proceed.

Currently we have models for the simulation of FDDI, DQDB, and CSMA/RN [Foudr90b, Khann90a, Game90a] that are available for analysis. To support the distribution of modules detected, we will use tools developed during the last year that provide interprocessor communication and server model synchronization. The tools, described in [Pater90a], may need to be extended to allow for two-way data flow and synchronization under a request and deliver scheme.

References

Foudr90a.
Foudriat, E. C., K. Maly, C. M. Overstreet, S. Khanna, and F. Paterra, *CSMA/RN—A Universal Protocol for Gigabit Networks,* Computer Science Dept., Old Dominion University, April 1990.

Foudr90b.
Foudriat, E. C., K. Maly, C. M. Overstreet, S. Khanna, and F. Paterra, "A Carrier Sensed Multiple Access Protocol for High Data Rate Ring Networks," TR 90-16, Computer Science Dept., Old Dominion University, March 1990.

Game90a.
Game, David and Kurt Maly, *Performance of Gigabit FDDI,* Computer Science Dept., Old Dominion University, April 19, 1990.

Khann90a.
Khanna, S. and E. C. Foudriat, *Modeling High Data Rate Communications Network Access Protocols,* Computer Science Dept., Old Dominion University, April 17, 1990.

Maly88a.
Maly, K., C. M. Overstreet, Xiao-ping Qiu, and Deqing Tang, "Dynamic Resource Allocation in a Metropolitan Network," *SIGCOMM-88 Proceedings*, pp. 13-24, August 1988.

Maly89a.
Maly, K., E. C. Foudriat, D. Game, R. Mukkamala, and C. M. Overstreet,

"Traffic Placement Policies for a Multi-Band Network," *SIGCOMM-89 Proceedings*, pp. 94-105, Sept 1989.

Maly90a.
Maly, K., L. Zhang, D. Game, E. C. Foudriat, and S. Khanna, "Fairness Problems at the Media Access Level for High-Speed Networks," TR 90-15, Computer Science Dept., Old Dominion University, March 1990.

Maly90b.
Maly, K. and D. Game, "Extensibility and Limitations of FDDI," TR 90-14, Computer Science Dept., Old Dominion University, March 6, 1990.

Mukka90a.
Mukkamala, R., E. C. Foudriat, K. J. Maly, and V. Kale, *Alternate Parallel Ring Protocols*, Computer Science Dept., Old Dominion University, April 17, 1990.

Newma88a.
Newman, R. M., Z. L. Budrikis, and J. L. Hullett, "The QPSX Man," *IEEE Communications Magazine*, vol. 26, no. 4, pp. 20-28, April 1988.

Pater89a.
Paterra, F., C. M. Overstreet, and K. Maly, *Distributed Simulation of Network Protocols*, Computer Science Department, Old Dominion University, 1989.

Pater90a.
Paterra, F., C. M. Overstreet, and K. Maly, *Distributed Simulation: No Special Tools Required*, Computer Science Department, Old Dominion University, 1990.

Tinke89a.
Tinker, Peter A. and Jonathan R. Agre, "Object Creation, Messaging, and State Manipulation in an Object Oriented Time Warp System," in *Proceedings of the 1989 Distributed Simulation Conference*, pp. 79-84, Society for Computer Simulation International, Mar. 1989.

SUBMITTED PAPERS

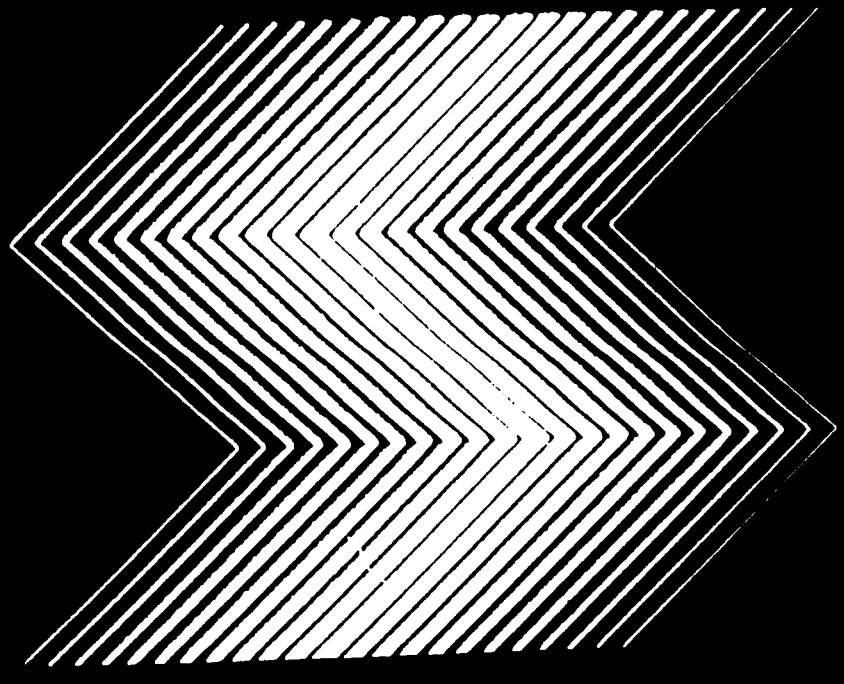
Computer Communications Review
Volume 19, Number 4
September 1989

acm PRESS

SIGCOMM '89 SYMPOSIUM

Communications Architectures & Protocols

Austin, Texas
September 19–22, 1989



SPONSORED BY ACM SIGCOMM WITH SUPPORT GIVEN BY THE INFORMATION SCIENCES AND TECHNOLOGY CENTER OR SRI INTERNATIONAL.

Traffic Placement Policies for a Multi-Band Network*

K. J. Maly E. C. Foudriat D. Game
R. Mukkamala C. M. Overstreet
Department of Computer Science
Old Dominion University
Norfolk, Virginia 23529-0162

Abstract

Recently protocols have been introduced that enable the integration of synchronous traffic (voice or video) and asynchronous traffic (data) and extend the size of local area networks without loss in speed or capacity. One of these is DRAMA, a multiband protocol based on broadband technology. It provides dynamic allocation of bandwidth among clusters of nodes in the total network. In this paper, we propose and evaluate a number of traffic placement policies for such networks. Metrics used for performance evaluation include average network access delay, degree of fairness of access among the nodes, and network throughput. The feasibility of the DRAMA protocol is established through simulation studies. DRAMA provides effective integration of synchronous and asynchronous traffic due to its ability to separate traffic types. Under the suggested traffic placement policies, the DRAMA protocol is shown to handle diverse loads, mixes of traffic types, and numbers of nodes, as well as modifications to the network structure and momentary traffic overloads.

*This work was sponsored in part by contracts CITT-596041 from CIT and NAG1-1-908 from NASA Langley Research Center.

1 Introduction

Several recently introduced protocols illustrate the change in performance that results from subdividing network capacity into multiple channels. In addition, demand for integration of video and voice traffic with data traffic has resulted in protocols that allow for both synchronous and asynchronous traffic, such as DRAMA. The DRAMA protocol not only takes advantage of the multichannel efficiency but also allows for synchronous and asynchronous traffic over a large distance without significant loss in speed and capacity. This protocol is introduced in [15,16], and is based on a broadband technology, allowing for dynamic allocation of bandwidth among clusters of nodes, called local area network groups (LANGs), and dynamic allocation of synchronous/asynchronous traffic bandwidth.

Marsan and Roffinella [12] evaluate multichannel multiple-access schemes such as CSMA/CD for local networks, but standard CSMA/CD protocols do not provide for transmission of synchronous data as addressed in this paper. Chlamtac and Ganz [4] propose a multichannel design which statically allocates bandwidth capacity to avoid the simultaneous delivery of frames to a node from different channels, but does not allow for efficient use of unwasted bandwidth due to unbalanced traffic patterns. Merakos and Bisdikian [10] divide the capacity of the network into channels for intra- and inter-LAN traffic, but does not provide for dynamic allocation of the bandwidth allocated to each traffic type. Minimum delivery times for synchronous traffic cannot be guaranteed due to the use of bridges for interconnection of LANs. Wong and Yum [17] allocate channel bandwidth by a contention-based reservation protocol which provides circuit switching services for all traffic types, whereas, DRAMA only provides circuit switched services for synchronous traffic.

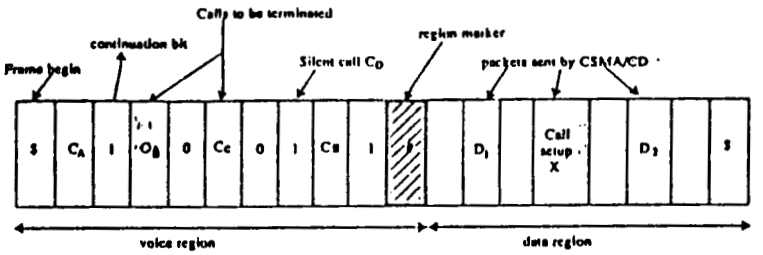


Figure 1: Sample Frame

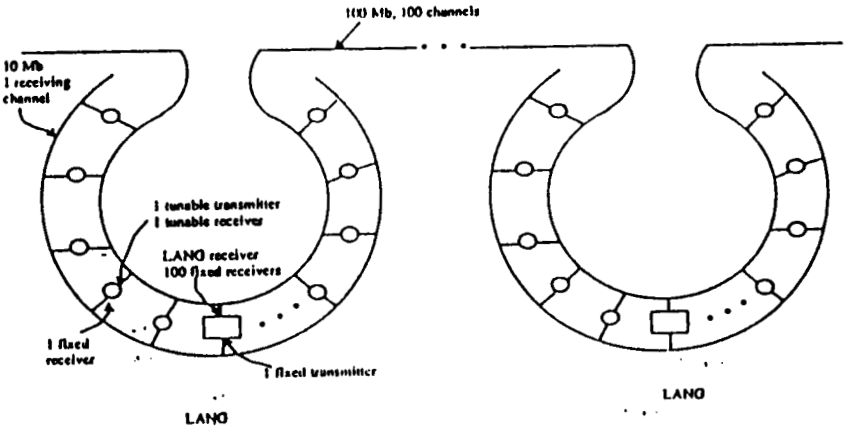


Figure 2: A possible overall network configuration

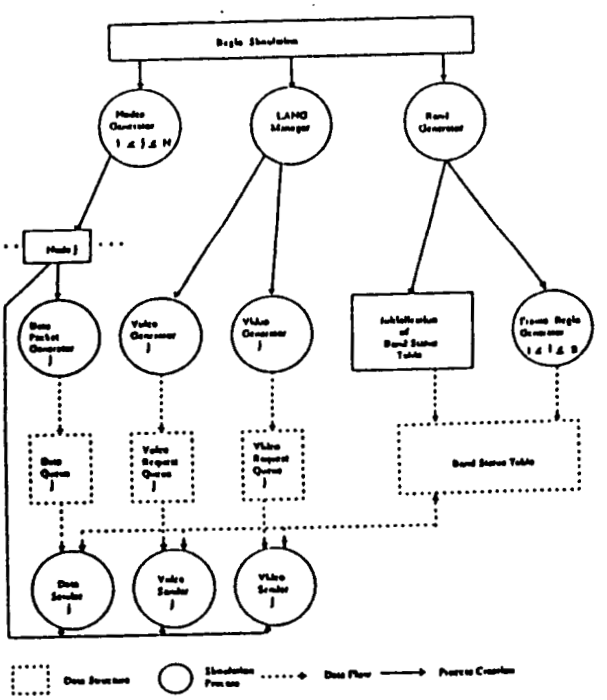


Figure 3: Structure of Simulation program

In [11], we presented the results of a collection of simulation studies of the DRAMA protocol. That paper focused on the ability of the protocol to reallocate bandwidth among LANGs in order to respond to changes in individual LANG workloads. The current paper reports on simulation studies modeling an individual LANG; in it, we analyze network performance in handling loads and evaluate how traffic placement policies affect performance. We assume a multichannel network based on CSMA/CD and wish to decide if the approach is feasible.

This paper is organized as follows. Section 2 describes the DRAMA protocol; Section 3 describes the simulation model used in these experiments; Section 4 presents the traffic placement policies studied here; and Section 5 discusses the results of these studies.

2 The DRAMA Protocol

The DRAMA protocol is designed for the dynamic sharing of bandwidth of a single broadband bus among groups of nodes in a large, integrated voice-video/data network. The amount of bandwidth available is assumed to be large, say 350-500 MHz. The nodes, each capable of transmitting all traffic types, are clustered by distance and function into LANGs. This type of clustering is typical for various locations of a company within a particular city, installations on a large ship or military base, or among different departments within a university. The cable bandwidth is frequency-divided into bands dedicated to particular LANGs and a global pool of bands that may be acquired by any of the LANGs. For each LANG, requesting, acquiring, or releasing a band depends on the current distribution and amount of traffic within that LANG relative to the current traffic within the entire network. A LANG is allowed to transmit only on those bands which have been assigned specifically to it, but is required to receive on all bands. For a more detailed discussion of DRAMA, including error recovery, the reader is referred to [15,16,11].

Basic design objectives of DRAMA are the integration of synchronous/asynchronous traffic and dynamic bandwidth allocation.

2.1 Synchronous/Asynchronous Transmission Protocol

This section briefly describes the DRAMA protocol for a LANG. In the protocol, the fraction of a band's capacity allotted to each traffic type depends on the current synchronous and asynchronous load. Time on each band is slotted into frames. Each band's frames are delimited by "frame-begin" markers broadcast by the band's current *band-leader* and are partitioned into voice/video and data regions. The boundary between the two regions varies from frame to frame, depending on the number of voice calls in talk-spurts during that frame. Either data or voice/video may consume the entire frame if no traffic of the other type is present.

The data region is composed of data packets and call-setup requests. The bandwidth in the data region is allocated using a CSMA/CD traffic placement policy. Normally CSMA/CD is not suitable for use among nodes separated by more than several kilometers because the interval during which a collision can occur is directly proportional to the propagation delay between the most distant nodes. DRAMA circumvents this problem by restricting transmission privileges on a band to exactly one LANG at a time, while allowing all LANGs to receive the transmissions. In this way, the CSMA/CD-based protocol can be used over the entire set of LANGs with the same efficiency as in a single LANG.

The voice/video region provides a virtual circuit for each established (one-way) voice call. In the multi-channel version this means that a two-way inter-LANG call uses two bands, one for each direction of the call. One varying size slot is allocated in the voice region to each (one-way) voice circuit. The slot contains a varying size voice packet followed by control information, called the *control byte*. The slot size may differ because silence periods, which comprise roughly 60% of an average voice conversation [3], are not transmitted. The control byte informs the other nodes whether the call will terminate after this frame or will be continued. The slots for the different voice calls are contiguous and precede any data transmitted in the frame.

Figure 1 shows a frame in which five calls are ongoing (A,B,C,D,E), one of which is silent (D). Three packets are transmitted in the data region using CSMA/CD (data packets 1 and 2, and a successful call-setup for the next frame).

Since in CSMA/CD the amount of bandwidth wasted due to collisions increases dramatically with load, it is important to allocate bandwidth in a fair manner. Therefore, reducing load on heavily used channels at the cost of increasing it on lower utilized channels should reduce total collisions.

2.2 Dynamic Bandwidth Allocation

In proposed broadband systems such as [9] and [13], as well as in currently available commercial systems, the broadband frequency spectrum is statically parti-

tioned by user group and/or by traffic class. For example, CableNet [14], Sytek's LocalNet [6], and Mitre CableNet [8] each partition the bandwidth into fixed bands for particular applications of specific groups of users; some bands are permanently reserved for video channels, some are reserved for time division multiplexing for a set of closely located users, and some bands are dedicated to specific functions such as process control. However, a common characteristic of a network that supports diverse traffic classes such as voice, data, and video is that the bandwidth requirements of both an individual node and a LANG fluctuate widely over time compared with the LANG's average requirement. In such diversified systems, static partitioning according to average requirements will often waste idle bandwidth; at other times, it will be insufficient to satisfy a LANG's traffic requirements while bandwidth is available elsewhere in the network.

In the DRAMA protocol the bandwidth is frequency-divided into M + 1 bands. One band is reserved for a slotted band-control channel used by all the LANGs to coordinate band-sharing, and the remaining M fixed-size bands (say, 10MHz each) are available for voice, data and video transmissions. The M bands are partitioned into a set of *dedicated bands* and a set of *available* bands. Dedicated bands guarantee that no LANG "starves" and that each LANG's performance is at least that of a normal, solitary LANG using a baseband cable.

The available bands are either assigned to a LANG or are in a global pool. Bands in the pool are shared via a dynamic, fully-distributed, band-sharing policy that allows each LANG to obtain global bands based on three factors: its current needs, the current needs of other LANGs, and the current availability of global bands. When a LANG acquires a band, that LANG's nodes have exclusive transmission rights to the band; each node in the system must be able to receive all bands that might contain packets addressed to it. Because all bands can be received at all nodes, a uniform communication mechanism exists among all nodes in all LANGs; this is preferable to the use of gateways, which introduce additional delays and buffering requirements since transferred traffic must compete with local traffic when it is sent between nets. Simple wide-band repeaters may be required over a long distance network to overcome attenuation and/or signal distortion. The details of this band allocation can be found in [11].

2.3 Traffic Placement

When a node wishes to transmit, it chooses among the bands on which its LANG currently has transmission privileges; the node seeks a band with no traffic. If the LANG owns 10 bands, a typical figure, we estimate this to take about 20 microseconds. We refer to the policy for choosing among the free transmission bands as the *traffic placement policy*.

Given the frame format in Figure 1, two primary objectives in formulating a placement policy exist. First, it is important to keep bands free of voice/video traffic if possible since a band cannot be freed if its release would interrupt synchronous traffic. Second, in order to minimize delay, data traffic is spread as evenly as possible over all the LANG's current bands. The systems and algorithms that control this traffic placement policy and its resulting performance are the subject of this paper.

2.4 DRAMA Implementation

In the original DRAMA system each node needed as many receivers as there were channels and enough tunable transmitters to service the channels assigned to its LANG. This is not economically feasible. Figure 2 presents a possible solution to this cost problem by showing how nodes in a LANG can share receivers and how the number of transmitters per node can be limited. The network sketched has 100 Mb total capacity, divided into 100 channels. The LANG illustrated has 10 bands assigned to it. Each node needs at least one tunable transmitter/receiver pair to be able to determine whether a selected band is free and to detect a possible collision on that band after it has begun transmitting. Nodes can have more than one tunable transmitter if they need to send information on more than one channel simultaneously. All incoming information is handled by the LANG central receiver system and is forwarded to the nodes in that LANG by a secondary channel that connects all nodes, or, alternatively, by twisted pair (not shown). In the system illustrated, the LANG receiver has 100 fixed receivers to listen to all channels; it filters all messages belonging to its LANG (including the ones sent from within the LANG) and channels them to the LANG net. Hence the total number of signaling devices in a LANG with n nodes is $3n + 100$.

3 Simulation Model

We used simulation to study performance of the DRAMA protocol with different traffic policies. The network model and related protocols were written in SIMSCRIPT II.5. The simulation program is highly parameterized to allow experimentation with different loads and different traffic placement policies.

Initial experimentation showed that the model achieved steady-state in 1.8 seconds of simulation time; thus for each experiment, data collection began after this point. Three techniques were used for model validation. First, traces – that is, printing sequences of significant model state changes – satisfied our concerns about correct program implementation. Second, as indicated by figures in this paper, the model replicated standard CSMA/CD behavior. Third, as we altered input parameters the model responded appropriately. We did occasionally encounter what was, at least initially, counterintuitive behavior in experiments, but further analysis showed in each case that the model had behaved appropriately; it was our intuition that failed us.

4 Model Objectives and Structure

The main objectives of our simulation experiments are:

- To create an implementation of the DRAMA protocol at the packet level as it would operate in a single LANG,
- To compare several traffic placement policies, and
- To measure performance under a variety of loads and traffic placement parameters.

Our earlier studies [11] dealt with performance issues corresponding to band allocations among the LANGS in a network. In those studies, a LANG was modeled as an abstract entity with varying needs for bands. In this study, however, we concentrate on the performance issues related to a single LANG (as opposed to a network of LANGs). Accordingly, a LANG is modeled as a group of nodes with the ability to communicate to nodes across the network. Restricting simulation to a single LANG considerably reduces the execution time required for each run. The conclusions determined for a single LANG are easily generalized to the multiple LANG case since each LANG operates independently.

In our earlier study [11], we have shown that bands can be rapidly reallocated such that the band utilization at every LANG is kept within a small percentage of the total network average. In this study, we want to hold the number of bands constant and study the effects of traffic placement on the network.

As shown in Figure 3, the simulation model includes:

- Band Status Table: Each node maintains a band status table for all bands. Each entry of this table indicates the status (dedicated/global) of each band.
- Request Queues: Each node maintains three queues, one each for data, voice setup, and video setup. A first-in-first-out policy is used to service each queue.
- Sender Units: In addition to the request queues, each node maintains units to control the sending of voice, video and data blocks. These buffers contain the packet that is being currently transmitted as data.

The simulator initially generates the n nodes corresponding to a single LANG. For each of these n nodes, data traffic is created in terms of data packets as opposed to data messages of variable length. The data generator assumes Poisson arrivals of packets that are then placed in a data queue at the appropriate node. For efficiency, voice and video traffic are generated at the LANG level (described as LANG manager in Figure 3), and then randomly assigned to nodes in the LANG. Both video and voice traffic have Poisson arrivals and exponential service times. When a data packet, voice request packed, or video request packet is placed in its respective queue, the data frame controller checks with the band status table to determine whether a band is available for the packet. The time required to check all bands is called the *band choice thinking time*. A packet is not removed from its queue until after it has been placed and the collision interval has passed.

The video/voice/data traffic mix generated in a LANG is described as a percentage of the total load in the LANG. Once these percentages are chosen for a particular experiment, the average arrival rate for each traffic type at each node (λ_{DT}, λ_{VO}, λ_{VI} respectively) can be determined using the following system of equations:

$$\lambda_{DT}\mu_{DT} + \lambda_{VO}\mu_{VO} + \lambda_{VI}\mu_{VI} = \frac{C_u}{n} \quad (1)$$

$$p_{DT} + p_{VO} + p_{VI} = 100 \quad (2)$$

$$\lambda_{DT}\mu_{DT} = \frac{p_{DT}C_u}{100n} \quad (3)$$

$$\lambda_{VO}\mu_{VO} = \frac{p_{VO}C_u}{100n} \quad (4)$$

$$\lambda_{VI}\mu_{VI} = \frac{p_{VI}C_u}{100n} \quad (5)$$

where

- C_u is the assumed average channel capacity used by the LANG,

- p_{DT}, p_{VO}, and p_{VI} are the desired percentages of data, voice, and video traffic on the channel respectively, and
- μ_{DT}, μ_{VO}, and μ_{VI} are the the given average service times of data, voice, and video traffic respectively.

Some simulation experiments studied the effect of traffic mix. Equations (1) through (5) were used to generate arrival rates. For example, a traffic mix of 15% voice, 25% video and 60% data and a total network load of 60% results in individual network loads of 9%, 15% and 36% for voice, video and data respectively. The data arrival rate necessary to generate its appropriate load is further dependent upon the length of the specific packet. Data traffic is then uniformly distributed among the nodes of the network.

Other simulation experiments generated varied mixes of voice, video and data traffic in order to test the DRAMA system's ability to handle different traffic conditions. However, in many runs where other features of the protocol were being studied, a typical traffic mix of 15% voice, 25% video and 60% data was used. Standard transmission rates of 64 Kb/s for voice and 500 Kb/s for video were used for each circuit. The justification for using the higher video load is that a single video transmission occupies half a band, making it harder to place this kind of traffic than to place equivalent voice traffic. We felt that the increased video traffic would cause greater disruptions since these calls would occupy a large block on a band, making it more difficult to place data traffic and would tend to occupy more bands less effectively. The results would, in turn, have a greater tendency to show where problems in traffic integration would occur. Finally, most of the tests used a capacity of 10 Mb/s for the entire network, implemented as ten 1 Mb/s bands.

We considered the following performance metrics for system evaluation:

1. *Access Delay*: Due to the significance of network access delay in our experiments, we refer to this as *access delay*. Thus the access delay does not include the transmission and propagation delays. δ_i indicates the average access delay at node i.
2. *Fairness*: The network access delay for each of the n nodes should be independent of its position in the network and should be close to the average access delay E of the network. This is measured by the degree of fairness D_f, where

$$D_f = \sqrt{\frac{\sum_{i=1}^{n} (\delta_i - E)^2}{n}} \qquad (6)$$

where

- δ_i is the mean access delay at the ith node,
- E is the mean access delay in the LANG,
- n is the number of nodes in the LANG

Ideally, D_f should be zero. Note that this equation is used to measure fairness within a single LANG. Fairness among LANGs was studied in [11].

3. *Throughput*: We measure the throughput of a LANG as a function of the offered load, which is measured as a percent of network capacity. In these studies offered load ranged from 0 to 200 percent. Here, throughput is the percent of the network capacity taken up with successful traffic.
4. *Recovery time*: We were interested in the response of a single LANG to impulse traffic. To this end, in some experiments a sudden pulse of traffic was generated in order to determine how long the system would take to return to within 5 percent of the nominal load. Ten percent of the nodes were given a burst of additional data traffic in order to build up the queues at these nodes.

5 Traffic Placement Policy

As previously stated, the traffic distribution is a mix of video, voice and data traffic. Our simulation has separate traffic generation procedures for each type of traffic. In an effort to minimize the number of bands carrying synchronous traffic, new synchronous traffic is assigned to the lowest numbered band carrying synchronous traffic upon which it will fit. Video and voice traffic is handled essentially as described in the DRAMA protocol. Because of the short length of the simulation runs relative to the length of video and voice transmissions, these traffic demands are fairly static. We paid more attention to the placement of data traffic because its dynamic nature has greater effect on the network. We also studied the response to the load changes that would be induced by starting or stopping a video transmission.

By examining a number of placement strategies, we intended to determine the sensitivity of the network's performance to various methods of placing data traffic on the network.

5.1 Traffic Placement Strategies

Traffic placement policies are concerned with techniques to compete for transmission time and to handle

unsuccessful transmissions. More specifically, a traffic placement policy must address the following situations:

1. A node would like to transmit but finds all bands busy. The basic question is how long the node should pause before reexamining the bands. Strategies vary from no delay to waiting a large amount of time under the assumption that the network is currently saturated. The time required to scan all bands, which we call *scan time*, is the minimum possible delay. In some policies an additional delay is added in consideration of other nodes' attempting to transmit. Having a multiband network with no available bands is a much better indication of the state of the network than finding a single-band Ethernet network busy. We believed that this information might be used to devise better placement policies.

2. A node attempts to transmit but a collision occurs. A collision does not necessarily indicate that the network is busy, only that a band is busy. When a node attempts to (re)transmit a packet, it examines all bands to determine which ones are not busy. With light traffic it is better to attempt immediate retransmission. Under high loads, a delay may reduce collisions.

3. A node transmits successfully and has additional packets in its queue to be sent. This state is similar to the condition for busy bands, but has no information about the state of the network.

We studied four basic strategies. The simplest, *fixed*, is a static policy that delays a constant amount of time before retransmitting. We used a minimum of 20 microseconds for this figure. Since the policy assumes no knowledge of network load, it provides a baseline for comparison with other methods that use local or global knowledge in order to improve performance. Each of the three situations above employs this delay.

Speedup bases the delay between attempts upon the node's own queue size, using only local knowledge. The delay is inversely proportional to the queue length. In this approach the node transmits packets more frequently as its queue grows and incorporates an opposite philosophy of *backoff*. Under light network loading, a node will empty its queue more quickly by transmitting more often. We assumed that this method would reduce delays at low loads but that performance might suffer at high loads. *Speedup* uses this calculation for each of three situations.

Backoff employs the binary backoff approach, in which the delay time increases as more collisions occur. This method has been shown to be very successful on CSMA/CD networks during heavy loads, and would give us a point of comparison for alternative strategies under similar adverse conditions. In non-collision situations *backoff* waits for a time not less than *scan time* and not more than twice the value of *scan time*.

Given our intuition that *speedup* would be preferable in light loads and *backoff* in heavy loads, we also investigated a combination of the two, which we called *tempered backoff*. If a node is experiencing few collisions, *tempered backoff* will approximate *speedup*. As the number of collisions increases, the delay incorporated into the formula for *backoff* will quickly dominate the term for *speedup* and will give the characteristics of a *backoff* approach.

6 Results

Figure 4 illustrates DRAMA's performance compared with equivalent single-channel CSMA/CD protocols. The most important aspect shown in Figure 4 is that the access delay of the multiband DRAMA system is considerably lower than that of equivalent or higher bandwidth single-band CSMA/CD systems. Figure 5 substantiates this point and demonstrates that as the number of bands in the LANG increases from 1 to 20, the access delay for integrated voice/video/data traffic decreases significantly. One can conclude that multiple-band local area network systems are able to successfully handle integrated traffic using a CSMA/CD-type protocol. Also, DRAMA's performance is for networks that can cover up to 100 km whereas single-band higher frequency CSMA/CD protocols work only for a few km.

Specifically, the curves of Figure 4 depict the average access delay of a data package. As expected, the delay is high even for low loads in a 50 Mb/s channel without synchronous traffic. Here, the collision slot time is a large percentage of the time it takes to send a packet so collisions become very costly. The 10 Mb/s channel with synchronous traffic uses a framed Ethernet, where framing is necessary in order to provide guaranteed access for the synchronous synchronous information. The cost of incorporating synchronous traffic is significant, since the framing creates a point, located immediately after the voice/video frame terminates, where the probability of collisions is high.

In contrast, DRAMA effectively separates the synchronous effects caused by framing so that data packets generally have immediate access to at least some bands. In addition, DRAMA provides for efficient recovery of that portion of the frame which is not used

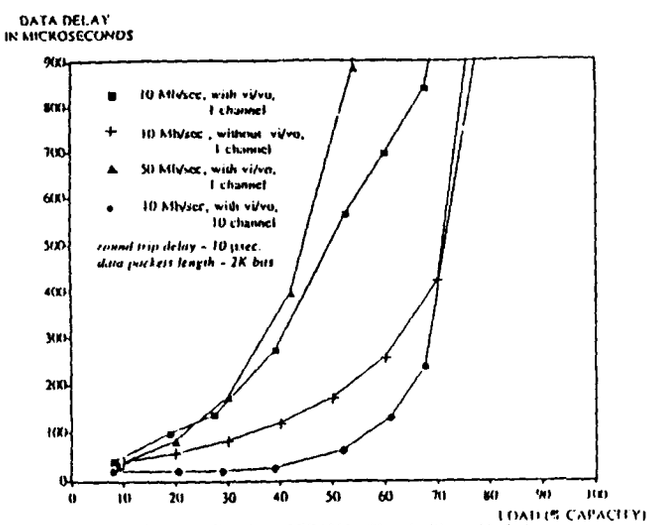


FIGURE 4 : One channel DRAMA with and without video/voice compared with multichannel DRAMA

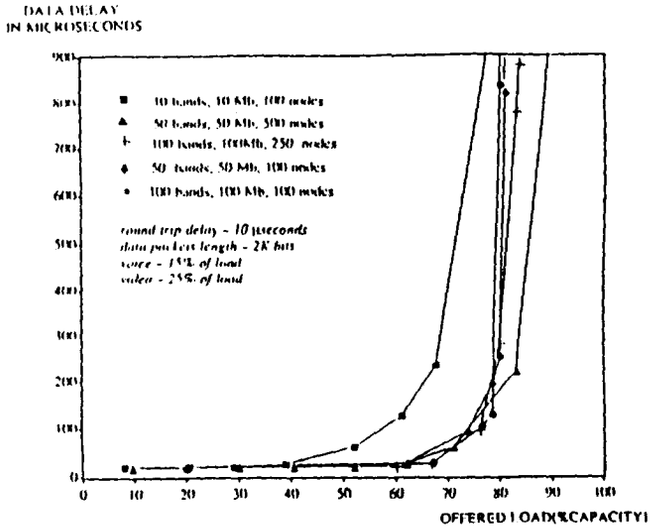


FIGURE 7: Scaling of network by bandwidth, number of bands and nodes

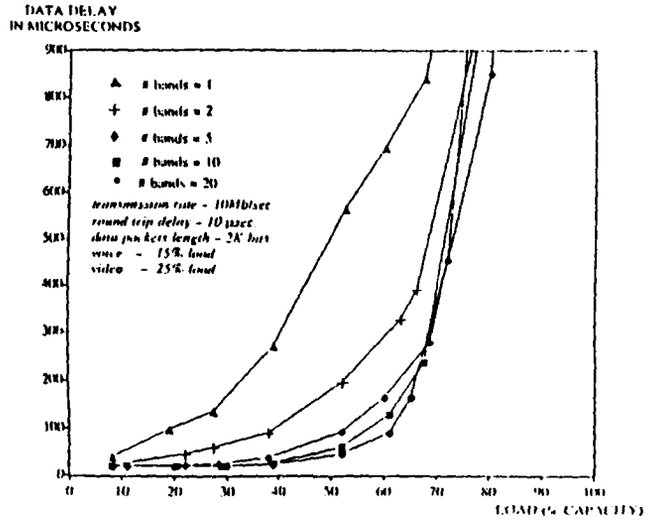


FIGURE 5: Multiple channel versus single channel

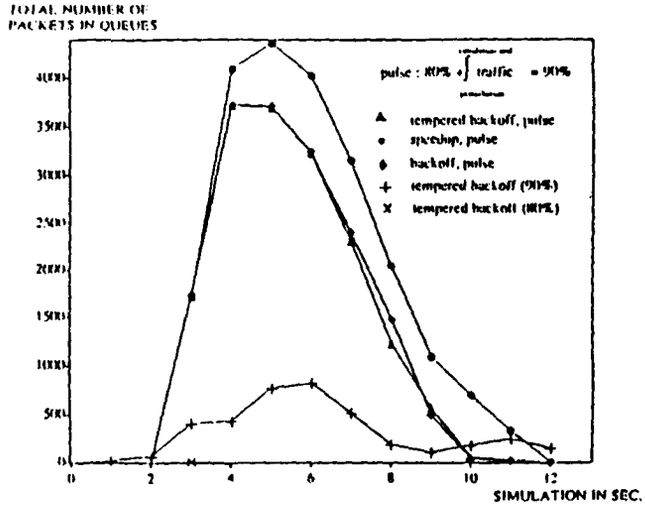


FIGURE 8: Reactions of various algorithms to strong overload at 80% of nodes

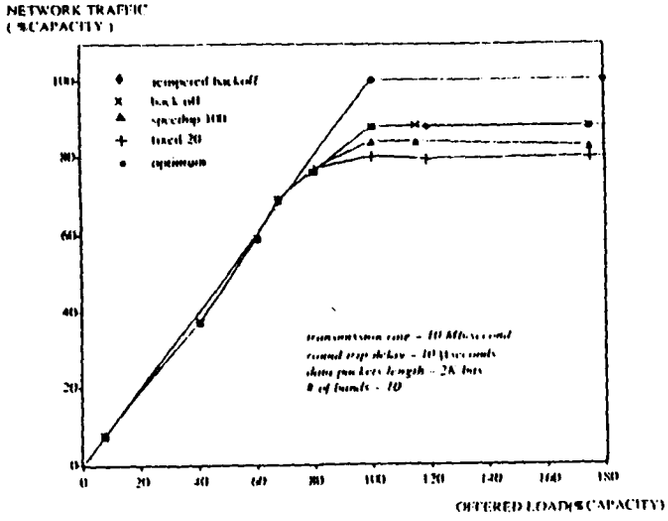


FIGURE 6: Network throughput for various traffic placement algorithms

for synchronous messages.

Figure 5 shows the effect of changing the number of bands assigned to a LANC. For instance, with two channels of 5 Mb/s each (of which 2 Mb/s of load are due to voice and video), we have approximately the same delay as on one 10 Mb/s channel without synchronous traffic (see Figure 4). For the LANC with 100 nodes, ten 1 Mb/s bands appear to be a good compromise, since delay does not improve greatly after that point. There is clearly an optimal number of bands; as the bandwidth per band gets smaller, the negative impact of both framing, as we shall see below, and delivery delay become more evident.

Note that we are comparing access delays only. In the overall performance of a network, additional transport delay (5 μsec/km) and bandwidth delay (bits/sec) affect the arrival of the complete packet at its destination. Thus, a 2k packet at 10 Mb/s would take 205 μsec to arrive at a station 1 km from the source, while at 1 Mb/s it would take 2050 μsec. In fact, one can conclude that single-band CSMA/CD is always superior in overall delay to multi-band CSMA/CD if all traffic is asynchronous. Hence, the price one pays in order to have an integrated CSMA/CD that can handle voice, video and data effectively is to accept this increase in delivery delay. However, a 2 msec transmission delay for a 2K packet, while it may be serious in some high-performance distributed systems, is still considerably less that the software delays that occur in many higher-level communications protocols [2].

Figure 6 illustrates the stability of the DRAMA protocol to handle high-load conditions without choking. It compares the traffic placement policies described in Section 4 and demonstrates that all the policies are effective when traffic overloads occur on the network; however, *backoff* and *tempered backoff* work somewhat better at high loads. Up to about 75% of capacity, practically all traffic on the net is message traffic with a negligible (<1%) amount of noise. At about a 90% load the message traffic levels off and stays constant, independent of the amount of traffic offered. The last data point we measured is with an offered load of 185% of capacity. The amount of noise, i.e. collision, is about 5% and the additional capacity wasted is between 5% and 15% depending upon the placement policy.

Table 1 presents the data shown in Figure 6 in order to compare the traffic placement policies. Up to 80% load we see that all policies provide approximately the same throughput. Above that point, the policies with *backoff* show a slight improvement over *speedup* and *fixed*. Delays are a little better for the *backoff* policies starting about 60%, and show more improvement as load increases. Thus, policies with *backoff* should be used for traffic placement in the DRAMA protocol.

An interesting – and not totally unexpected – property is revealed in Figure 7. As we scale up from 10 bands with 100 nodes to 50 and 500, respectively, the performance improves. The increase in available bands allows further separation between synchronous and asynchronous traffic so that the latter has more bands to choose among. At the upper node and band count, little deterioration in access delay occurs up to 80% offered load. Most other protocols fail to reach this level of performance and most get worse as the node count increases.

One important question discussed in Section 3 is how DRAMA handles a combination of localized, bursty overload traffic. Figure 8 shows how quickly the system reacts under the various traffic placement algorithms in order to clear the nodes' queues. The total number of packets in the system is shown at various simulation times for three situations: (1) with 80% offered traffic for *tempered backoff*; (2) with 90% offered traffic for *tempered backoff*; and (3) at 2 seconds in the simulation of an 80% run we offer an overload to 10% of the nodes. The overload consists of as many packets as would have been needed to raise the total offering to 90% over the entire simulation period. The network response to this severe transient is good in that packet queues return to previous levels within 6 to 10 seconds.

The DRAMA protocol shows a marked difference from normal CSMA/CD behavior in the effect of round trip delay (potential collision slot time). Here, performance is not the same for different values of transmission time and round trip delay even though the ratio of the two is the same. Figure 9 shows that only at higher values (approximately 20 km length) does the round trip delay decisively increase the access delay. Two factors in DRAMA that do not exist in single-band CSMA/CD help to reduce the effect of round trip delay and hence improve the LAN spanning distance. First, with the reduced bandwidth the effect of a collision is considerably lessened, since the collision slot time is a much smaller percentage of the packet transmission time. Second, the probability of collisions is further reduced since, with multiple bands, a collision occurs only if two packets arrive within the slot and select the same empty band. Thus, besides extreme flexibility in configuring a network as noted above, DRAMA provides the additional feature that LANs can span both much greater and widely differing distances.

In our experiments, we have seen a major impact on performance by virtue of DRAMA's frame structure. In these runs the frame structure has been enforced so

	7.5%	40%	60%	67.5%	80%	100%
Speedup	20	30	135	315	1749	1167949
Backoff	20	28	123	284	1909	713018
Fixed	20	28	127	311	2277	1668377
Tempered Backoff	20	28	126	305	2050	897411

Table 1: Average Delay versus Load

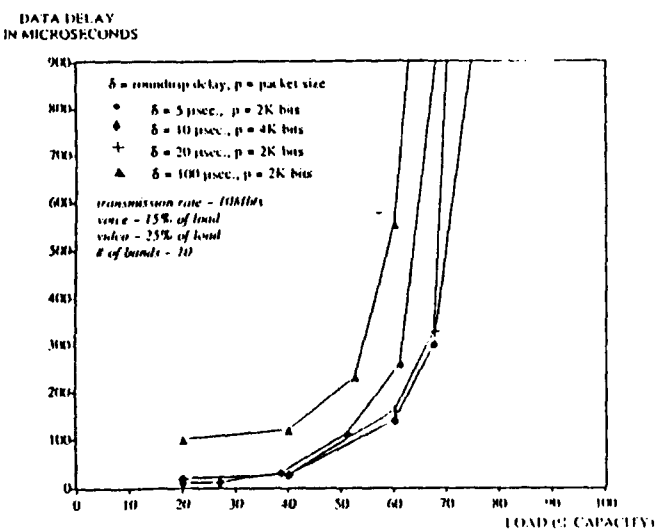


FIGURE 9: Data transmission delay versus load with varied normalized to 2Kb packet size and round trip delay

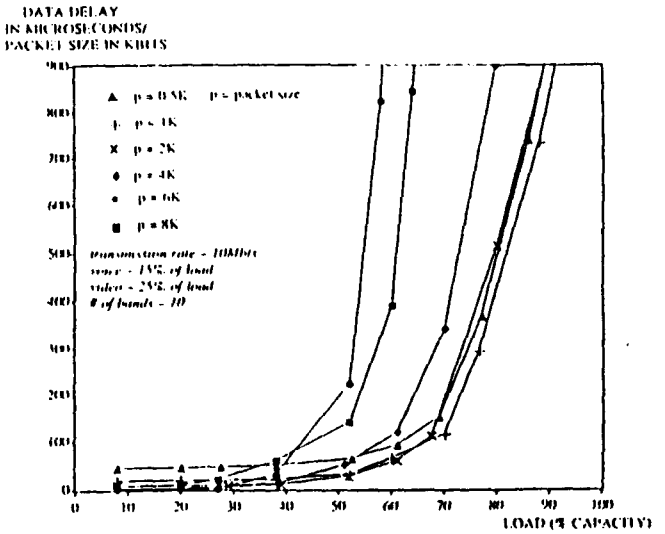


FIGURE 11: Data transmission delay versus load with normalized packet size

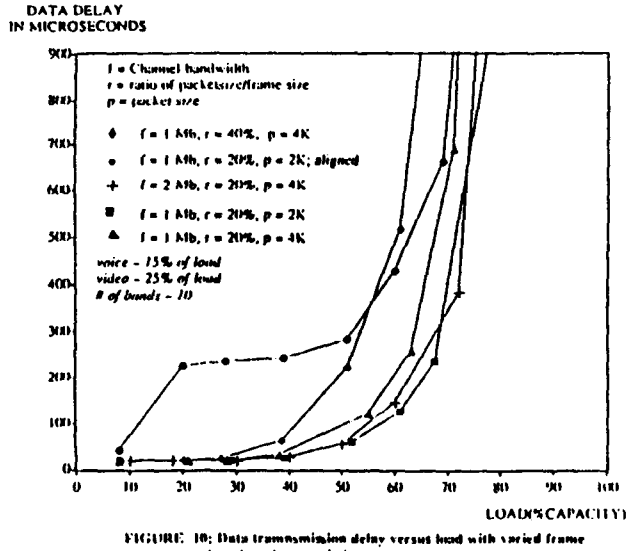


FIGURE 10: Data transmission delay versus load with varied frame length and transmission rate

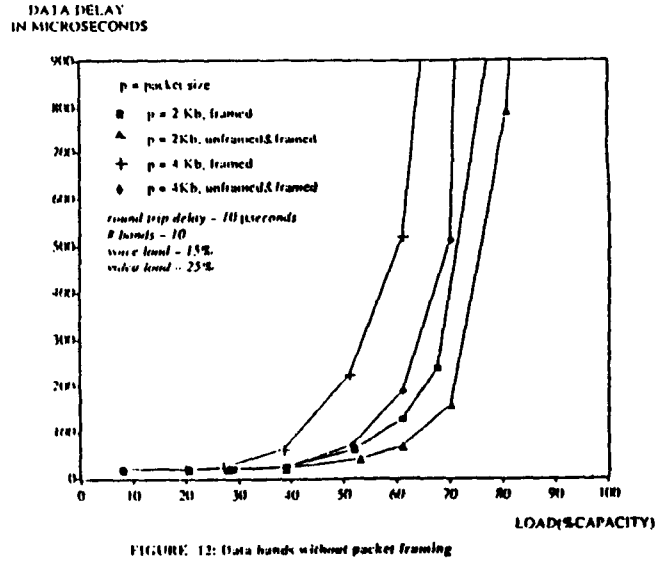


FIGURE 12: Data bands without packet framing

that packets which are too large to fit at the end of a frame are denied access until after the frame mark has occurred. This can cause additional delays especially as load increases. Figures 10 -12 examine the relation between frame size, packet size, transmission rate, and frame begin alignment. In order for packets to fit at the end of a frame they should be small in comparison with the frame length. Figure 10 shows that if we keep packet to frame ratios to 20% then the delay performance is relatively stable under changes in band structure or frame size. However, if we increase this ratio to 40%, delay at higher loads is noticeably increased. Further, if we force all packets to be aligned on frame boundaries performance even at low loads is much poorer. Figure 11 illustrates the problem by change in packet length from .5K to 8K bits. In this figure, the delay has been normalized to a 2K packet delay since one is required to send more packets at lower packet size in order to transmit the same information. Again, we see that for ratios 20% or less the performance is relatively constant, but above that delay increases appreciably. Note that at 6K and 8K packet sizes it is possible to fit only one packet into a frame.

Finally, Figure 12 shows that performance can be increased significantly by changing the protocol algorithm slightly. In this modification each bandleader elides the frame begin marker whenever no voice/video calls are assigned to the band. Essentially, this makes the protocol unframed CSMA/CD for data until voice/video calls are placed on the band. Dynamic framing with overflow and timing recovery on those bands which have synchronous traffic assigned, similar to the FDDI overflow system [1,5], would provide further recovery of wasted capacity.

These studies also addressed a concern left from our earlier study [11]. That study focused on reallocation of bands among LANGs and assumed that the allocation could be "easily" done; this will not be true if synchronous traffic tends to spread across most bands allocated to a LANG. This study shows that, using the DRAMA protocol, data remains reasonably well concentrated on a small number of bands. Even under high loads (e.g. 100%), in each experiment we ran, some channels never carried any synchronous data throughout the run. This allows the LANG to release some bands if network conditions require reallocation.

Fairness is another metric for a local area network. In the DRAMA system there are two considerations of fairness: (1) that across the whole network all LANG have equal access delay, and (2) that within a LANG each node have equal access delay. The first fairness factor, across the network was reported in [11] and showed equal LANG access. The second fairness factor was measured in this simulation study for the individual nodes. Statistics were computed to observe mean delay and standard deviation of delay about the mean for each node in the network. In 90% of all cases observed, the means tended to be equal for all nodes and the standard deviation of the delay was always less than the mean. In the remaining 10% of the cases, the standard deviation never exceeded 125% of the mean. We note that beyond 100% load these statistics become meaningless since the queues grow unbounded. This indicates that nodes are being fairly treated and that no node is being denied access excessively due to some quirk in the protocol or node location.

7 Conclusions

The analysis of the DRAMA protocol reported here and in [11] indicates that the protocol has several important features. The most impressive characteristic is its extreme flexibility. A single metropolitan area network using the DRAMA protocol can support:

- LANGs with very different numbers of nodes,
- LANGs spread across a wide geographic area,
- Dramatic fluctuations of load, and
- Widely varying mixtures of traffic types.

The protocol provides this flexibility since:

- A network can quickly rebalance loads by reallocating bands among the LANGs (taking on the order of 30 to 300 ms),
- The protocol effectively integrates voice, video and data on a CSMA/CD network,
- The traffic placement policies work well with dynamic resource allocation since the number of bands with synchronous traffic is kept to a minimum, and
- The network is stable even at very heavy loads and with momentary overloads at some nodes.

These studies also suggest that, even with a limited bandwidth (say 10 Mb/s), use of a multiband network rather than a single band (say ten 1 Mb/s bands versus one 10 Mb/s band) can significantly lower average access delay (though transmission time for large packets on a 1 Mb band will be longer). In addition the multiband approach allows integration of synchronous and asynchronous traffic.

As a result of the these studies we have additional encouraging performance information on the DRAMA protocol; we know that both *backoff* and *tempered backoff* provide satisfactory traffic placement.

An area that needs further research with the DRAMA protocol is the problem of providing tunable transmitter hardware for each node so that a node is able to transmit at various bands and enough receivers so that the node can receive voice/video and data packets sent to it in a random fashion. Further study is underway to assess the feasibility of hardware to accomplish the reception function without have rapidly tunable receivers to detect when information is arriving for a particular node. We are also studying buffering mechanisms to handle the problem of a single node receiving a large influx of information on several bands simultaneously.

Certainly the DRAMA protocol provides one alternative for supporting the multitraffic-type flexible gigabit networks which are becoming progressively more important. We are now in the process of developing a simulation testbed with the capability of generating comparative performance data for protocols such as DRAMA, FDDI-II, QPSX, CSMA/RN [7], and tree-structured MAN protocols.

References

[1] FDDI token ring media access control (MAC). *ANSI Standards Document ASC X3T9.5 Rev. 10*, February 28, 1986.

[2] T. Mueller, B. Bhargava, and J. Riedl. Experimental analysis of layered Ethernet software. *Proceedings of the ACM-IEEE Computer Society 1987 Fall Joint Computer Conference*, 559-568, October 1987.

[3] P.T. Brady. A technique for investigating on-off patterns of speech. *Bell System Technical Journal*, 44, January 1965.

[4] Imrich Chlamtac and Aura Ganz. A multibus train communication (AMTRAC) architecture for high-speed fiber optic networks. *IEEE Transactions on Communications*, 6(6):903-912, July 1988.

[5] Doug Dykeman and Werner Bux. Analysis and tuning of the fddi media access control protocol. *IEEE Transactions on Communications*, 6(6):997-1010, July 1988.

[6] G. Ennis and P. Filice. Overview of a broad-band local area network protocol architecture. *IEEE Journal on Selected Areas in Communications*, SAC-1, No.5:832-841, November 1983.

[7] E.C. Foudriat, D. Game, L. Founds, K.J. Maly, R. Mukkamala, and C.M. Overstreet. A carrier sensed multiple access ring for gigabit networks. *Technical Report TR-89-9, Dept. of Computer Science, Old Dominion University*, February 1989.

[8] T.B Fowler and S.F. Holmgren. A wideband cable bus local area network. *COMPCON*, 405-414, Fall 1982.

[9] M. Hatamian and E.G. Bowen. Homenet: a broadband voice/data/video network on a CATV system. *AT&T Technical Journal*, 64:347-367, February 1985.

[10] G. Exley, L. Merakos, and C. Bisdikian. Interconnection of CSMA local area networks: the frequency division approach. *IEEE Transactions on Communications*, 730-738, July 1987.

[11] Kurt Maly, C. Michael Overstreet, Xia-Ping Qiu, and Deqing Tang. Dynamic resource allocation in a metropolitan area network. *Proceedings, SIGCOMM Symposium*, 13-24, August 1988.

[12] M.A. Marsan and D. Roffinella. Multichannel local area network protocols. *IEEE J-SAIC*, 885-897, Nov. 1983.

[13] A.N. Netravali and Z.L. Budrikis. A broadband local area network. *Bell System Technical Journal*, 64, no. 10:2449-2465, December 1985.

[14] M.W. Rahm. Cablenet: a user perspective. *COMPCON*, 337-342, Fall 1982.

[15] S. Sharrock, K. Maly, S. Ghanta, and H. Du. A broadband integrated voice/data/video network of multiple LANs with dynamic bandwidth partitioning. *Proceedings, INFOCOM '87*, 417-425, March 1987.

[16] S. Sharrock, K. Maly, S. Ghanta, and H. Du. A framed, movable-boundary protocol for integrated voice/data in a LAN. *Proceedings, SIGCOM '86*, 9 Pages, August 1986.

[17] P.C. Wong and T.S. Yum. Design and analysis of a contention - based lookahead reservation protocol on a multichannel local area network. *IEEE Transactions on Communications*, 36, no.2:234-238, Feb. 1988.

N90-22280

CSMA/RN - A Universal Protocol for Gigabit Networks

A White Paper

by

E. C. Foudriat, K. Maly, C.M. Overstreet, S. Khanna and F. Paterra
Old Dominion University
Norfolk, VA 23529
Fri, May 4, 1990

1.0 Introduction

Networks must provide intelligent access for nodes to share the communications resources. During the last eight years, more than sixty different media access protocols for networks operating in the range of 50 to 5000 Mbps have been reported[1]. At 100 Mbps and above, most local area [LAN] and metropolitan area networks [MAN] use optical media because of the signal attenuation advantage and the higher data rate capability. Because of the inability to construct low loss taps other than star couplers, fiber optics systems are usually point-to-point.

In the range of 100 Mbps - 1Gbps, the demand access class of protocols have been studied extensively. Many use some form of slot or reservation system and many the concept of "attempt and defer" to determine the presence or absence of incoming information. Local sensing of the existence of information is used in slot, train and reservation systems such as Cambridge Ring [2], Expressnet and Fastnet [3], and DQDB (formerly QPSX) [4]. In slotted systems, long messages must be broken into slot size proportions contributing to wasted network capacity, since the slot size selected is always a compromise over the wide range of integrated (voice, video and data) traffic that high data rate networks must carry. Also, recent studies indicate that reservation systems have fairness difficulties when servicing nodes at the ends of the bus under high load conditions[5]. Other demand access systems may use a token, like FDDI, but waiting for the token to rotate can cause slow access especially in longer and higher data rate rings. In addition, most demand access systems use a master controller mechanism, like in FDDI II, for handling synchronous traffic [6, 18].

The random access class of protocols like shared channel systems (Ethernet), also use the concept of "attempt and defer" in the form of carrier sensing to alleviate the damaging effects of collisions. In CSMA/CD, the sensing of interference is on a global basis. However, as bandwidth increases, a message spans a smaller portion of the global bus length so network collisions can reduce throughput significantly, especially at higher load [7]. This, coupled with the fact that optical broadcast systems have a difficult time building effective low loss taps, makes global sensing impractical for high speed networks.

Some systems have used a delay line [8] or a buffer, like the register-insertion system [9, 10], for alleviating the corruption of data because of simultaneous access. The tree LAN system [8] uses "attempt and defer", while the register-insertion system uses "attempt and defer" or "attempt and hold" under full or empty buffer conditions, respectively. Finally, a hybrid system [11] uses "attempt and abort" under some conditions but reverts to a master controller at high loads when aborting begins to waste needed network capacity.

All systems discussed above have one aspect in common, they examine activity on the network either locally or globally and react in an "attempt & whatever" mechanism. Of the "attempt + " mechanisms discussed, one is obviously missing; that is "attempt & truncate". As noted above, the amount of space occupied by a packet decreases as network rate increases. For example, at 100 Mbps, a 2K bit packet occupies a space of approximately 4 km along the network ring; at 1 Gbps, this space is reduced to 0.4 km. Thus, a 1 Gbps, 10 km network can potentially have 25 separate 2K bit packets simultaneously in existence over its span.[1] Thus, as bit capacity of the network increases with data rate, it would seem reasonable that, at least for some load conditions, truncating a message when it is about to interfere with a message on the system and resuming it later when free space is available would be a reasonable access protocol to consider.

"Attempt and truncate" has been studied in a ring configuration called the Carrier Sensed Multiple Access Ring Network (CSMA/RN). In this paper, we will describe the system features of CSMA/RN including a discussion of the node operations for inserting and removing messages and for handling integrated traffic. We will then discuss the performance and operational features based on analytical and simulation studies which indicate that CSMA/RN is a useful and adaptable protocol over a wide range of network conditions. Finally, we will outline the research and development activities necessary to demonstrate and realize the potential of CSMA/RN as a universal, gigabit network protocol.

2.0 Carrier Sensing and Control in Ring Networks

Local carrier sensing and collision avoidance is used in all "attempt & whatever" mechanizations. It has been implemented using a delay line for a tree LAN optical network operating in the Gbps range[8]. This network has a number of receiving links and a transmitting link at the node points of the tree. Each receiving link can have an incoming signal but only one outgoing signal can be propagated. The key to sensing selection is based on a delay line that gives the selection switch advanced warning of the incoming signal and hence, a chance to exercise intelligence to select a single receiving line and avoid a collision before the signals arrives. This same form of advanced information detection and control is the key to the CSMA/RN operation.

2.1 Basic Operation

Figure 1 illustrates the characteristics of a node in the carrier sensed ring network. The incoming signal is split into two streams, one through a delay line or buffer. Note, the delay can be relatively short if high speed logic is used in the controller system. For example, a 100 bit delay at 1 Gbps is approximately a 20 meter piece of fiber and causes a 100 nanosecond delay. The node controller, based upon information accumulated, is required to make a number of decisions. First, it must detect the presence of incoming data; if its exists, the node must always propagate incoming information as the outgoing signal to the next node on the ring because it would be impossible to recreate the packet unless a much larger storage system is provided [9]. If no incoming packet exists, the node is free to place its own data on the ring if its queue is not empty. However, during the time this latter data is being transmitted, if an incoming packet arrives, then the node, within the time limits dictated by its delay size, must discontinue its transmission and handle the incoming packet. When truncating a packet, the node can place a terminator block at

[1] Some demand access systems realize this sharing of physical network space by having multiple trains or slots distributed over the network length.

the end to assist the receiving node with re-accumulating a fractured message.

Packets are tested at each node to determine if the incoming packet is destined for this node and should be copied to its receiving data buffer (not shown in Figure 1). Since address decisions are required by the controller, packets are nominally removed at the destination, since, as discussed in Section 4, destination removal increases network capacity significantly. As a protection against packets circulating continuously in case of node failure or address errors, packets are removed by the source after one or more rotations by comparing addresses against an established list.

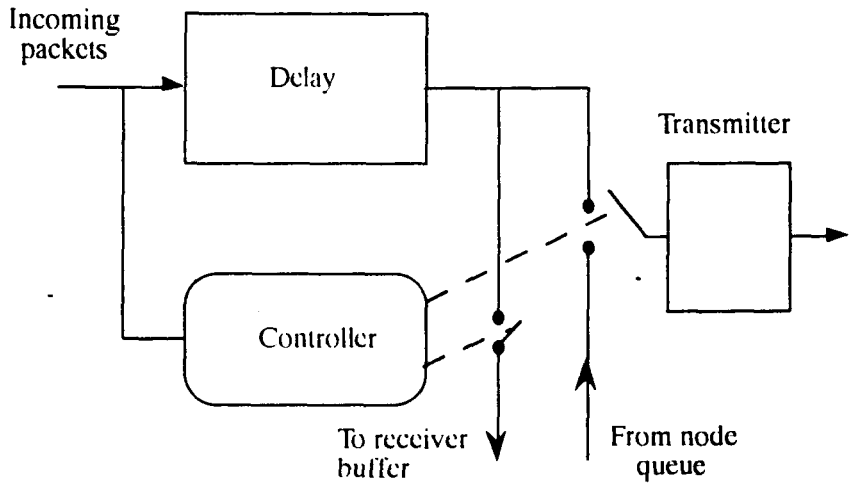


Figure 1 CSMA/RN Access Controller Logic

2.2 Synchronous Traffic Operation and Guaranteed Access

A most difficult problem for random access protocol systems occurs when integrated traffic and guaranteed access are required. In CSMA/RN, we have solved both problems by the use of a circulating packet reservation (CRP) system[2] which is similar to the concept we developed in DRAMA [12].

In the reservation system, a special small packet, about 100 bits, circulates continuously around the ring for nodes having synchronous traffic or requiring guaranteed access. The synchronous traffic is attached to this packet. To set up a call, the node informs the rest of the network of the bandwidth needed and the expected inter-arrival time of messages. After checking that this additional call does not require any parameter changes to the protocol, the node proceeds to send messages as follows. On the cycle prior to the node's need to submit synchronous traffic, the node through the packet requests that a block of space following the packet be freed for its use on the next cycle. The node is free to set this indicator whenever it has synchronous traffic and the reservation packages is *free*. When the nodes makes the reservation, it also sets the flag to *busy*; after a complete cycle the node changes the flag to *free* and sends its synchronous message(s) in the free space behind the CRP. In most cases, the capacity reduction due to a circulating reservation packet is very small so that the net can have one or more packets depending node count and ring length. The separation of circulating reservation packets will

[2]We have not considered implementing an isochronous traffic system as this would requrire a 125 μsec. cyclic control. Our intention is not to compete with the telephone network to handle virtual circuits by framed time-division multiplexing but to integrate periodic traffic into a multi-area, gigabit data network.

limit the size of regular data packets so the number of reservation packets on the system should be regulated. The block size requested by a circulating packet depends upon the synchronous traffic bandwidth required and the inter-arrival time of the useable circulating packets.

As the CRP request circulates around the network, the space is freed as the packets arrive at their designated destination. However, this space can be used further if the destination node or any subsequent node has a message for any node between and up to the node controlling the circulating packet. In addition, the node controlling the circulating packet does not need to request space to a greater extent than needed. For example, a telephone call which goes into the non-talk phase, no reserved space or only a minimal block would be required for that cycle. Hence, synchronous data block space is used only as needed and unused space does not need to be recovered. As a result, the CRP system and handling of synchronous traffic should cause minimal impact on the CSMA/RN network system.

Guaranteed access works in conjunction with the CRP system. Access is guaranteed one revolution after receipt of the free circulating packet. To guarantee that nodes have access to free circulating packets requires that a node cannot use a circulating packet on a number of successive revolutions. By passing it packets on, every node is will get to see a free circulating packet within a fixed number of ring revolutions. Access time depends upon the conditions established for network operations based upon the number of nodes, the number of circulating reservation packets and the ring length. If the network can not guarantee the access time specified with the one reservation packet, it can introduce another one into the ring. This will impose a new limit on the maximum message length but will lower the current worst case access time by half after the same synchronous traffic is redistributed equally. Alternatively, if circulating reservation packets are not being used, they can be eliminated from the system if access times are acceptable.

2.3 Fairness

Fairness in any basic CSMA system can be a problem since access is based upon statistical probability. Fairness problems are most likely to occur when a node up stream has a long or a lot of messages to send and fills all the packets, or a node down stream is sent many messages by a nodes up stream. For example, when we have non-uniform load such as a node being a file server or a bridge which both sends and receives more than other nodes, actual starvation at either side of this node can occur.

To solve the fairness problem, we propose to use a scheme first investigated for DRAMA [12]. In DRAMA, a multichannel protocol, a small channel was set aside for transmitting global information which among other things transmitted network averages of all channels to all nodes. It is more efficient to do so than to have each node monitor every channel since at any time a node participated only on a few channels. Nodes use this information to adjust their bandwidth usage so that every node would experience approximately the same access delay. That is, nodes with lots of information to send obtained lots of bandwidth and those with little got little bandwidth assigned. The algorithm was totally distributed and was able to totally reallocate bandwidth upon strong disturbances within 30 msec.

CSMA/RN is not a multichannel protocol, but, analogous to the global communications channel in DRAMA, we can reserve a small, 100 bit, circulating fairness control message in CSMA/RN. The information in this fairness message is the current network throughput averaged over all nodes and the average wait time per message, call it nm. That is, it is the average of all nodes

perception of the traffic on the network - remember that, in general, no node will see the same amount of traffic pass by because messages are taken off at the destination.

To keep *ntn* current each node keeps two copies - one from the last cycle and one from the current cycle. When the fairness control message comes around, the value is replaced by the one the node has calculated during the last cycle. The value which was taken off is used to calculate the value for the next cycle as follows:

$$ntn = ntn - last/n + current/n$$

where n is the number of active nodes. If a node finds that its current throughput is more than the $p\%$ of global-node-throughput, and its wait-time per message is less than $q\%$ of the network wait-time per message, the node is forced to wait for x Kbits hole in that ring cycle. With this control, a hyperactive node is forced to abstain from sending messages on the ring and giving other nodes a chance to send their messages.

3.0 Performance of CSMA/RN

Both analytical and simulation performance studies of CSMA/RN have been conducted. The analysis results show that a simple queuing theory model displays excellent correlation with the simulation results. The model assumes each node to be statistically independent with its message service time a function of the probability of the arrival of a free message block based upon the network load. Under these conditions, the queuing theory model is basically M/G/1 so that the wait time for messages in the queue can be found directly from the Pollaczek-Kintchine analysis [13]. Once the message has been serviced and is placed on the network, the travel time from source to destination is fixed by the network propagation speed. Hence, the response time for a network, which is the sum of the wait, service and travel times, is easily obtained.

The major factor in obtaining accurate analytical results for the above model is a good estimate of the arrival of free message blocks so that service time results are accurate. Two models were developed which give probability of a free packet based upon load factor. One ignores and the other models the effect of packet size based upon packet fracture results obtained from the simulator. Interestingly, the former model was found to provide better results because, although packets may fracture, the smaller packets arrive more frequently, so the net result is that the service time is approximately the same, independent of whether many small or a few large empty blocks arrive.

The simulation was built to study the parametric aspects of CSMA/RN which are not modelled in the analysis. During the simulation, runs were made to examine the statistical properties of the results so that run-times would provide accurate data. General conditions for the initial simulator runs include:

(1) packets were removed at the destination and the empty space used by the node to send queued messages;
(2) additional header bits required because of packet fracture were not added to the message,
(3) nodes are uniformly spaced around the ring;
(4) all message arrivals are uniformly distributed among the nodes;
(5) all message destination addresses are uniformly distribute among the nodes other than the source node; and

(6) all messages are fixed length.
Additional runs were made where conditions 2) and/or 6) have been removed.

3.1 General CSMA/RN Performance[3]

The simulation results for a 10km, 10 node ring, Figures 2 and 3, indicate that under nominal conditions CSMA/RN provides excellent performance as an access protocol. First, access or wait time approaches zero at no load and remains relatively flat until the load approaches 140% of the network load. As load increases, wait time, which is dependent upon service time, becomes unstable, nominally at loads > 200%. Service time remains close to the minimal, no load service time throughout most of the load range: it remains within a factor of 2 for load levels up to 120% network load and with a factor of 4 for loads up to 200%. Finally, since travel time for a message on the ring is fixed by the media propagation speed, the total response time in MAN and larger LAN networks is mainly dependent upon the source to destination length. In any case, the CSMA/RN access protocol does not slow the travel time, so that a message, once on the network will move as quickly as possible to the destination.

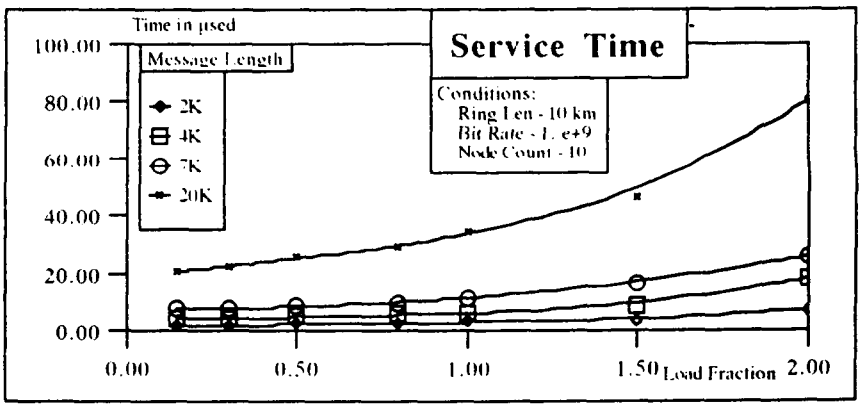


Figure 2 Service Time for CSMA/RN Access Protocol

As mentioned previously, the performance of the CSMA/RN is improved by using destination removal of messages. This is immediately apparent from Figures 2 and 3 ---the 1Gbps network is capable of handling up to 1.75 Gbps (175% load factor) without saturating, because, on an average, messages travel only half way around the ring. Thus, load performance for CSMA/RN and other destination removal networks systems, like register-insertion [9, 10], can double the basic net capacity.

In the initial simulator studies of CSMA/RN, runs were made varying a number of conditions. Node counts were varied from 10 - 200 nodes, ring lengths from 2km - 10000 km, and message lengths from 2K - 2 Mbits. In all cases, CSMA/RN performance was considered to be excellent and to correlate closely to the expected results from the analytical studies up to the maximum load factor of 200% (2 Gbps).

Additional features were examined using the simulator system. Message fractures were determined for all runs. In most cases, mean message fracture ratio was below 2.5 for all conditions above when load factors was less than 150% and usually below 4 for loads up to 200%. The maximum mean message fracture was noted for high node counts (short inter-node

[3]Details on CSMA performance studies can be found in [15]

distances of 0.25km) where mean message fracture was 7 at 200%. Simulator runs made with overhead added for message fracture (condition 2 above removed) showed a small increase in wait, service and response times and message fractures for load conditions up to 175% .

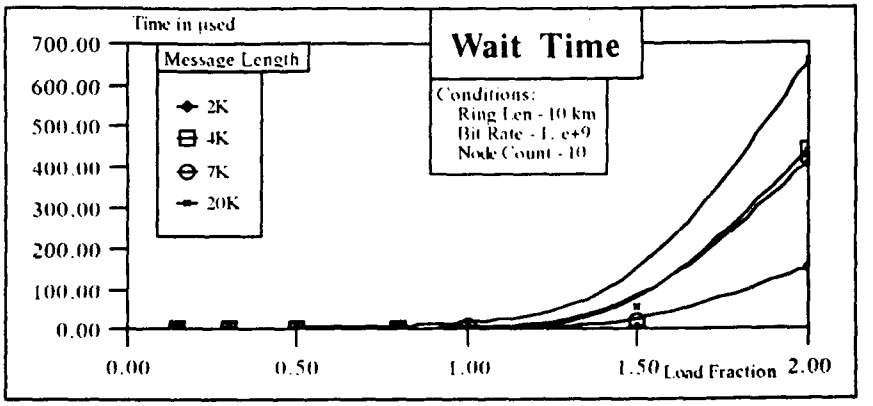


Figure 3 Wait Time in Queue for CSMA/RN Access Protocol

Simulator runs were made to test throughput at overload conditions. At 250% input load, the network delivered over 190% of capacity. Simulator runs were made at lower network data rates to determine whether CSMA/RN is an effective protocol in the megabit range. The results showed that protocol performance was good and as predicted by the analytical model at data rates to 100 Mbps. We anticipate that this limit can be decreased for longer length networks. Additional simulator runs have been conducted with random message sizes ranging from 2000 - 6000 bits. The results show no significant increase in service time or packet fracture but some increase in wait times at the higher load fractions, as predicted by the Pollaczek-Kintchine formulation of the analytical model, due to the increase variance in the service time.

Finally, a scaling factor suggested by the simulator model has been shown to be accurate in predicting CSMA/RN performance for wide area ring networks. In scaling, the ratio of network length to message length is fixed and the number of nodes remains constant. Simulator runs were made for 4 conditions up to 10000 km and 2 Mbit message lengths and compared to performance based upon scaling up from 10 km and 2 Kbits. Comparisons showed results were almost identical. Thus, one can predict performance of WANs to be the same as those from scaled LANs with the exception that travel time once the message is on the network will be greater. Using these scaling conditions, CSMA/RN is shown to provide an excellent access protocol for a National Research and Education Network[14].

3.2 Synchronous Traffic Performance

Simulation runs have been made with the above model to test the impact of the circulating reservation system's circulating packets on the networks asynchronous performance. As noted previously, the CRPs will limit the maximum size that a normal packet can have when the CRPs are space uniformly along the network. Tests were made on a 10 km, 10 node ring with 4 Kbit messages. Each circulating packet was 100 bits long. Each circulating packet will reduce the network capacity by 0.2%. More important, each circulating packet will recur at a node every 50 µsec. Tests were run with 1, 2, and 5 circulating packets. Five circulating

packets limit the maximum size of message blocks to 10 Kbits and CRP inter-arrival times to 10 μsec.

The results of the runs showed that the maximum impact of the the circulating packets was to increase the message fracture, especially at low loads. Here, 5 CRPs produced a fracture ratio of 1.9 packets per message at 30% load, for 1 CRP, it was 1.21 and with no CRPs the fracture was 1.15. The service time at low loads was also increased by not nearly as significantly. At high loads, the circulating packets did not have as great an effect since packets already tend to be quite fractured and the interruptions by the circulating packets made minimal additions. The results indicate that, for nominal circulating packet inter-arrival times of 50 - 200 μsec., CRPs should not nave a significant effect on the data traffic that the network is carrying. In addition, we have analyzed a group of synchronous traffic scenarios. In all cases, the maximum guaranteed access was less than 1 msec. and in rare cases the maximum message length was reduced to 10 Kbits.

3.3 Fairness Tests[4]

The CSMA/RN simulator presently available does not model non uniform and highly variable traffic from a node. However, some examples of the fairness problem were observed and initial tests were made on the fairness control model to determine its effectiveness. At 200% load four nodes suffer severe starvation, sending from 166% to 176% and at 225% load nodes, twelve nodes had between 180%-190% and five nodes had below 180% throughput. Since the results are based upon random distribution, conditions will vary over different runs and over different intervals in a run.

We have run experiments with a network in which a specific node is starving for a certain period of time under a nominal load condition of 250%. During a 30 msec. duration of starvation, the node's waiting time increases significantly from 425 msec. to 1038 msec. In this specific case, the node is unable to sent its messages even at expense of higher wait-times per message, mainly due to the fact that the node does not receive many messages and hence, does not see empty packets on the ring. In such an overloaded net the bus will be practically busy all along its length, so a node gets 75% to 98% chances of sending from being able to take off messages addressed to it and thereby creating a hole to send its message. We have implemented the fairness control scheme described in section 2.4 with p and q being 20% each. Now the same node as in previous case starves only for 20 msec. The wait-time is still high because of the overload situation being experimented. Most significant, however, the node is able to send 86 messages in a 1 msec. period with the fairness control enforced; without fairness control the node only sent 53 messages in the identical period. This test indicates that the fairness control system we have adapted from DRAMA has the potential of solving node starvation in the CSMA/RN system.

3.4 Comparison to Metropolitan Networks

CSMA/RN, by its basic operating premise, is a protocol which works better at higher speed and longer length networks. Although 100 Mbps is at the low end of effectiveness for CSMA/RN, we feel it useful to give some calibration of performance by comparing it with well known high-speed protocols such as FDDI and DQDB. We have already shown earlier that CSMA/RN

[4]Further results on fairness can be found in [5]

exhibits the very low wait time at lower loads due to the nature of CSMA-type protocols and that the instability point is well beyond 150% offered load. In Figure 4, we have plotted the wait times of FDDI, DQDB, and various forms of CSMA/RN for the conditions of a 50 km, 50 node network with uniformly distributed offered load. The legend indicates the basic data rate that a particular versions can handle; that is, *CSMA 0.1* means that a node can send up to 100 Mbps. Figure 4 shows that *CSMA 0.1* outperforms FDDI significantly although circuit speed is comparable. *CSMA 0.15* ,which has the same circuit speed as DQDB but one bus as compared to DQDB's two buses, performs equal to DQDB at high loads and better at lower loads. Beyond *CSMA 0.15*, all versions are better than either FDDI or DQDB.

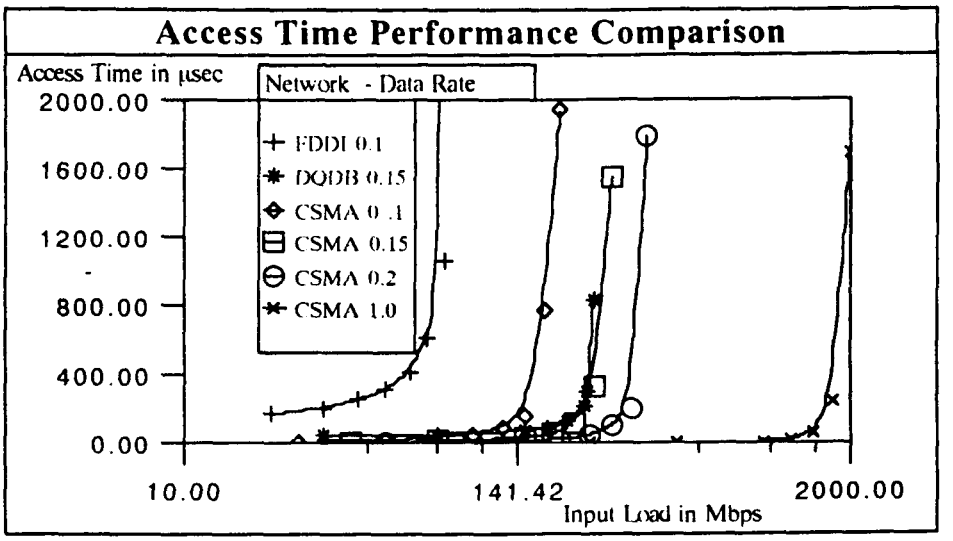


Figure 4 Performance Comparison between High Data Rate Access Protocols

4.0 CSMA/RN Operational System

The analytical and simulation studies indicate that CSMA/RN is a media access protocol which can operate effectively over a wide range of network conditions. In the following we, will briefly discuss the operational aspect of the access controller. The access controller must perform it operations rapidly; for a 1 Gbps system with a 100 bit delay buffer, the total operational time is 100 nanosec. A suggested packet frame is shown in Figure 5.

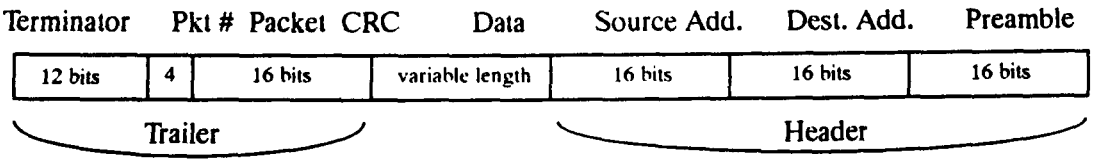


Figure 5 Packet Frame Structure

It is assumed that the network would use some form of encoding, possibly 5:4 bit encoding similar to that used in FDDI systems. If a node is transmitting, then after the first 4 bits of the preamble or 4 nanosec, it is alerted that an incoming message is arriving. The preamble is a unique 16 bit code. After the 16 bit preamble, the two 16 bit addresses must be decoded. After 48 nanosec., the node can decide whether an outgoing packet must be truncated. If the outgoing packet is to be terminated the access controller has 20 nanosec. to make the decision and switch

to place the trailing information into the packet. It is assumed that the packet count and terminator block can be prepared before hand, since, in anticipation of a possible switch, after 4 nanosec., the packet CRC system is able to select the last bits in the message buffer and complete its CRC block calculations. Hence, the access controller system has 64 nanosec. to finish the CRC calculation in anticipation that it may be needed.

The access controller must contain additional logic related to the circulating packet reservation and firness control systems. It must remember whether the last reservation packet was set, maintain a count of the free space requested and the location of the requesting source so that it can decide whether to free or use the space behind the CRP. For fairness control, it must be ready to remove and re-insert information in the fairness control packet. However, the logic for these actios can be made in a somewhat more leisurely time frame.

CSMA/RN can be implemented as the low level access protocol interfacing with the physical transport media of the network. However, equally important it could be implemented above the present or future frame and channel structure of a telephone carrier system. The basic synchronous mode transfer (STM) frame/channel structure prescribes that channels of fixed length are imbedded within a frame. Asynchronous mode transfer (ATM) structures are variable in length but provide unique header and trailer terminators [16]. Each block or channel carries a call which at the receiving end is directed to a transmitter system which imbeds the information into a new channel in a new frame for travel to the next receiver. The call information continues this process from source to termination in what is known as a virtual circuit from the source to the destination phone. STM frames recur every 125 μsec., the isochronous repetition rate, and carry 8 bits for a data rate of 64 Kbps.

For virtual CSMA/RN, we propose to reserve channels within a frame. Channels are allocated to form a virtual circuit from node to node in such a manner that a virtual ring is formed. At the receiving node, those channels allotted to the ring are examined as they arrive and if empty up to the delay time window, the node is free to start or continue inserting its queued messages. When an incoming message arrives it is checked for destination and deleted or forwarded as required. Synchronous traffic would use the circulating reservation packet system described above and isochronous traffic, the underlying frame/channel system. From the CSMA/RN standpoint, the only difference between the physical and virtual implementation is that, in the latter, there will be channels occupied by other messages which will bypass the controller logic completely. The additional problem created by ATM framing would be that the system would have to identify the frame as belonging to the ring. The advantage is that blocks can be considerably longer than the basic 8 bit channel so that virtual circuit operations may be smoother. It may be desirable to implement circulating frames within the ATM system thereby maintaining a reasonably fixed bandwidth for the network. This condpet is similar to FDDI II which embeds data traffic in unused channels in its frame structure [18].

Virtual CSMA/RN can be viewed to have a number of significant advantages. From an implementation standpoint it should be able to use logic similar to that now being used in the telephone switching system. Unlike physical CSMA/RN, the arrivals may have delays between distinct channels so the node logic may be better implemented with a delay buffer as opposed to delay line. In addition, it would be anticipated that the ring network could add or release channels as its load changed thereby keeping the network resources used to a minimum based upon good performance. Such a system could potentially provide a constant, uniform network

service over a wide range of network and load conditions. Finally, using the virtual circuit ring system, multiple rings could be established adding to over all reliability and to the possibility that for time-critical transmissions travel time could be optimized.

5.0 Research Issues for CSMA/RN

Clearly, CSMA/RN is beyond the conceptual state. To date, studies have demonstrated its capabilities, including:

1) virtually immediate access and minimal message fracture for loads up to 150% of capacity;
2) ability to handle widely varying message sizes;
3) up to 175% of rated network capacity without overload (1.75 Gbps traffic for a 1 Gbps network data rate);
4) synchronous traffic with little overhead, <1%, with no global master controller and automatic recovery of unused synchronous traffic bandwidth;
5) guaranteed maximum access time < 1 msec.; and
6) capability to span distances from 2 km - 10,000 km and wide range of node counts.

Hence, CSMA/RN approaches a universal media access protocol for gigabit networks.

There remain many questions and research issues which must be investigated in order to fully understand and use this media access protocol to implement gigabit networking. First, better analytical and simulator models of CSMA/RN performance are required in order to fully document its capabilities under the wide range of conditions existing in high data rate networks. Loads must include both synchronous and asynchronous, non-uniform traffic like that experienced at servers, gateways, and bursts to and from supercomputers and over longer durations where synchronous traffic is initiated and terminated These are conditions where access and fairness can become a reality. Alternative forms for handling synchronous traffic and for fairness control should be examined and compared and the best scheme from both an operational and performance standpoint implemented. Further, conditions where guaranteed access is required should be studied and documented. Load conditions should also simulate complex message traffic including broadcast and multi-cast, error handling for those messages whose addresses are corrupted, test of performance under software and hardware implemented acknowledge schemes and the study of the protocol's influence on upper level protocols, like TCP/IP. All of these investigations should be conducted over the wide range of network conditions which CSMA/RN is capable of handling.

Second, the controller logic should be built, tested and demonstrated so that its operations as noted in section 4 are better understood. While the first breadboard model can be built at a lower speed, later versions should built to handle gigabit rates. A computer logic simulation should accompany the hardware logic model so that tests for alternative and better logic procedures can be examined. This is especially true, if the controller logic is required to perform the fairness calculations within the nanosecond time frame that would be required in some forms of fairness control.

The implementation and integration of CSMA/RN into a frame/channel virtual circuit telephone system under both STM and ATM conditions requires further investigation in order to document both the performance and the hardware requirements for this form of operational CSMA/RN systems. The hardware integration should consider the basic controller logic, how it differs from that of the physical system and what systems can be used or modified within the present and

future telephone systems to support implementation. In addition, it should consider the operational features of virtual CSMA/RN and how they differ from a physical implementation. An interesting feature of virtual circuit CSMA/RN is the possible load balancing trade offs between virtual circuit capacity and network load conditions so that the total system will perform at an ideal data rate based upon performance and resource cost. One could visualize that over a wide range of loads, using such a load balancing scheme, the system performance will be virtually constant and hence, extremely predicable by the user for both his asynchronous and synchronous data interchange operations. For ATM broadband ISDN systems, there is the additional possibility where CSMA/RN can provide a longer lasting connectivity service, i.e., a dedicated network based upon a lower level, normally more transient, packet-switched, asynchronous data service [17].

6.0 References

1. Skov, M.: "Implementation of Physical and Media Access Protocols for High Speed Networks," IEEE Comm. Magazine; June 1989; pp 45-53.

2. 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.

3. Tobaji, F.A.; Fine, M.: "Performance of Unidirectional Broadcast Local Area Networks: Expressnet and Fastnet," IEEE Jour. on Selected Areas in Communication; Vol SAC-1; No 5; Nov 1983; pp 913-925.

4. Newman, R.M.; Budrikis, Z.L.; Hullett; J.L.: "The QPSX Man," IEEE Communications Magazine; Vol 26, No 4; April 1988; pp 20-28.

5. Maly, K; Zhang, L.: Game, D.: "Fairness Problems in High-Speed Networks," Old Dominion University, Computer Science Dept. TR- 90-15; Mar. 1990.

6. Ross, F:"FDDI - A Tutorial," IEEE Communications ; Vol. 24; No. 5; May 1986; pp 10-17.

7. Bux, W.: "Local Area Subnetworks: A Performance Comparison," IEEE Transactions on Communications; Vol. Com-29; No. 10; Oct. 1981; pp. 1465-1473.

8. Hilal, W.; Liu, M.T.: "Analysis and Simulation of the Register-Insertion Protocol," Proc. of Computer Networking Symposium; Dec. 10, 1982; pp 91-100.

9. Liu, M.T.; Hilal, W.; Groomes, B.H.: "Performance Evaluation of Channel Access Protocols for Local Computer Networks," Proc. Computer Networks ; Compcon '82; Sept. 20-23,1983; pp 417-426.

10.Suda T., et. al.; "Tree LANs with Collision Avoidance: Protocol, Switch Architecture and Simulated Performance"; ACM 0-89791-279-9/88/008/0155

11. Bhargava, A; Kurose, J.F.; Towsley,D: "A Hybrid Media Access Protocol for High-Speed Ring Networks," IEEE Jour. on Selected Areas in Communications; Vol. 6; No.6; July 1988; pp 924-933.

12. Maly, K; Foudriat, E.C; Game, D.; Mukkamala, R.; Overstreet, C.M.: "Traffic Placement Policies for a Multi-band Network," SIGCOMM Symposium; Sept. 1989.

13. Jaiswal, N.K.: Priority Queues; Academic Press; NY; 1968.

14. Wintsch, S.: "Toward a National Research and Education Network," MOSAIC; Vol 20; No. 4; Winter 1989; pp 32-42.

15. Foudriat, E.C.; Maly, K.; Overstreet, C.M.; Khanna, S.; Paterra, F.: "A Carrier Sensed Multiple Access Protocol for High Data Rate Ring Networks," Computer Science Department TR 90-16; Old Dominion University; Norfolk, VA. 1990.

16. Asatani, K.: "Lightwave Subscriber Loop Systems Toward Broad-Band ISDN,"Lightwave Technology, Vol. 7, No. 11, Nov. 1989, pp. 1705 -1714.

17. Partridge, C.(edit.).: "Workshop Report: The Internet Research Steering Group Workshop on Very High-Speed Networks," Jan 24-26, 1990.

18. Draft Proposed American National Standard. "FDDI Token Ring Media Access Control (mac)," asc x3t9.5 rev. 10; Feb. 28, 1986.

Distributed Simulation, No Special Tools Required*

Frank Paterra, C. Michael Overstreet, and Kurt Maly

March 30, 1990

Abstract

In this paper the authors present a toolkit of C language functions that can be linked with SIMSCRIPT programs to provide the data communication primitives necessary for distributed simulation. The authors' test case is discussed and some timing data are presented. Additionally some metrics, developed to determine the applicability of the server model decomposition for particular simulations, are discussed.

1 Why Distribute Simulations

Computer simulations are often computationally intensive tasks requiring long runs in order to obtain useful results. The runtime requirements of a simulation model can be a problem both during model development and validation and while performing production runs of the simulation.

The development of computer models to simulate a real world objects is a well understood problem and number of tools exist to aid the model developer [1]. Often times, the initial runs of a simulation model provide more questions than answers and the focus of study is changed. This results in an evolutionary process for model development, with refinements directed at different attributes as the object or its environment becomes better understood. Often the complexity of the model also increases during this process.

As the model is evolving, many runs may be needed to better understand the object and to verify the correctness of the simulation. The runtime requirements of complex models can greatly increase the time needed to

*This work was supported in part by CIT under grant INF-89-002-01, by NASA under grant NAG-1-908, and Sun Microsystems under RF596043.

Once a model has evolved to the point that production runs are being made, the runtime requirements again come into play. Often the output from each run may only consist of a single data point for a graph. In this case multiple runs of the same simulation with different inputs are needed. This can force the investigator to reduce the number of data points collected in order to reduce the time needed to generate a graph, resulting in incorrect conclusions about the simulated object.

Complementary to the problem of long, computationally intensive, runtimes is the fact that many times other computers are sitting idle and can provide basically free processor cycles to the simulation. In an effort to utilize some of these free cycles, much research has gone into developing algorithms for performing a single simulation on a number of losely coupled, cooperating processors.

The idea of distributing a simulation model among cooperating processors involves difficult problems. Principal among these are the identification of an effective decomposition of the simulation model, and the maintenance of processor synchronization to insure that the program is being executed in the correct order. The use of very tightly coupled functions and dependence on shared data, common in simulation programs, makes these problems are very acute to distributed simulation.

Significant research has gone into these two problems and the results are promising depending on the model being simulated. It is not our intention to address these problems in this paper, but rather to select an effective decomposition and synchronization scheme that will be used to demonstrate distributed simulation using our communication toolkit and standard simulation and operating system tools. A comprehensive treatment of the processor synchronization and model decomposition problems can be found in [2,3,4]. The problem of processor synchronization is more easily solved in very tightly coupled processors that support very high speed communication [5,6].

2 Model Decomposition

The model decomposition most easily supported by the tools here is to distribute some special types of model components, here called servers and receivers, on different machines. The term server is borrowed from object oriented design: a component is a server submodel if it can be represented as only sending to other model components.

This decomposition can be thought of as a collection of data servers and and receivers with no cycles. With no cycles synchronization becomes easier and the problems with deadlock such as that described in [7] is avoided. This is an easy and usually useful decomposition for complex, tightly coupled models, because the extensive data interaction among the model's parts are not interfered with. Other, potential more effective decompositions are outside the scope of this paper.

High performance scientific workstations sharing a LAN are becoming common. Since shared memory is not available and message passing among workstations in the network is slow, a decomposition of the simulation task is only likely to to be effective if messages are infrequently passed among workstations. This toolkit has been developed with these constraints in mind.

The tools support a "warehouse" approach. Information to be sent to receivers is "batched" and sent periodically as a single large message rather that as several smaller messages. In addition, the receiver workstation maintains inventories of data from servers, and based on anticipated consumption "orders" additional data periodically so that new data should arrive before current supplies are exhausted.

If data provided by the servers requires significant computation and data sent to the receivers also requires computing time, than significant parallelism can result since the required computation is off loaded from the receiver workstation. No possibility of deadlock exists in this approach and synchronization is particularly simple.

3 Distributed Simulation with Standard Tools

In this paper we present a toolkit of functions that allows distributed simulation to be carried out in a loosely coupled, general purpose, workstation environment without the use of special purpose operations systems, programming languages, or hardware.

The environment for which this software was developed contains a collection of Sun workstation computers connected via an ethernet LAN. These are very loosely coupled UNIX workstations with no shared memory and only communicate via a shared bus (the ethernet LAN). SIMSCRIPT was selected as the simulation language because of it wide use in the simulation community. All processors cooperating in the simulation run programs written in SIMSCRIPT and call external functions for processor communication.

The processor communication functions are provided via the UNIX Interprocess Communication (IPC) functions [8]. These functions are standard with the BSD UNIX operating system and allow communication among processes both within the same computer and those residing on separate computers. Because the IPC functions are designed to provide communication among a large number of varying processor types, a significant amount of overhead is inherent with data communication. This could be reduced by writing replacement functions that only provide for the needs of this simulation, however a design goal was to use as little custom software as possible.

4 The Toolkit

The toolkit consists of a collection of functions, written in the C language and linkable with SIMSCRIPT programs. The basic functions provided by the toolkit are interprocess data communications and sufficient processor synchronization to allow a simulation to be broken into a collection of data servers and receivers.

To use these tools, one must first determine what in their model can be thought of as a source or generator of precomputable objects. In order for an object to be precomputed, no information about current simulation time or access to local variables can be required. The most obvious precomputable object is random numbers, however, more complex objects may be precomputed based on the simulation model at hand. Once the data sources have been identified, the simulation is written as usual, except that the identified source objects are written as a separate SIMSCRIPT program. This results in the simulation being implemented as a data generator program and a simulation program. Two C language functions must be called by both the simulation program and the generator program to install the communications interrupt handler and to identify each of the generators participating in the simulation. Each of the SIMSCRIPT programs must also contain a function that the C routine will call to transfer generated data to and from SIMSCRIPT variables. Each of these functions are described below.

C Functions

- inst_int(host,mode)
  - char *host - The name of the host running the receiver program

- char *mode - Must equal "server" or "receiver" for the server and receiver programs respectively.

This is a C routine that is called by both the receiver and server programs. Called only once, and before any the link server function described below, this function opens a socket for reading, installs the communications interrupt handler, and initializes the variables used to maintain the server information.

- link_server(server,host,mode) -
  - char *service - The name of the service being identified
  - char *host - The name of the host where the server resides
  - char *mode - Must equal "server" or "receiver" for the server and receiver programs respectively.

This C routine is called by both the receiver and server processes to identify the services that are being used in this simulation. The function creates a record of the identified server's information, opens a sending socket for the server, initializes the list of messages to that server as null, and adds the server to the list of participating servers.

- request(service,command) -
  - char *service - The name of the service being requested
  - int command - A command to be sent to the server. This command integer is not examined by the toolkit; it is completely definable by the model and server developers.

This C routine is called by the simulation module to request more data items from a server. After the request is sent, control is return to the simulation software. When the requested data are received, the simulation code will be interrupted and the toolkit will make a call to user provided accept_data routine, described below.

SIMSCRIPT Functions

- accept_data given service, data, length
  - service - text variable containing the name of the service supplying the data. This field is used to route data to the correct inventory.

- data - memory for objects created. This memory space will be formatted by the server to hold the data in the correct SIMSCRIPT format.
- length - This is the length in bytes of the data area.

The accept_data function is called by the C routine that performs the socket reads when new data arrives. Because the arrival of data causes and interrupt to be serviced and this function is called during that interrupt, the code may be executed at any time. This will have an impact on the simulator's use of pointers or indices to the inventory of data.

- fill_request given service, data, yielding length
  - service - Text variable containing the name of the service being requested.
  - data - memory for objects being created. This is unformatted memory and can be interpreted and filled according to the objects being requested.
  - length - Integer variable returning the length in bytes of the data to be supplied.

  This SIMSCRIPT function is the server complement to the accept data function. When a request for objects is received by the communications handler, this function is called to fill the request. As before, because the communications are interrupt driven, this function can be called at any time.

After the receiver code has been moved to a separate program, additional SIMSCRIPT code will have to be added to the receiver program to manage the remotely generated data. This additional code keeps track of the available inventories of remotely generated data, placing requests for additional data when the local inventory falls below some threshold. How this threshold is calculated is discussed in a later section.

The receiver program may itself be a data source, for example generating simulation data that are sent to additional programs that provide statisical analysis and summary reports or to other servers for graphical display.

5 Timing Data and Decomposition Considerations

Message passing overhead must be considered when designing any type of distributed processing. To decide if any speedups can be expected for a simulation using the server model decomposition, some analysis for message passing times verse local computational costs should be performed. The following definitions are used to perform this analysis.

- CIO - Overhead induced by servicing a communications interrupt. This includes the time required to transfer data from the communications buffer to a SIMSCRIPT variable.
- CSO - Overhead induced by actively sending a message to another server.
- CTT - Time for a command message to travel between two hosts.
- DTT - Time for a data message to travel between two hosts.
- MST - Minimum time required to supply objects.
- OS - Order size. The number of items shipped in each order.
- ECR - Expected consumption rate for generated items.
- DCC - Distributed computation cost.
- LCC - Local computational cost for generating one object.
- RGT - Remote generation time. The time required by the server to generate the values. This value is determined by OS and LCC.
- TBO - Time between orders.

Assume that the generators have precomputed more of the objects than will be requested so that the time that the generator will spend processing a request is 0. With this assumption we can define

$$DCC = CSO + CTT \tag{1}$$

$$LCC = CIO + DTT \tag{2}$$

$$MST = DCC + LCC \tag{3}$$

Clearly the formulae below must hold or it will always be faster to compute the objects locally.

$$OS \geq MST * ECR \tag{4}$$

$$OS * LCC > CIO + CSO \tag{5}$$

In many cases, unless LCC (the cost of computing the data locally) is significant, OS will have to be large to make this approach feasible. Practically speaking, since for most simulations the actual consumption rate can vary, $OS * LCC$ should be significantly larger than $CIO + CSO$.

In order to assist in determining the potential effectiveness of using this approach for distributed simulation, some timing data was collected for LCC, CIO, CSO, and CT. The variable ECR is model dependent and, with the other variables fixed, OS can be determined.

Timing data for message passing among Sun workstations connected via ethernet networks and bridges was collected. The times required for message passing are not significantly affected by message length as long as are less than the maximum packet length for the ethernet (1500 bytes). Messages were passed between processors that resided on the same physical network and those on separate, bridged networks. As can be seen below, messages that had to go across bridges took twice as long as those that stayed on a single network. All data was collected when the network was lightly loaded.

Packet Size: 100 bytes

Number of Packets	Single Net (seconds)	Bridged Nets (seconds)
100	1	3
500	5	9
1,000	9	23
5,000	45	112
10,000	87	206
50,000	435	944

Packet Size: 500 bytes

Number of Packets	Single Net (seconds)	Bridged Nets (seconds)
100	1	3
500	6	18
1,000	14	26
5,000	68	137
10,000	118	256
50,000	513	1,113

Packet Size: 1000 bytes

Number of Packets	Single Net (seconds)	Bridged Nets (seconds)
100	2	3
500	8	20
1000	16	33
5,000	79	171
10,000	156	332
50,000	793	1,721

Analysis of the above data gives the following table of average times and throughput rates. Times are given in seconds/byte and throughput is given in bytes/second.

Packet Size	Average Time (sec.)		Throughput (bytes/sec)	
	Intra-net	Inter-net	Intra-net	Inter-net
100	0.000087	0.000194	11,400	5,100
500	0.000022	0.000047	46,000	21,500
1,000	0.000015	0.000034	63,000	29,000

To obtain values for the variables LCC, CSO, and CIO, the UNIX prof command was used. This is a standard UNIX tool that profiles executable code and generates reports on number of times each function is called, time spent during each call, and total time spent in the function during program execution. For more information on the prof command see [9].

6 Example

As an example, consider the generation of normally distributed random numbers. The machines used are Sun 3/60 workstations with 8 megabytes of memory. LCC was found to be 0.1 ms; the CSO and CIO were both 0.025 ms. The table below shows values for OS with corresponding ECR values.

ECR/minute	OS	Orders	RGT	TBO	MST
10,000	50	200	0.005	0.30	0.002
	100	100	0.010	0.60	0.013
	500	20	0.050	3.00	0.061
	1,000	10	0.100	6.00	0.121
	5,000	2	0.500	30.00	0.601
100,000	50	2,000	0.005	0.03	0.002
	100	1,000	0.010	0.06	0.013
	500	200	0.050	0.30	0.061
	1,000	100	0.100	0.60	0.121
	5,000	20	0.500	3.00	0.601
500,000	50	10,000	0.005	0.006	0.002
	100	5,000	0.010	0.012	0.013
	500	1,000	0.050	0.060	0.061
	1,000	500	0.100	0.120	0.121
	5,000	100	0.500	0.600	0.601
1,000,000	50	20,000	0.005	0.003	0.002
	100	10,000	0.010	0.006	0.013
	500	2,000	0.050	0.030	0.061
	1,000	1,000	0.100	0.060	0.121
	5,000	200	0.500	0.300	0.601

Two points can be made from this table. First, when the order size becomes large, the communication time for transferring the numbers from the server to the simulator becomes larger than the time required to compute the values locally. As long as the time needed to consume the numbers is greater than the MST, a speedup is possible with the server decomposition.

Secondly, when the ECR becomes very large, the remote server cannot keep up with the ECR, the receiver will be forced to wait for the server to generate numbers. If the time spent waiting is significantly less than what is required to compute the values locally, then the server decomposition still provides speedup.

7 Conclusion

The toolkit that we have developed can be use to develop distributed simulation applications without having to invest in new environments or training. SIMSCRIPT and the UNIX operating system are widely available, allowing easy access to these tools. The toolkit is composed of 650 lines of C

code and requires about approximately 50 lines of additional SIMSCRIPT code per server/receiver pair to be added to the simulation model. The user of these tools need only be concerned with three C function calls and two SIMSCRIPT routines, so the complexity of the simulation program is not significantly affected.

Use of these tools requires decomposing a model into components in which information flow is unidirectional such as traffic arrival generators, statistical analysis procedures, or graphical displays.

References

[1] Richard L. Gimarc. Distributed simulation using hierarchical rollback. In *1989 Winter Simulation Conference Proceedings*, pages 621–629, 1989.

[2] Richard M. Fujimoto. Parallel discrete event simulation. In *Proceedings of the 1989 Winter Simulation Conference*, pages 19–28, 1989.

[3] David Jefferson. Distributed simulation and the time warp operating system. *ACM SIGOPS*, 77–93, Nov. 1987.

[4] B. A. Cota and R. G. Sargent. *Concurrent Programming in Discrete Event Simulation: A Survey.* Technical Report, Syacuse University, Dec. 1986.

[5] Fred J. Kaudel. *A literature Survey on Distributed Discrete Event Simulation.* 1987.

[6] Douglas W. Jones, Chien-Chun Chou, Debra Renk, and Steven C. Bruell. Experience with concurrent simulation. In *Proceedings of the 1989 Winter Simulation Conference*, pages 756–763, 1989.

[7] Rajive L. Bagrodia, K. Mani Chandy, and Jayadev Misra. A message-based approach to discrete-event simulation. *IEEE Transactions on Software Engineering*, SE-13(6):654-665, June 1987.

[8] ***Unix Programmer's Manual, Supplementary Documents 1.***

[9] ***Unix User's Manual, Reference Guide.***

Alternative Parallel Ring Protocols*

R. Mukkamala E.C. Foudriat K.J. Maly V. Kale
Department of Computer Science
Old Dominion University Norfolk, Virginia 23529-0162

April 17, 1990

Abstract

Communication protocols are known to influence the utilization and performance of communication network. In this paper, we investigate the effect of two token-ring protocols on a gigabit network with multiple ring structure. In the first protocol, a node sends at most one message on receiving a token. In the second protocol, a node sends all the waiting messages when a token is received. The behavior of these protocols is shown to be highly dependent on the number of rings as well as the the load in the network.

1 Introduction

The use of parallel communication channels to achieve a gigabit network is a very interesting concept. Especially, if a gigabit network needs to be built from an existing common carrier system, a network of parallel channels may be a viable alternate. However, its appropriateness can only be ascertained after determining its behavior under different load conditions. To this end, we chose a parallel ring network operating on token-based protocols.

Currently, our studies are restricted to two token-ring protocols. With each of the protocols, a token is assigned to each of the rings in the network. All tokens rotate in the same direction. While a node is holding a token for transmission, it cannot hold any other token.

1. Exhaustive Policy: Under this policy, when a node obtains a token, it transmits all the messages in its queue, and then releases the token.

*This work was supported in part by CIT under grant INF-89-003-01, by NASA under grant NAG-1-908, and by Sun Microsystems under RF596043.

2. Non-exhaustive Policy: Under this policy, a node can transmit at most one message when it receives a token.

In this report, we describe the results obtained from simulations of these two protocols. From these results, we make some comments regarding the appropriateness of parallel ring structures operating at gigabit speeds.

2 Input Parameters

In order to determine the behavior of the multiring token-based protocols, we have run a number of experiments. Following is a summary of the input data:

- Since we are only interested in gigabit networks, we have considered the total bandwidth of the network to be 10^9 bits/sec.
- When there are R rings in the system, each ring has a bandwidth of $10^9/R$ bits/sec.
- Since token-ring protocols are only relevant for local area nets, we considered a total ring length of 30 Km, with 30 equally spaced nodes.
- The propagation delay on the network is taken to be half the speed of light (i.e. 150Km/msec).
- For simplicity, we assume constant length messages (10K bits). Each message is transmitted as one entity (i.e. no fragmentation).
- The load on the network is expressed in terms of mean-time between arrivals (m) of messages at any node. Low loads are represented by $m = 1.0$, which denotes that one the average a message may be expected once every 1 msec at each node. High load is represented by $m = 0.31$. We have used $m = 0.5, 0.4, 0.35$ as other values. Message arrivals are assumed to be Poisson.

3 Results

The performance of the protocols is based on simulation of the network. The simulation was carried out for 10 seconds. The system was let to stabilize in the first 5 seconds, and the statistics were then taken in the second half of the experiment.

We make the following observations from the obtained results:

- In Figure 1, we compare the average time that a message at the *head* of a queue at a node needs to wait before a token on any of the rings is captured. This time is also generally referred to as residual time of the token inter-arrival time (RH_T). When the inter-arrival time of messages (at each node) is at least 0.5, the residual time is less than 0.3 msec. As the mean-time between arrivals decreases, this time increases significantly. Certainly, having multiple rings results in a reduced RH_T.

- Figure 2 illustrates the effect of number of rings on the average waiting time of a message. Under the exhaustive policy, the average waiting time (RT_T) is less than 1 msec for $m \leq 0.35$. The benefit due to the presence of multiple rings is more apparent at high loads. For low loads ($m \geq 0.5$), the average waiting time is almost the same as RH_T. This is obvious since the queue sizes are generally very small.

  Under the non-exhaustive policy, single-ring networks cannot tolerate situations where $m \leq 0.5$. (This can also be proved analytically). When there are at least 8 rings, the system can tolerate values of $m \geq 0.35$. When $m = 0.31$, however, the system has very large queues (hence not shown in the figure), resulting in very large average waiting times.

- Figure 3 summarizes the relationship between average token interarrival times and the number of rings. Obviously, the average token inter-arrival time ($E[T]$) at a node decreases with the number of rings, since the number of tokens is now increased. This decrease is more pronounced at high loads ($m \leq 0.31$). Under the non-exhaustive policy, the token inter-arrival times are quite low (even under high loads). This is not surprising knowing that in this protocol at most one packet per node per token is only transmitted. With the exhaustive policy, however, the token inter-arrival times for the tokens could be significant for smaller number of rings.

- Figure 4 illustrates the randomness of the measured inter-arrival times of tokens. This randomness is expressed in terms of the ratio of the standard deviation to the mean of the token inter-arrival times. The ratios are higher for the non-exhaustive policy. Since the non-exhaustive policy sends at most one packet at a time, while the exhaus-

tive policy sends all the pending packets with a token, this observation is counter intuitive. We are attempting to explain this phenomenon through some probabilistic analysis. Generally, this ratio seems to increase with the number of rings (at least up to 8 rings). Beyond eight rings, the behavior of this ratio seems to depend on the load factor (m).

- Figure 5 describes the relationship between response time and the number of rings for different loads. Response time includes the time to wait in the queue, the time for transmission, and the time for propagation from source node to destination node. For low loads, since the waiting times are approximately constant, there is a slight increase in the response time with the increase in number of rings. The increase in response time may be explained by the increase in transmission delay due to reduced bandwidth per ring. The behavior of the average response time for both non-exhaustive and exhaustive policies is similar to that of the average waiting times in the queue.

- Figure 6 summarizes the variances in the response time. This seems to be quite different from the variances in the waiting times. First, the variances of the exhaustive and non-exhaustive policy are now comparable. Second, the variances are much less than those in Figure 4. Once again, we are investigating the possible causes for this phenomenon.

- Average token rotation time for each ring is also an important performance metric. This metric is shown in Figure 7. In the case of exhaustive policy, as expected, the token rotation time is independent of the number of rings (since increase in number of rings also increases the transmission delay by the same extent). The behavior of the non-exhaustive policy needs more investigations.

- Determining the probability with which an arriving token is used to transmit messages (f_T) is useful in describing the utilization of the network. This relationship is summarized in Figure 8. Clearly, the behavior of f_T differs under the exhaustive and non-exhaustive policies. For low loads (e.g. $m = 1.0$), both policies exhibit similar behavior.

- Figure 9 summarizes the behavior of the average waiting time in the queue under the two policies. This is similar to Figure 5. Figure 11 describes this information for messages that were not in the head of

the queue (obviously relevant only with the exhaustive policy). Figure 10 displays the average time a message waited from the time it arrived to the time its last bit left the node. Clearly, this is similar to the times in Figures 5 and 9.

- The number of packets that were transmitted per node for each arrival of a token is also a metric of interest. Figure 12 describes this metric ($E(L)$). Not surprisingly, the patterns in this figure are similar to the ones in Figures 5,9, and 10. The plots for the non-exhaustive policy are not relevant since it only transmits at most one packet each time with a token.

- Network utilization is a very important metric in determining the ability of a protocol to function at high loads. Figure 13 summarizes this metric for the two policies. With the non-exhaustive policy, with a single ring the network (actually with a capacity of 1 gigabit/sec), can't have more than 60% utilization of the network capacity (even when the input load is high). This is certainly a restriction. Similarly, a two-ring network cannot have more than 75% utilization. The exhaustive policy, however, seems to place no such restriction, and the utilization appears to be independent of the number of rings.

4 Conclusions

From the above results, we make the following conclusions:

- At all loads, irrespective of the policy, there is a gain in having multiple rings. Whether this gain reduces when the number of rings increases (and hence the bandwidth of each ring decreases) beyond a certain point, is yet to be seen. We propose to experiment with 32, 64, and 128 rings to make stronger conclusions about this impact.

- Gigabit speed networks with a small number of parallel rings (1,2, or 3) will limit the utilization of the network. Thus, the total capacity of the network can never be utilized. The exhaustive policy, however, seems to perform well with any number of rings.

- The reduction in response time seems to be significant with multiple rings and at high loads. Certainly, there is a noticeable reduction in response even at medium loads with the multiple ring structures.

- The token inter-arrival seems to be significantly affected by the number of rings and the choice of the policy. This fact is very significant when parallel networks are used to support real-time applications.

In summary, the results are very interesting, but some more studies with other policies, bigger networks (more nodes, larger lengths), and more rings need to be carried out before a well established guidelines are set towards the design of parallel gigabit networks.

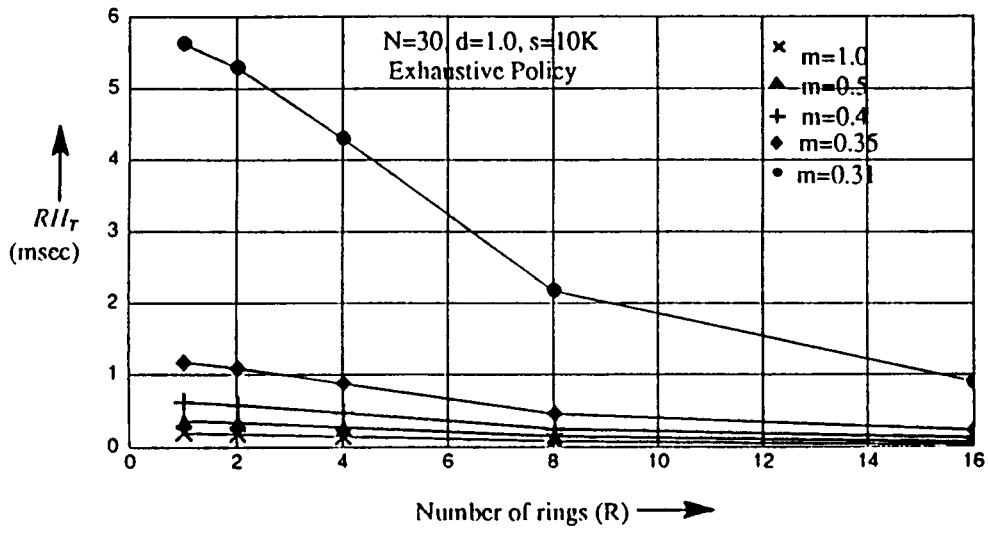


Figure 1. Mean Residual Time for Messages at the Head of the Queue

Figure 2a. Exhaustive Policy

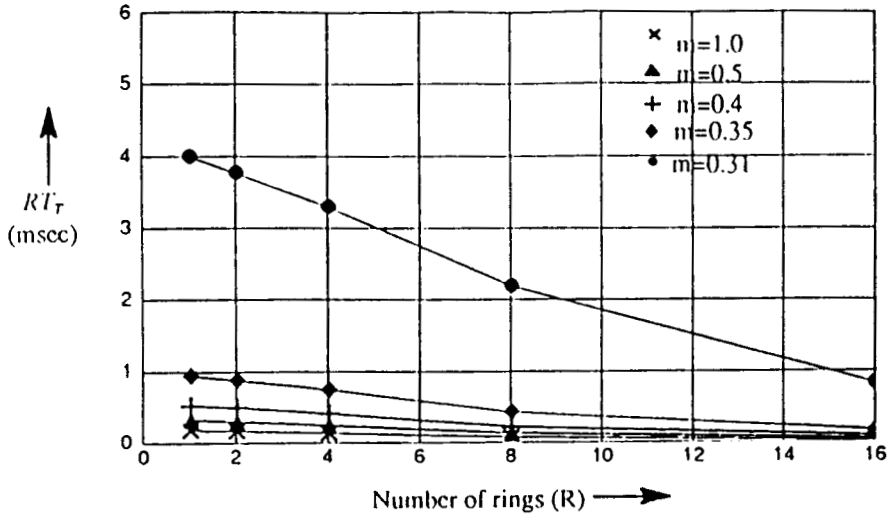


Figure 2b. Nonexhaustive Policy

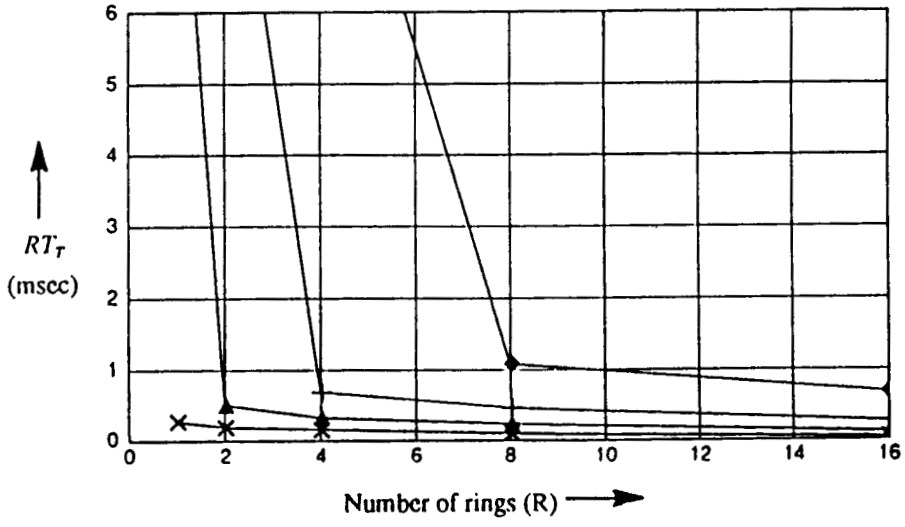


Figure 2. Average wait time (of a message) in a queue before a relevant token was received by its source node (N=30, d=1.0, s=10K)

Figure 3a. Exhaustive Policy

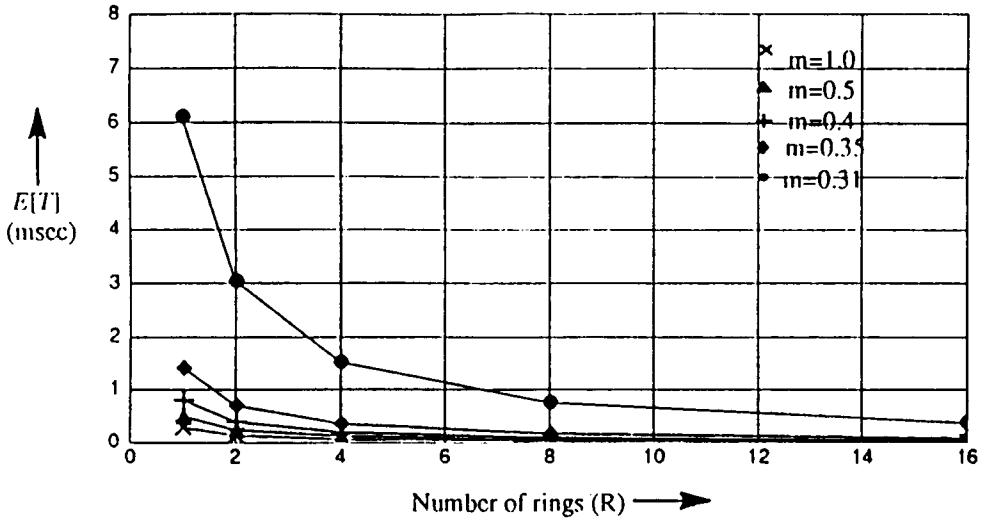


Figure 3b. Nonexhaustive Policy

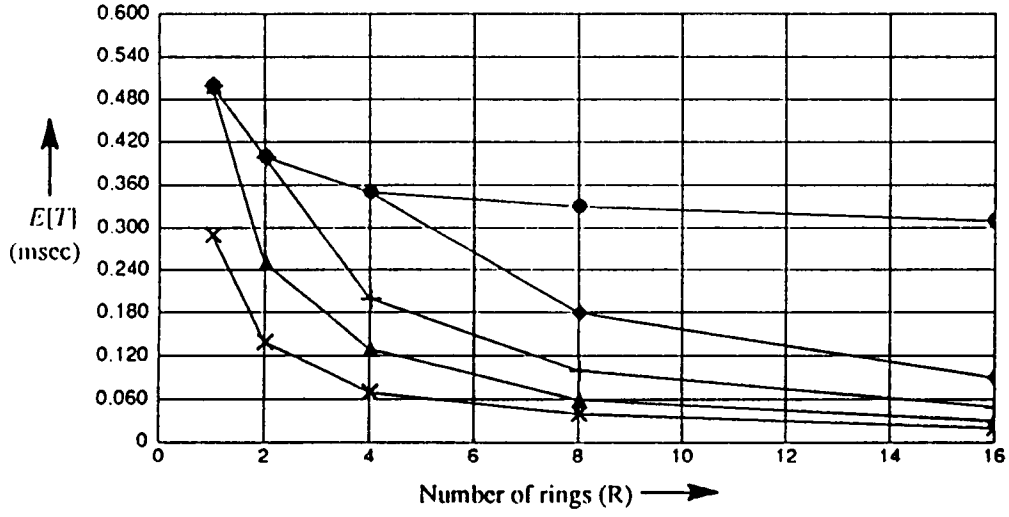


Figure 3. Average Token Inter-arrival Times at Each node (N=30, d=1.0, s=10K)

Figure 5a: Exhaustive Policy

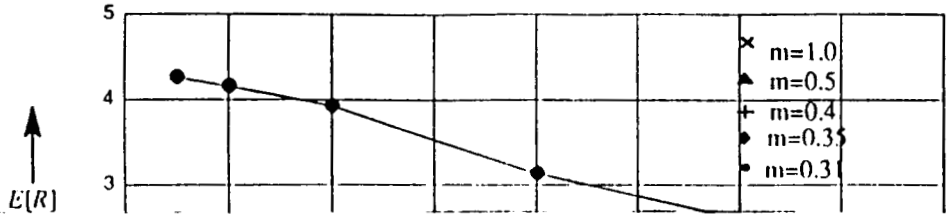


Figure 4a: Exhaustive Policy

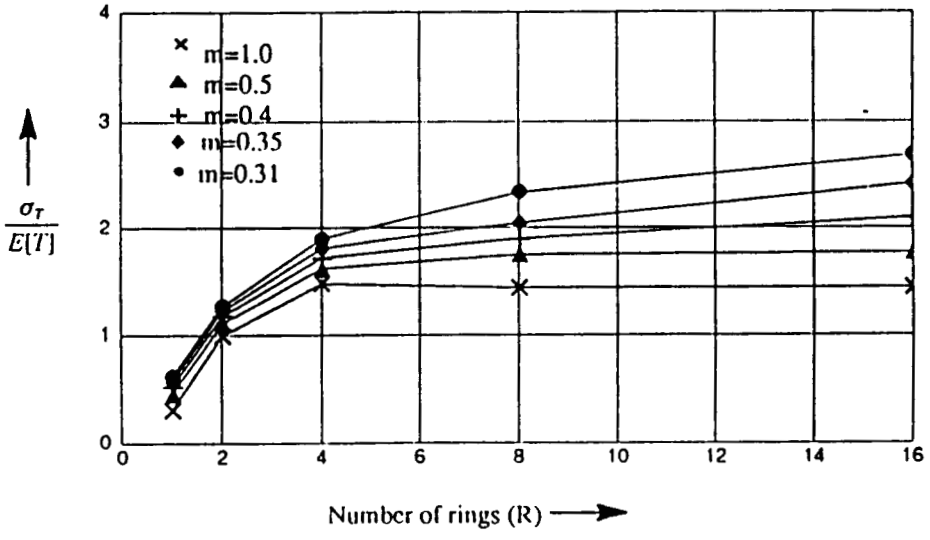


Figure 4b: Nonexhaustive Policy

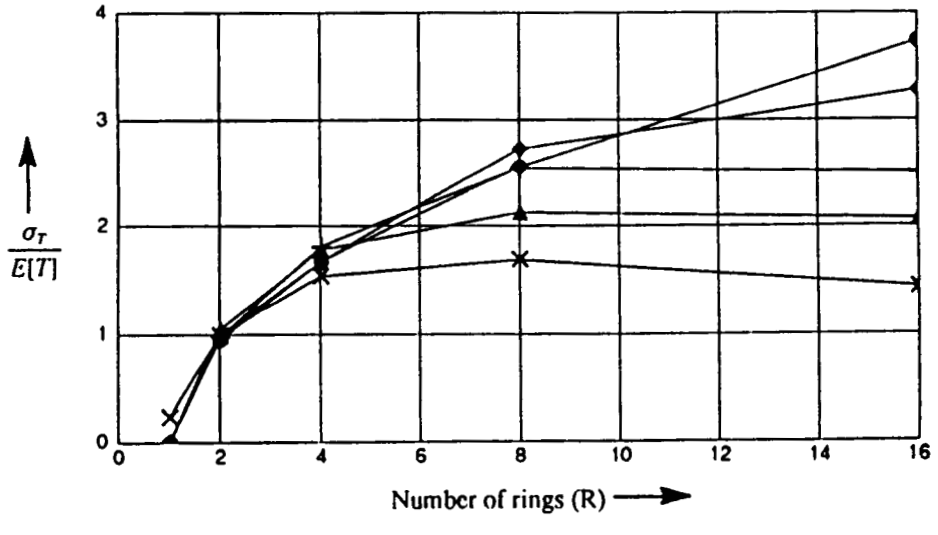


Figure 4. Standard deviation/Mean ratios for Token Inter-arrival Times
(N=30, d=1.0, s=10K)

Figure 6a: Exhaustive Policy

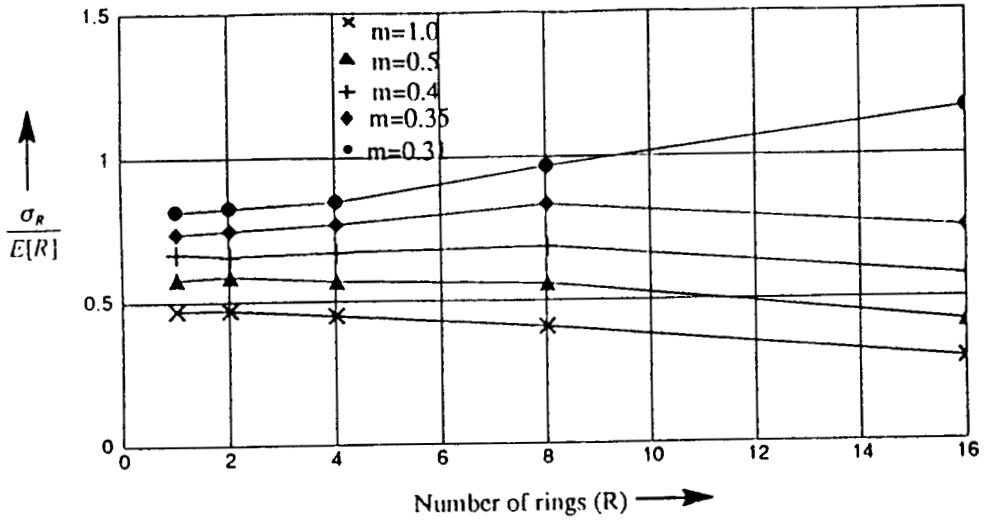


Figure 6a: Nonexhaustive Policy

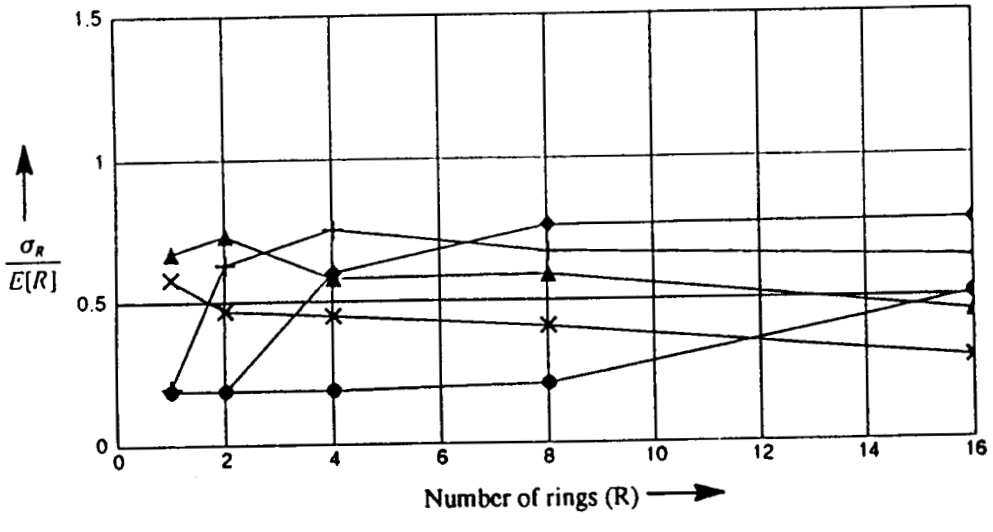


Figure 6. Standard deviation/Mean ratio for Packet Response Times (N=30, d=1.0, s=10K)

Figure 7a: Exhaustive Policy

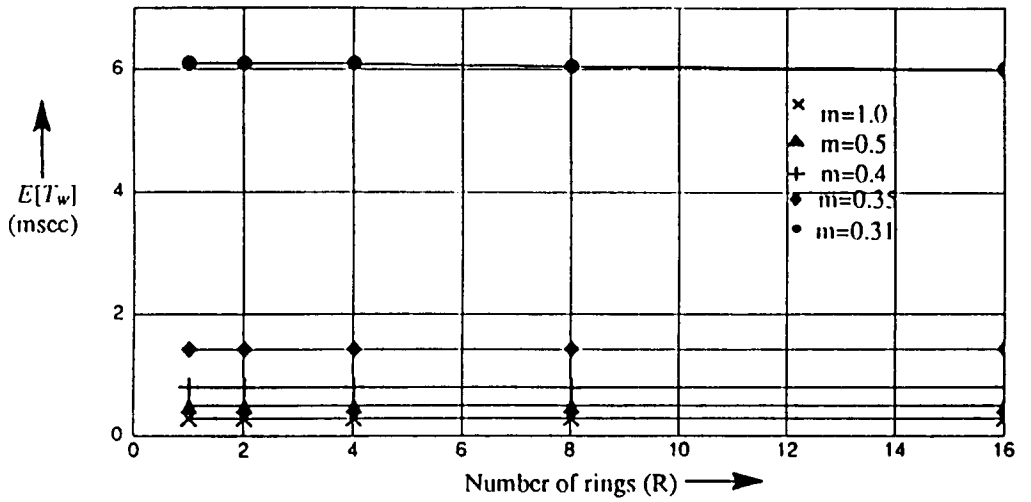


Figure 7b: Nonexhaustive Policy

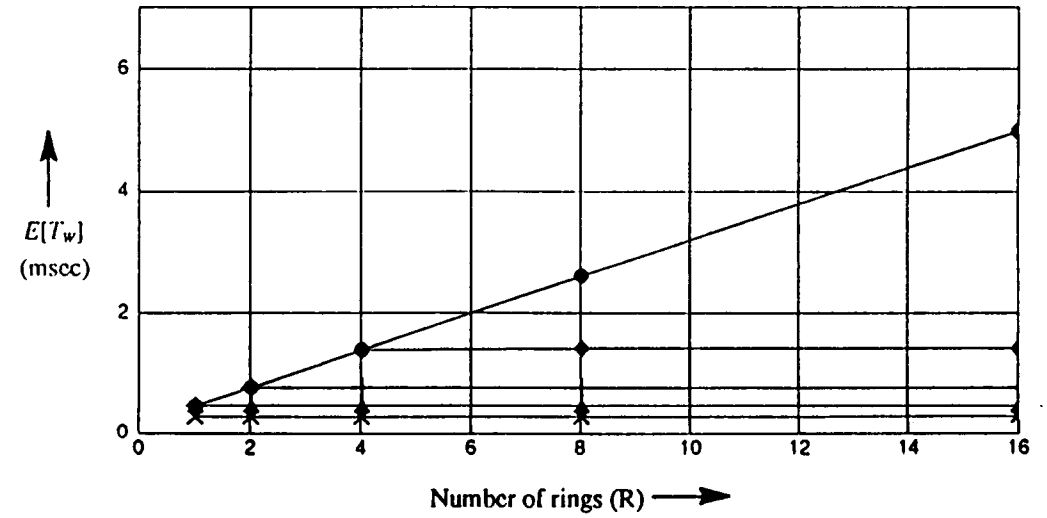


Figure 7. Average Token walk-times for each Ring (N=30, d=1.0, s=10K)

Figure 8a: Exhaustive Policy

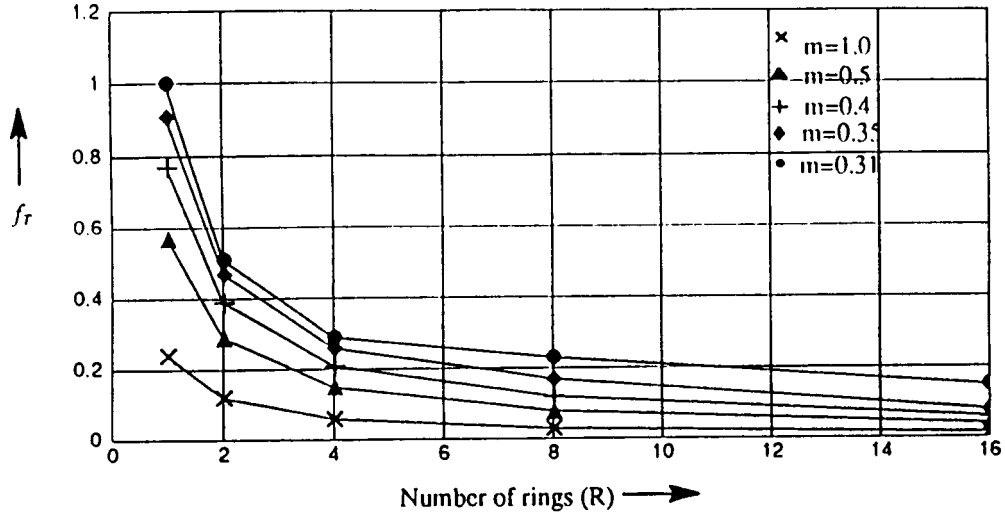


Figure 8a: Nonexhaustive Policy

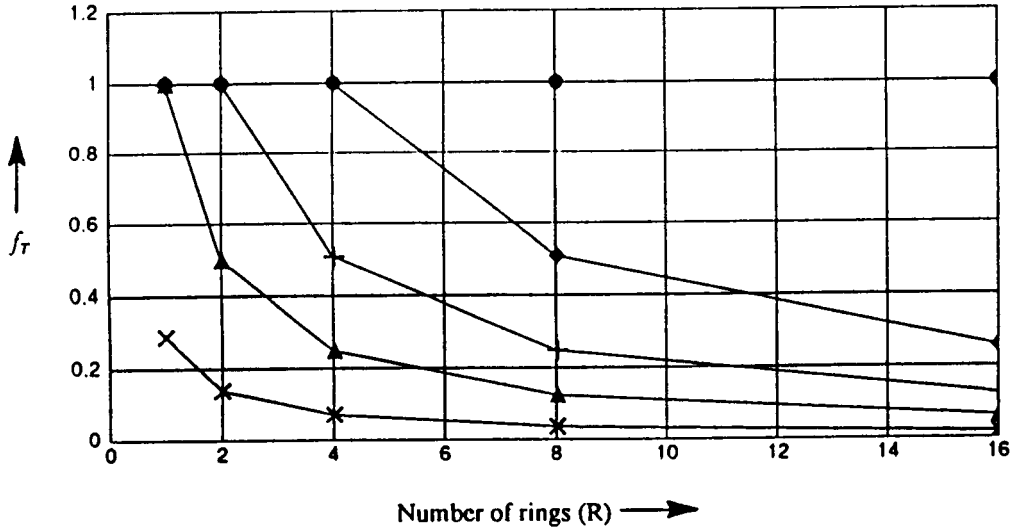


Figure 8. Fraction of times that an Arriving Token is Used by a Node for Transmission

(N=30, d=1.0, s=10K)

Figure 9a. Exhaustive Policy

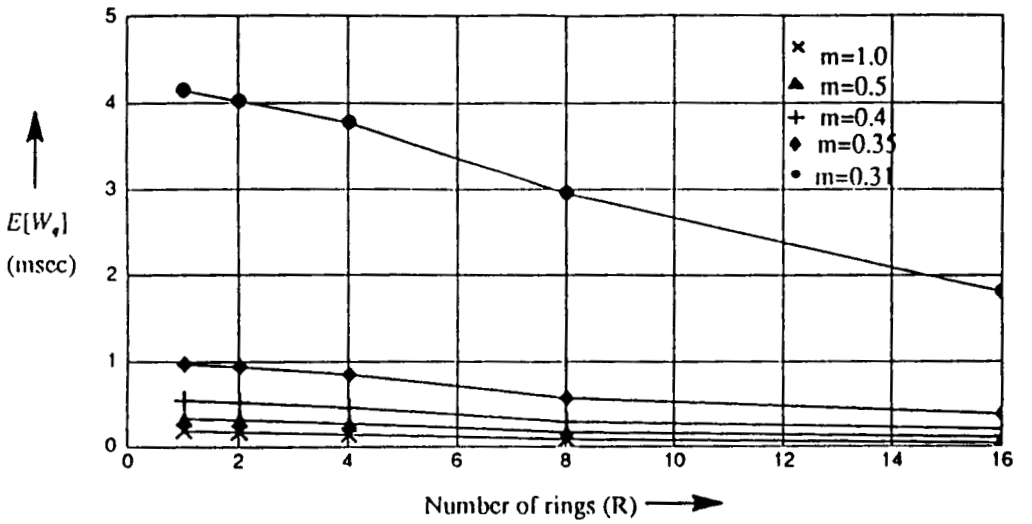


Figure 9b. Nonexhaustive Policy

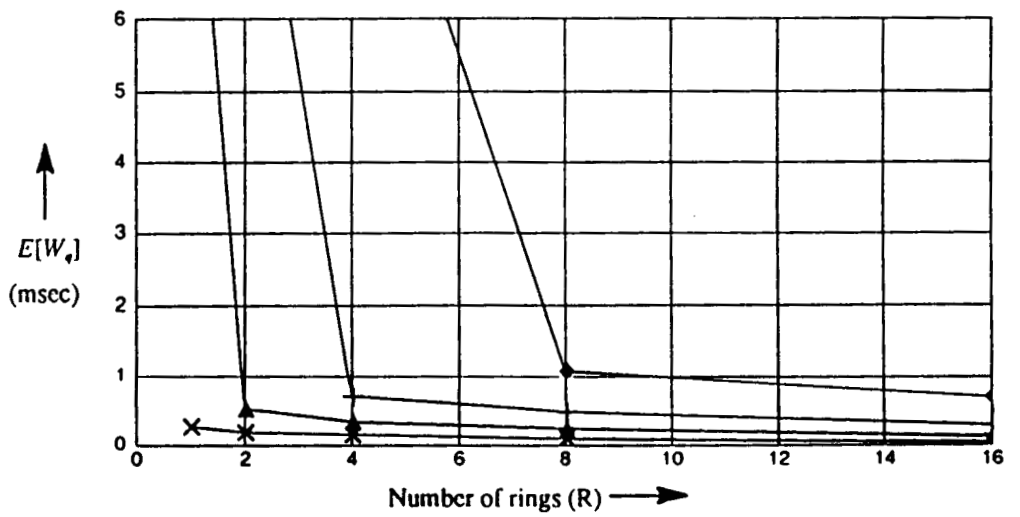


Figure 9. Average Waiting Times for Each Message (at a Node)
(N=30, d=1.0, s=10K)

Figure 10a. Exhaustive Policy

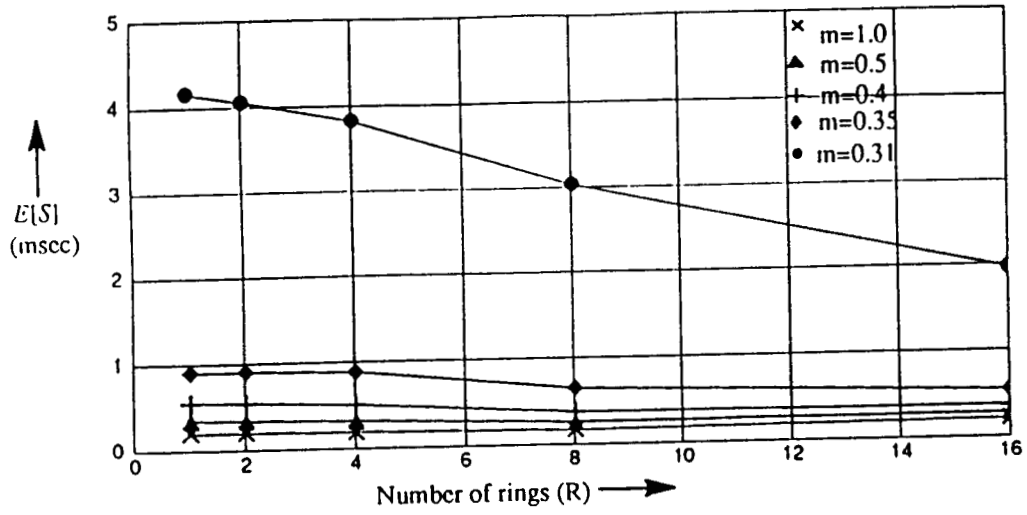


Figure 10b. Nonexhaustive Policy

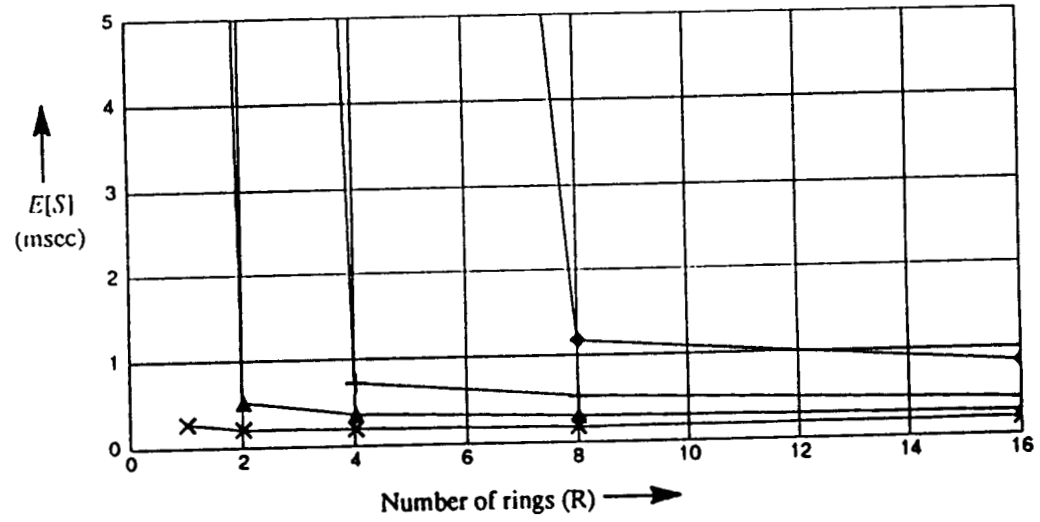


Figure 10. Average Time Between Message Arrival and Its Leaving the Node

(N=30, d=1.0, s=10K)

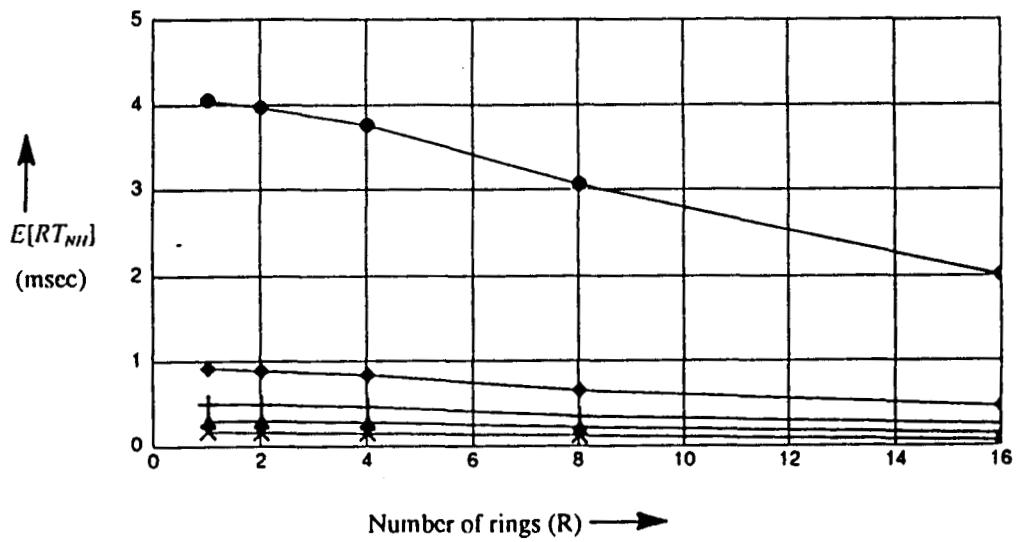


Figure 11. Average Response Time for Non-header Messages in a Queue
(Exhaustive Policy; N=30, d=1.0, s=10K)

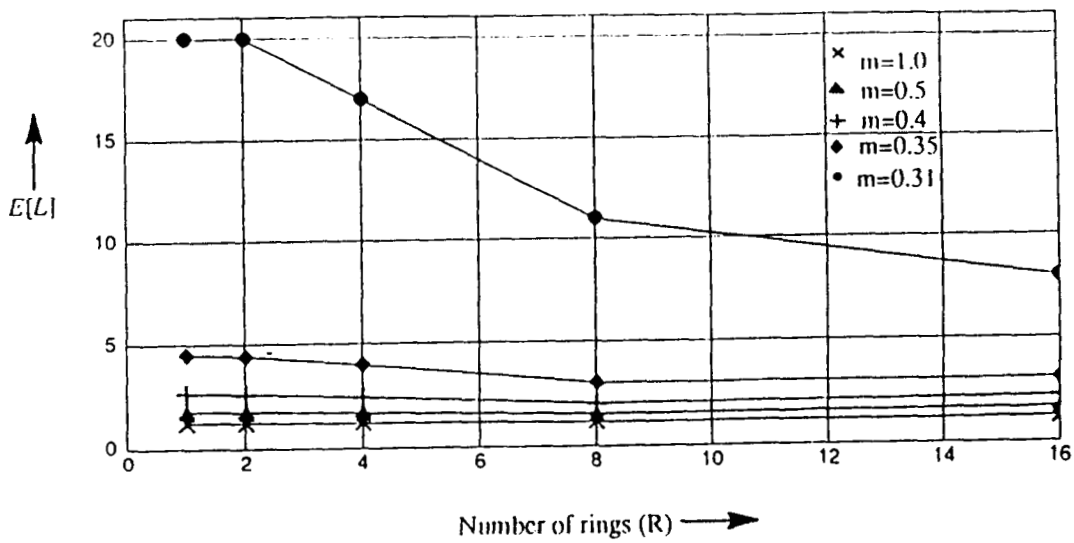


Figure 12. Average Number of Messages Transmitted per Node Per Token

(Exhaustive Policy; N=30, d=1.0, s=10K)

Figure 13a. Exhaustive Policy

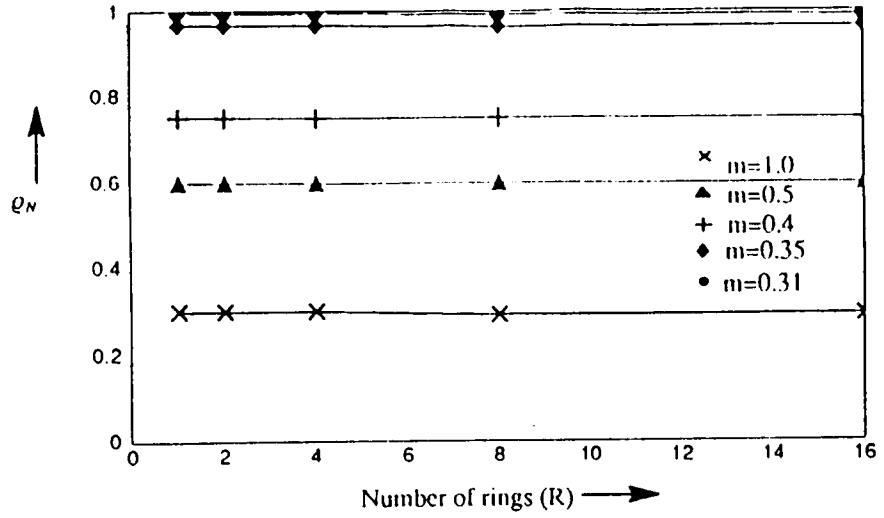


Figure 13b. Nonexhaustive Policy

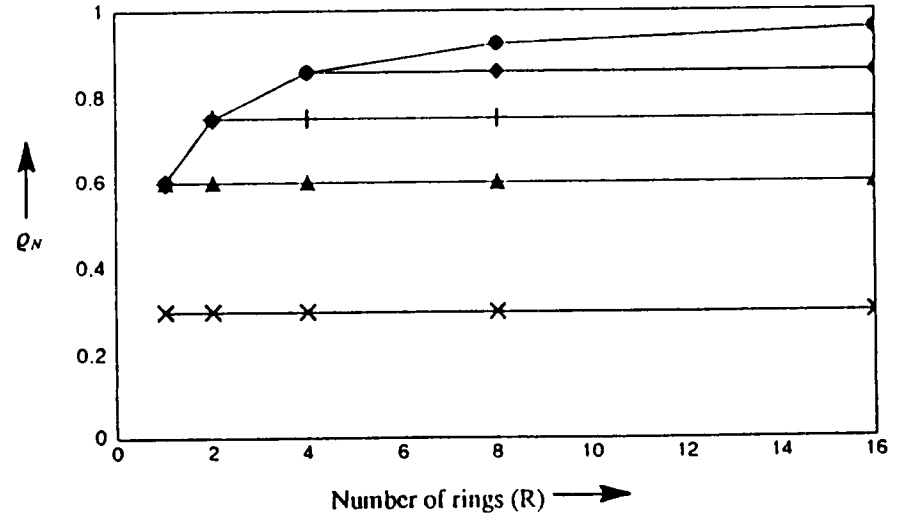


Figure 13. Network Utilization under the Two Policies
(N=30, d=1.0, s=10K)

Performance of Gigabit FDDI

David Game
Kurt Maly
Department of Computer Science
Old Dominion University
Norfolk, Virginia 23529-0162

April 23, 1990

Abstract

Great interest exists in developing high speed protocols which will be able to support data rates at gigabit speeds. Hardware currently exists which can experimentally transmit at data rates exceeding a gigabit per second, but it is not clear as to what types of protocols will provide the best performance.

One possibility is to examine current protocols and their extensibility to these speeds. This paper investigates scaling of FDDI to gigabit speeds. More specifically, delay statistics are included to provide insight as to which parameters (network length, packet length or number of nodes) have the greatest effect on performance. [1]

[1]This work was supported by CIT grant RF-89-002-01, NASA grant NAG-1-908 and Sun Microsystem grant RF 596043.

1 Introduction

Lasers, optical fiber, and related optics technology have recently redefined the bottleneck in data communications[2,4,6,7] such that the communications channel is no longer the limit to information processing. Instead, the issue has become one of whether or not a computer can generate and/or process information at the rates which are now available. Applications for interactive video such as a surgeon examining a 3-D display of an organ can readily use such high bandwidth, but the computer itself is not able to generate images at a rate which will consume such a high bandwidth for an extended period of time.

Nonetheless, great interest exists in extending the communications capacity. Although a single computer may not be able to use such high rates, a large number of nodes can, and a national research initiative is ongoing in an attempt to develop a gigabit channel for applications such as a national research network[3].

2 FDDI

FDDI[9] is a 100 Mbps fiber optics ring which is commercially available and currently being used primarily as a backbone for internetwork communication. The cost (about $10,000 per node) is a major factor prohibiting its use in workstations, but this is expected to drop significantly as the product matures. Given its likely widespread use, we investigate in this paper the effect of using a gigabit transmitter in this type of network.

FDDI is fundamentally a token ring network. The distinctive characteristics of the network are its use of fiber optics and associated high data rates, a dual counter rotating ring topology and a token holding timer algorithm to determine the length of time for which a node may hold the token and transmit data. Although FDDI is a dual ring, the second ring is primarily intended to allow for healing in the event of a damaged link[8]. For this reason, only one ring is modelled.

The token holding timer algorithm is one whereby each node keeps a local timer as a means of determining how long it can hold the token for transmission. It is intended to place a bound on access delay for synchronous traffic. Each time the token arrives, the clock is reset. If a sufficiently small amount of time has expired (less than an amount negotiated among the nodes called the target token rotation time, TTRT), data may be transmitted for TTRT minus the lapsed time on the timer when the token returned. At that point the token is released[1]. Consider the case where the TTRT value is set precisely at the level which will let every node transmit its data on each cycle(rotation) of the token. If TTRT is reduced by one-half, half of the nodes (actually less) will be able to transmit during each rotation. The overhead of passing the token becomes more significant and utilization is decreased. For a more detailed discussion, see [1]. In order to minimize this as

a factor, the TTRT value was set arbitrarily high (20 milliseconds) in these runs. Only asynchronous traffic is condsidered.

3 Parameters and Metrics

Clearly, extending the rates will improve performance over standard FDDI. Packet transmission times will be proportionally reduced and propagation delays will remain the same. The question is to determine which factors will have the greatest impact on such a network so that the environment in which it can best be utilized can be better understood.

It is anticipated that the predominant factors which will affect performance are

1. number of nodes,
2. length of the network, and
3. packet length.

For the simulation, each of the three parameters above have been tested over a range of three values each as follows.

Nodes	10, 100, 1000
Network Length	1Km, 10Km, 100Km
Packet Length	5K, 10K, 15K

Given the large bandwidth of the network, it is anticipated that large numbers of nodes can be supported. Length has been considered to include LAN, MAN and WAN scenarios and packet length varies from 5000 to 15000 bits.

The metric used to evaluate performance is delay. This is a measure of the time between arrival of the message at the node to delivery of the last bit of the message at the destination. Other metrics frequently used in network analysis include access delay(time to beginning of message transmission), throughput and fairness. Access delay is not graphed because we are concerned with the impact of distance on the network and want to include the impact of propagation delays as the distance is lengthened. In the results shown here, the only cases considered are those in which the network is less than fully loaded so that throughput is equal to offered load. Fairness of FDDI has been shown in [5]. There is no reason to assume that an increased transmission rate will affect fairness so it too has been ignored here.

4 Results

For the purpose of comparison, each of the following graphs use the same scaling. The x-axis represents load on the system in percent of the transmission rate of the

network. The y-axis represents delay in thousands of microseconds. The results of selected runs from the set of runs described previously are shown.

4.1 Nodes

In a typical token ring network, the number of nodes affects performance in two primary areas. First, there exists a delay introduced on the ring at each node which is using the network. Second, for each node capturing the token on a cycle around the ring, the token arrival to other nodes is delayed by an additional token retransmission. In addition, there is a delay between recognition of the token and transmission of the queued packet. This is explained in [1]. For these reasons the number of nodes has an adverse effect on performance, but the impact of number of nodes varies as explained below.

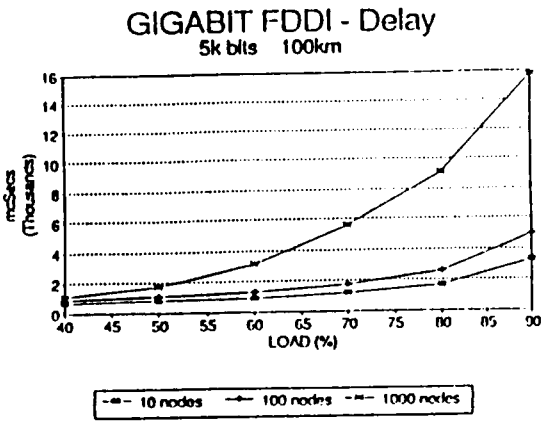
Figure 1 shows the effect of varying the number of nodes in four different scenarios. Vertically the graphs have the same packets size and horizontally they have the same distance. Note that in every case, the number of nodes has a negative effect, however the effect varies with certain combinations of the other parameters. A comparison of the graphs horizontally shows that if the load is distributed in smaller packets, the number of nodes has a greater effect than in if the load the packet size is larger. This can be explained by the fact that the overhead time required for a token capture does have an impact. As the packet size is smaller and thus distributed to more nodes, additional nodes capture the token on each cycle, introducing additional delays.

4.2 Packet Length

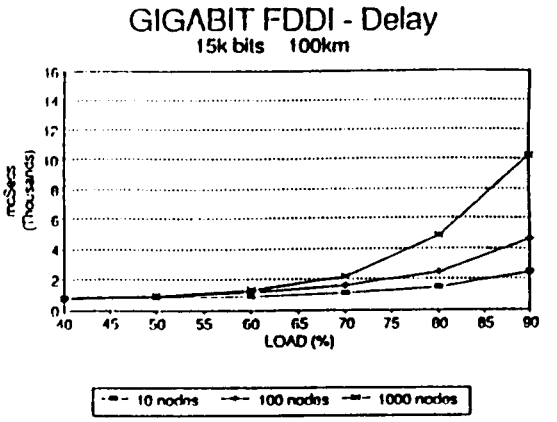
As described in the previous section, packet length and number of nodes, in combination, can have an effect on performance. Figure 2 reinforces the previous results. The three graphs show scenarios where the number of nodes equals 10,100 and 1000. Notice that in cases a and b, the effect of packet size is practically insignificant. However, when the number of nodes increases to 1000, the delay varies significantly. Case c shows that increasing the packet size from 5000 bits to 15000 bits cuts the delay in half for up to 80% and by a significant quantity for 90%.

4.3 Network Length

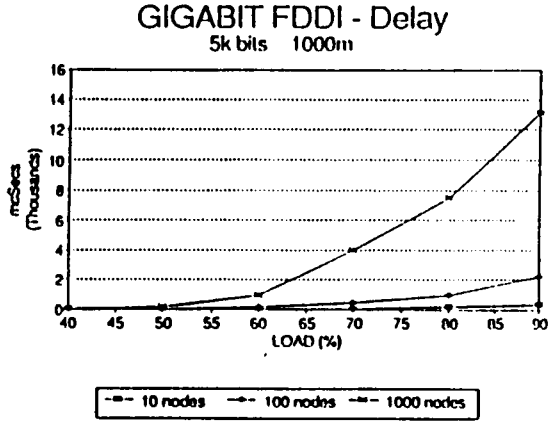
The last set of graphs in Figure 3 shows four scenarios similar to figure 1. One would likely anticipate that the impact of propagation delay is simply a matter of being relatively large for the 100Km case and proportionally less for the other two cases. Cases a and b indicate that although the increased length has a negative effect on the network, it has a worse effect as the number of nodes increase in conjunction with



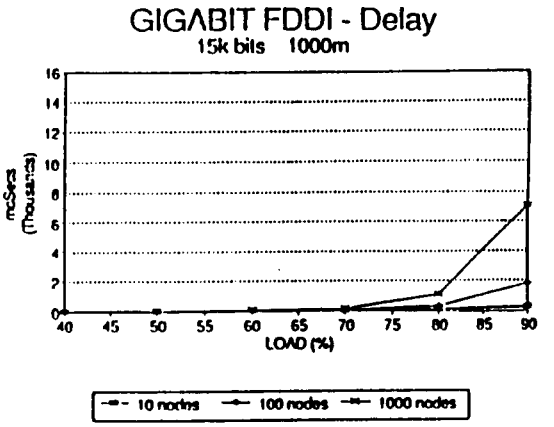
(A)



(B)

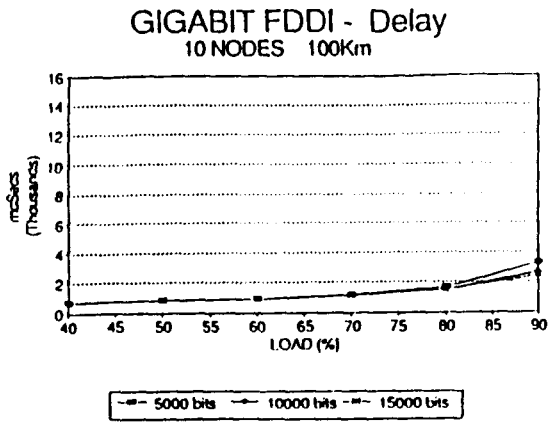


(C)

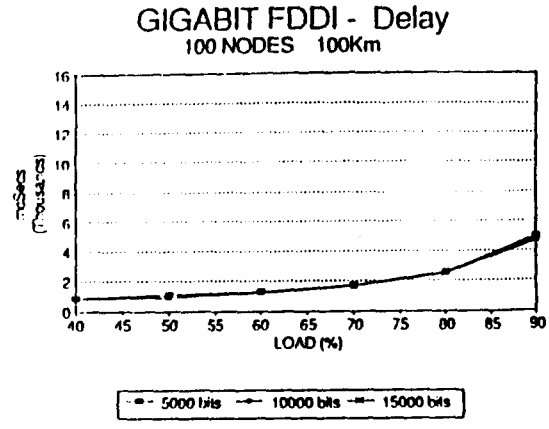


(D)

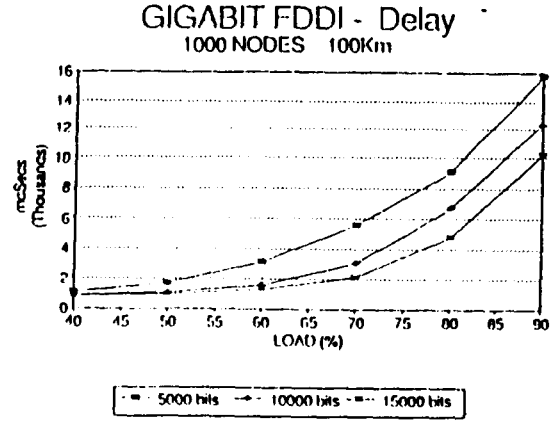
Figure 1: Impact of Nodes



(A)



(B)



(C)

Figure 2: Impact of Packet Size

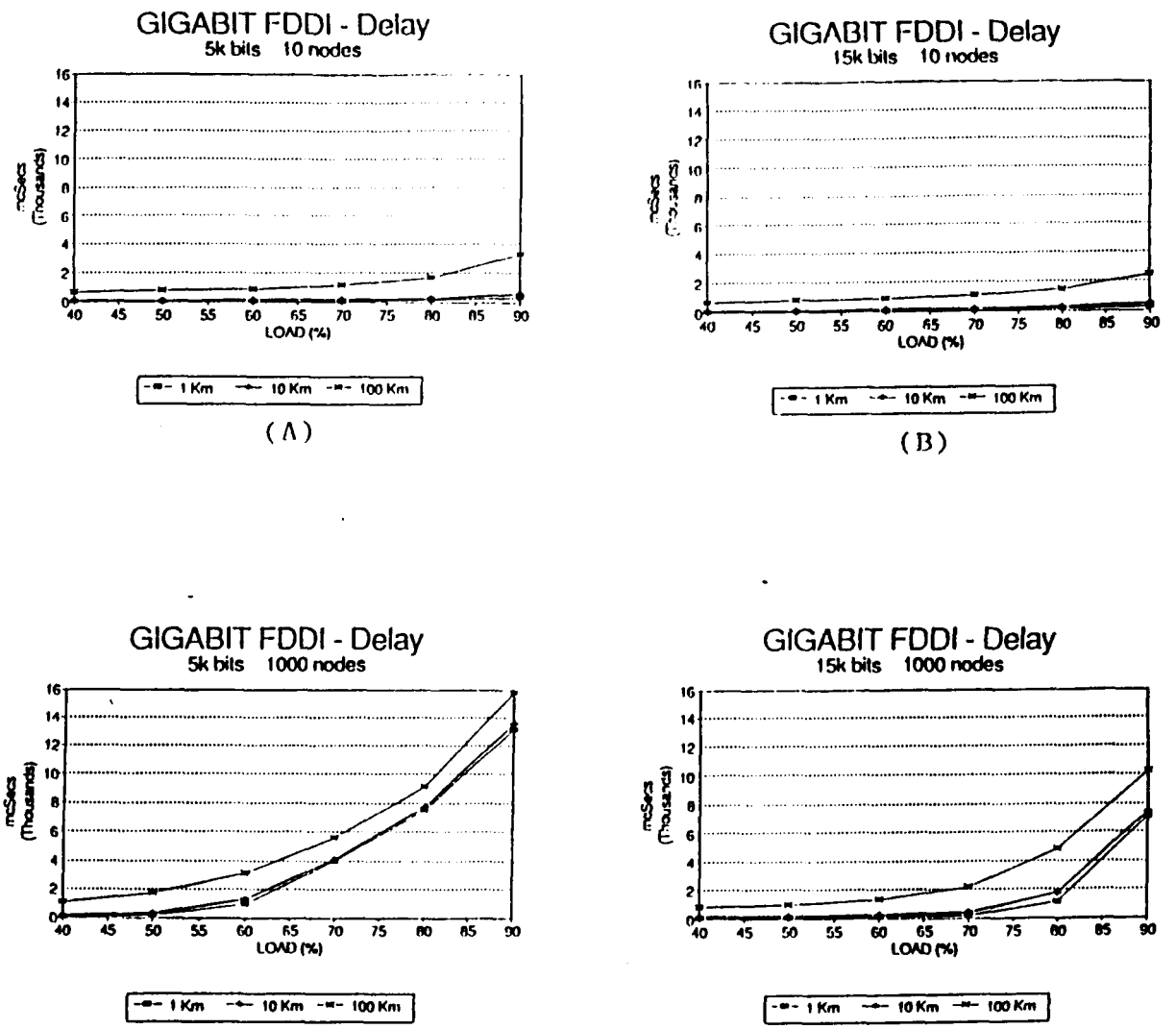


Figure 3: Impact of Network Length

the network length than if the packet size size is decreased in conjunction with the increased network length. In the cases a and b, delay is at or below 2 milliseconds for all values of distance over the entire range of loads.

5 Conclusions

This paper shows the effect of the number of nodes, network length and packet length for FDDI at gigabit speeds. Most of the results show that over the range of parameters examined, delay is on the order of a couple of milliseconds for loads below the 60% level. As load increases above 60%, the delay degrades at different rates depending on the specific case examined. The number of nodes is the one factor which has the greatest effect on performance of the three parameters considered. In addition, the number of nodes compounds the problems worse when increased in conjunction with reducing packet sizes.

Recent research has shown that the number of nodes can in fact be used to reduce the delay and increase throughput through a modification to FDDI. The reader is referred to [1]. Further research should investigate the degree to which the advantages of increased numbers of nodes in a modified FDDI can balance the disadvantages mentioned above and how to what extent increasing packet size will have an advantageous effect on performance.

References

[1] D. Game and K. Maly. *Extensibility and Limitations of FDDI.* Technical Report, Old Dominion University, February 1990.

[2] H.S. Hinton. Architectural considerations for photonic switching networks. *IEEE Transactions on Communications*, 6(7):1209–1226, August 1988.

[3] R.E. Kahn. A national network: today's reality, tomorrow's vision, part 2. *EDUCOM Bulletin*, 14–21, Summer/Fall 1988.

[4] M. Maeda and H. Nakano. Integrated optoelectronics for optical transmission systems. *IEEE Communications*, 26(5):45–51, May 1988.

[5] K. Maly, D. Game, L. Zhang, E. Foudriat, and S. Khanna. *Fairness Problems at the MAC Level for High-Speed Networks.* Technical Report, Old Dominion University, February 1990.

[6] T. Nakagami and T. Sakurai. Optical and optoelectronic devices for optical fiber transmission systems. *IEEE Communications*, 26(1):28–33, January 1988.

[7] K. Nosu. Advanced coherent lightwave technologies. *IEEE Communications*, 26(2):15–21, February 1988.

[8] F. Ross. Fddi - a tutorial. *IEEE Communications*, 24(5):10–17, May 1986.

[9] Draft Proposed American National Standard. Fddi token ring media access control (mac) asc x3t9.5 rev. 10. February 28, 1986.

Distributed Simulation of Network Protocols*

F. Paterra, C. M. Overstreet, and K. Maly

April 16, 1990

Abstract

Simulations of high speed network protocols are very CPU intensive operations requiring very long runtimes. Very high speed network protocols (Gigabit/sec rates) require longer simulation runs in order to reach a steady state, while at the same time requiring additional CPU processing for each unit of time because of the data rates for the traffic being simulated. As protocol development proceeds and simulations provide insights into any problems associated with the protocol, the simulation model often must be changed to generate additional or finer statistical performance information. Iterating on this process is very time consuming due to the required runtimes for the simulation models. In this paper we present the results of our efforts to distribute a high speed ring network protocol, CSMA/RN[1].

1 Introduction

Computer simulations of real world entities can be computationally intense tasks taking many hours or even days to run. Simulation analysis of network topologies and protocols is of this type. Because communication media speeds are ever increasing, a need exists for protocols which can fully use the newly available bandwidth. Their development, however, relies on the use of computer-based simulations. One promising method for lowering the time

*This work was supported in part by CIT under grant INF-89-002-01, by NASA under grant NAG-1-908, and Sun Microsystems under RF596043.

One such protocol currently understudy at Old Dominion University is a gigabit ring network called CSMA/RN. This a fiberoptic, carrier sensed, multiple access protocol that operates nominally at 1 gigabit per second. Because of the amount of simulated traffic that must be processed before the network reaches a steady state is very high, long simulation times are required. In addition, because each second of simulation time accounts for a large amount of data, hence simulated events, each second of simulation time take longer to process.

2 Model Decomposition and Processor Synchronization Schemes

While we are interested in using distributed simulation as a tool for studying high performance networks, questions concerning the effective distribution of a simulation model must be addressed: decomposing the model into cooperating processes, and insuring that the results of the distributed simulation match those obtained in a single processor version.

Key to efficiently utilizing concurrent computing techniques is determining an effective decomposition of the model. Because the environment which is most commonly available is a number of losely coupled workstation computers, connected via a low speed network, the overhead of interprocessor communication can easily exceed any time gains resulting from the concurrent computation. This implies that a successful decomposition will attempt to minimize the amount of interprocessor communication.

Three basic methods for performing model decomposition have been identified as workable for certain problem domains: server decomposition, physical model decomposition, and arbitrary decomposition.

The server method requires that the developer identify support functions to the simulation, such as random number generation or events list maintenance, that can be isolated and placed on separate processors. In this scenario one processor is left to run the main simulation while other processors provide the support functions. [2] and [3] have used this method successfully to reduce simulation times by as much as 80%.

To allow further study of the server model, Old dominion University has developed a suite of tools, described in [4], that can be linked to simulations

written in Simscript, C, or Pascal to provide the necessary communication and handshaking software for server model decompositions.

In contrast, physical model decomposition breaks the model into submodels corresponding to the components of the physical system being simulated. Different components share a single processor or may be execute on separate processors. This method results in a loosely coupled distribution which can, under some circumstance may reduce the communication requirements among processors. The drawback of only be able to utilizes many processors as there are concurrently operating components in the corresponding physical entity. This method seems easily applicable some communication network protocol simulations such as the CSMA/RN protocol described in [1].

The arbitrary method of model decomposition is to simply divide the model into as many parts as available processors, making the breaks where ever convenient from a programming point of view, such as programming modules. This results is a decomposition which may require more interprocessor communication, but could also allow greater concurrency during the run. There is little chance for reduction in simulation time with this method given the runtime environment. A better environment for simulations employing this decomposition would be a tightly coupled set of processors working with a pool of shared memory.

While each of these may provide some reduction in serial computer time, each only works well for some small problem domains and have not been shown to function well in the general case.

The second major problem faced by distributed simulation researchers is that of processor synchronization. Two methods are much discussed in the literature, the "conservative" and the "optimistic" methods. However, as with model decomposition, neither appears to work well for all problems.

Both the conservative and optimistic methods rely on developing a usable decomposition of the simulation model, and running each component on separate processors. The conservative method allows the parallel execution of events as long the events can be insured to be safe. A safe event is one who's inputs are fully defined and will not be directly or indirectly affected by the output of any other event. A more complete description of the conservative method synchronization can be found in [5].

The optimistic method, also called Time Warp, allows all processors to execute the events as soon as they are available, regardless of their relative logical times and the state of their inputs. As inappropriate timing sequences

are detected, the simulation for the node with the inappropriate sequence is rolled back to a early time when the simulation was known to be correct and then restarted. In order to "roll back," the state of the computation on each processor must be saved at various points, so that that process state can be restored if that processor must be rolled back. Saving the states may incur more overhead than the benefits gained by the distributed computation. A good introduction to the optimistic or Time Warp method of synchronization can be found in [6].

3 Experiences

Four possible decompositions of the CSMA/RN model have been studied, all are described below.

3.1 Physical decomposition, one node per processor

The first attempt resulted in a decomposition of the events list based on node. Each node in the network was assigned to a physical computer and each computer maintained its own events list.

We found that this decomposition resulted in a I/O bound, very tightly coupled model that was executed in lock step form, with all processors waiting for a single resource, the ring media. Because CSMA/RN is a carrier sensed network, messages being transmitted may be interrupted if another messages passes the sending node. As the simulation runs, new messages enter the ring and can either 1) be successfully placed on the ring or 2) are interrupted by passing messages. This means that each node, before transmitting a message, must determine that the ring in front of the node is not occupied and must have knowledge of the next time a message will pass in front of the it in order to properly terminate the transmission (as complete or interrupted). In addition, because the network has a ring topology, any node can influence any other node, so global information about the state of the ring is required by all nodes before each transmission. Finally the amount of computation at each processor between interprocessor communication operations is very small therefore the processor spend most of their time waiting for I/O and comparatively little time performing the simulation. This decomposition resulted in longer runtimes than the single processor model.

3.2 Physical decomposition, one node per processor with replication

The second method studied also assigned a separate processor to each node in the network, however because global information is required by each node, all parts of the simulation that could effect the node was run on each local host.

This decomposition resulted in a very loosely coupled model that required virtually no interprocessor communication. Each processor was loaded with very computationally intensive code, but resulted in most of the simulation being replicated on each processor with corresponding replication of computations. In order to determine which operations, all operations of the simulation were independently studied for interoperation dependencies. The complete analysis can be found in the appendix, however, the collection of statistically data was identified as the only opertations who's execution did not require global information and therefore could be executed separately for each node in the network. If statistics collection consumes a large percentage of the runtime, then the amount of statistics time would be divided by the number of nodes in the network. Unfortunately, as shown in the analysis, statistics collection represents a very small percentage of the runtime so observed speedup would have been minimal.

3.3 Segmented ring

The third method studied was an attempt to increase the amount of processing on each physical processor and reduce the amount of interprocessor communication required. The network ring was to be segmented, as shown in the figure below, and placing a number of simulated nodes on each processor.

This method to suffers from the same some of the same problems as the first decomposition described. There is a high level of computation than in the first method, however, at the points in the simulated network where the ring must pass to a new physical processor, the execution be comes lock stepped again. The resulting simulation allowed one processor, representing a bank of connected nodes, to operate for the length of time equal to the simulated network propagation delay for the two nodes that reside on the beginning and end of the adjacent ring segments.

With this and the first decomposition studied, the problem of deadlock

Figure 1: Segmented Ring

had to be addressed. Each processor in cooperating in the simulation must communicated with the processor representing the nodes behind its nodes on the ring. Because this is a ring interprocessor breaks of the, there is no concept of beginning or ending to the network so the dependencies are circular. A scheme to insure that the simulation progress was developed, however because the the speedup would be limited, if observed at all, it was not fully detailed.

3.4 Server Decomposition

One observation made during our tests was that if data could be provided by one module to a second one without a time dependent cycle developing, the synchronization needs are much less. The major problem being addressed currently is the development of methods for model decomposition that reduce the amount of intermodule time based dependency.

Using static code analysis tools previously developed at Old Dominion University, we are performing data flow analysis of existing simulation models, written in the SIMSCRIPT, C, and Pascal languages, to determine the prevalence of code sections which either supply and/or consume time inde-

pendent data objects during the simulation run. One major problem maybe the the relative costs of computing the data objects. Simple objects, such as random numbers, can be computed quickly, so the overhead of communication for can easily exceed ny benefits derived from computing the values on other machines. As discussed in [4], one solution to this problem may be to bunch the data objects and send them in quantity. This results in an inventory model with a reduction in communication overhead.

Additionally, we believe that code containing time dependent data cycles can be distributed if there is sufficient computation time between data requests to allow for the synchronization to occur or that the dependencies are not tight, one generation per synchronization, so that values can be precomputed and the simulation can be made to proceed.

4 Conclusion

Distributed simulation is a very hard problem [7]. Simulations of very tightly coupled systems such as network protocols that share a common resource have proven to be more difficult due to the amount of shared information that is required. Initial efforts in developing decompositions along physical lines has proven fruitless. In order for a distributed simulation to provide reductions in runtime, the modules must be designed so that the can perform compute intensive operations and require very little intermodule communication.

Currently we have simulation models for the FDDI[8], DQDB[9] (formally QPSX), and CSMA/RN[10] protocols available for analysis. To support the distribution of modules detected, we will use tools described in [4] to provide interprocessor communication and server model synchronization. These tools may need to be extended to allow for two way data flow and synchronization under a request and deliver scheme.

A Appendix – Analysis of CSMA/RN Replication Distribution

A.1 The data set

The data set was developed to show four scenarios when simulation the network; A message immediately sent, A message forced to wait, A message interruption, and a neighbor to neighbor transmission.

From	*To*	*Length*	*Time*
A	B	2	1
C	B	2	2
A	C	3	4
C	B	2	5
A	B	2	10
B	C	2	10

The network being simulated has three nodes, equally spread 1 time unit apart to form the ring

$$A \Longrightarrow B \Longrightarrow C \Longrightarrow A$$

A.2 Events and operations

Seven operations have been identified for the operations analysis of this algorithm.

1. A new message begins transmission
2. The start of a message passes through a node without causing an interruption
3. A message interrupts another
4. A message reaches its destination
5. A message completes transmission [1]
6. The end of a message is passed through a node
7. An interrupted message is restarted

Each of these operations can be further broken into suboperations to which execution times can be attached.

[1] That is the message has completely left the originator

1. A new message begins transmission
   (a) Copies of a Start event are placed on the *Actual* list for all nodes from the originator to the neighbor preceding the destination.
   (b) An End event is placed in the *Possible* list for the neighbor of the originator
   (c) The *Transmitting* array is updated to indicate that the originator is currently transmitting.
   (d) The delay for the Start is added to the packet delay
2. The start of a message passes through a node without causing an interruption
   (a) The Start event for the executing node is removed from the *Actual* list
   (b) The *Transmitting* array is updated to show that the node is no longer transmitting.
3. A message interrupts another
   (a) Fracture count is incremented
   (b) Copies of an Interrupt event are placed on the *Actual* list for all nodes from the originator to the preceding neighbor of the destination.
   (c) Remove the Start event, that caused the interruption at the current node, from the *Actual* list
   (d) Compute the number of bits transmitted
   (e) Compute the number of bits forced to wait
   (f) Add the number of bits transmitted to the count of bits delivered so far
   (g) Remove the interrupted message's End event from the *Possible* list
   (h) Place new Start and End events on the *Possible* list
4. A message reaches its destination
   (a) NO ACTION REQUIRED
5. A message completes transmission
   (a) The corresponding End event is moved from the *Possible* list to the *Actual* list and copies of it are posted for each of the nodes from the originator to the neighbor preceding the destination node

(b) The total delay for the message is added to the total message delay
(c) The number of bits in the last packet message is added to the total bits delivered
(d) The transmitting flag is reset

6. The end of a message is passes through a node

   (a) The End or Interrupt event is removed from the *Actual* list
   (b) The *Transmitting* array is updated

7. An interrupted message is restarted

   (a) Copies of a Start event are placed on the *Actual* list for all nodes from the originator to the neighbor preceding the destination.
   (b) An End event is placed in the *Possible* list for the neighbor of the originator
   (c) The *Transmitting* array is updated to indicate that the originator is currently transmitting.
   (d) The delay for the new Start is added to the packet delay

All of the subevents are weighted as taking 1 unit of execution time except subevents 1.a, 3.b,5.a, 7.a which take $(n/2)$ and 4.a taking 0 units[2].

A.3 The simulation

The following is a table of the operations executed and their ordering for the data set given in section A.1.

[2]n is the number of nodes in the network being simulated

This information can be gotten from section A.3.

Operation Class	Number Required	Units per Instance	Total for Class
1	6	5	30
2	3	2	6
3	1	9	9

Time	Operation/ Node	*Transmitting* Array A	B	C
1	1-A	1	0	0
2	1-C	1	0	1
2	4-B	1	0	1
3	5-A	0	0	1
3	2-A	1	0	1
4	4-B	1	0	1
4	5-C	1	0	0
4	6-A	0	0	0
4	1-A	1	0	0
5	1-C	1	0	1
5	2-B	1	1	1
6	4-C	1	1	1
6	3-A	1	1	1
7	6-B	1	0	1
7	4-B	1	0	1
7	5-C	1	0	0
8	6-A	0	0	0
8	7-A	1	0	0
9	5-A	0	0	0
9	2-B	0	1	0
10	4-C	0	1	0
10	6-B	0	0	0
10	1-A	1	0	0
10	1-B	1	1	0
11	4-B	1	1	0
11	4-C	1	1	0
12	5-A	0	1	0
12	5-B	0	0	0

[3] J. Comfort. The simulation of a microprocessor bassed event set processor. In *Proceedings of the Fourteenth Annual Simulation Symposium*, pages 17–33, 1981.

[4] F. Paterra, C.M. Overstreet, and K. Maly. Distributed simulation: no special tools required. 1990. Submitted to the 1990 Winter Simulation Conference.

[5] Jayadev Misra. Distributed discrete-event simulation. *ACM Computing Surveys*, 39–65, 1986.

[6] David Jefferson. Distributed simulation and the time warp operating system. *ACM SIGOPS*, 77–93, Nov. 1987.

[7] Peter A. Tinker and Jonathan R. Agre. *Object Creation, Messaging, and Stat Manipulation in an Object Oriented Time Warp System.* Society for Computer Simulation International, Mar. 1989.

[8] W. E. Burr. The fddi optical data link. *IEEE Communications*, 24(5):18–23, May 1986.

[9] R. M. Newman, Z. L. Budrikis, and J. L. Hullett. The qpsx man. *IEEE Communications*, 26(4):20–28, April1988.

[10] E. Foudriat, K. Maly, C. M. Overstreet, D. Game, S. Khanna, and F. Paterra. Csma/rn a protocol for high speed fiber optic networks. 1990. Submitted for publication WHERE? — ED IS THIS THE CORRECT TITLE??

Modelling High Data Rate Communication Network Access Protocol *

S. Khanna, E. C. Foudriat, F.Paterra,
K. Maly, C.M. Overstreet
Old Dominion University
Norfolk, VA 23529

April 23, 1990
Draft

1 Introduction

Modelling of high data rate communication systems is different from the low data rate systems. Unlike the low data rate systems a message does not fill the whole network structure, so there can be many messages on the system at one time. It implies that more than one event may take place at a time and it is impossible to model the network by treating messages as entities which start and end before other events take place. Previous experience with simulations of networks where only a single message was considered indicated that protocol models are complex and simulations usually take a long time to run on the computer.

Three simulations were built during the development phase of CSMA/RN modelling. The first was a model using Simscript was based upon the determination and processing of each event at each node. The second simulation was developed in C based upon isolating the distinct object that can be identified as the ring, the message, the node, and the set of critical events. The

*This work was supported in part by CIT under grant INF-89-002-01. by NASA under grant NAG-1-908, and by Sun Microsystems under RF596043

third model further identified the basic network functionality by creating a single object, the node which includes the set of critical events which occur at the node. The ring structure is implicit in the node structure. This model was also built in C.

In this paper, we will discuss each model and compare their features. It should be stated that the language used was mainly selected by the model developer because of his past familiarity. Further the models were not built with the intent to compare either structure or language but because the complexity of the problem and initial results contained obvious errors, so alternative models were built to isolate, determine and correct programming and modeling errors. The next section discusses the CSMA/RN protocol in sufficient detail to understand modelling complexities. In the following sections, each model is described along with its features and problems. Following this the models are compared and concluding observations and remarks presented.

2 Description of CSMA/RN protocol operations

The network access controller for CSMA/RN is shown in Figure 1. The incoming signal is split into two streams, one through a delay line or buffer. The node controller, based upon information accumulated in the buffer, is required to make a number of decisions. First, it must detect the presence of incoming data; if it exists, the node must always propagate incoming information as the outgoing signal to the next node on the ring because it would be impossible to recreate the packet unless sufficient storage is provided. If no incoming packet exists, the node is free to place its own data on the ring if its queue is not empty. However, during the time this latter data is being transmitted, if an incoming packet arrives, then the node, within the time limits dictated by its buffer size, must discontinue its transmission and handle the incoming packet. Hence, packets once on the ring take precedence over the insertion of new packets.

Packets are tested at each node to determine if the incoming packet is destined for this node and should be copied to its incoming data buffer (not shown in Figure 1). In addition, to improve the network operation, packets are removed at the destination so the node can use the free space to send

information waiting in its queue. Destination removal improves performance under uniform loading by a factor of two [].

Figure 2 illustrates the events that can occur at each node based upon the travel of empty and full packets of data around the ring as time progresses.

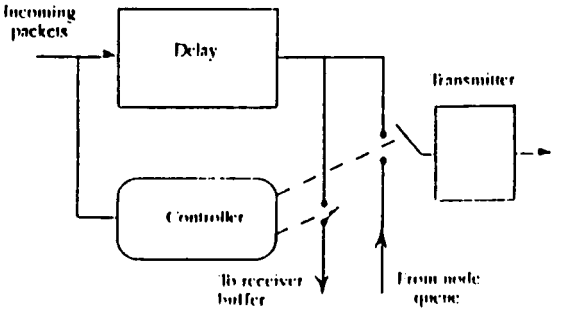


Figure 1 CSMA/RN Access Controller Logic

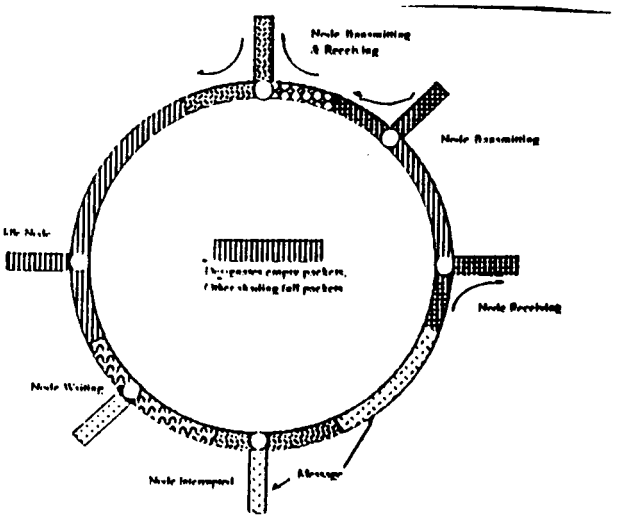


Figure 2. Illustration of Network Operational Conditions

3 Simulator I - Simcript - Node Event List

3.1 Model structure

The first model attempted for the simulation of CSMA/RN was a SIMSCRIPT based model consisting of approximately 900 lines of SIMSCRIPT code. The simulation was written to be event driven, rather than process driven, and has four event types. Because of memory constraints, outgoing messages are only generated when there exists and opportunity to transmit. because of this, all events are initiated by external traffic passing in front of the node. The events are defined as follows:

- start.of.message given node.id, destination, and message.id – executed when a new message is seen passing in front of a node. This event

terminates any internally generated message that is currently being transmitted and places it in a queue for retransmission, sets a busy flag. If the message seen is not for the current node, a **start.of.message** event of posted for the next node in the ring.

- **end.of.message given node.id, destination, length, and message.id** – This event is executed when the end of a messages passes in front of the node. The busy flag set in **start.of.message** is reset and statistics are updated. As with **start.of.message**, an **end.of.message** event is posted for the next node in the ring. After this event completes execution, the current nodes has an opportunity to transmit any data pending.

- **interrupt.message given node.id, destination, length, and message.id** – This event is similar to the **end.of.message** event, but is initiated when a message is interrupted during transmission. As this event is propagated through the nodes, the processing is the same as that of the **end.of.message** event.

- **check.flag given node.id** – When a locally generated message begins transmission at a node, a flag is set to indicated that the node is currently transmitting. This flag inhibits the node from trying to transmit more than one message concurrently, but it must be reset when the message completes transmission. To reset this flag at the proper time, the event **check.flag** is scheduled to execute when the message terminateds transmission. This event resets the flag and, if the message was not interrupted, posts an **end.of.message** event for the next node in the ring.

In the above events, the parameters are defined as follows:

- **node.id** – This is the number of the node that must execute the event.

- **destination** – This is the destination of the message.

- **length** – This is the length of the message, or partial message in the case of an interruption that was sent.

- **message.id** – This parameter is used to insure that all messages are processed in order.

4 Three Object Model

4.1 Model Structure

In this model of CSMA/RN three distinct structures, the ring, the node and the event list, were manipulated by code related to each structure. While they were not identified as objects per se, as in C++, they were separate code blocks in the later versions of the program. In the following sections, we will describe each unit.

4.1.1 Ring Structure

The ring is defined by the number of bits that can exist simultaneously. For example, a 10 km length ring with a 1 Gbps data rate assuming media speed of 5 microsec/km contains 50,000 bits. The ring is modeled as a doubly linked list of data structures containing the necessary data for a packet (a packet is a contiguous portion of a total message). The packet data includes the bit position of its begin and end location on the ring, its length in bits, the packet condition (e.g., free, in use, etc.), message information including source, destination and message number, and left and right pointers to the packets on the ring. Ring operations include updating the packet locations by the increment of time for the next event and linking,delinking,creating and combining packets.

4.1.2 Node Structure

The nodes were defined to be at fixed bit locations on the ring. The nodes were modeled as an array of structures with each succeeding node at a higher bit location. The node data structure contained considerable information including:

1. node number, location and distance to previous node;
2. message details e.g. destination, length, timing data;
3. operational statistics on network performance; and
4. a pointer to the packet presently at the nodes location.

The procedures relating to node operations include collection of operational information, new message generation, updating the packet pointer as packets progressed around the ring and handling the events which occurr at the node.

4.1.3 Event Structure

The event structure and its operations were fairly standard. The event list was a doubly linked list with an event type, pointers to the node and/or packet related to that event, and pointers to complete the list. Procedures related to event processing consisted of creating the event and linking it into the event list either from the head or tail.

The main program is an event handler. The next event is read, the ring and nodes updated as needed, and the event processed. The event can change the ring and/or node conditions. After the event is handled, those nodes which are ready but do not have free packets assigned for their ready message are processed so that if a free packet is available then it can be assigned.

4.2 Experiences

The major difficulties in developing the ring system were in programming the correct wrap around conditions for the ring position and for the various tests relating node locations to packets on the ring. The combining of empty packets is necessary to reduce the number of events as empty spaces are filled more readily. For updating packet pointers, it was found that it had to be checked each time packets were created, removed or moved, since any of these conditions could change the node to which packet pointed.

Table 1 shows the breakdown of subroutine code for each major structure. The general code includes I/O, initialization routines and generic procedures.

Item/Code Unit	Ring	Node	Event	General
Procedure Count	5	7	3	5
Lines of Code	100	332	121	252

Table 1. Procedure Code to Support Network Simulation System

The major problem of coding and debugging was the identification of the event interface between the packets on the ring and the nodes. Initially, two

events were described, the insertion of a new packet at a ready node and the removal of a packet which had reached its destination. In order, to schedule packet insertion, the node, when ready, looks at the packet at its location. If it is empty, a new packet is created starting at the present location; if the packet at the location is full, the node searches arriving packet to identify an empty packet. If an incoming empty packet is found, an event is created for its arrival time at the node. Arriving full packets are emptied and the node checked to see if it has a ready message.

The major event handling difficulties arise due to potential interaction of these events with adjacent up stream nodes. First, if an arriving full packet length encompasses the previous node(s) the packet can not be completely emptied or else the previous nodes may incorrectly identify the packet as empty and use it. Thus, the packet must be truncated at the previous node and a pseudo arrive event created to take care of new event. For very long packets a number of subsequent pseudo events may be necessary.

Very similar problems occur for filling empty packets and searching for empty packets for a node to use. Packets can not be created for a length greater than the previous node since that node may become ready and occupy a part of the empty packet. Alternatively, a node can not identify for use an empty up stream packet which a prior node may use before the packet arrives at the node in question. This creates handling problems which make the originally simple events quite complex and account for much of the code and most of the programming problems.

5 One Object Model

The model is based on the fact that the carrier sensing is local to a node and the nodes implicitly refer to the ring. The main events which result from the local carrier sensing are :

- A node receives an upstream message not destined for itself and gets interrupted and starts reposting the upstream message.
- A node is transmitting a message which is on the head of its queue.
- A node is idle i.e. node has an empty queue.

The unit of measurement in the model is a bit.

5.1 Node Structure

The ring is modelled as an array of nodes and broadly speaking each node has the following structure -

- The Node Status: Transmitting, Idle or Reposting;
- Head of the Queue: Information about the message which is on the head of the imaginary queue at a node. For example, arrival time of the message, message length, its destination etc. In this model a queue never exists but a new arrival is scheduled whenever a message is fully transmitted which may be well in future or way back in past;
- The Interrupt Timetable: A linked list of interruption times for a node and the duration of interruption. It lists the time a node will pre-empt the current message if it is transmitting (and resume later) and start reposting for the duration mentioned in the list;
- Statistics: Network performance satistics.

5.2 Interaction

The operation of the model is mainly goverened by a schedular,Next Time Generator, which looks at local conditions at all the nodes and then decides the next time when a set of events will fire at different nodes. The resulting flow is depicted in the figure 3.

5.3 Experiences

Two or more events may occur simulatneously on the ring. The schedular has a tight job to decide about the next time it will return. Also, the interrupt timetable has to be maintained carefully enough else the packets on ring will start cramping on each other.

The ring under heavy load may have a state where all the messages can be in a bumper to bumper situation. If a node finds this state of the ring in its interrupt timetable, it will continue to repost till it finds a hole on the ring.

For packets of sizes equal to the bitlength between two consecutive nodes, a proxy-completion of the message is caused which is identical to $buf = 0$ state in the flow given in fig.3. The proxy-completion avoids a phenomenon where each node checks for its transmit complete and next node's interruption before interrupting and finishing transmission.

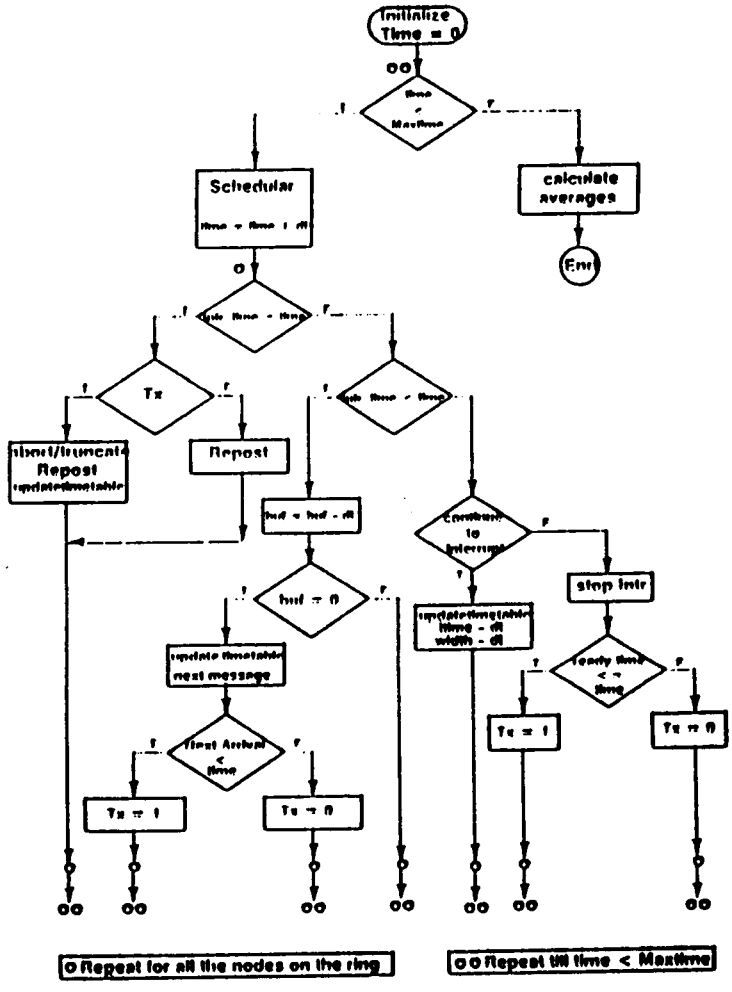


Fig.3 Flowchart for one object model

6 Discussion

Decisions are based on the local conditions at a point in the network and not on global conditions. Even if the decisions are simple, complexity may occur because prior decisions when propagate, influence the present network condition at some other point in the network. Conditions on the network evolve. Each bit or indivisible block of network should be modlled so that it and its effects on the network conditions can be developed.

Various types of runs were made to study simulator confidence. Data were collected for intervals during a run and compared as to their variability and to the mean of all data collected for the run. We plotted wait time, the most sensitive of the variables, taken at the end 10 intervals, and the cumulative average taken of the active period of the run. Load fractions of 1.0 and 1.5 were used since at the higher loads, fluctuations tend to be greater. First, it was found that the ring tended to reach steady state values rather quickly, but its results still varied considerably between intervals. It was found that in order to obtain data with a 90% confidence in the mean accuracy, the ring had to cycle a number of times, where a cycle is the time for information to completely traverse the ring. In general, about 1000 - 5000 cycles was found to be sufficient elasped time. Figure 4 shows a comparitive result for wait time analysis for a 10 nodes, 10 Km, 1Gbps, and 2 Kbits messages.

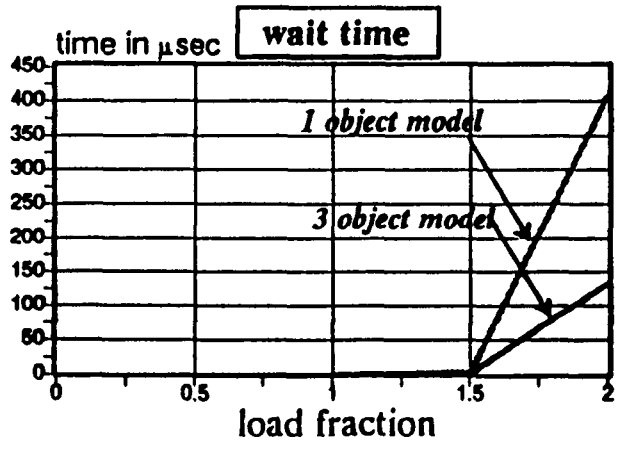


Fig.4 Comparitive results for wait times

7 Concluding remarks

To date, CSMA/RN studies have been limited to simple asynchronous data operational conditions. Additional study is required to document its performance for messages with variable lengths, for non-uniform load conditions, for conditions where ring domination by a few nodes can occur, and for large node count conditions where message fracture is most likely. Protocol procedures must be developed and studies must be done for CSMA/RN to effectively handle integrated traffic, i.e., synchronous traffic consisting of voice and video data in conjunction with asynchronous messages. It means that the model's capability to handle complex decisions needs expansion as operational features of the protocol become known; thus adding capability to the model further increases its complexity.

TECHNICAL REPORTS

DEPARTMENT OF COMPUTER SCIENCE

Technical Report # TR-90-14

Extensibility and Limitations of FDDI

David Game, Kurt Maly

Old Dominion University
Department of Computer Science
Norfolk, Virginia 23529-0162
U. S. A.

03-06-90

Old Dominion University
Norfolk, VA 23529-0162

Extensibility and Limitations of FDDI

David Game
Kurt Maly
Department of Computer Science
Old Dominion University
Norfolk, Virginia 23529-0162

March 6, 1990

Abstract

Recently two standards for MANs, FDDI and DQDB, have emerged as the primary competitors for the MAN arena. Great interest exists in building higher speed networks which support large numbers of node and greater distance, and it is not clear what types of protocols are needed for this type of environment. There is some question as to whether or not these MAN standards can be extended to such environments.

This paper investigates the extensibility of FDDI to the the Gbps range and a long distance environment. It does this first by showing which specification parameters affect performance and providing a measure for predicting utilization of FDDI. A comparison of FDDI at 100Mbps and 1Gbps is presented. Some specific problems with FDDI are addressed and modifications which improve the viability of FDDI in such high speed networks are investigated. [1]

[1]This work was supported by CIT grant RF-89-002-01, NASA grant NAG-1-908 and Sun Microsystem grant RF 596043.

1 Introduction

Network data rates are commercially available at rates in the 100Mbps per channel class. The two most prominent of those are competing MAN standards FDDI[20] and DQDB(QPSX)[5] and much research is currently ongoing in an effort to better understand their performance capabilities and limitations[8,2]. Research, however, is going forward and a national research initiative is underway to develop Gbps networks to be employed as a backbone for a national research network[9].

Many questions still remain as to the approach which can best suit the requirements of such a network. A national network will likely transport synchronous and asychronous traffic, support large numbers of nodes (at least 100 and likely over 1000), and be spread over very large distances (over 1000 Kilometers). The impact of considering these types of parameter ranges can be very negative for token rings due to increased token cycle time and for CSMA/CD due to the increased slot times. No known research exists to show how FDDI and DQDB are affected. Current national networks are very slow packet networks on the order of 56Kbps. As we move towards Gbps speeds, the networks will be much more expensive and efficiency will become a much more important factor.

Increased data rates could be accomplished by focusing on the development of transmitter/receiver devices which are capable of functioning at such high rates[7, 11,16,18], i.e. just build gigbit speed lasers. There are of course numerous problems associated with such high speed devices other than the transmitter/receivers themselves, such as how to build computers which can process data at the rate of the network and what types of protocols would work best at these rates. For example, [9] suggests that it might be necessary to structure packet sizes to be large in order to minimize overhead impact.

Another approach is to examine how current transmitter/receiver technology can be optimized to improve characteristics such as throughput and delay. Parallel channels show some promise for improved performance [13,12,14] and is the subject of some research. Many architectures have been proposed in an attempt to design more efficient networks. Strategies have included 'train' protocols [22,21,10], hybrid CSMA/CD protocols[3,15,13,12,4], slotted and register insertion rings[6], and numerous others.

In this paper I will examine the viability for scaling FDDI, a 100Mbps token ring protocol, to the type of environment mentioned above. A number of problems will be defined and a suggestion for performance improvement will be given. The suggestion for performance enhancement is applicable to token ring networks at any speed and distance and provides a framework for improving other types of networks.

2 FDDI

2.1 Basic Token Ring

A token ring network is distinguished by the manner in which transmission rights are granted to nodes on the network. The token packet circulates on the ring, passing by each node. If a node's queue is empty, it simply continues circulating the token to the next node. However, if the node has a message in its queue to transmit, it will effectively remove the token by changing some information in the packet. The token itself is not actually removed, but is transmitted in some altered form such that subsequent nodes will not see it as a token and will not attempt to transmit. The modified token continues circulating and is eventually consumed (not retransmitted) when it reaches the node which is in the process of transmitting (holding the token).

Figure 1 illustrates how the token is 'removed'. Node 4 has a message for node 2 and is waiting for the token before initiating transmission. Figure 1.*b* shows that the token has been modified so that it will not be recognized as such and the message has been placed on the network. Part *c* shows that the token has been retransmitted on the network and is available for subsequent use.

As the token continues around the ring, subsequent nodes remove the token and append another packet. This scenario is illustrated in Figure 2 where node 1 has a message for node 5. In order to send the message, the token is modified, the message for node 5 is transmitted and a new token is made available to the other nodes. The network becomes filled with messages and *old tokens*. At most one real token is on the network at any instant of time.

The exact time at which the *old tokens* and messages are removed is dependent upon the distance of the network, the length of the packet and the data rate of the network. The most important factor to note is that all nodes except the node holding the token are forwarding messages. The node holding the token does not forward the incoming message but instead forwards its own message. Incoming messages are lost. Eventually, the *old token* will encounter a node which is in the process of transmitting a message and be removed.

Data packets are removed in a slightly different manner. It is important that they be removed by the sender so that a receiver will not accept the message a second time if it recirculates. The tail of the message is removed(modified) at the sender once the address is recognized, whereas the fragments of the headers of these packets are removed as mentioned above[19].

2.2 Token Rotation Time

The decentralized access mechanism of a token ring protocol can place limitations on how long a station has access to the network once it obtains the token. The

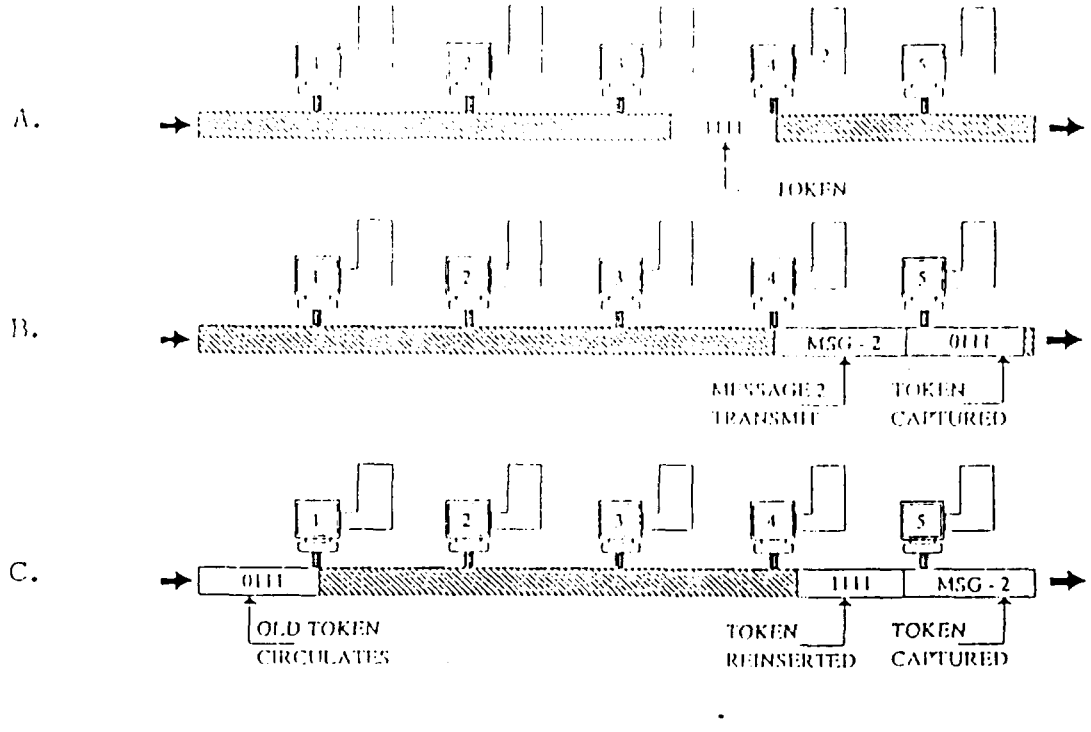


Figure 1: Token Ring Protocol Token Capture

approach in FDDI is to limit the amount of *time* for which one staion can hold the token[23]. Each node has a timer which is reset when the token arrives. When the token returns, the node may capture it only for an amount of time which will assure that the token will return within a specific time period. This time period is the *Target Token Rotation Time*, $TTRT$. The value of this parameter is negotiated amongst all nodes on the network and is essentially the smallest value selected by any node. This defines the maximum amount of time between access to the network and should provide for synchronous traffic.

Although it can be shown that the algorithm will guarantee that the token will return within the negotiated time frame on the average[8], it can not be guaranteed that the node will be able to hold the token at all once it returns. This has serious implications for periodic traffic and maximum throughput and will be examined in the next section.

3 Impact of Token Rotation Time

Johnson[8] provides analysis concerning the timing requirements of FDDI for both the ideal and the non-ideal(including overhead) cases. For the ideal case, the token can always be guaranteed to return to the station within $2 * TOPR$ where $TOPR$ is the current operating value of the $TTRT$, within which the token should return. For example, if the currently negotiated value of TOPR is 125 μseconds, then the token can only be guaranteed to return within 250 μseconds. It would appear then that one would simply negotiate for one-half of the desired $TTRT$ and then the proper availability of the token could be assured. For reasons of maximizing utilization of

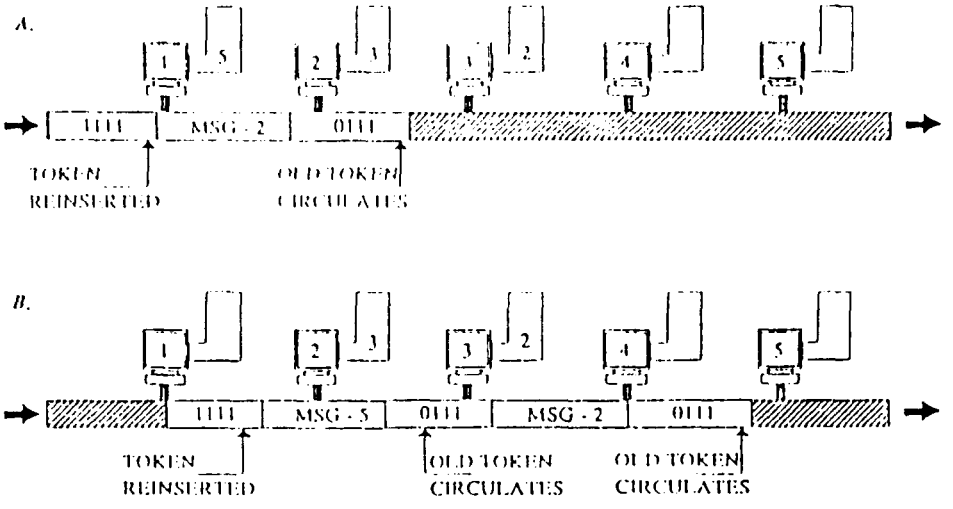


Figure 2: Token Ring Protocol Without Removal at Destination

the network, there is compelling motivation to have a large value of $TTRT$ thus resulting in a tradeoff between the design objective to support synchronous traffic and the need for high utilization. The following section illustrates the impact of $TTRT$ on utilization.

3.1 Parameters affecting TTRT

As cited in [8], the primary components of ring overhead are as follows.

- Total Propagation Delay (D_{prop}) is determined by multiplying the propagation delay for fiber optic media (5085 ns/km) by the length of the network.
- Latency (L_n) occurs at each node and is effectively the delay between the time a bit arrives as a node and departs the node. Therefore, if one examines the round trip of a single bit around the network, the delay is increased by the latency at each node times the number of nodes. L_{tot} represents the total latency of the network($L_{tot} = N \times L_n$).
- The number of nodes to capture the token, N_c, increases the delay of the token rotation. In the minimal delay case, no node needs the token and this component is nothing, but no information is transmitted. By focusing upon the process of transmitting a frame at a single node, this overhead becomes apparent. The head of the token arrives at the node and is passed on to the rest of the network while the node waiting to transmit identifies this as the token. Recognition takes place before the tail of the token is retransmitted, providing the capturing node the opportunity to modify the end of the token,

transforming it into a non-token frame, and thereby capturing the token. The delay required to accomplish this is incorporated into Latency, L_n. The node then proceeds to transmit its packet and *retransmit* the token to its neighboring node as explained in the section Basic Token Ring and Figure 2.

- As each node captures the token and retransmits it, an additional delay equal to the Token Transmission Time (T_t) will be incurred. Note that the delay from message transmission does not contribute to *overhead* delay.

- The design specifications of FDDI[20] allow for a maximum Transmitter Idle Time(T_i). This represents the time which is required between recognition of the token by the node and beginning of transmission of the frame. As in the previous item, this is only a factor when nodes are actually capturing the token.

Specified times for these delay components can be found in [20].

Latency per connection	600 ns
Token Transmission Time	880 ns
Max Transmitter Idle Time	3500 ns

Consider three scenarios for FDDI as a basis for evaluating the impact of these parameters:

1. 10 nodes separated by a distance of 100 meters each (1 km total) - representing a backbone for interconnecting local area networks,

2. 3 nodes separated by a distance of 10 meters each (30 meters total) - representing the connection of two mainframe/supercomputers or peripheral equipment which is in a close physical proximity,

3. 500 nodes each separated by a distance of 100 meters each (50 Km total) - representing a HSLDN or MAN.

Table 1 illustrates the delays[2] inherent in each of the scenarios. The difference between MAX and MIN TOTAL DELAY is the number of nodes transmitting on a token rotation.

One can see that as latency improves below 60ns, the effect will be significant only in the MIN TOTAL DELAY for mainframe environments.

[2]a one bit delay is equivalent to 10 ns for a 100Mbps network

Latency : 600 ns per node

	1.Backbone	2.Mainframes	3.HSLDN
Prop Delay (D_{prop})	5.085μs	0.1526μs	2543μs
Latency (L_{tot})	6μs	1.8μs	300μs
Max Token Trans (T_t)	8.8μs	2.64μs	440μs
Max Trans Idle (T_i)	35μs	10.5μs	1750μs
MAX TOTAL DELAY	54.885μs	15.0926μs	5033μs
MIN TOTAL DELAY	11.085μs	1.9526μs	2843μs

Latency : 60 ns per node

	1.Backbone	2.Mainframes	3.HSLDN
Prop Delay (D_{prop})	5.085μs	0.1526μs	2543μs
Latency (L_{tot})	.6μs	.18μs	30μs
Max Token Trans (T_t)	8.8μs	2.64μs	440μs
Max Trans Idle (T_i)	35μs	10.5μs	1750μs
MAX TOTAL DELAY	49.485μs	15.0926μs	4763μs
MIN TOTAL DELAY	5.685μs	.3326μs	2573μs

Table 1: Overhead Delay Parameters for FDDI Scenarios

3.2 TTRT vs Utilization

The purpose of this analysis is to develop a tool which is reasonably simple to use and which will be able to predict maximum utilization of an FDDI network. Practically all aspects of the derivation use average values of the random variables with focus given to heavily loaded conditions. Only asynchronous traffic is considered with justification for ignoring synchronous traffic in the analysis being given in the following section. The end of the analysis will provide results from a simulation of FDDI to illustrate the degree of accuracy of these approximations.

All nodes on a FDDI network use the same value of $TTRT$. If a node does not obtain the token in time to transmit its data and maintain the required timing restraints, it simply forwards the token to the next node. One way of viewing utilization (U) is to represented it as

$$U = \frac{TTRT - TRO}{TTRT} \tag{1}$$

where TRO represents the Token Rotation Overhead. During a single round trip of the token, only a certain percentage of the time can be spent sending data. The rest of the time essentially represents the amount of time required to transmit the token around the ring. Part of TRO is fixed and independent of the network load. All other factors remaining fixed, one can see that for a heavily loaded network, increasing $TTRT$ will increase utilization ··· at the expense of delay. Here we examine an estimation for utilization as a function of $TTRT$.

Using the terms developed in the previous sections, TRO can be expressed as follows

$$TRO = N_c \times (T_t + T_i) + D_{prop} + I_{tot} \tag{2}$$

The number of nodes to capture the token on a rotation is dependent upon the available bandwidth for transmission ($TTRT - TRO$), the packet length, load and the number of packets transmitted by each node which captures the token. All of the variables except N_c are static values, however, for this derivation, separate the components of TRO into two terms as follows, TRO_s representing the component of TRO which is independent of the number of packets transmitted and the dynamic part TRO_d.

$$TRO = TRO_s + TRO_d \tag{3}$$

$$TRO_s = D_{prop} + I_{tot} \tag{4}$$

$$TRO_d = N_c \times (T_t + T_i) \tag{5}$$

The dynamic component is determined by the number of nodes which capture the token. Each time the token is captured, there is a transmitter idle delay and a retransmission of the token. It is possible that a node can transmit multiple packets for a single token capture. If each node capturing the token transmits twice as many

messages, the token retransmission(T_t) and transmitter idle delays(T_i) would occur half as often.

Consider the range of values which N_c has with the load uniformly distributed among the nodes. Under low loads, few nodes have messages to transmit. As network load increases, N_c increases, approaching the number of nodes on the network N, then it decreases as the network becomes overloaded. In the last case, as nodes have large queues, one token capture results in a large number of packet transmissions. Eventually, each node holds the token for a period of time which precludes other nodes on the network from capturing the token until its next rotation.

As the queues at each node overload, the utilization actually increases as there are fewer token captures per rotation; however, we are interested in determing the maximum traffic which the network can support without queue buildup. In such an overloaded situation, the network cannot support the traffic levels even though overall utilization may be higher. Therefore, it is assumed that traffic is distributed such that on the average a node only has a single packet(or less) to transmit per token rotation and that the maximum value of N_c is N. The number of packets which can be transmitted is dependent upon the number of packets which can be transmitted during $TTRT - TRO_s$. N_c would be defined as

$$N_{pt} = N_c = \frac{TTRT - TRO_s}{\frac{P}{R} + T_t + T_i} \tag{6}$$

where
P is the packet length
R is the transmission rate of the network and
and N_{pt} is the number of packets transmitted.

As the load increases, the number of nodes capturing the token would have a limit of

$$N_c = N \tag{7}$$

and the number of packets transmitted with maximum token captures N_{ptM} would be

$$N_{ptM} = \frac{TTRT - TRO_s - N \times (T_t + T_i)}{\frac{P}{R}} \tag{8}$$

Substituting Equation 8 back in to Equation 1 , U can be expressed as

$$U = \frac{TTRT - TRO_s - N \times (T_t + T_i)}{TTRT} \tag{9}$$

Figures 3 and 4 illustrate the predicted and real maximum utilization for the backbone and MAN scenarios listed above.

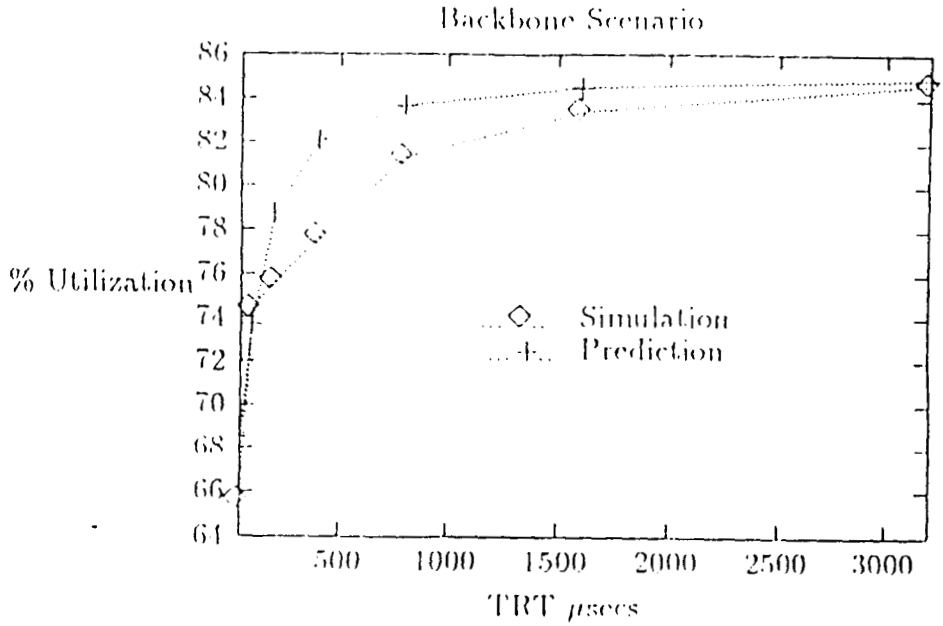


Figure 3: Backbone Predicted Utilization vs TRT

3.3 TTRT Impact on Synchronous Traffic

Synchronous traffic has not been included in the previous analysis. Because synchronous traffic by its nature places a uniform load on the system per token rotation, we can initially consider it as an overhead of the token rotation. After calculating the maximum utilization as described above, add the percentage of synchronous traffic to the previous utilization value to obtain the true utilization. The only approximation involved is due to the number of token captures which could be higher if, for example, synchronous traffic packets were small and distributed to a large number of nodes. This would further reduce maximum utilization.

Application of the previous results to typical synchronous traffic requirements indicates that FDDI does not support synchronous traffic without significant decrease in utilization. ISDN compatability requires that synchronous traffic must be delivered at the rate of once every 125μs. In order to guarantee this arrival rate, a $TTRT$ of 62.5μs must be established. For a conservative estimate, assume that $TTRT$ is 125μs, knowing that on the average 125μs will be attainable although in some instances packets may be lost due to the inability to guarantee $TTRT$ can be met. Comparing this with the lower node latency and ignoring packet overhead, the maximum utilization is illustrated in Table 2.

The table indicates that FDDI could not support an acceptable ISDN interface in most configurations and that it would be extremely unlikely that synchronous traffic with comparable periodicity could be supported in a long distance (MAN) environment. It is also interesting to note that in the scenario for a backbone, the

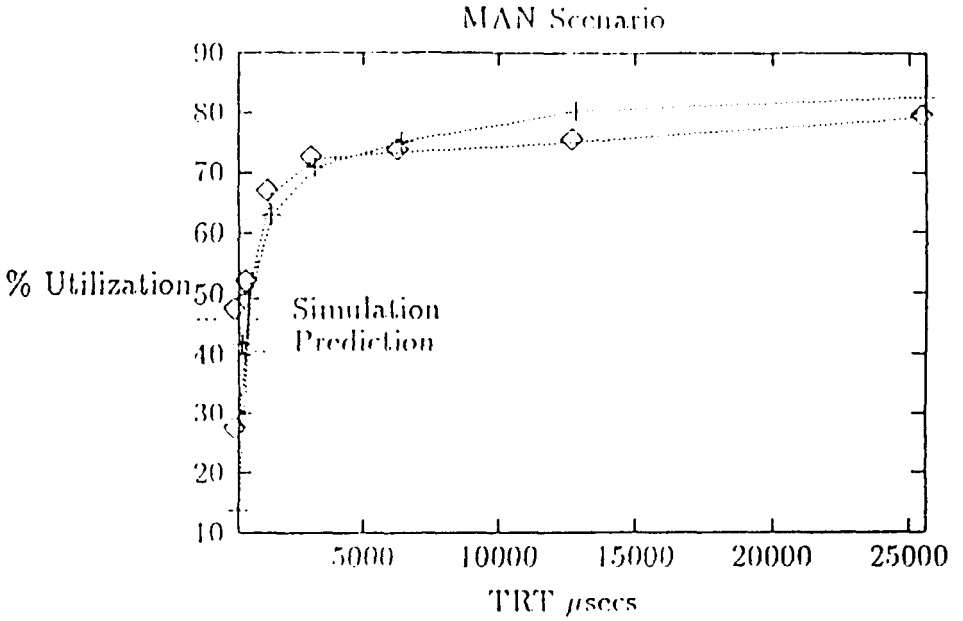


Figure 4: MAN Predicted Utilization vs TRT

	1.Backbone	2.Mainframes	3.HSLDN
MAX TOTAL DELAY	60%	88%	****
MIN TOTAL DELAY	95.5%	99.9%	****

Table 2: Maximum Utilization for 125μs $TTRT$

utilization could drop significantly depending upon the number of nodes which can capture the token on one rotation. This is of course also dependent on the packet size being transmitted.

4 Scaling FDDI to Gigabit Speeds

With the disparity between the development of the speed of transmission technology (optical systems) and the speed of processing elements in a computer, one might reasonably question the degree to which gigabit networks can be used at all. This disparity between the speed of transmitter technology and processing elements is tolerable if large numbers of nodes can be viewed as using the gigabit speeds if only for short durations. This is one reason to examine the scalability of FDDI in terms of the *number of nodes* it will support. The previous section also provided some insight as to the impact which number of nodes will have on performance. The second most important factor to examine is *network length*. Token ring networks are usually dependent upon short propagation delays for providing fast network access.

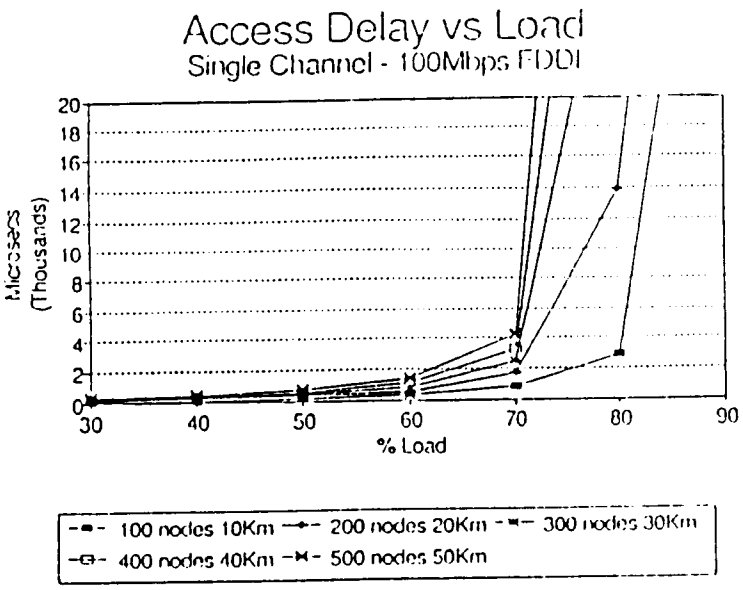
Given that this is not a detailed investigation, the two factors will not be addressed independently. Instead, the network configurations examined will look at increasing the number of nodes while maintaining the same internode distance, thereby increasing the network length simultaneously.

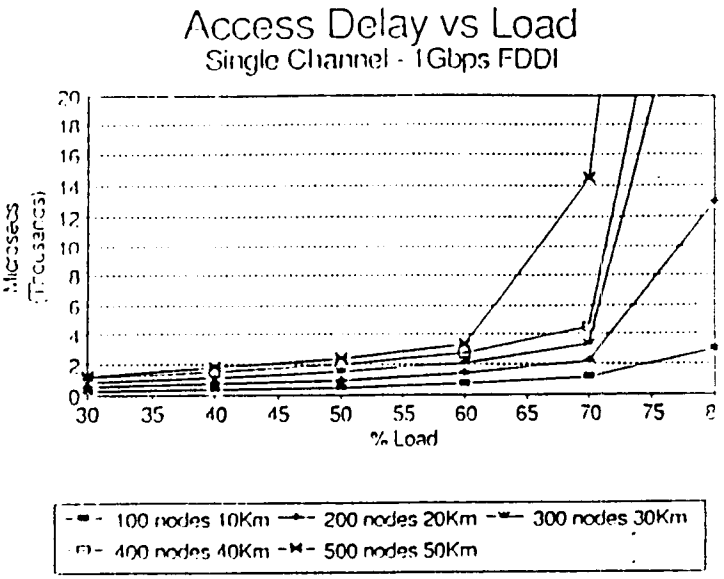
When considering the scalability of FDDI, one could view a scaling of the transmitter with proportional scaling of the speed of the nodes, or the transmitter speed could be scaled leaving the node processing speeds (values of Transmitter Idle Time, etc. discussed in the previous section) at the same rate. The data which follows is an examination of the latter given its greater probability of occurrence.

The parameter space examined here varies the number of nodes between 100 and 500 maintaining internode distance at 100m, resulting in network lengths between 10Km and 50Km. The $TTRT$ has been chosen at a level which allows for maximum utilization in the 75-90% range in general(10000 μs). The delay terms mentioned in the previous section were constant in all runs. Packet size was fixed at 1000 bits.

Figure 5 shows access delay for standard(100Mbps) and gigabit FDDI. The vertical axis scaling has been chosen to be twice $TTRT$(10000 μs). This shows how performance suffers dramatically in long-distance networks when more than one token rotation is required to deliver data. As expected, performance degrades in each graph as nodes and distance are increased. The two graphs reveal that gigabit FDDI begins to degrade at approximately 70%, whereas standard FDDI degrades slightly after 70%. One might expect that a faster transmission rate would mean shorter delays, however, these graphs tend to show comparable results. The reason is that with a constant packet size there are more packets at 70% load in gigabit FDDI, and more nodes are also transmitting. The previous section indicated the overhead associated with token capture and how it relates to the number of nodes.

The final graph in figure 5 combines the 100 and 500 nodes curves from the previous two graphs. Surprisingly, it does not appear that the scaling of the transmitter is the issue. Both speeds indicate similar performance curves. Number of nodes and distance are the factors which distinguish the shapes of the curves.





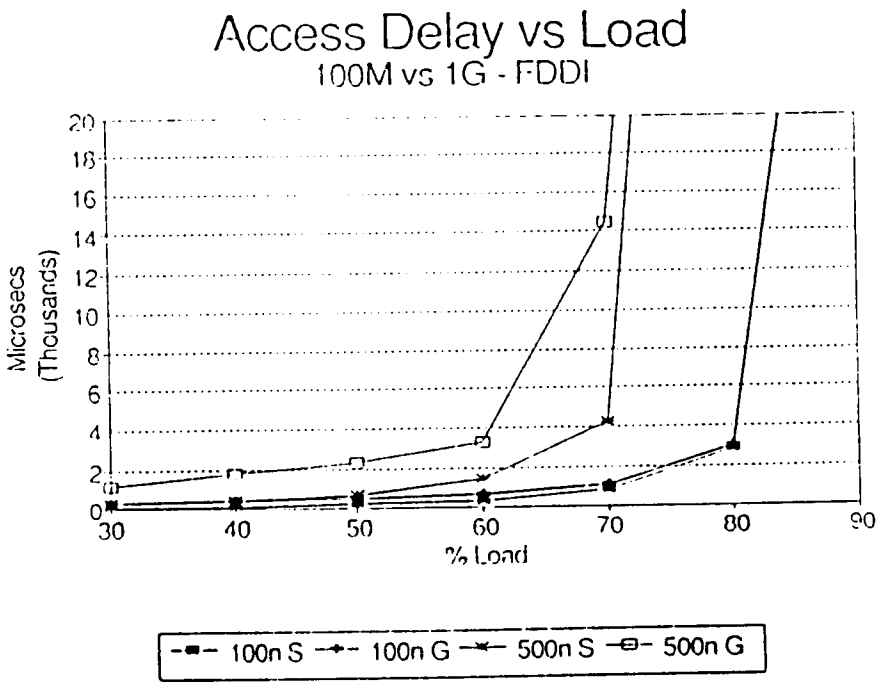


Figure 5: Standard FDDI vs 1Gbps FDDI

5 Performance Improvement

As the data transmission rates and distance covered by the networks increase, the number of simultaneous packets on a network increases dramatically. The parameter a where

$$a = \frac{length\ of\ data\ path}{length\ of\ packet} \tag{10}$$

represents this concept and frequently arises as a crucial parameter in network performance. For example, a network of 10 Mbps capacity, 2000 packet length and LAN length of 1 km can only hold .003125 packets at a time. Network designs with capacities such as 1 Gbps, 100 km and packet lengths of 2000 bytes, could contain approximately 31 packets simultaneously.

Increasing the number of packets on a network as shown above demands that greater attention be given to the management of these packets. Most LANs do not have this problem of simultaneous packets. Networks such as DQDB[17] which employs a slotted scheme or DSMA/RN[4] which uses a hybrid CSMA/CD technique may provide greater opportunity for optimization of this packet capacity.

In addition, the metrics for network performance, or at least our view of them, needs to be reconsidered. When analyzing the performance of a data communications network, one typically uses utilization as a metric of evaluation. If a 100Mbps network is capable of delivering 100Mbps of data, then the network is assumed to have 100% utilization (ignoring packet overhead).

This 100% limitation is based upon the assumption that only one node is transmitting at a time. If it is only possible for one node to transmit at a time as in most CSMA/CD networks or token ring networks, then this is a reasonable assumption. Even in situations where link level protocols provide for the existence of multiple packets on the network simultaneously as in *selective retransmission* and *go-back-n*, and in systems which allow for the building of a train of packets such as Expressnet, no consideration is given to the possibility of simultaneous transmission by nodes. Register insertion rings[6], DSMA/RN[4] and others[1] have shown that utilization greater than 100%(throughput greater than 1.0) is achievable.

Assume that nodes are numbered from 1,...,N and placed in a ring. If node 1 can send to node 2 while node 3 sends to node 4, 200Mbps is being transmitted. To understand the inefficiency of a ring running at close to 100% utilization, consider not only whether the network is filled with packets, but whether or not the packets are doing useful work, where work which is not useful occurs when packets take up network capacity but have already been delivered to the receiver.

The focus of the suggestion for improvement in performance of FDDI in this paper is on recovering the unused packet capacity by removing the packet at the destination and inserting new packets in their place which I will call *destination insertion*. *Destination insertion* will also allow for multiple simultaneous transmitters on the network and increased throughput. The technique is also applicable to

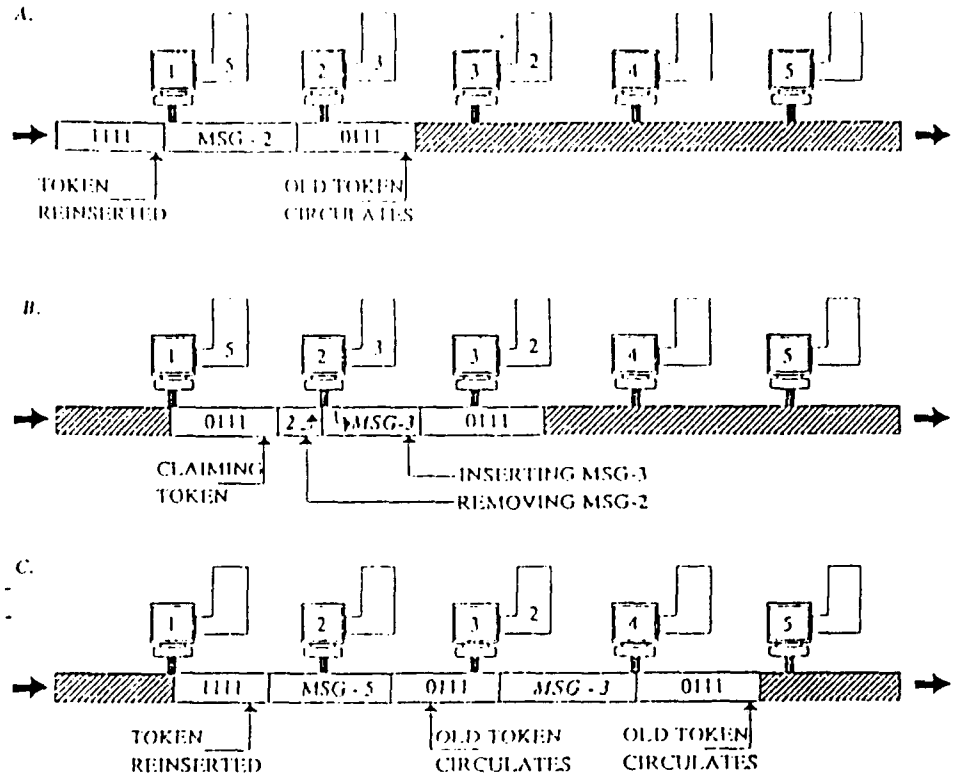


Figure 6: Removing Packet at Destination

DQDB and token rings in general, although the analysis does not address DQDB.

5.1 Destination Insertion

Figure 2 illustrates the removal of the message at the receiver. In the second part of this diagram, *MSG - 2* has reached the destination node and is about to arrive back to the original sender, node 4. It has been shaded to emphasize that the slot is only delivering an acknowledgment and that the capacity of the network is not being used in an optimal fashion. In this case the slot could have been used by node 2 to transmit the message to node 3. I.e., two messages could have been delivered instead on one with this packet-slot.

Figure 6 depicts node 2 removing the message from node 4 and at the same time inserting its message to node 3. The removal of the token at node 1 and transmission of the message from node 1 to node 5 is unaffected by this squeezing of the message from node 2 to node 3 into the train of packets. The squeezed data can not be longer than the message which it is replacing.

Destination insertion works as follows. When a node transmits a message on the network, the message proceeds until it reaches its destination. At that point it is marked as received and the slot containing the message is available for further use. As long as the slot is on the ring, it can be used by other nodes. This raises two additional questions:

- how long will the slot remain on the ring, and
- which nodes(messages) are candidates for reusing the slot.

5.2 Restrictions on slot reuse

This message would normally travel at least as far as the original sender and could conceivably travel even further if the original sender releases the token before the packet returns. If the time of transmission of the message is at least as long as the propagation delay of the network($a < 1$), the message will terminate at the sender. If the time of transmission is much less that the propagation delay($a > 1$), the token will leave the node before the message has made a loop and the message may not reach the node holding the token for a significant period of time.

When $a < 1$, the packet arrives at the sender before transmission is complete. Assume that node 1 holds the token and has a message for a node i. When node i receives the slot, it is reusable by any node up to node 1, but not beyond node 1. If an attempt is made to send a message past node 1, it will be absorbed by node 1, the holder of the token. If $a > 1$, additional reuse could be made of the packet, but the opportunity for reuse will be terminated at the original sender in this analysis.

Length of packet also presents a problem, as mentioned above. Obviously, a message which is inserted into the free slot must of length less than the length of the delivered message. For this analysis, the assumption is made that all packets are of equal length with the effect of varying packet lengths to be examined in subsequent research.

5.3 Objective

If *destination insertion* recaptures these packets, one expects throughput to increase and delay to decrease. The central questions are how much these measures would be affected and how feasible the implementation would be. Recent papers[6,1,4] have used similar techniques to show increases on the order of 1.5 to 2 times 100%, but these papers have not included any generalizations as to how this might apply in an arbitrary case, rather the simulation results are for specific cases. I intend to provide an analysis which will allow one to predict the degree to which a method such as this can improve performance in FDDI or token rings, and to show that a feasible strategy can be developed which does not meet the maximum, but can approach it.

5.4 Advantages

One might assume that the effect of *destination insertion* would be simply to provide for an increase in throughput at high loads and have little effect at low loads.

However, the method can be shown to improve performance in the following areas.

- Throughput will be able to sustain traffic at a much higher load
- One of the major problems with token ring networks is the access delay for obtaining the token. These extra slots will reduce average access delay to the network.
- Because of the large distance and number of nodes inherent in proposed wide area research nets, normal token ring access delays are likely prohibitive. These additional packets will reduce access delay as mentioned above and reduce the sensitivity of a token ring network such as FDDI to longer distances.

5.5 Expected Maximum Throughput Increase

For this analysis, the assumptions will be that all nodes always have at least one packet in the queue, that destination address space is uniformly distributed among the nodes, and that packets are fixed length. If all messages were destined for the neighbor, throughput could be increased by a factor of N, but this is an unlikely scenario. The result derived is a function of the number of nodes, $\mathcal{E}(n)$, which states the factor by which throughput can be expected to improve under heavily loaded conditions. For example, if utilization is currently 80% and the expected throughput increase, $\mathcal{E}(n)$, is 1.4, then utilization should be able to reach 112% under the conditions specified in the assumptions above for n nodes.

In order to determine the expected throughput for such a network, consider the traversal of a packet around the network. Assume that node n removes the token from the network and transmits a packet. The packet must be destined for one of the nodes $n-1 \ldots 1$. Assume that the packet is destined for node $j, n < j \leq 1$. Upon receiving the message, either

1. node j has a packet available for transmission to node k where $j < k \leq< 1$ or $k = n$ which states that the message can be squeezed into the now available slot and removed before it passes the original sender
2. node j has a packet available for transmission to node k where $n - 1 < k < j$ and it can not be squeezed without a possibility of being removed by a node which has the token (specifically node n may still be transmitting, so we assume that it is in order to guarantee viability of the slot)
   or
3. there is no packet in the queue (which this analysis is ignoring).

Define $\mathcal{E}(i)$ to be the expected increase in throughput given that the slot has as its destination node i. Using a recursive derivation, start with $\mathcal{E}(1)$.

$$\mathcal{E}(1) = \frac{1}{N} \tag{11}$$

because the expected value of increased throughput is the probability of the message in the head of the queue being destined for node n (the original sender of the slot and the only possible node to which node 1 can send), $\frac{1}{N}$, times 1 (the number of messages it can send in the slot) plus the probability that the message at the head of the queue is for some other node, $\frac{N-1}{N}$, times 0.

The expected increase at the second node is composed of three terms

1. the probability that the message at the head of its queue is for node n times its expected increase, $\frac{1}{N}$ times 1 as above,
   plus

2. the probability that the message at the head of the queue is for node 1, $\frac{1}{N}$, times the expected increase which is 1 plus $\mathcal{E}(1)$ because the message is delivered at node 1 and can be reused again at node 1 with expected increase $\mathcal{E}(1)$
   plus

3. the probability that the message at the head of the node is for neither node 1 nor n, $\frac{N-2}{N}$, times the expected increase which is also $\mathcal{E}(1)$ because the slot will proceed to node 1 available for reuse

Therefore,

$$\mathcal{E}(2) = \frac{1}{N} + \frac{\mathcal{E}(1)}{N} + \frac{N-2}{N} \times \mathcal{E}(1) = \frac{1}{N} + \frac{N-1}{N^2} \tag{12}$$

For arbitrary node j, the formula can be generalized to

$$\mathcal{E}(j) = \frac{1}{N} + \sum_{i=1}^{j-1} \frac{1}{N} \times (1 + \mathcal{E}(i)) + \frac{N-j}{N} \times \mathcal{E}(j-1) \tag{13}$$

$$= \frac{j}{N} + \frac{N-j+1}{N} \times \mathcal{E}(j-1) + \sum_{i=1}^{j-2} \mathcal{E}(i) \tag{14}$$

for $n < j \leq 3$
and

$$\mathcal{E}(j) = \sum_{i=1}^{j-1} \frac{1}{N} \times (1 + \mathcal{E}(i)) \tag{15}$$

for $j = n$

The first term in Equation 13 represents the expected increase if the message is for node n, the original sender. The second term represents the expected increase if the message is for a node positioned between the current node j and node 1 which is the squeezed message itself plus any expected increase once that message is delivered. The third term assumes that the slot could not be used so it is passed on to node $j-1$. In the case for $j=n$, the first term is omitted because it would never send a message to node n, itself.

5.6 Overall effectiveness

The following graph show the increase in throughput expected from a traffic placement strategy as described above. The result of interest from the above derivation is the value of $\mathcal{E}(n)$ which describes the number of expected messages delivered with each packet as it is transmitted from the node holding the token (n). Figure 7 shows $\mathcal{E}(n)$ versus n. One can observe that the effect of such a technique has a much greater effect as the number of nodes increases.

Recall that the proper interpretation of this graph is that throughput can be increased by the factor given. Results show that for a 100 node problem which operates at 90% maximum utilization, this method will increase throughput by a factor of 3 to 270%. A number of curves are provided. The curve marked *Analysis* is the result of calculating $\mathcal{E}(n)$ for various values of n as derived above. The analytical results can be compared with simulation results in the curve *Simulation*. The simulation model did not require using an FDDI model and was only modelling the passing of messages from node to node without delay statistics. Here a packet was allowed to be reused an arbitrary number of times, an impractical assumption discussed later. The final two curves in this figure show maximum utilization when one limits the number of times which a slot may be used during a cycle around the network. *Max=2* indicates that the slot may be used twice (reused once).

5.7 Additional Simulation Results

A simulation model of FDDI, written in Simscript, has been developed on Sun workstations in order to test performance issues. A modification was made to the model to allow for the incorporation of *destination insertion*. It should be noted that the full advantage of this technique can not be seen in these results because of an inconsistency in the original design of the model and the design necessary for *destination insertion*. A new model which will be used to show the full potential impact for FDDI is under development. The results shown are a conservative estimate of the effect.

A number of benefits arise from removal of the message at destination and reuse of the slot, the first two of which are investigated in this paper.

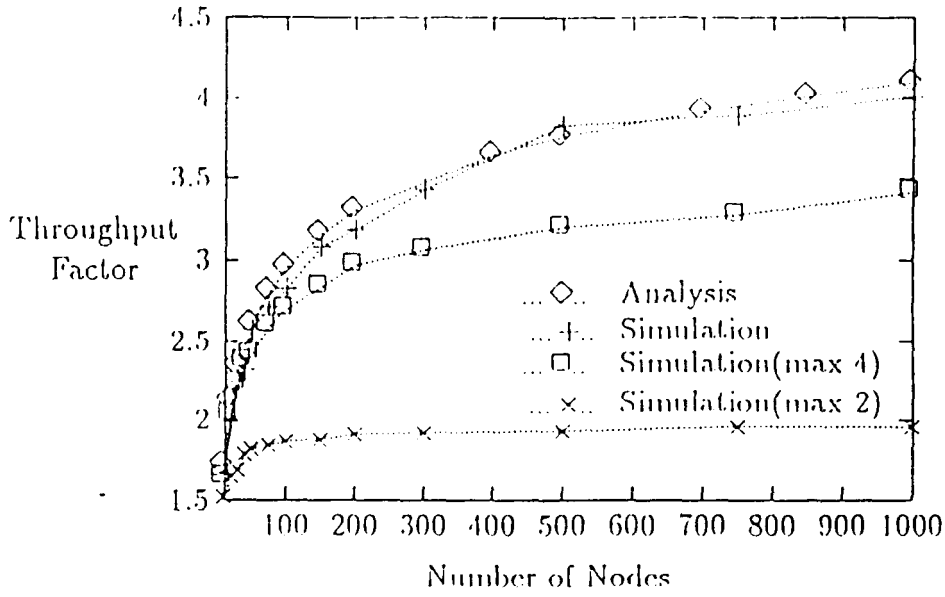


Figure 7: Expected Throughput Increase

1. As shown previously, throughput can be dramatically increased.

2. Even in scenarios where the system is not fully loaded, delay characteristics are improved.

3. The extensibility of an FDDI network is increased. One of the major limitations of extending an FDDI ring is the large propagation delays experienced as total network distance increases. These extra slots will provide additional opportunities for transmission beyond the token arrivals, decreasing access delay.

4. If synchronous traffic is required, it is possible to set $TTRT$ values at a higher value and still maintain the same average access delay. Raising the $TTRT$ will allow for higher utilization as shown previously.

The effect of access delay where load is less than 100% is shown in Figures 8 and 9. $TTRT$ is set to 5 ms, network rate(R) at 100 Mbps, packet length(P) is fixed at 5000 bits. Figure 8 shows how removal affects the 10 node case. Figure 9 incorporates the removal vs non-removal for 10, 50 and 100 nodes. Note that in every instance using the *destination insertion* technique, access delay at 100% load is comparable to access delay at very low loads. It should also be noted that these runs have not reached the assumption made in the analysis that all nodes have data in the queue; however, the effect is still significant.

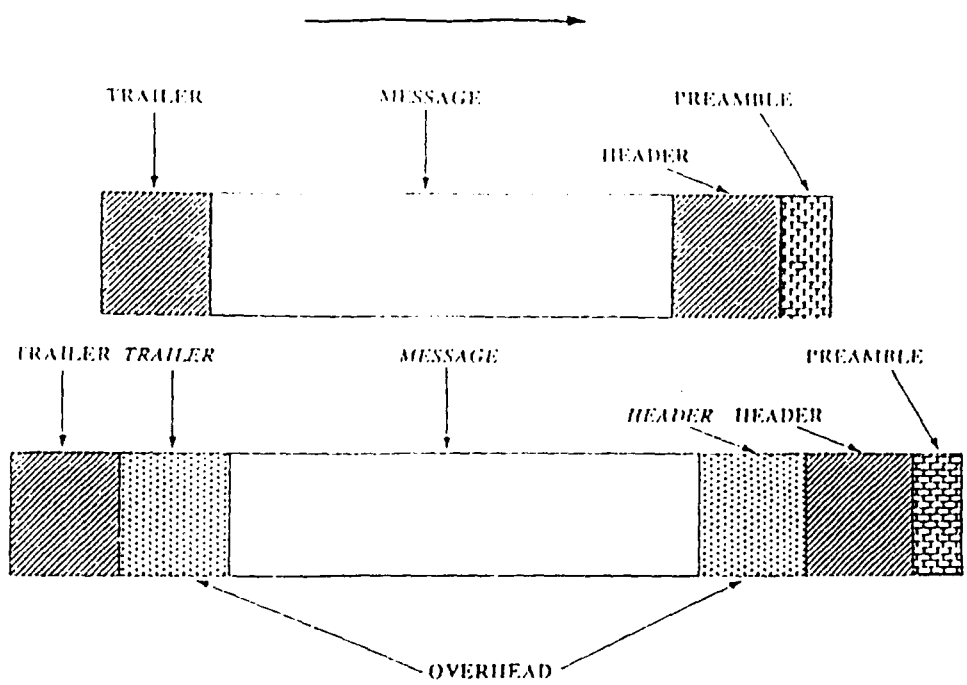


Figure 10: Packet Structure

5.8 Implementation

The implementation of this technique is not without a cost, but the cost is a nominal one. First, the receiver must be able to recognize the destination address and mark the packet as delivered. Second, the marking of the packet must be in such a manner as to allow for other nodes to determine that the packet has already reached the destination. Placing the header at the beginning of the packet allows for interpretation of the source and destination. This would make it possible for a receiver to determine if the packet had already been delivered, but the process of making the decision would be simplest if the receiver marked a 1 bit slot at the end of the header to indicate that the packet had been received. All subsequent nodes would recognize this similar to recognizing the token and transmit the data.

Token rings allow the original message to propagate back to the sender before being removed. This provides and acknowledgement mechanism that the message has been delivered and has circulated around the ring correctly. However, receipt of the original message is not required. Many protocols only require an acknowledgement of some type from the receiver. This can be done much more efficiently that using the entire packet slot for acknowledgment. If sufficient room is left at the end of the packet, the acknowledgement from the receiver can be accommodated. This would require an additional header/trailer combination for the reuse of the packet as shown in Figure 10. The number of these extra header/trailer combinations would be equal to the number of times of expected reuse of the packet. Of course, in the best case, all nodes would want to use the slot requiring $N \times (P_{trailer} + P_{header})$ overhead, an unacceptable price to pay. However, a value closer to the average number of uses of the slot could be used. For example, if on the average for a given number of nodes, the packet would be used 3 times, the overhead of 3 packet/trailer entries would be a reasonable one.

6 Conclusions

This paper has shown some of the limitations of FDDI. Those limitations are strongly a function of number of nodes and network distance. A tradeoff exists between maximum utilization and the network access as determined by the $TTRT$. Return of the token within an average specified time can be guaranteed but use of the token cannot. It has been shown that setting $TTRT$ lower reduces utilization. Reduced utilization in turn will *increase* access delay, the reverse of the desired effect.

Scalability of FDDI in the 500 node, 50Km range up to gigabit speeds is not without some proportional loss in performance. Simply increasing the transmitter speed by a factor of ten does not translate directly into being able to deliver ten times the data with similar access delays.

One method of improving the extensibility of FDDI is to remove packets at the destination and make those packets available for reuse as they continue circulating around the ring. This has been shown to have an expected increase of over twice the capacity of the network for more than 20 nodes when the network is fully loaded and to reduce access delay in cases where the network is less than fully loaded.

References

[1] Dobosiewicz and Gburzynski. Ethernet with segmented carrier. *Computer Networkin Symposium*, 72-78, April 1988.

[2] D. Dykeman and W. Bux. Analysis and tuning of the fddi media access control protocol. *IEEE Transactions on Communications*, 6(6):997–1010, July 1988.

[3] G.M. Exley and L.F. Merakos. Throughput - delay performance of interconnected csma local area networks. *IEEE Journal on Selected Area in Communications*, 1380–1390, Dec. 1987.

[4] E.C. Foudriat, K. Maly, C.M. Overstreet, S. Khanna, and F. Paterra. *A Carrier Sensed Multiple Access Protocol for High Data Rate Rings.* Technical Report, Old Dominion University, February 1990.

[5] IEEE 802.6 Working Group. Proposed standard: distributed queue dual bus man. August 7, 1989.

[6] W. Hilal and M.T. Liu. Analysis and simulation of the register-insertion protocol. *Proceedings, Computer Network Symposium*, 1982.

[7] H.S. Hinton. Architectural considerations for photonic switching networks. *IEEE Transactions on Communications*, 6(7):1209–1226, August 1988.

[8] M.J. Johnson. Proof that timing requirements of the fddi token ring protocol are satisfied. *IEEE Transactions on Communications*, 35(6):620–625, June 1987.

[9] R.E. Kahn. A national network: today's reality, tomorrow's vision, part 2. *EDUCOM Bulletin*, 14–21, Summer/Fall 1988.

[10] J.O. Limb and C. Flores. Description of fasnet - a unidirectional local-area communications network. *Bell System Technical Journal*, 61(7):1413–1440, September 1982.

[11] M. Maeda and H. Nakano. Integrated optoelectronics for optical transmission systems. *IEEE Communications*, 26(5):45–51, May 1988.

[12] K. Maly, E.C. Foudriat, D. Game, R. Mukkamala, and C.M. Overstreet. Traffic placement policies for a multi-band network. *SIGCOMM 89*, August 1989.

[13] K. Maly, C. M. Overstreet, X. Qiu, and D. Tang. Dynamic resource allocation in a metropolitan area network. *SIGCOMM Symposium*, 13–24, August 1988.

[14] M.A. Marsan and D. Roffinella. Multichannel local area network protocols. *IEEE Journal on Selected Areas in Communications*, 885-897, Nov. 1983.

[15] N.F. Maxemchuk. A variation on csma/cd that yields movable tdm slots in integrated voice/data local networks. *Bell Systems Technical Journal*, 61, no 7:1527-1550, September 1982.

[16] T. Nakagami and T. Sakurai. Optical and optoelectronic devices for optical fiber transmission systems. *IEEE Communications*, 26(1):28-33, January 1988.

[17] R.M. Newman, Z.L. Budrikis, and J.L. Jullett. The qpsx man. *IEEE Communications Magazine*, 26(4):20-28, April 1988.

[18] K. Nosu. Advanced coherent lightwave technologies. *IEEE Communications*, 26(2):15-21, February 1988.

[19] F. Ross. Fddi - a tutorial. *IEEE Communications*, 24(5):10-17, May 1986.

[20] Draft Proposed American National Standard. Fddi token ring media access control (mac) asc x3t9.5 rev. 10. February 28, 1986.

[21] F.A. Tobagi, F. Borgonovo, and L. Fratta. Expressnet: a high-performance integrated-services local area network. *IEEE Journal on Selected Areas in Comm.*, SAC-1, no. 5:898-913, November 1983.

[22] F.A. Tobagi and M. Fine. Performance of unidirectional broadcast local area networks: expressnet and fasnet. *IEEE Journal on Selected Areas in Comm.*, SAC-1, no. 5:913-925, November 1983.

[23] J. Ulm. A timed token protocol for local area network and its performance characteristics. *IEEE*, Feb. 1982.

DEPARTMENT OF COMPUTER SCIENCE

Technical Report # TR-90-15

Fairness Problems at the Media Access Level for High-Speed Networks

Kurt Maly, L. Zhange, and D. Game

Old Dominion University
Department of Computer Science
Norfolk, Virginia 23529-0162
U. S. A.

04-13-90

Old Dominion University
Norfolk, VA 23529-0162

Fairness Problems at the Media Access Level for High-Speed Networks *

Kurt Maly
L. Zhang D. Game
Department of Computer Science
Old Dominion University
Norfolk, VA 23529-0162

March 23, 1990

Abstract

Most lower speed ($\sim$ 10 Mb/s) local area networks use adaptive or random access protocols like Ethernet. Others at higher speed use demand assignment like token or slotted rings. These include Cambridge ring and electronic token ring systems. In this paper we discuss fairness issues in representatives of such protocols. In particular we selected FDDI as a demand access protocol using tokens, CSMA/RN a random access protocol and DQDB a demand access protocol using reservations. We focus on fairness at the media access level, i.e., attaining access or being excessively delayed when a message is queued to be sent as a function of network location. Within that framework we observe the essential fairness of FDDI, severe fairness problems in DQDB and some problems for CSMA/RN. We investigate several modifications and show their ameliorative effect. Finally we give a unified presentation which allows comparisons of the three protocols' fairness when normalized to their capacity.

*This work was supported by CIT grant RF-89-002-01, NASA grant NAG-1-908 and Sun Microsystems grant RF 596043.

1 Introduction

For the last ten years local area networks protocols have been dominated by Ethernet [9,1,14] and various token ring protocols with a nominal performance on the order of 10Mb/s. In 1990 implementation of metropolitan area networks are coming on the market with a performance on the order of 100Mb/s. For instance, FDDI [17,11,2,16] chipsets are now available with several vendors integrating them to make FDDI accessible with their products. FDDI is a token ring protocol designed to run at a speed of 100Mb/s. DQDB [10] is somewhat behind FDDI in reaching the market. It has been developed as a 300 Mb/s dual bus protocol (each bus is capable of handling 150Mb/s).

If predictions made in the Federal High Performance Computing Program [18,4] come true we will see within ten years operational gigabit widearea networks which will give individual users access to gigabit bandwidth while being separated thousands of kilometers. In a way we will have expanded local area networks from 10Mb/s to 1Gb/s and from few kilometers to thousands of kilometers.

The data rates mentioned above are realized at the lowest levels of the ISO model of communications. Currently the most commonly used protocol suit for higher levels - TCP/IP - reduces the effective speed by a factor of up to 10 in Ethernet. Most performance studies have been done at the medium access level. Considerably less information is available on the impact of overhead in layers above the transport layer. As new standards - OSI - are being developed to improve inter-operability of different computers, fears are expressed that the impact might prove too costly to be useful for gigabit networks. No matter what the future will bring for the higher levels it is absolutely essential that the media access protocol level be as efficient as possible. In this paper we shall concentrate on performance problems at the media access level and what, if anything, can be done about these problems at that level.

Traditional performance metrics for these protocols include delay metrics such as wait time for a message at a node, service time for a message, total response time (the time a message arrives at the destination minus the time a message arrives at a node), and throughput metrics such as bits delivered versus bits offered as a function of the load and the one we wish to concentrate on: fairness metrics.

Fairness in itself has many facets although in general it means that an entity should not have an advantage (as measured by some metric) over another entity. An entity may be a node, as for instance the node located physically at the end of the network should not have shorter access delays than a node in the middle of the network. An entity may be traffic of a specific type such as small messages versus large messages, or synchronous versus asynchronous. An entity may be load such as a node having to serve a large amount of traffic versus a node with little traffic and messages from either node should have the same access delay. In most cases fairness is only a problem over small time intervals. Most any protocol is fair if its behavior is observed over a long time period and statistics are calculated over that time period. On the other hand a faculty member editing a file remotely who gets a 30 second response time to a key stroke will be upset if his colleague at another node has a millisecond response time.

The general concept of fairness has been an issue in media access protocols since the beginning of local area networking. In carrier sensed systems, persistence, binary backoff, limited contention and a host of other techniques have been used to improve throughput and control access fairness [19]. Collision free protocols using basic bit mapping provide another method for each station to reserve a frame during the next transmission period. Since basic bit mapping is inherently unfair to the lower numbered nodes, BRAM, BRAP and other protocols were devised to alleviate the unfairness. [20]

Since, in contention protocols, there is the probability, however, remote, that a node's wait for access may be unbounded, token protocols were devised. Token protocols may maintain a priority system based upon actual token rotation time so that when heavily loaded, the ring will give higher priority messages better access. Later discussions of FDDI will examine token rotation access conditions in greater detail. Reference [6] has demonstrated that over time all stations on an FDDI ring have equal access to transmit asynchronous frames, regardless of the size of allocated synchronous bandwith for individual stations.

Slotted rings [21] provide control mechanism to establish access fairness. Each slot may contain access priority information or access to slots may be organized into rounds generated by a master station. Further access control can be based upon the fact that nodes must pass slots they empty to succeeding stations, or nodes may occupy only a single slot on the ring at any one time. In addition, some slotted rings provide different slot types, channel

and normal, which can and cannot be reused by the emptying node, respectively. Unidirectional bus systems provide a number of access mechanisms including those based upon trains, cyclic polling and non-cyclic reservation schemes [21]. In the paper, we will examine the fairness for one particular reservation scheme, DQDB, in much greater detail.

In addition to fairness controls at the media access level, some fairness issues may be resolved by other means. For example, a node which needs guaranteed access may designate the information as synchronous, thus providing regular interval access in an integrated network. This may result in considerably wasted throughput and/or increased response time in the case where the data being generated by the node is highly dynamic and should normally be delivered asynchonously.

This leads to various concepts of fairness which are not compatible with each other. When transmitting voice it is necessary that a certain amount of information be transmitted regularly. If necessary other nodes may need to be starved in order that the rate for voice traffic be sustained. Here an absolute level of fairness is not the goal, but one should be able to specify a level of fairness to allow both synchronous traffic and a reasonable amount of asynchronous traffic. The remainder of the paper discusses how various protocols fare from the fairness point of view.

In section two we describe three representative protocols and their handling of fairness. We have selected FDDI as a representative of demand access protocols using tokens and DQDB as a representative of a demand access protocol using reservations both serving metropolitan area networks. The third one presented is, *CSMA/RN*, a protocol developed for gigabit wide area networks [3]. It is representative of random access protocols which normally have unbounded access delays. It is known that DQDB has the problem that wait time is a function of node location and we present several strategies to improve performance in this regard. Section three provides a summary of the results and also gives a comparison of the protocols involving the three major metrics: delay, throughput and fairness.

2 Fairness in three protocols

Fairness problems normally are associated with non-uniform load distributions across the nodes is a network. A solution to this problem is load

balancing which attempts to match resources (bandwidth) with needs (offered load) at individual nodes. In order for this feature to handle sudden bursts, the reallocation of resources has to be done within milliseconds to be truly useful. One protocol which does so is DRAMA, [13,12,8,7], a protocol proposed for metropolitan areas. In none of the three protocols selected is load balancing fairness handled at the medium access level. We are currently in the process of developing distributed algorithms for FDDI, DQDB and CSMA/RN along the lines described in DRAMA, that is, each node, or group of nodes, keeps track of the overall network load and adjusts its own resources demands (bandwidth share) of the network accordingly. In future reports we shall describe our results in that direction.

All three protocols allow for different types of traffic, i.e., both synchronous and asynchronous traffic can be sent. Control of the allocation of bandwidth to nodes is handled in all cases at higher level. Problems with FDDI in handling isochronous traffic, i.e., guaranteed bandwidth at fixed intervals, have led to FDDI-II which we do not discuss here.

The problem we have studied in detail is the fairness to individual nodes compared to other nodes as measured by wait time and throughput even with a uniform load distribution across the network. To that end we have modeled all three protocols at the bit operational level. The models were implemented in Simscript for FDDI and DQDB and a mixture of C and Simscript for CSMA/RN. The details of the models and their validation can be found in [5,3].

Although we have made extensive studies of each protocol, we have selected for this paper only one case which illustrates the fairness problems we have found in these studies. The case selected is a 50 km long network with 50 equally spaced nodes. The traffic offered ranged from 0% - 100% for DQDB and FDDI and from 0% - 225% for CSMA/RN. Messages were 5000 bits long (with 10% variation) and were Poisson distributed; load was uniformly distributed among all nodes. For each protocol we will give a set of figures depicting wait time per message averaged over the entire network in μsec and throughput measured in percent of capacity as a function of offered load respectively. For each of these two curves we will give two or more snapshots, taken at various levels of load, of the same metrics averaged for each node and displayed for each node.

2.1 FDDI

FDDI is a token ring protocol. In Figure 1 a message is on the ring with the token appended to its end. Node D is the next one with a message and, ignoring the token rotation time for the moment, it will pick up the token and send its message as shown in Figure 2. Fair access and throughtput for individual nodes is obtained by means of a token rotation time (TRT) which guarantees that on the average each node will have access to the ring within TRT. In the worst case a node may have to wait 2*TRT. This, however, does not guarantee useful access to the ring in that time period. A node may get the token but no time may be left to send a message. In the worst case - all nodes having messages queued up - the token ring protocol will give each node in a round-robin way useful time to send messages although it may take $(n-1)$ *TRT where n is the number of nodes. For a detailed discussion see [6].

Figure 3 confirms the essential fairness of FDDI; at 80% the variation from the mean wait time of the net is about 10% for individual nodes. At higher loads the absolute amount of the variation increases significantly but the percentage grows much slower. For instance, at 90% the variation in wait time is 5,315 μsec with the net average being 22,309 μsec. As indicated earlier though, synchronous bandwidth is not handled at this level and it can happen that with nodal distribution varying over time that access delays will vary as a function of nodes. Secondly the variation in TRT does not allow for isochronous traffic. Thirdly the establishment of an effective TRT for a particular environment is not a trivial issue [5]. In this discussion we have ignored the fact that FDDI actually consists of dual counter-rotating rings for fault tolerant purposes because this does not effect the issue of fairness presented here.

2.2 CSMA/RN

CSMA/RN [3] is an extremely simple - hence it should prove inexpensive to implement - protocol which relies on the fact that a network looks quite differently at high speed ($>$ 100 Mb) than at low speed ($\sim$ 10 Mb). In the latter case only a fraction of one message occupies the entire ring while in the former case several messages can be on the ring, that is, if we assume uni-directional transmissions.

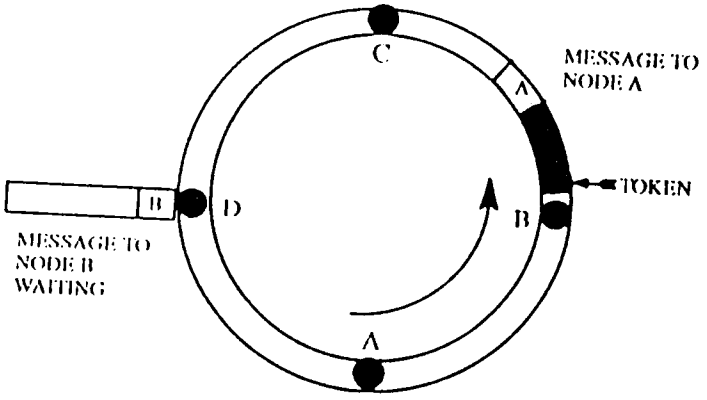


Figure 1: Message on token ring

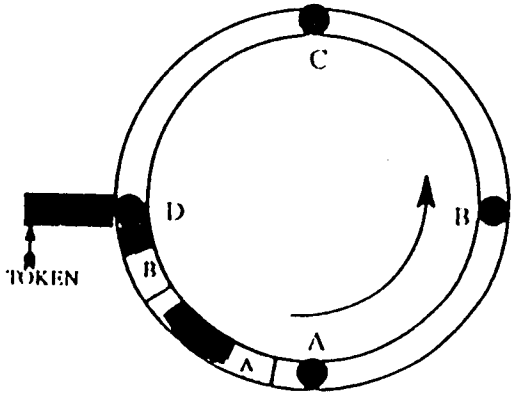


Figure 2: Node D captures token and sends its message.

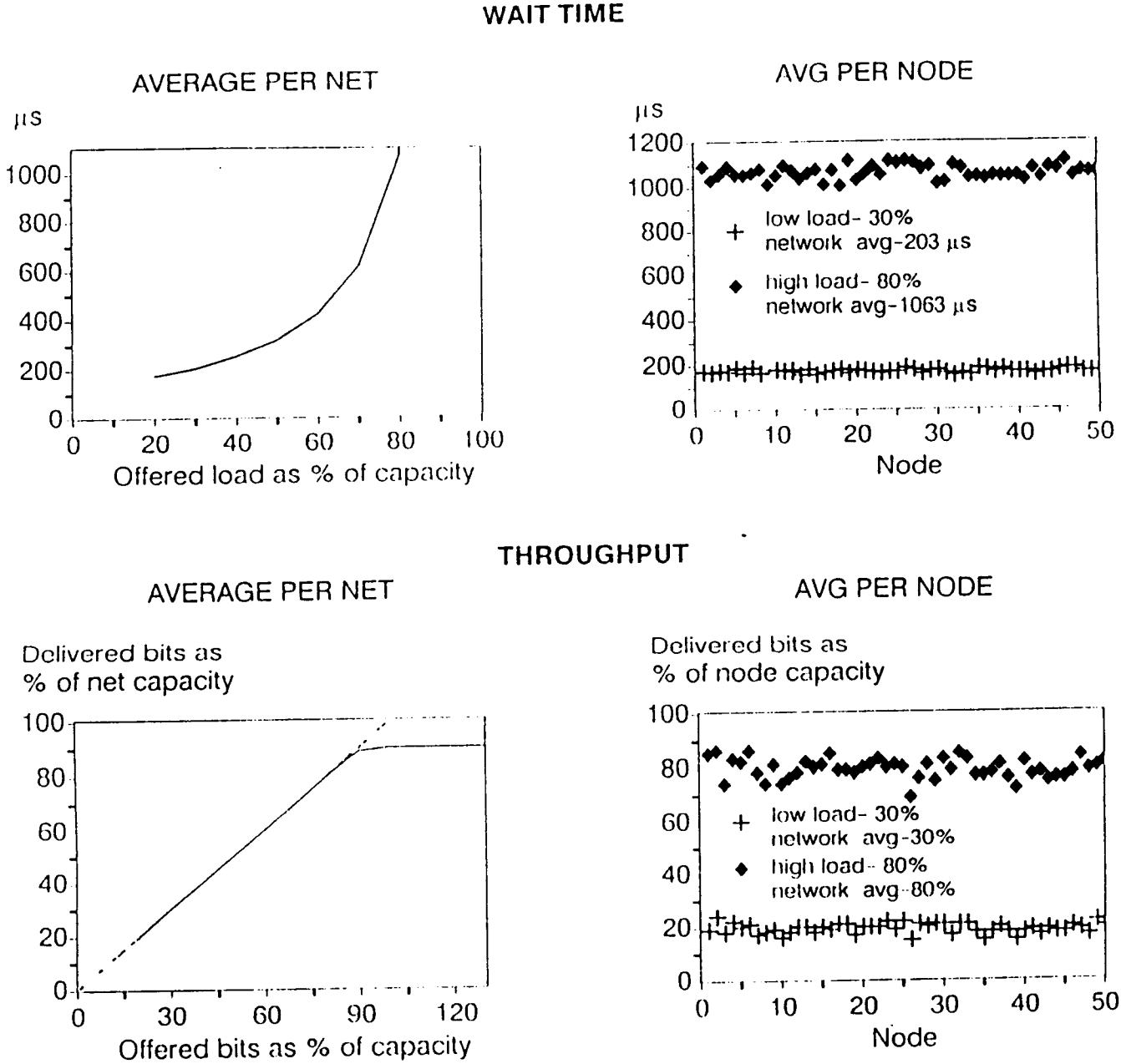


Figure 3: Fairness graph for FDDI

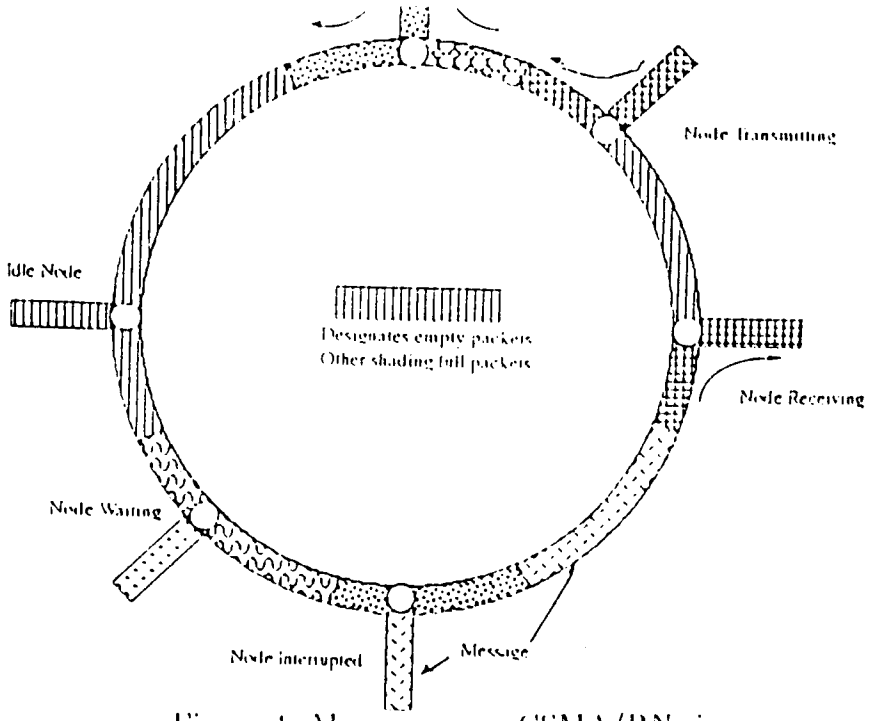


Figure 4: Messages on a CSMA/RN ring

The protocol is as follows: if a node has a message to send it does so if it does not detect another message passing by. If the ring is busy the node defers until it detects an idle ring at which time the node starts sending. If a sending node detects an incoming message whose destination is itself, the node continues sending while it takes the incoming message off the ring. If the incoming message is for another node, the sending node interrupts the message it is sending and lets the incoming message pass by. The remainder of the interrupted message is then treated as a new message. This implies that a receiver may have to reassemble the fractured messages. Figure 4 depicts various states the ring can be in. In [3] a detailed analysis of this protocol shows a truly outstanding performance over a wide range of parameter choices. For instance, with boards capable of handling one gigabit/s at each node, the net can actually deliver a throughput of two gigabits/s.

Fairness in most cases is quite good for up to 150% of offered load at 1Gbps (see Figure 5) but problems do occur. At higher loads the variation of the nodes' averages is beginning to increase. In specific cases actual starvation of some of the nodes can occur when they are unable to achieve their share of throughput even at higher wait times. Which nodes exhibit starvation does not depend on the node's physical location but who sends what message to who. Also this phenomenon is exhibited by some nodes for a short duration and starvation keeps drifting from node to node on the ring

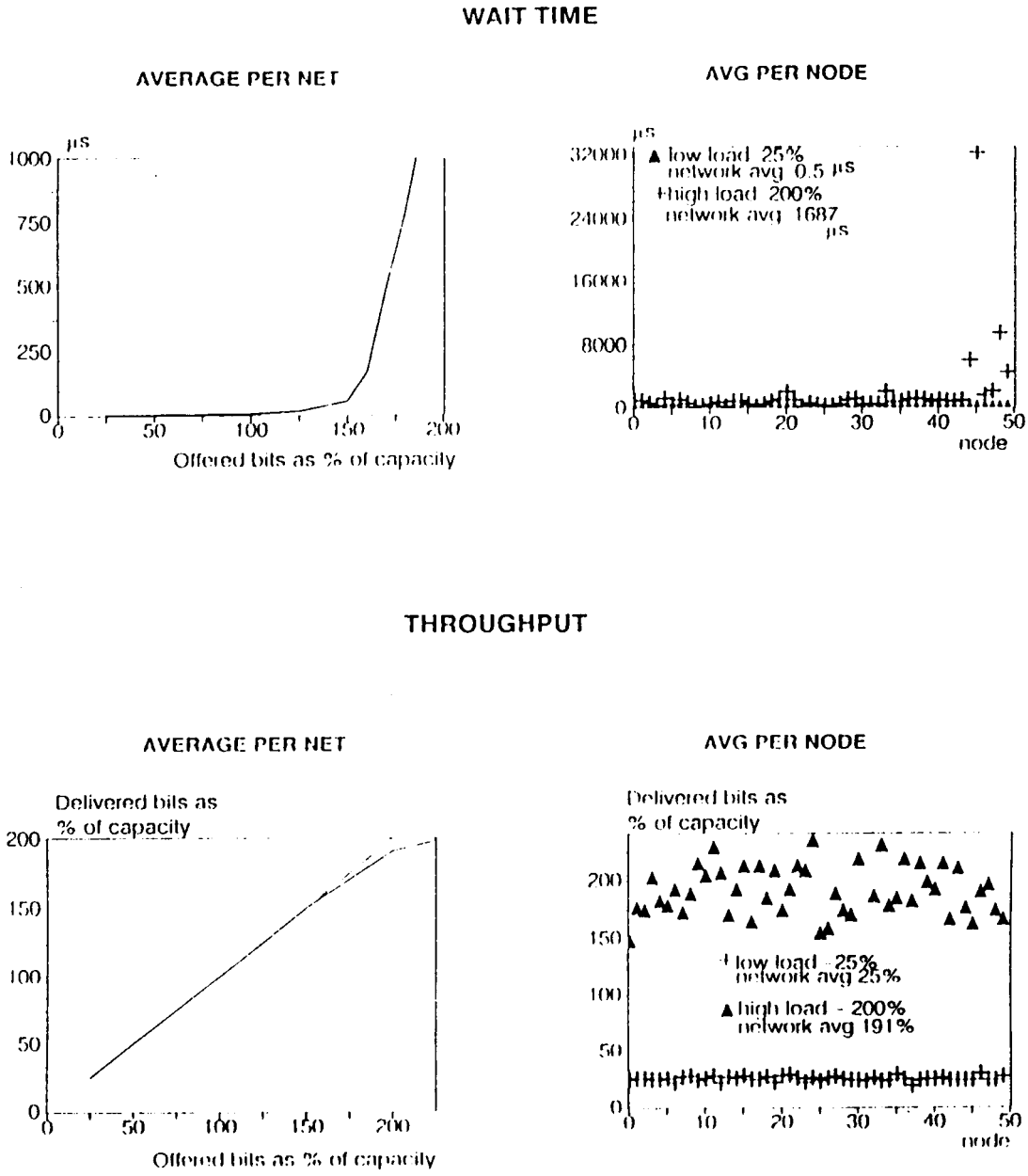


Figure 5: Fairness Graphs for CSMA/RN at 1Gb/s

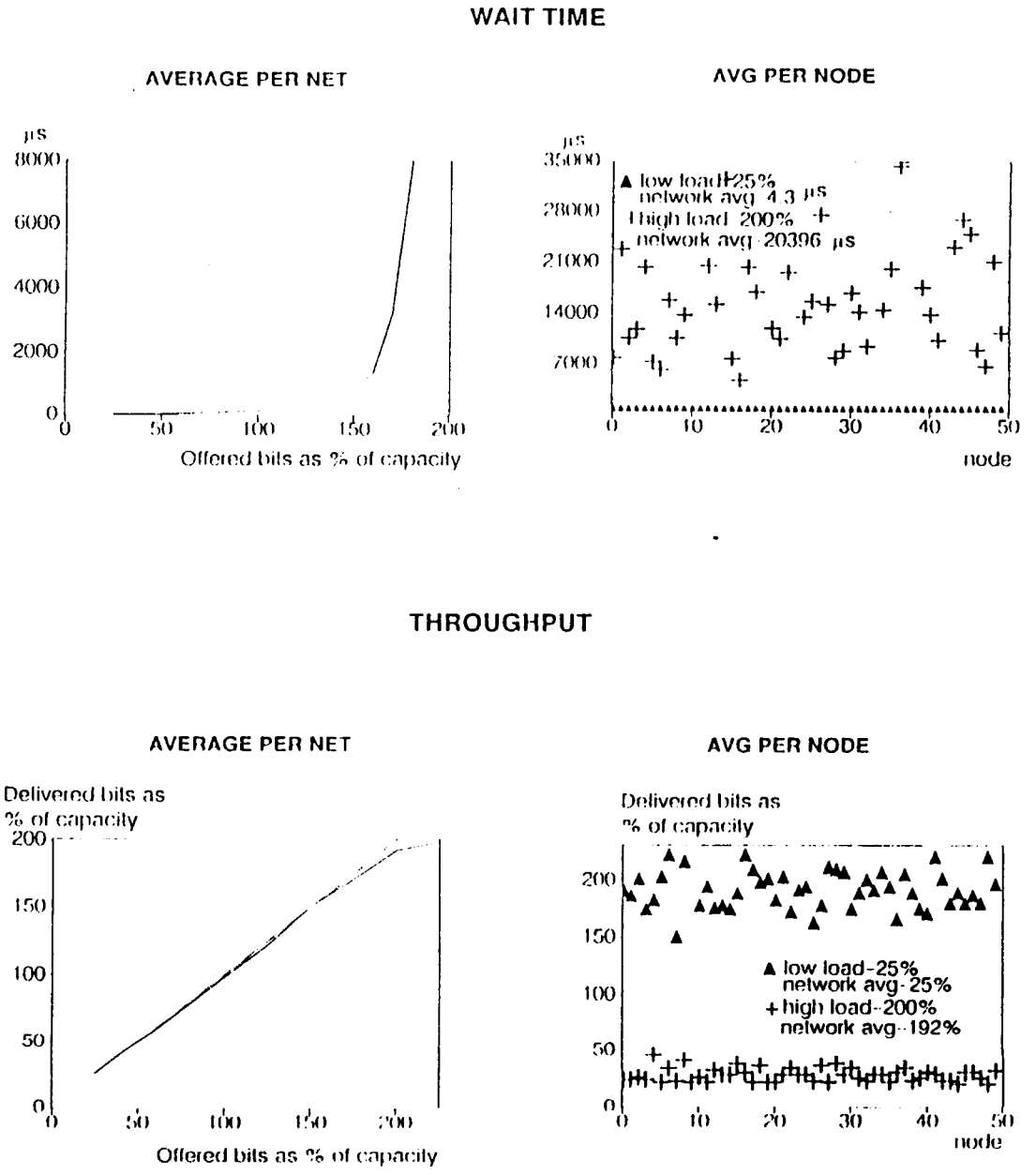


Figure 6: Fairness Graphs for CSMA/RN at 100 Mb/s

along the course of time. Not visible in the graph is the information that till 150% load no node starves, at 200% load four nodes suffer severe starvation sending from 166% to 176% and at 225% load nodes had throughput between (90% - 200%), twelve nodes had between 180%-190% and five nodes had below 180% throughput. Since the results are based upon random distribution, conditions will vary over different runs and over different intervals in a run.

In Figure 6 we have run CSMA/RN at the speed of FDDI - 100Mbps. As expected CSMA/RN performance has deteriorated although it is still better than FDDI in terms of throughput and wait time metrics. The point of instability lies between 150% and 175% of offered load instead of being beyond 200%. The starving node phenomenon is observed at 100% offered load and it deteriorates at higher loads. At 200% load sixteen nodes are severely affected and the variance of wait time and throughput for nodes is even higher than CSMA/RN operation at 1Gbps.

Research is ongoing to ensure that no node starves in this protocol. As indicated earlier, we are developing automatic load balancing algorithms at the media access level to solve unfairness problems due to non-uniform load distribution. As a side effect we expect that these algorithms will also solve the fairness problems for uniform load distribution.

2.3 DQDB

DQDB [10] uses a dual slotted bus to send segments (messages are partitioned and reassembled by the protocol) of 312 bits. A node uses one bus to send downstream and the other bus to send upstreams. Reservation bits are sent in the opposite direction to alert nodes to let empty segments go by to the nodes making reservations. Since the system is totally symmetric we shall illustrate it with sending in one direction only.

Each node has a reservation bit counter which keeps track of how many requests have been made downstream. If a segment arrives at a node the segment gets stamped with the counter value, say x, and the counter is reset to zero. When x empty slots have passed, this segment is sent in the next empty slot.

When a segment arrives, the node attempts to send its reservation upstream. DQDB uses one bit of the 312 bit segment as the reservation bit, hence the node has to wait till a segment with a free reservation bit arrives. Therefore, it is possible that the segment has been sent already while the

reservation bit is still waiting. As a consequence reservation bits can queue up while segments are actually being sent.

In Figure 7 we have shown a particular instance of DQDB. A segment with the destination for node B arrives at node A. Since two nodes have already made reservations (reservation bit counter = 2), the stamp is set to 2, and node A will be able to use the third empty segment, labeled 'a'. In the mean time, the reservation will have to wait till the old one (attempted reservation = 1) gets reservation slot 'b' after which node A will send its reservation in slot labeled 'c'. Thus, the segment will have been sent before a reservation was made. At this point the reservation bit counter will be up to a value of 3.

Under normal circumstances, DQDB will order all segments in the network waiting to be sent if we neglect the time it takes for signals to reach the other nodes. However, that does not apply to nodes where queues of segments (and messages) have built up. Each queue will be ordered and all the front segments of the queues will be ordered but elements in two different queues are not necessarily ordered with regard to each other. We shall denote this case as the *non-ordered* one.

We have developed models with different strategies for when to send the reservation bit and whether or not to fully order messages (and segments). The first strategy for sending the reservation bit is as described above and we shall refer to it as the *aggressive* one. In the second strategy we delay the stamping of the front segment until the reservation bit has been sent. We refer to this strategy as the *non-aggressive* one. A third strategy, the *combined* one, uses a Bernoulli trial to decide what strategy to employ for each message (not segment).

We also investigated the impact of fully ordering segments across all queues. In the ordered case, we attach to the state of the reservation bit counter each segment as it enters the queue and reset the counter to zero. When the front segment is stamped to start the count down, it is from this field that the value for the stamp is taken.

Figure 8-12 are the results of running the six combinations of reservation bit strategies and ordering across queues. Since we show these combinations for only one set of parameters, a short note on the impact of other parameters on DQDB's performance is appropriate. In the aggressive case, the wait time is strictly a function of how many reservation bits have come from downstream and how many segments arrive at upstream nodes before this

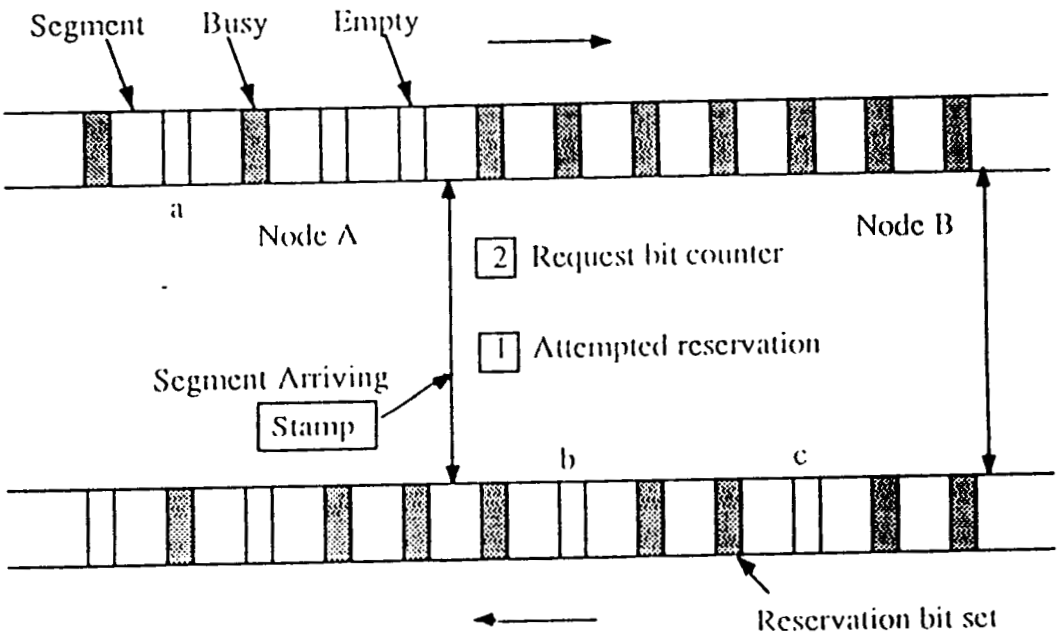


Figure 7: Instance of DQDB

segment's arrival can be signaled to those nodes. This, in turn, is a function of the number of nodes which want to send downstream and what their load is. In the non-aggressive case, the wait time is more directly tied to the availability of empty reservation bit slots coming from the downstream side. If enough nodes, have enough messages arriving, and are located within a short enough distance then it is possible to starve a node because no reservations bit will be empty.

For DQDB, we show the averages per node as a function of the node position and not just as a collection of discrete points. There clearly is a functional relation between node positions and node performance. We remind the reader that the wait time in the following figures are for messages and not for segments.

Figures 8 and 9 illustrate the impact that ordering has on aggressive DQDB. At 80% the curvature of the nodal curve flips from concave to convex; in unstable situations the curvature is concave in both cases. Ordering has also a marked effect on the network average. Comparing Figures 9 and 10 we can observe the effect of changing from an aggressive to a non-aggressive strategy. The non-aggressive strategy produces worse network averages, better performance for the middle nodes and worse performance at the end nodes. This holds true for both throughput and wait time. Figure 11 shows that a simple uniform combination does not flatten out the curve although it produces better performance for more nodes and the number of badly affected nodes is reduced. The cost for this improvement is a slightly worse network average.

For completeness sake we give the two remaining non-ordered cases in Figures 12 and 13 noting that their performances are worse than their ordered counterparts.

Table 1 gives a comparison of the various cases for wait time averages for the entire net at 90%. It should be noted that in all cases the net is on the boundary of being overloaded and wait time per se is not a meaningful figure. Since we made the runs for the same length of simulation time we still can make meaningful relative comparisons.

Figure 14 shows finally a flat curve or in other words a fair protocol. It is obtained by running a non-uniform combination strategy. At the endnodes the probability for using the aggressive strategy is .90 which is reduced parabolically down to .5 for most of the middle nodes. This type of com-

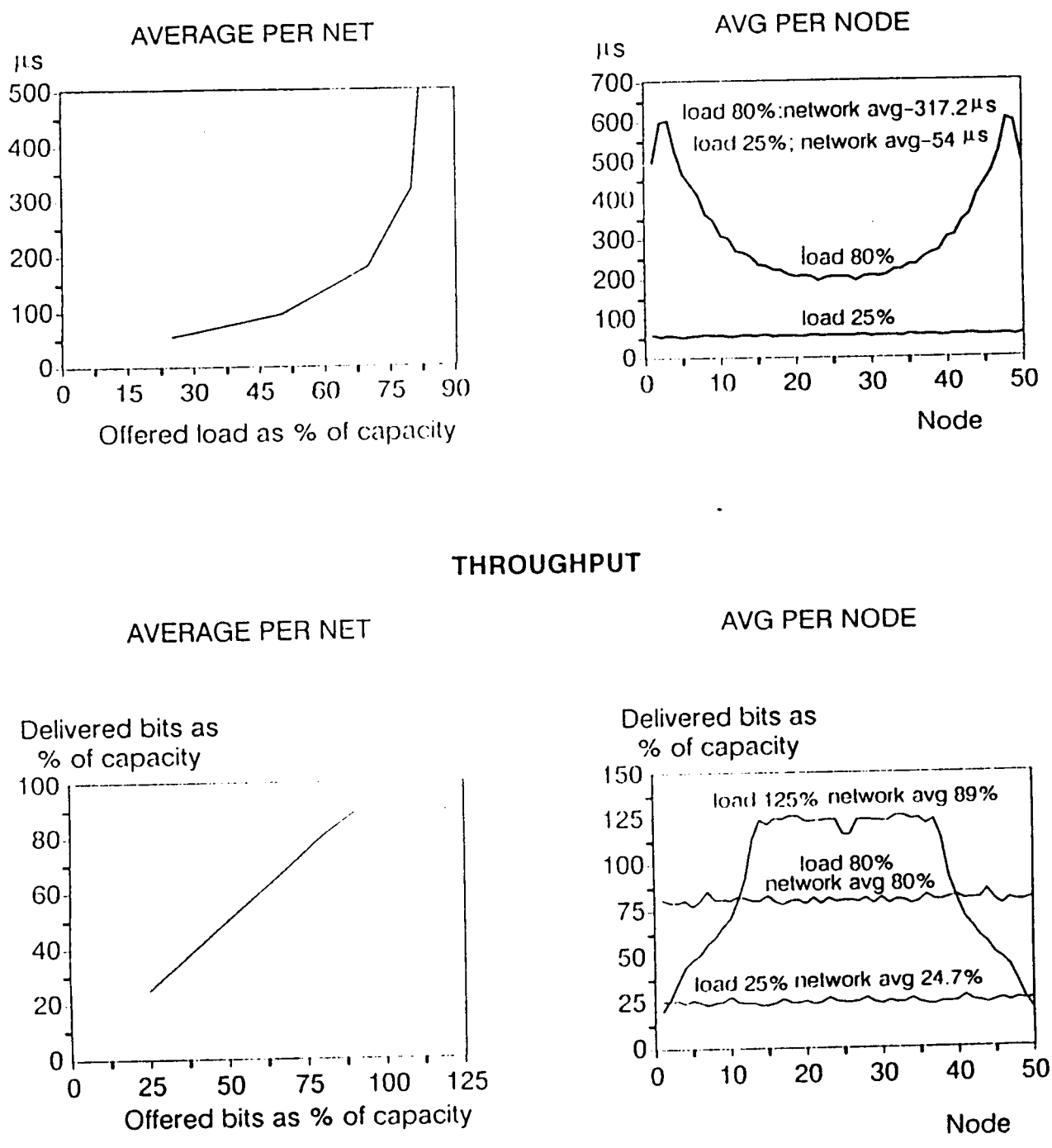


Figure 8: Fairness in non-ordered, aggressive DQDB

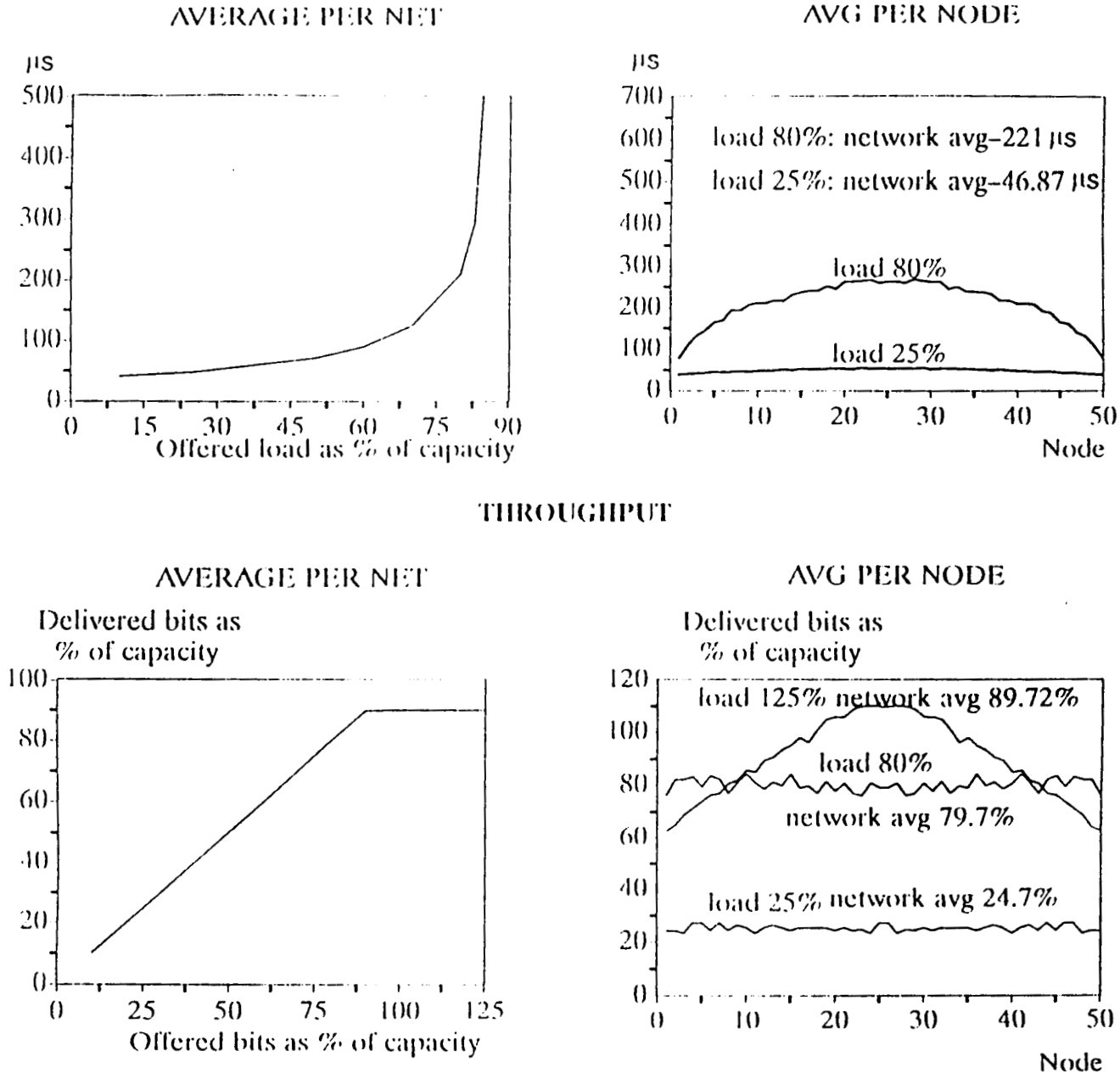


Figure 9: Fairness in ordered, aggressive DQDB

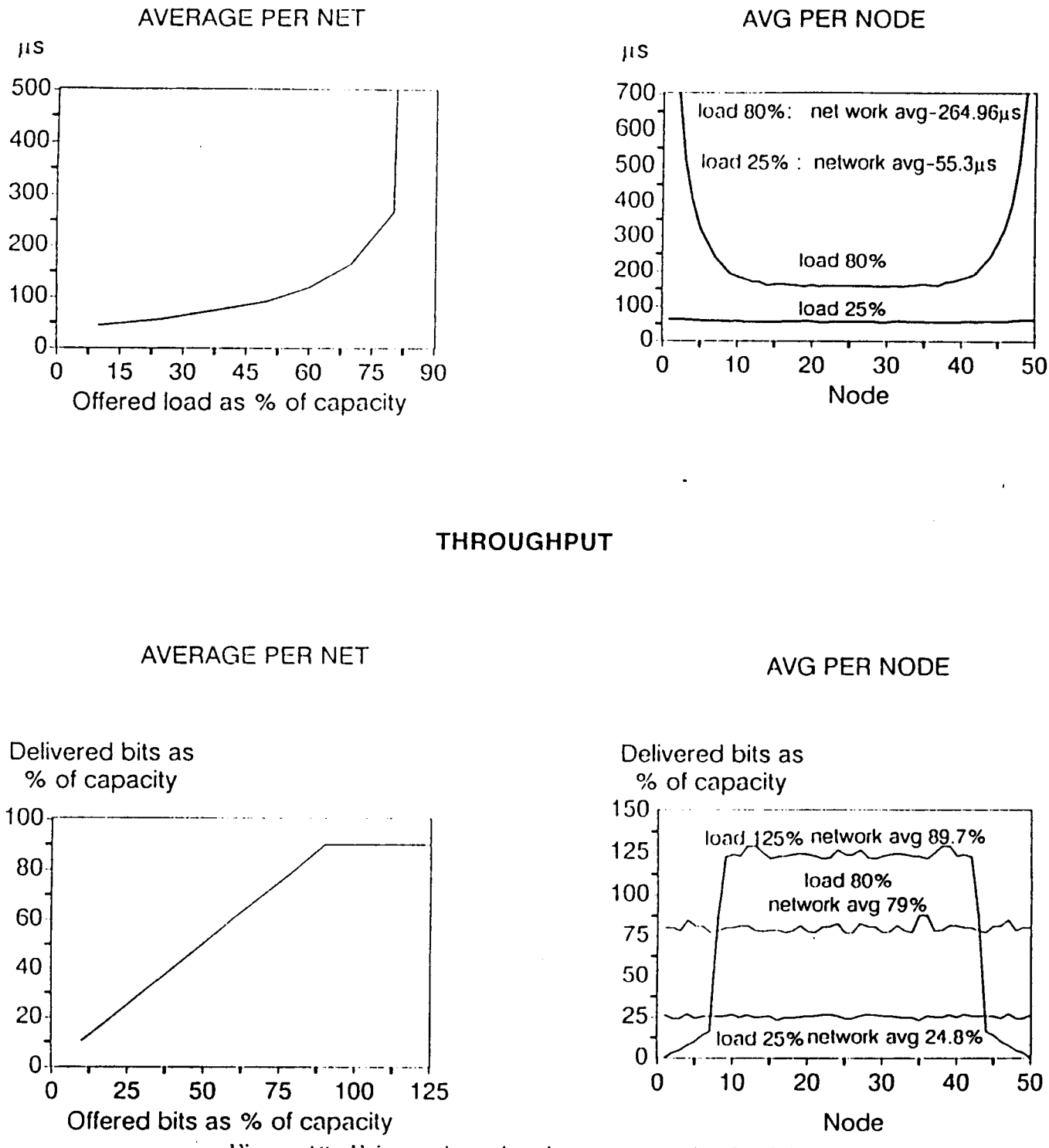
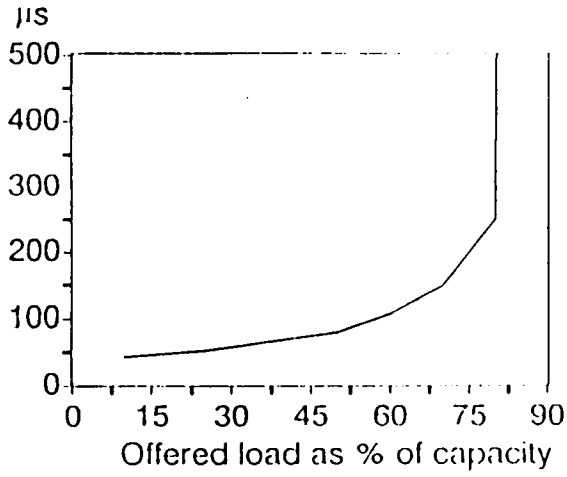


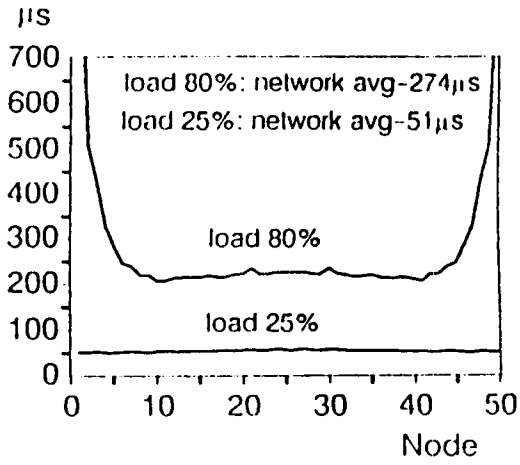
Figure 10: Fairness in ordered, non-aggressive DQDB

WAIT TIME

AVERAGE PER NET

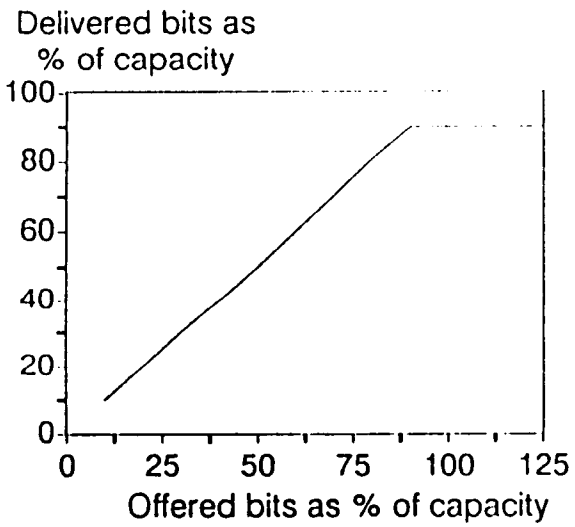


AVG PER NODE



THROUGHPUT

AVERAGE PER NET



AVG PER NODE

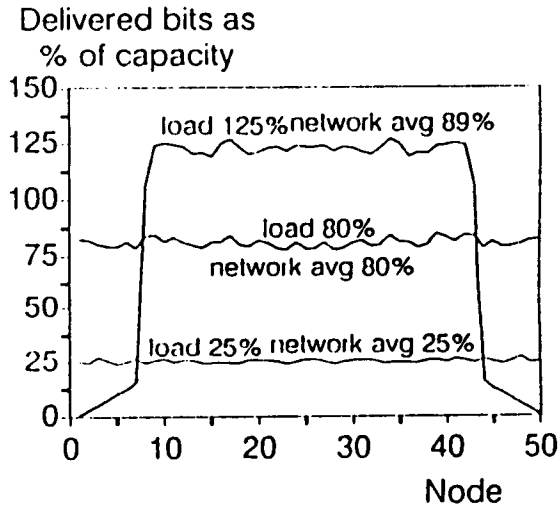
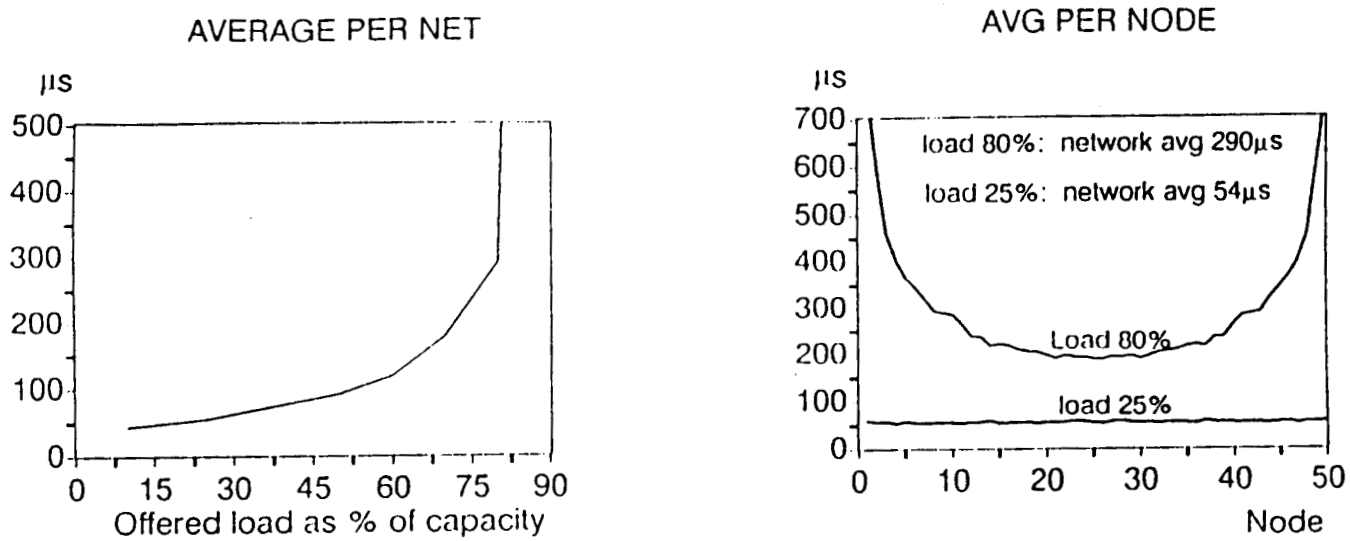


Figure 11: Fairness in ordered, combined DQDB



THROUGHPUT

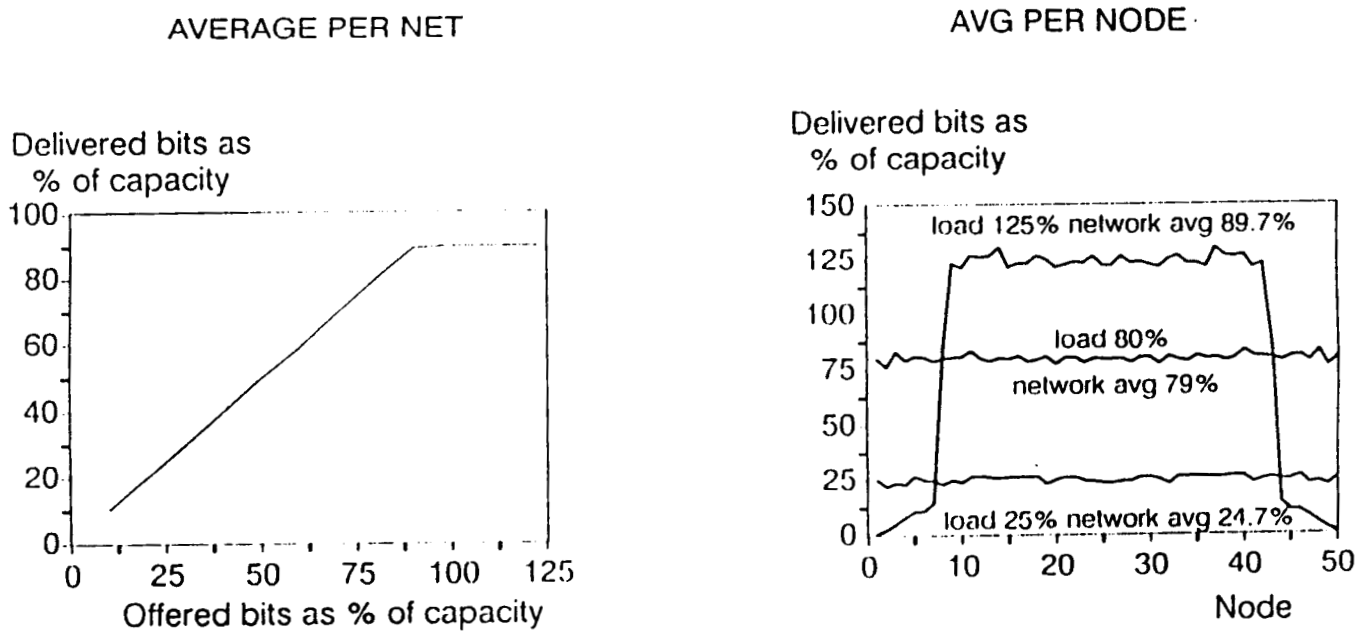
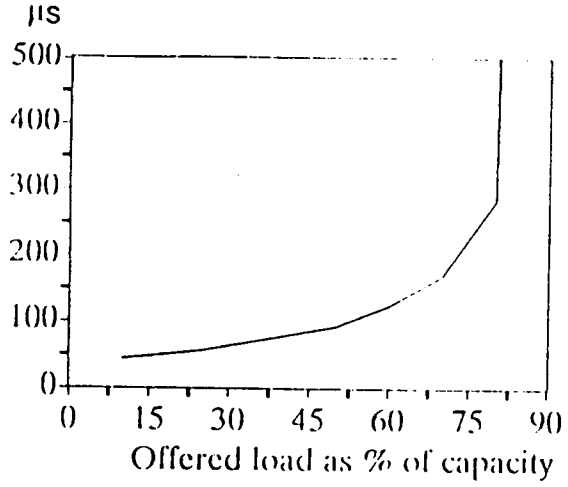


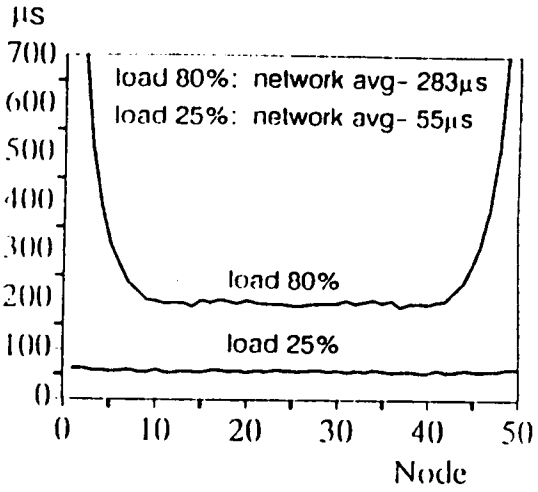
Figure 12: Fairness in non-ordered, combined DQDB

WAIT TIME

AVERAGE PER NET

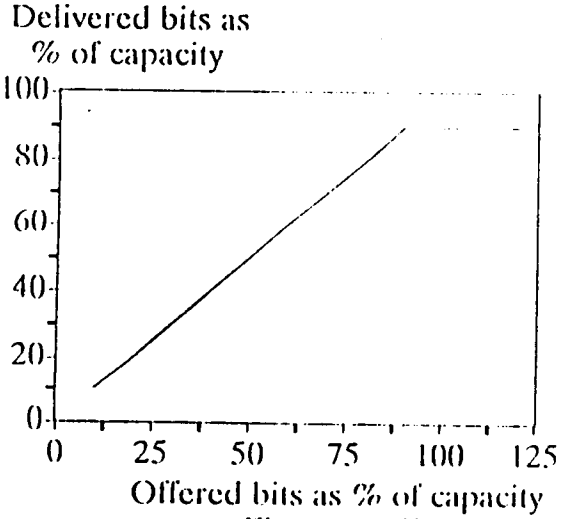


AVG PER NODE



THROUGHPUT

AVERAGE PER NET



AVG PER NODE

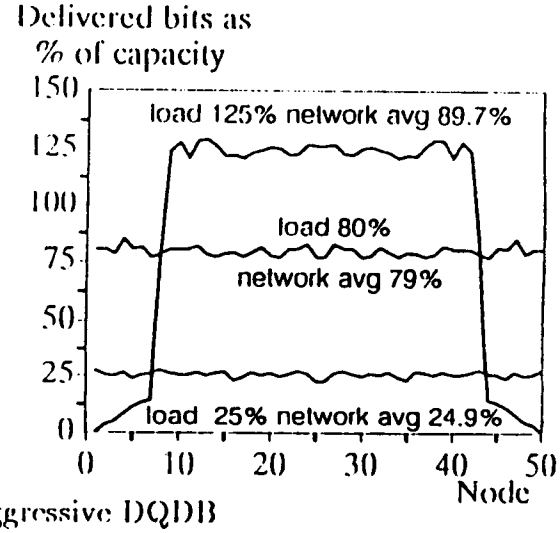


Figure 13: Fairness in non-ordered, non-aggressive DQDB

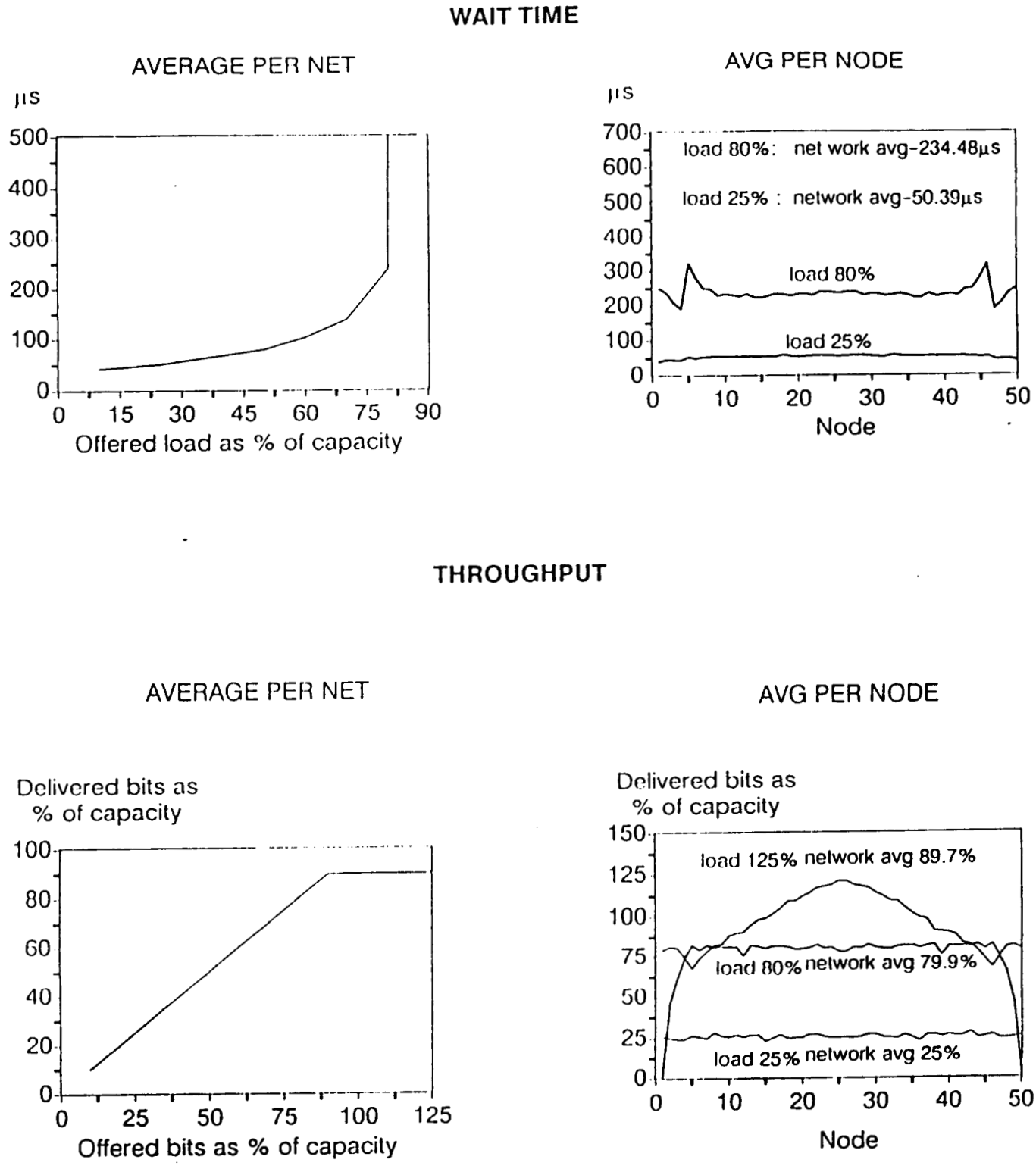


Figure 14: Fairness in ordered, non-uniform combined DQDB

Type of Network Model	Avg Wait Time
Non-ordered Non-aggressive	4440.06 μs
Non-ordered Aggressive	5310.6 μs
Non-ordered Combination	3091.56 μs
Fully Ordered Non-aggressive	4388.48 μs
Fully Ordered Aggressive	9843.61μs
Fully Ordered Combination	8858.03μs
Fully Ordered Non-uniform Combination	7200.7 μs

Table 1: Average Wait Time per Network at 90% Load

bined strategy produces the fairest of the ones investigated although even this combination produces unfair situations at overload. These results are convincing enough to make us believe that DQDB can be made fair at least to the degree FDDI is fair. The avenue we are pursuing, as with CSMA/RN, is to solve the problem of automatic load balancing for non-uniform load at the cost of taking some bandwidth for global communication. This solution in combination with the approach above should enable us to make DQDB fair in all cases.

3 Comparisons and conclusions

FDDI develops random variations in performance across nodes at overload but the variations are well within acceptable limits (about 20%). CSMA/RN shows an increase in variations of the performance metrics as load increases. The variation becomes significant (greater than 100%) at loads greater than 150% for as much as 20% of the nodes and in extreme cases can lead to temporary starvation of nodes. The variations are not a functions of the location of a node but occur in random positions. DQDB shows unfairness at higher

loads as a function of a nodes' position in the network. Depending on the strategy selected end nodes are better or worse off than middle nodes. One particular, non-uniform combination of an aggressive and a non-aggressive DQDB version produces fair performance up until overload.

Clearly, comparing the performance of a 100 Mb/s, a 300 Mb/s, and a Gb/s protocol is akin to comparing apples and oranges. But we wanted to see how the protocols compare in terms of fairness for a fixed data rate. Therefore, Figure 13 and 14 give first a comparison for the network averages of the three protocols followed with nodal snapshots at 80 Mb/s, 270 Mb/s and 2 G/bs. When viewed in isolation FDDI was the most trouble free but when compared with the other protocols at constant capacity it actually is the worst. Other factors such as reliability and cost though will change this picture for different environments and no one absolute statement can be made.

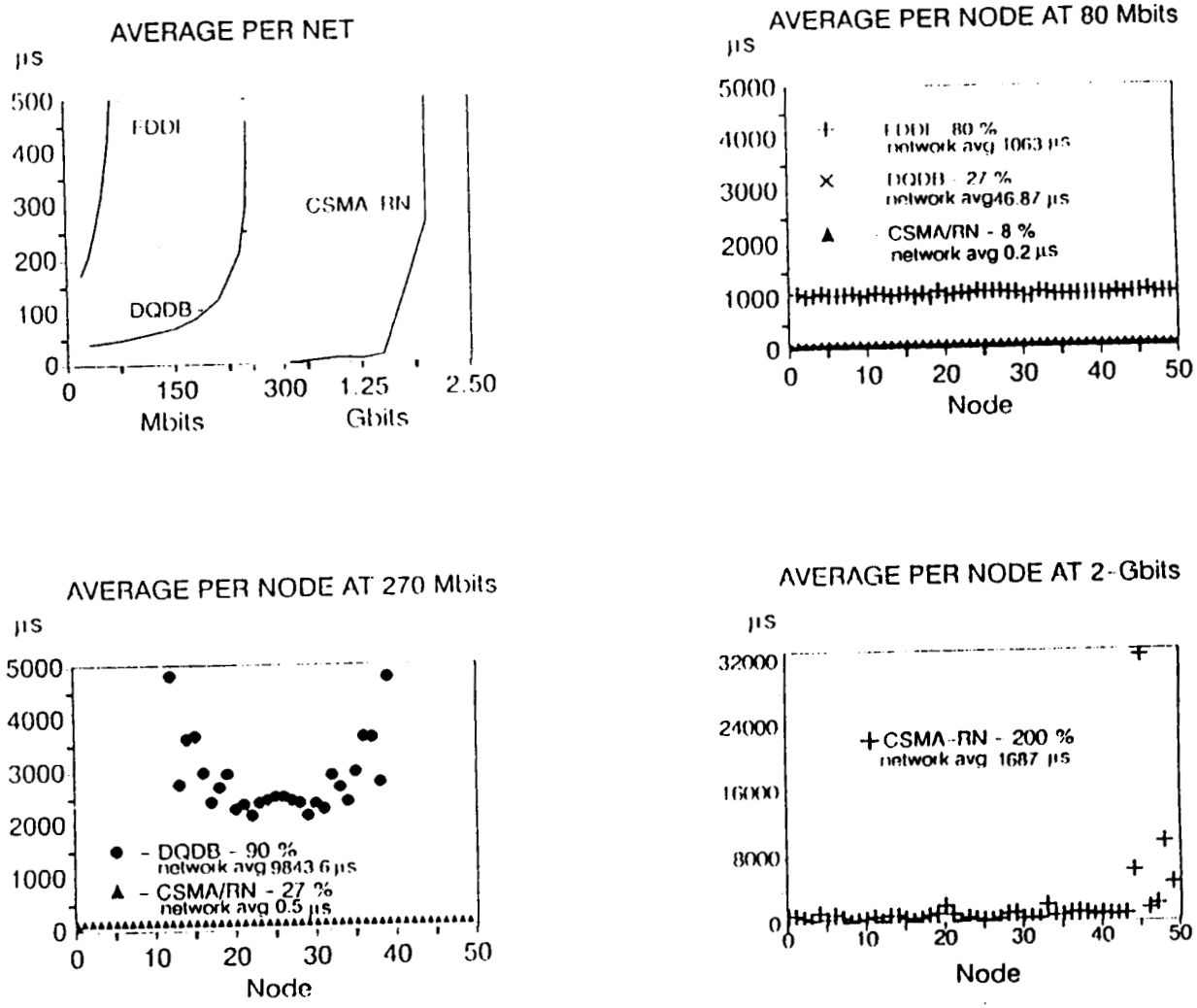


Figure 15: Comparison of access delays for FDDI, DQDB, and CSMA/RN

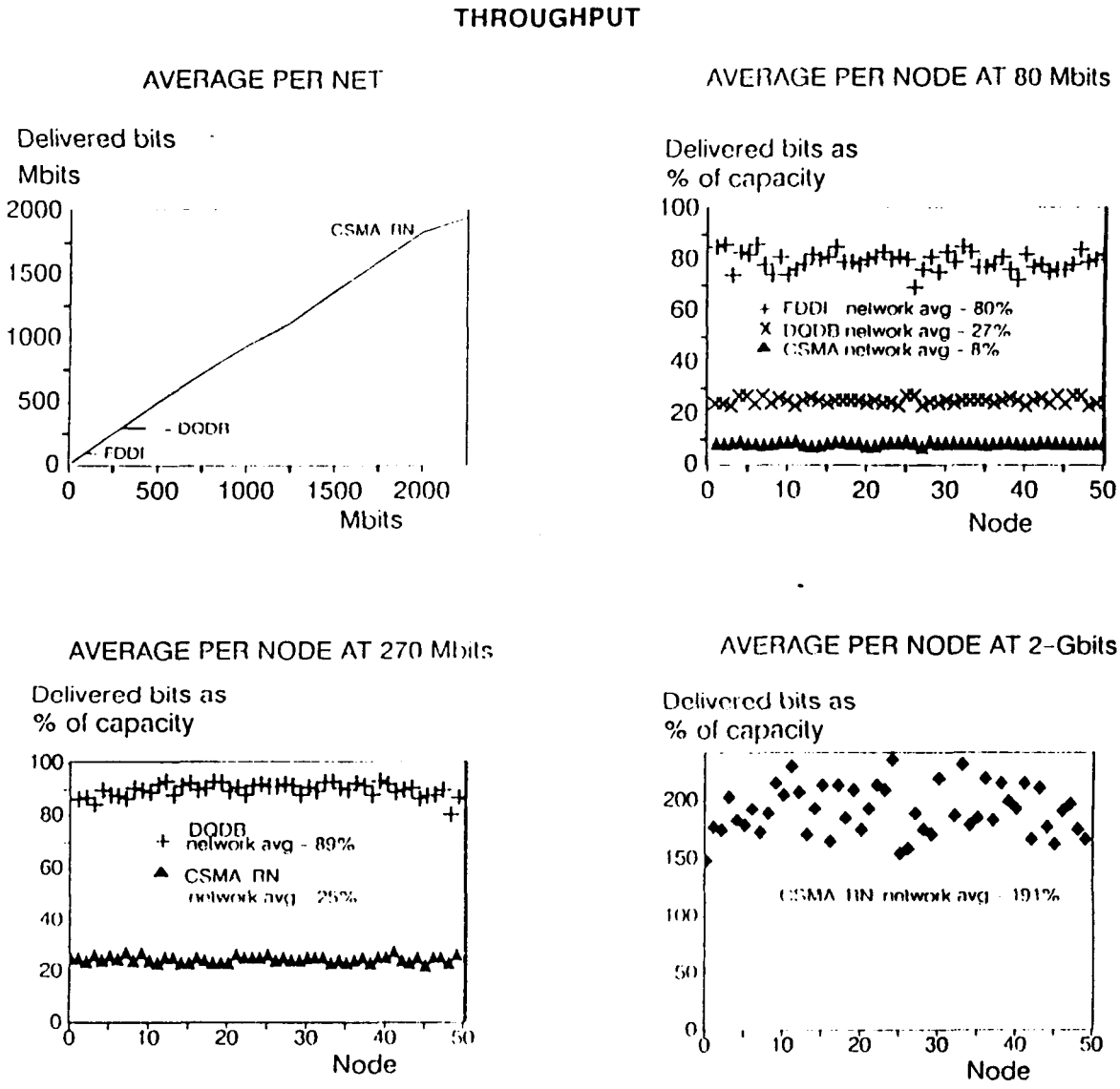


Figure 16: Comparison of throughputs for FDDI, DQDB and CSMA/RN

References

[1] D.R. Boggs, J.C. Mogul, and C.A. Kent. Measured capacity of an ethernet: myths and reality. *SIGCOMM Symposium*, 222–234, August 1988.

[2] W.E. Burr. The fddi optical data link. *IEEE Communications*, 24(5):18–23, May 1986.

[3] E. Foudriat, K. Maly, C.M. Overstreet, Sanjay Khanna, and Frank Paterra. *A Carrier Sensed Multiple Access Protocol for High Data Rate Rings*. Technical Report, Old Dominion University, Feb 1990.

[4] Peter Freeman. High performance computing a brief analysis. *Computing Research News*, 2, no.1:1,19, 1990.

[5] D. Game and K. Maly. *Extensibility and Feasibility of FDDI*. Technical Report, Old Dominion University, Feb 1990.

[6] M.J. Johnson. Proof that timing requirements of the fddi token ring protocol are satisfied. *IEEE Transactions on Communications*, 35(6):620–625, June 1987.

[7] K. Maly, E.C. Foudriat, D. Game, R. Mukkamala, and C.M. Overstreet. Traffic placement policies for a multi-band network. *SIGCOMM Symposium*, September 1989.

[8] K. Maly, C. M. Overstreet, X. Qiu, and D. Tang. Dynamic resource allocation in a metropolitan area network. *SIGCOMM Symposium*, 13–24, August 1988.

[9] R.M. Metcalfe and D.R. Boggs. Ethernet: distributed packet switching for local computer networks. *Commun. ACM*, 19, July 1976.

[10] R.M. Newman, Z.L. Budrikis, and J.L. Jullett. The qpsx man. *IEEE Communications Magazine*, 26(4):20–28, April 1988.

[11] F. Ross. Fddi - a tutorial. *IEEE Communications*, 24(5):10–17, May 1986.

[12] S. Sharrock, K. Maly, S. Ghanta, and H. Du. A broadband integrated voice/data/video network of multiple lans with dynamic bandwidth partitioning. *INFOCOM '87*, 417-425, March 1987.

[13] S. Sharrock, K. Maly, S. Ghanta, and H. Du. A framed, movable-boundary protocol for integrated voice/data in a lan. *SIGCOMM '86*, 9 Pages, August 1986.

[14] J. F. Shock and J. Hupp. Measured performance of an ethernet local network. *Communications of the ACM*, 711-721, December 1980.

[15] M. Skov. Implementation of physical and media access protocols for high-speed networks. *IEEE Communications Magazine*, 45–53, June 1989.

[16] Draft Proposed American National Standard. Fddi token ring media access control (mac) asc x3t9.5 rev. 10. February 28, 1986.

[17] S.L. Wallach. Fddi tutorial: lan industry gets another standard. *LAN Magazine*, 44-47, March 1987.

[18] Paul Young. Challenges for computing research. *Computing Research News*, 1, no.2:1-3, 1989.

[19] Jason S. J. Chen and Victor O. K. Li. Reservation CSMA/CD: A Multiple Access Protocol for LAN's. *IEEE Journal on Selected Areas in Communication*, 202 211, February 1989.

[20] Andrew S. Tanenbaum Computer Networks and Communication. *Chapter 3.*

[21] Morten Skov. Implementation of Physical and Media Access protocols for High Speed Networks. *IEEE Communications Magazine*, 45–53, June 1989.

[22] Mirjana Zafirovic-Vukotic, Ignas G. Niemegeers and Durk S. Valk. Performance Analysis of Slotted Ring Protocols in HSLAN's. *IEEE Journal on Selected Areas in Communication*, 1011-1024, July 1988.

DEPARTMENT OF COMPUTER SCIENCE

Technical Report # TR-90-16

A Carrier Sensed Multiple Access Protocol for High Data Rate Ring Networks

E. C. Foudriat, K. J. Maly, C. M. Overstreet,
S. Khanna, Frank Paterra

Old Dominion University
Department of Computer Science
Norfolk, Virginia 23529-0162
U. S. A.

03-1-90

Old Dominion University
Norfolk, VA 23529-0162

A Carrier Sensed Multiple Access Protocol for High Data Rate Ring Networks*

E. C. Foudriat K. Maly C.M. Overstreet S. Khanna
F. Paterra
Old Dominion University
Norfolk, VA 23529

March 1, 1990

Abstract

This paper presents the results of the study of a simple but effective media access protocol for high data rate networks. The protocol is based on the fact that at high data rates networks can contain multiple messages simultaneously over their span, and that in a ring, nodes need to detect the presence of a message arriving from the immediate up stream neighbor. When an incoming signal is detected, the node must either abort or truncate a message it is presently sending. Thus, the protocol with local carrier sensing and multiple access is designated CSMA/RN.

The performance of CSMA/RN with "attempt and truncate" is studied in this paper using analytic and simulation models. Three performance factors, wait or access time, service time and response or end-to-end travel time are presented. The service time is basically a function of the network rate; it changes by a factor of 1 between no load and full load. Wait time, which is zero for no load, remains small for load factors up to 70% of full load. Response time, which adds travel time while on the network to wait and service time, is mainly a function of network length, especially for longer distance networks.

*Research support has been provided by Sun Microsystems, RF 596044, NASA, Langley Research Center grant NAG-1-908, and Center for Innovative Technology grant INF-89-002-01

Simulation results are shown for CSMA/RN where messages are removed at the destination. Destination removal on an average increases network load capacity by a factor of 2, i.e., a 1 Gbps network can handle a 2 Gbps load. A wide range of local and metropolitan area network parameters including variations in message size, network length and node count are studied. In all cases performance is excellent, and message fracture usually remains less than a factor of 4. Throughput, even at overload conditions, remains high for the protocol. The nominal network rate is 1 Gbps; however, performance remains good for data rates as low as 200 Mbps. Finally, a scaling factor based upon the ratio of message to network length demonstrates that the results of this paper, and hence, the CSMA/RN protocol, are applicable to wide area networks.

1 Introduction

Networks must provide intelligent access for nodes to share the communications resources. During the last eight years, more than sixty different media access protocols for networks operating in the range of 50 to 5000 Mbps have been reported [1]. At 100 Mbps and above, most local area [LAN] and metropolitan area networks [MAN] use optical media because of the signal attenuation advantage and the higher data rate capability. Because of the inability to construct low loss taps, fiber optics systems are usually point-to-point. One of the most straight forward access protocols provides each node with a separate frequency by using wavelength division multiplexing (WDM) either with passive or active star couplers [2],[3]. These systems provide excellent capacity but limit the number of nodes which can be supported since each node must have its own broadcast wavelength. In passive star couplers, the division of signal strength also limits the number of nodes. To overcome these limitations, multi-hop techniques can be used to increase the nodes available but at the expense of slower signal travel time due to staging [4].

In the range of 100 Mbps - 1Gbps, the demand access class of protocols use some form of token, slot or reservation system. Protocol schemes like FDDI [5] use a token, which when received, permits the node to transmit information. Waiting for the token to rotate can cause slow access especially in longer and higher data rate rings. Local sensing of the presence of data or an empty slot is used in slotted and reservation systems. In Expressnet [6], a few bits in the preamble are available to be corrupted if a packet arrives when another packet is being sent by the station. Expressnet however, has

separate transmit and receive sections on the bus and therefore, extends the length of the network by a factor of 2 or 3. Other systems like Fastnet [6] and DQDB [7], [8] (formerly QPSX) provide a pair of unidirectional busses to link the nodes. Fastnet provides a train-like operation started at the master stations at the end of the bus, while DQDB provides a empty/full slot indicator. A slot reservation system is used in DQDB so that down stream nodes have a chance at an empty slot. Recent studies indicate that DQDB have fairness difficulties when servicing nodes at the ends of the bus under high load conditions [9]. Slotted ring access protocols work similar to those of dual bus systems but in a ring configuration. Cambridge-like rings [10] can operate with either master assignment or and empty/full access control mechanism. Finally, some systems have used a delay line [11] or a buffer, like the register-insertion system [12], [13], for alleviating the corruption of data because of simultaneous access.

Broadcast or shared channel communication systems, like Ethernet, use information (carrier) sensing to alleviate the damaging effects of collisions. However, as bandwidth increases and the message size spans a smaller portion of the global bus length, the network throughput is reduced [14]. As the frequency goes up and effective bandwidth increases, the round trip propagation time as measured in terms of packet lengths increases and a larger percentage of the packet or even a number of packets can exist simultaneously over the network span [15]. A resulting collision over the network span wastes time causing lower throughput. Thus, global carrier sensing even with collision detection loses effectiveness. This, coupled with the fact that optical broadcast systems have a difficult time building effective low loss taps, makes global sensing impractical for high speed networks.

As noted above, the amount of space occupied by a packet decreases as network rate increases. For example, at 100 Mbps, a 2K bit packet occupies a space of approximately 4 km along the network ring; at 1 Gbps, this space is reduced to .4 km. Thus, a 1 Gbps, 10 km network can potentially have 25 separate 2K bit packets simultaneously in existence over its span. Ring and dual bus systems realize this sharing of physical network space by having multiple trains or slots distributed along the network length. These blocks can be treated independently and locally, if at any access point :

1. the system can sense and operate on the existence of a data packet or "carrier" *at that point*; and

2. packets, once on the net, are not corrupted by collision with incoming packets during their passage through the node.

In a ring network, packets propagate unidirectionally and synchronously. If each node is able to sense the information or "Tcarrier" in its locality, i.e., in the concept of "Tsense and take action", then the control actions that occur locally and independently can provide an effective means for media access.

In this paper, we investigate the concept of local sensing and control as a means to access a ring network with out corrupting the incoming signal. The system is called Carrier Sensed Multiple Access - Ring Network, CSMA/RN. The paper first describes how carrier sensing can be implemented operationally and some of the possible features which can be used to control the node/network operation. Next, we present an analytical model based on queuing theory to describes the fundamental operational parameters which influence CSMA/RN. We then briefly describe a simulator which has been used to verify the analytical model and to study the network protocol capabilities. Finally, we present results which demonstrate CSMA/RN's ability to provide excellent operational features over a wide range of network conditions and indicate the direction for future work.

2 Carrier Sensing and Control in Optical Ring Networks

Local carrier sensing and collision avoidance using a delay line or buffer have been considered for a tree LAN optical network operating in the Gbps range [11]. This network has transmitting and receiving nodes at the leaves and junction points of the tree. Groups of nodes are clustered into collision avoidance broadcast units. Each unit has a selector system to choose only one packet from the group of nodes for further propagation. The packet that wins the contention continues to propagate to the broadcast link or to the next higher junction if the tree is multi-level. The key to sensing selection is based on a delay line that gives the switch advanced warning as to the future arrival of packets and hence, a chance to exercise intelligence to select a single incoming line and avoid a collision before the packets arrives. This same form of advanced information detection is the key to a collision avoidance and control scheme for the ring network.

2.1 Basic Carrier Sensed Multiple Access Ring Network-(CSMA/RN) Operation

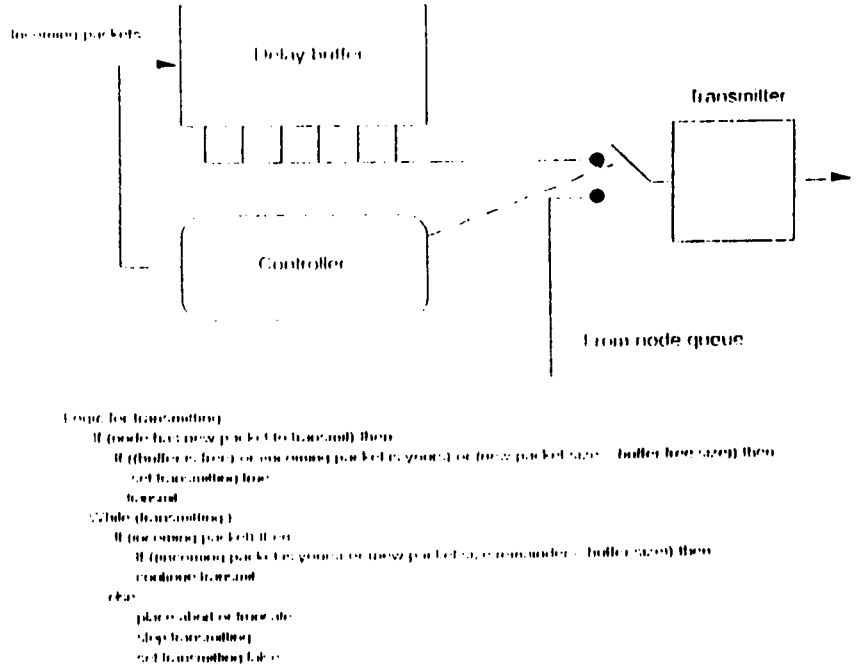


Figure 1: CSMA/RN Logic Operation

Figure 1 illustrates the characteristics of a node in the carrier sensed ring network. The incoming signal is split into two streams, one through a delay line or buffer. A buffer is illustrated here instead of a delay line as used in [11], because it allows greater operational and control flexibility. Note that the delay can be relatively short if high speed logic is used in the controller system. For example, a 100 bit delay at 1 Gbps is approximately a 20 meter piece of fiber and creates a 100 nanosecond delay. The node controller, based upon information accumulated in the buffer, is required to make a number of decisions. First, it must detect the presence of incoming data; if its exists, the node must always propagate incoming information as the outgoing signal to the next node on the ring because it would be impossible to recreate the packet unless sufficient storage is provided. If no incoming packet exists, the node is free to place its own data on the ring if its queue is not empty. However, during the time this latter data is being transmitted, if an incoming packet arrives, then the node, within the time limits dictated by its buffer size, must discontinue its transmission and handle the incoming

packet. Hence, packets once on the ring take precedence over the insertion of new packets.

Packets are tested at each node to determine if the incoming packet is destined for this node and should be copied to its incoming data buffer (not shown in Figure 1). In addition, in order to make room for new packets, old packets must not be allowed to circulate continuously. At the very least, after one revolution of the ring, the source node should delete its packet. Alternatively, the destination node can sense the arrival of information and remove the packet as has been done in some slotted and the register-insertion rings[13].

These controller functions will dictate the buffer delay length and to some extent the packet structure. There must be sufficient delay and logic in the unit, Figure 1, to determine the address of the arriving packets prior to the need to send it on to the next node. The logic is no more difficult for either source or destination removal, but it does dictate to some extent the information structure of the packet header; i.e., the addresses should be placed as near the front in the packet header structure as possible. With high speed GaS logic, it seems reasonable that 100 bit delay buffers at each node would be reasonable as a base line design [1].

2.2 Additional Control Features in a CSMA/RN

Additional operational features can be built in to the buffer control system to assist in network regulation and control. First, if the minimum size information packet is N bits long, then it is always possible to place a message on the system if the node's buffer has N bits of empty space and it does not sense an incoming signal at the time it starts transmitting. Second, if a node is sending a long message and an incoming signal occurs, the node has the option either to abort the entire packet or to truncate and continue the message later. Both these actions are interesting from a network control standpoint. For:

1. **abort** - the operation is similar to "attempt and defer" mechanisms used in many demand access protocols [1], [6], [10], [15], [16]. However, in the case of CSMA/RN, a considerable portion of the message may already have been transmitted and the resource would be wasted as it travels the ring [15]. Hence, the node should place an abort byte at

[1] In addition, there is another delay caused by the transmitter triggering. Using electro-optical computer components, this delay should only be a few bits.

the end of the packet. To recover the resources, a node receiving the packet could detect the abort signal and eliminate part or all of the packet by removing that part still in its buffer. If a part of the message had already been transmitted, the node can place an abort byte at the new end point. This would allow subsequent nodes to remove further portions of the packet until it was completely eliminated from the system. Hence,the network can recover at least some of the capacity otherwise wasted by aborted packets.

2. truncate - the operation would cause message breakup similar to most slotted systems except that slots here are variable length. The node would continue the message in the next free block. The node could place a unique identifier at the end of the packet, so that the receiving node would be alerted to look for subsequent, correlated packets in order to accumulate the total message. This latter mode is the condition studied in this paper.

Packet removal provides interesting opportunities for enhancing network operation [10], [13]. When a node has packets queued for transmission, the detection of an incoming packet destined for that node is an excellent time to start transmitting a queued packet. Implemented at the destination the scheme allows two nodes to communicate, thus establishing a fixed bandwidth, full duplex circuit. More important, however, is that destination removal can increases the effective network capacity by a factor of 2 or better depending on the message origin and destination patterns [13]. In the interest of fairness, this logic may not be desirable especially when done at the source node, since under heavy loads, once a node captured a slot, it would tend to keep it and not give other nodes fair access. Additional logic schemes would allow nodes to cooperate in the use and control of free blocks on the ring [10].

For synchronous or isochronous traffic on the network, a circulating frame system is used in order to guarantee a node sufficient bandwidth to handle its data [10], [17]. In CSMA/RN, this can be accomplished by a node attaining sufficient data blocks with source removal. Upon receiving the return block, the node would replace it with the newly generated information. Thus, a node, over a time period, could accumulate sufficient data blocks to provide both the timing and the bandwidth to handle its required periodic traffic load. As noted above, some restrictions, such as a master controller assignment for blocks [10], [17] could be used so that node do not

accumulate a share of "locked-in frames" sufficient to make the remainder of the ring's operation unacceptable.

3 Comparison to Other Ring Access Protocol Systems

CSMA/RN has features which make it very similar to slotted ring [10], [14] and register-insertion protocols [12], [13], [18]. With regard to slotted rings, it can be considered to be a ring with a slot size of one bit, although slots this small are unusable by CSMA/RN and are passed on as empty. However, since its slot size is variable, CSMA/RN can take advantage of sending large messages without arbitrarily having to break them into smaller blocks. Conversely, it can send small messages without wasting part of a large slot. However, in fixed size slotted rings, the number of blocks needed for a message is known but can not be determined apriori in CSMA/RN. Finally, it does not need to wait for the head of an empty slot, potentially giving better access. Here, CSMA/RN acts more like a random assignment or contention protocol than a demand assignment system [1]. Hence, it should be more efficient and adaptable to a wider range of network conditions than slotted systems.

With its buffer and controller system, CSMA/RN could be modeled to behave identically to the register-insertion (RI) system studied primarily at Ohio State University [12], [13], [18]. First, the idea of message removal at the destination is adapted from the RI system as it provides a factor of 2 improvement in throughput. The RI system gives non-preemptive priority to the locally generated message with the incoming message delayed. Conversely, in CSMA/RN, the buffer is strictly a delay line, in order to enable truncation or aborting of the outgoing message and to enable detection of incoming messages destined for the node. Thus, packets experience predicable delays when traversing the ring, i.e., message travel time is a fixed, predicable quantity. In doing so, CSMA/RN suffers from the fact that packet sizes on the ring may vary and that unpredictable message fracturing can occur. For low data rate networks, where registers are necessary for RI to operate at all, message fracturing is a significant problem. At high data rates, where the ring can contain many messages simultaneously and large blocks of empty space may be available, reasonable message fracturing should not severely hamper ring operations.

A hybrid media access protocol which senses a carrier is presented in

reference [15]. This system operates in two modes, multiple train mode which is very similar to CSMA/RN but where TTattempt and abort" without recovery is used. At high loads, throughput is greatly reduced because of collision, so the network transfers to a single train mode of operation to avoid collisions.

In all cases, CSMA/RN differs in that it uses the concept of TTattempt and truncate" instead of TTattempt & defer" used in demand access protocols or TTattempt and abort" used in the hybrid ring.

4 Access Analysis

A study was conducted to build an analytical model for CSMA/RN operation. The analysis considers only the basic CSMA/RN system without the additional control or abort features. In doing so, a number of analysis configurations were examined including those which were used to model the register-insertion ring [12], [18], [19] and others

based upon priority and preemptive queuing models [20]. After examining these, it was found that a relatively simple queuing theory model can provide acceptable results.

Only a single node need be modeled since logic operations at each node are independent. The message traffic is represented by a Poisson arrival process based on the network load. The analysis evaluates the capability of the node to insert its fixed length message into the ring data stream. The insertion of the total message is defined as the service time for the node. The service time includes the condition where a message may be delayed several times for packets on the ring arriving at the node. Thus, the task is to define a model for calculating service time versus message load based upon the expected arrival of empty packets and to determine what effect the fracturing of packets will have on the service time.

To simplify the analysis, the ring is assumed to be large enough so that an TTinfinite" number of packets the size of the message can be place upon the ring. The probability of an available packet arriving at the node is:

$$Pr(x = available) = p = (1 - lf(n-1)/n + lf(n-1)/n^2) \qquad (1)$$

where lf= *loadfactor*; > n= *numberofnodes*; and

where an available packet is describe as one which is either empty or one whose destination is the node under consideration.

Considering each packet condition to be statistically independent, the probability that the kth packet is the first one available is:

$$Pr(t_k) = (1 - p)^{k-1}p \qquad (2)$$

As load increases the empty space on the ring tends to fracture. This can further increase the service time, since now more than a single packet is needed to service the message. Packet fracturing is modeled by assuming a statistical distribution of packet size. It is based upon observations made during the simulation runs (to be discussed in greater detail later) that a portion of the packets have sizes which are uniformly distributed up to the message length and that the remainder have sizes equal to or greater than the message length. Thus, the probability density of packet size is model as:

$$p(s) = \begin{cases} k_u + k_m\delta(s_m) & \text{for } 0 < s \leq s_m \\ 0 & \text{for } s > s_m \end{cases} \qquad (3)$$

where > s = packet size, and
> s_m = message size, and
> δ is the dirac delta function.

The values of k_u and k_m are interrelated, by $\int_0^{s_m} p(s)ds = 1$ so that: $k_u = (1 - k_m)/s_m$.

It was observed in the simulator runs that k_m varied as load changed; at low loads, $k_m \rightarrow 1$; at high loads, $k_m \rightarrow 0$. A simple linear equations representing these conditions is:

$$k_m = \begin{cases} (1 - lf) & \text{if } 0 \leq lf \leq 1 \\ 0 & \text{elsewhere} \end{cases}$$

The service time is:

$$S = S_{nl}E\{t_k\}/E\{s\} \qquad (4)$$

where S_{nl} = no load service time, > $E\{t_k\}$ = expected arrival time for an empty packet, and

The value for $E\{S\}$ in equation (4) was calculated both with $E\{S\}$ based on equation (3) and with $E\{S\} = 1$ and compared to simulator runs. Figure 2 shows the comparison for the condition $E\{S\} = 1$ which provided superior results.

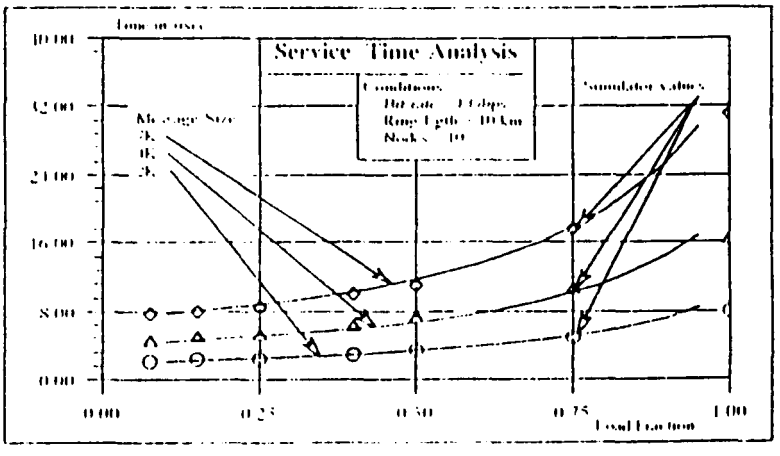


Figure 2: Service Time Comparison Between Analytical and Simulator Models

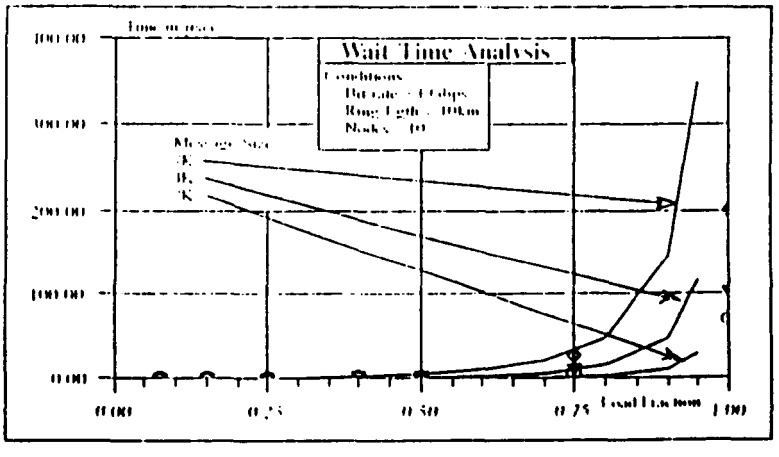


Figure 3: Wait Time Comparison Between Analytical and Simulation Models

Figure 2 demonstrates that service time can be simply and accurately estimated for CSMA/RN by a model which takes into account no load service time and the expected arrival time for empty packets. While packet fracturing may be a factor in service time, it is effectively compensated for by the fact that as packets fracture and become smaller they tend to arrive at the node more rapidly. Hence, in a ring where the message size is small in comparison to ring size, node performance can be modeled by expected available packet arrival.

Based on the service time analysis, the expected wait time for a message is calculated using the Pollaczek-Kintchine formula [20]:

$$E\{w\} = \lambda E\{S^2\}/2(1 - \rho) \tag{5}$$

where $\rho = \lambda E\{S\}$. The calculated values for $E\{S\}$ and $E\{S^2\}$ were taken from equation (4) and by calculating $E\{l_k^2\}$ from the distribution given in Equation (2), respectively. The comparison between calculated and simulated results is shown in Figure 3.

In Figure 3, we have plotted the calculated value of wait time only up to 0.95 load fraction since the value goes unstable for load fractions that approach $\rho = 1.0$. For these conditions, the calculated results accurately represent those obtained from the simulations runs. Hence, we can with confidence model CSMA/RN as an available packet arrival queuing system for those conditions where packet length is small in comparison to ring length.

The response time is defined as the time from message arrival at the source until the time the message reaches the destination. Hence, the expected response time is given by:

$$E\{R\} = E\{w\} + E\{S\} + E\{T\} = E\{w\} + E\{S\} + L/2 \tag{6}$$

where T = travel time of the message on the ring, and
L = travel time to complete one cycle of the ring.

$E\{T\} = L/2$ assumes uniform selection of destinations amongst the nodes. Simulation results presented later will illustrate that equation (6) accurately models the response time, especially for large size rings and low to medium loads, where the response time is primarily the travel time for a message.

5 Simulation System

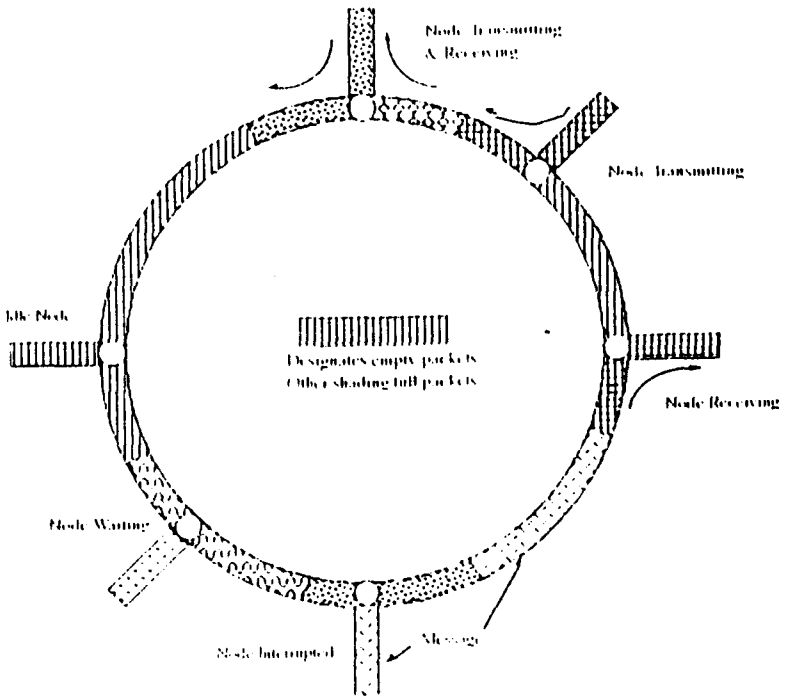


Figure 4: Illustration of Network Operational Conditions

In order to determine the capability of CSMA/RN as an access protocol, a discrete event simulation model was built. This model is designed to enable the rapid determination and handling of the events that can occur at each node based upon the travel of empty and full packets of data around the ring as time progresses. The events are those which can occur at a node by the arrival of a message, by the condition that an empty packet is passing the node when a message arrives, by the arrival of an empty packet at a node which has a queued up message, by the arrival of a packet destined for a node and by interrupting a node which is transmitting. The set of conditions which create events on the ring is illustrated in Figure 4. It is interesting to note that, although the logic conditions at each node are simple, the

logic conditions for the ring become quite complex when it contains a large number of nodes and the decisions at a node eventually influence other nodes on the ring.

In the simulation, the ring is modeled as a linked list of packets; each packet containing information on its length, its location conditions, etc. Thus, the condition of each bit of the ring based upon the network bit rate, length and propagation speed are embodied in the ring data. Events are formulated by examining the condition of each node, as noted in Figure 4, to determine the time that the node should transmit based upon its ready condition and the condition of packets approaching its location. Once a packet is placed by the node, its travel time is known so that the event related to its arrival can be calculated at placement time. A time ordered list of events is maintained. Time increments are related to bit size; for each new event, all packet locations on the ring list is incremented and the new event processed to change ring state. Nodes are modeled as fixed data structures and maintain the information as to their present status and their past message handling events.

In conducting simulation runs, questions arose as to how long the simulation runs should be in order that conditions on the ring had stabilized to steady state conditions and that sufficient time elapsed so that statistical data collected was reasonably accurate. A series of tests were conducted to assess the confidence which could be placed on the data collected during a simulation run.

Figure 5 illustrates the type of runs made to study simulator confidence. Data were collected for intervals during a run and compared as to their variability and to the mean of all data collected for the run. In Figure 5, we have plotted wait time, the most sensitive of the variables, taken at the end 10 intervals, and the cumulative average taken of the active period of the run. Load fractions of 1.0 and 1.5 were used since at the higher loads, fluctuations tend to be greater. First, it was found that the ring tended to reach steady state values rather quickly, but that it results still varied considerably between interval. It was found that in order to obtain data with a reasonable confidence in the mean accuracy, the ring had to cycle a number of times, where a cycle is the time for information to completely traverse the ring. In general, about 1000 - 5000 cycles was found to be sufficient elasped time.

Still at the 1.5 load condition, results may vary considerably because of the sensitivity of wait time to load and service time variations.

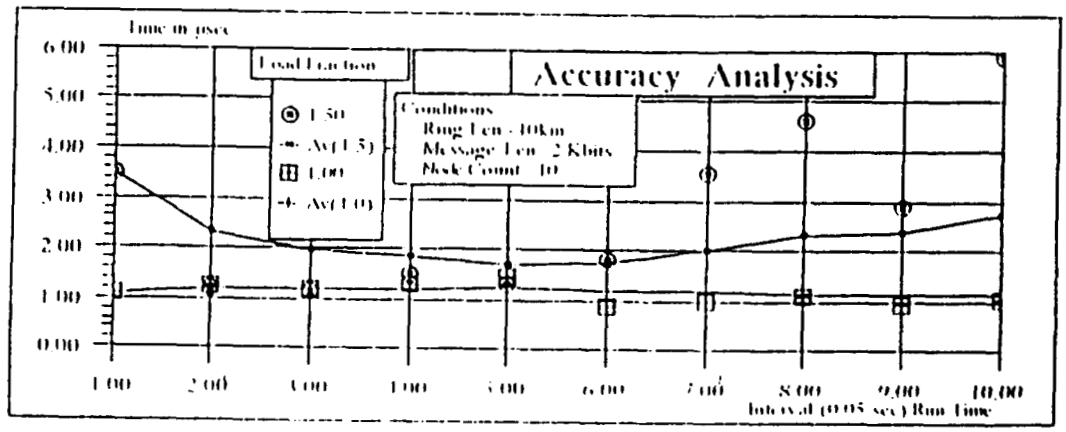


Figure 5: Simulator Run Convergence and Variations

6 Protocol Performance and Operational Features

The simulation system was used to study and document the access performance of the CSMA/RN system. The basic performance measures which were obtained from the runs are wait, service and response time as defined in Section 4.0. In addition, message fracture was obtained in order to estimate the conditions under which fragmentation of the ring could be found to occur. In addition, the ability to handle overloads without degradation in throughput, is critical for network performance. Specific runs were made to study throughput at high loads. Finally, runs were made to determine conditions under which the results can be scaled to related but unstudied conditions.

General conditions for all simulator runs include:

1. packets were removed at the destination and the empty space used by the node to send queued messages,

2. additional header bits required because of packet fracture were not added to the message,

3. nodes are uniformly spaced around the ring,

4. all message arrivals are uniformly distributed among the nodes,

5. all message destination addresses are uniformly distribute among the nodes other than the source node, and

6. all messages are fixed length.

The analysis results, Figures 2 and 3, indicate that under nominal conditions CSMA/RN can provide excellent performance as an access protocol. First, access or wait time approaches zero at no load and remains relatively flat until the load approaches 70% of the network load. As load increases, wait time, which is dependent upon service time, becomes unstable as $\rho \rightarrow 1$. Service time is close to the minimal, no load service time throughout most of the load range; it remains within a factor of 2 for load levels up to 60% network load and with a factor of 4 for loads up to 95%. Finally, since travel time for a message on the ring is fixed by the media propagation speed, the total response time is dependent upon ring length in MAN and larger LAN networks. In any case, the CSMA/RN access protocol does not slow the travel time, so that a message, once on the network will move as quickly as possible to the destination.

The simulation studies were done to determine its performance under a range of system parameters. Three conditions are of major interest: message length, ring length and the number of nodes in the system. Three sets of runs were made to examine the effects of these variations.

Figures 6a - 6d present data for a 1 Gbps, 10km, 10 node ring for message lengths ranging from 2K to 20K bits. As noted previously, one of the best performance features of the CSMA/RN is immediately apparent in this and all subsequent figures -- the 1Gbps network is capable of handling up to 2 Gbps without saturating, because, on an average, messages travel only half way around the ring. Thus, load performance for CSMA/RN and other destination removal networks systems like register-insertion [13] is at least double the basic net speed bandwidth. We see from Figures 6 that average performance characteristics for CSMA/RN are not detrimentally altered by message length. First, mean wait time is very consistent with that predicted by the analysis.

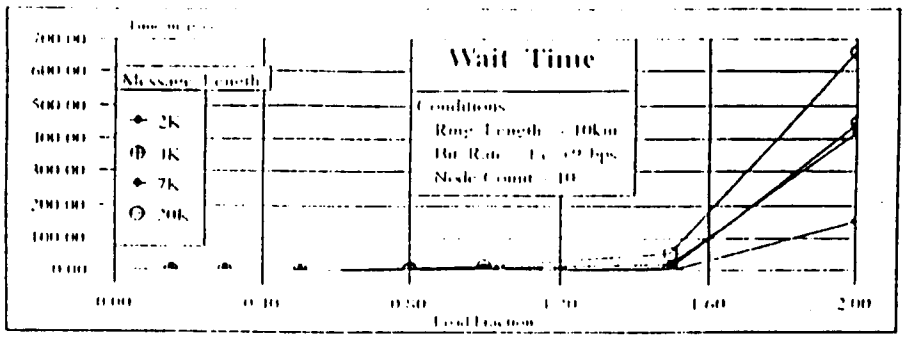


Figure 6a: Wait Time for Various Message Lengths

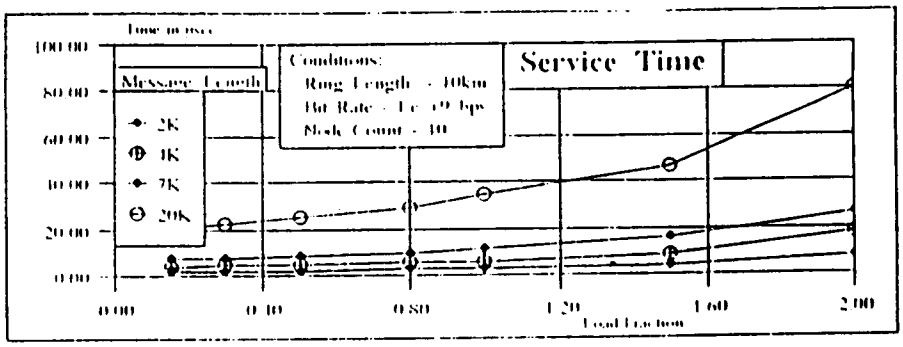


Figure 6b: Service Time for Various Message Lengths

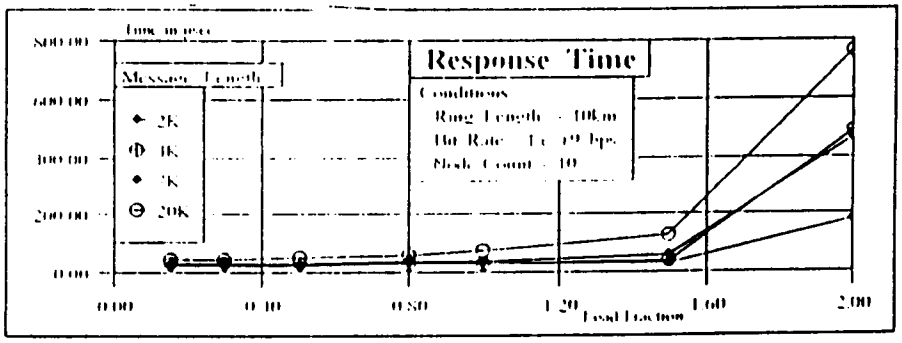


Figure 6c: Response Time for Various Message Lengths

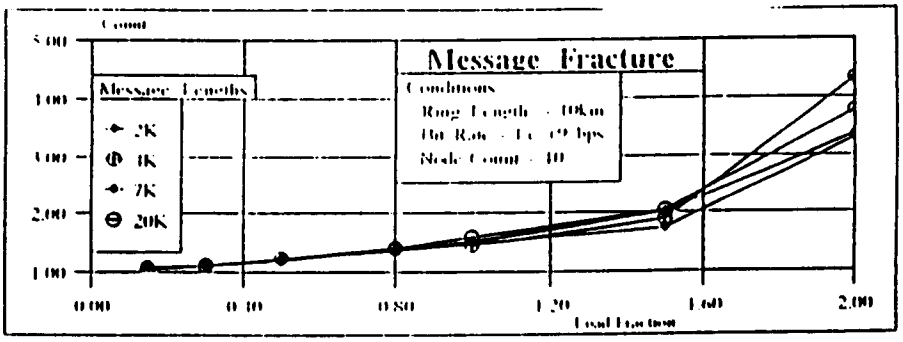


Figure 6d: Message Fracture for Various Message Lengths

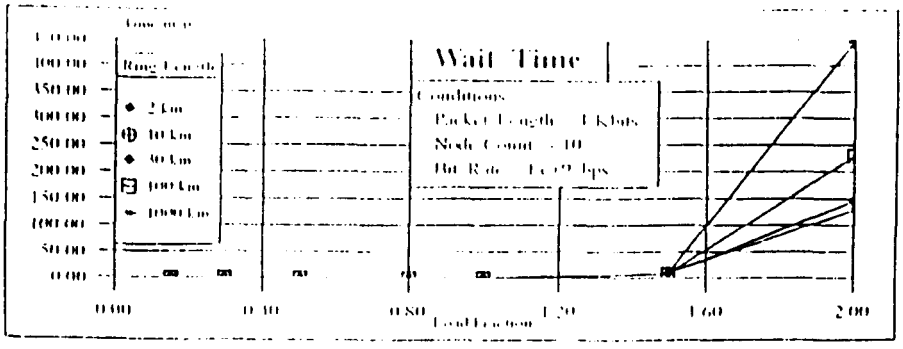


Figure 7a: Wait Time for Various Ring Lengths

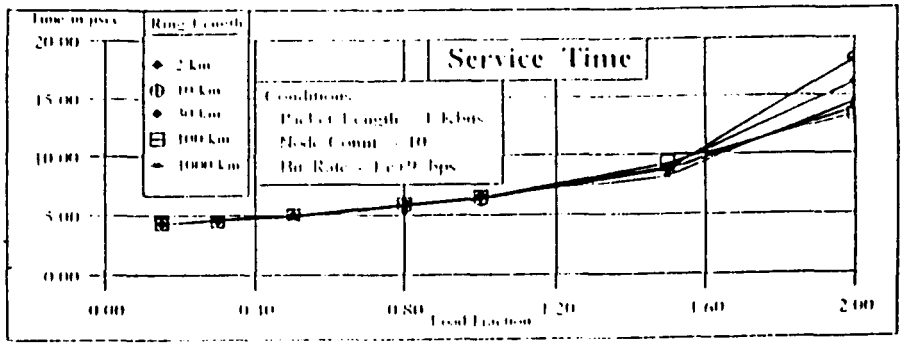


Figure 7b: Service Time for Various Ring Lengths

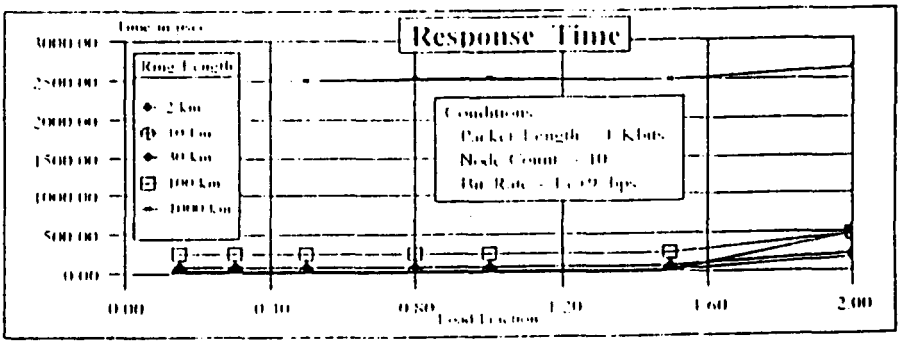


Figure 7c: Response Time for Various Ring Lengths

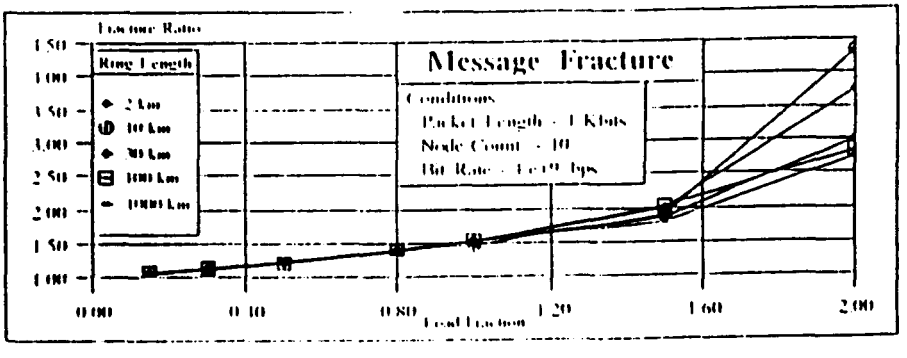


Figure 7d: Message Fracture for Various Ring Lengths

As in the analysis, no load service time is a function of packet length but the shape of each curve is similar to that predicted by the analysis results. 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 Figures 7a - 7d to show the effect of ring lengths. Here, rings lengths range from 2 km to 1000 km. In all cases, ring length affects mean wait time only after the load has reached 80%. After this point, wait time does not seem to have a consistent variation with ring length. The differences are probably due to the variances in random load generation and the sensitivity of wait time to service time in this region which can cause the system to become unstable. The average service time shows little difference for the wide range of ring lengths. In general length should not have much effect as the service time is mainly dependent on the existence of arriving available packets. Response time shows significant dependence upon ring length, mainly due to the travel time necessary from source to destination. In the case of the longer length rings, this factor dominates, so service and wait time become insignificant. It is only at the lower lengths, 2 km and 10 km, that other ring factors make any difference. Finally, packet fracturing is not affected by ring length in any significant fashion, illustrating, at least, for the uniform ring loads and node locations that the CSMA/RN protocol provides excellent operations over a range of conditions.

Figures 8a - 8d 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. Message length for these runs is set at 2 Kbits. For the ranges considered, the operation of CSMA/RN is very good. Mean wait and response times correspond to the previous runs and to the analytical results. At a large 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 and subsequently increase service time.

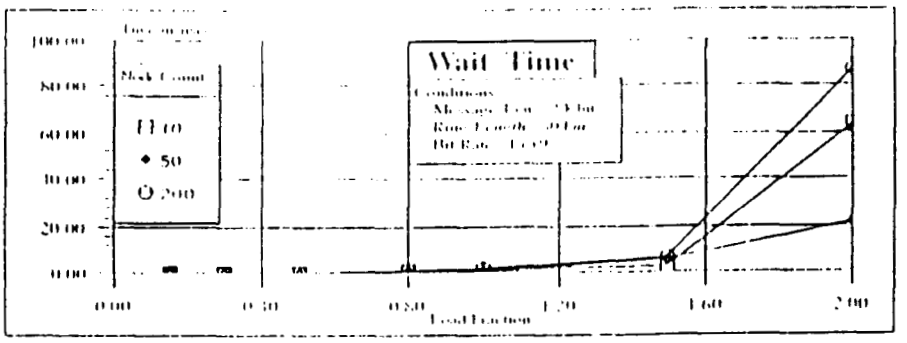


Figure 8a: Wait Time for Various Node Counts

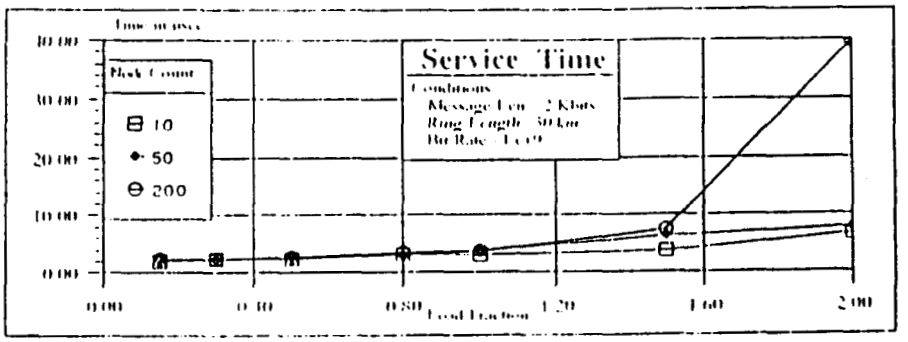


Figure 8b: Service Time for Various Node Counts

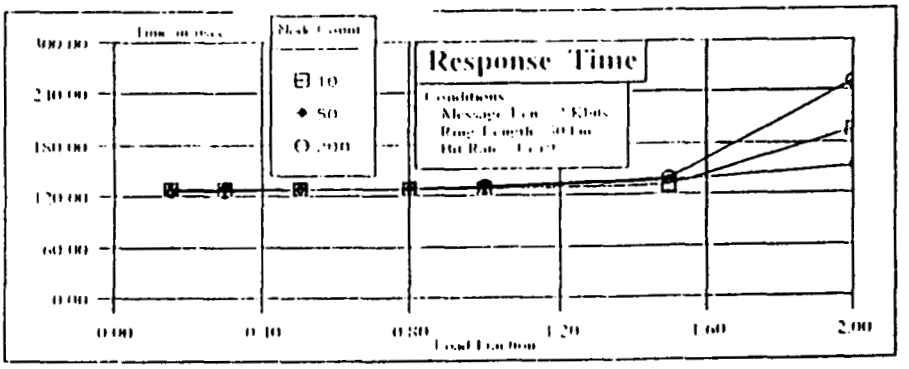


Figure 8c: Response Time for Various Node Counts

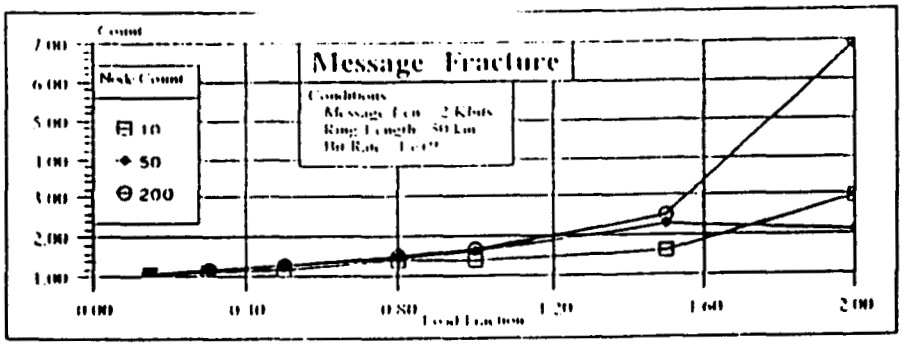


Figure 8d: Message Fracture for Various Node Counts

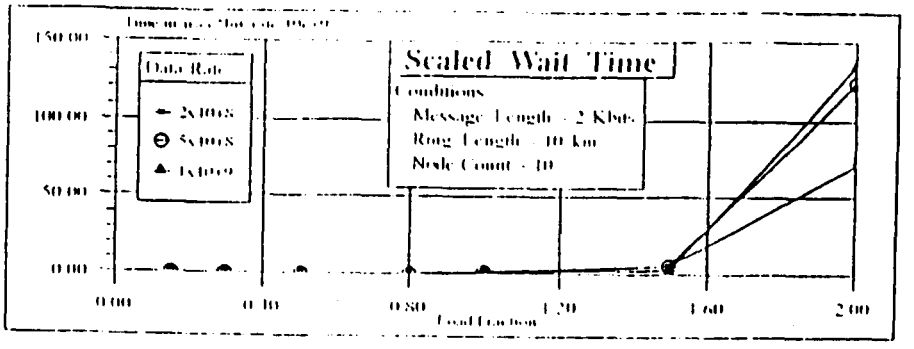


Figure 9a: Scaled Wait Time for Various Data Rates

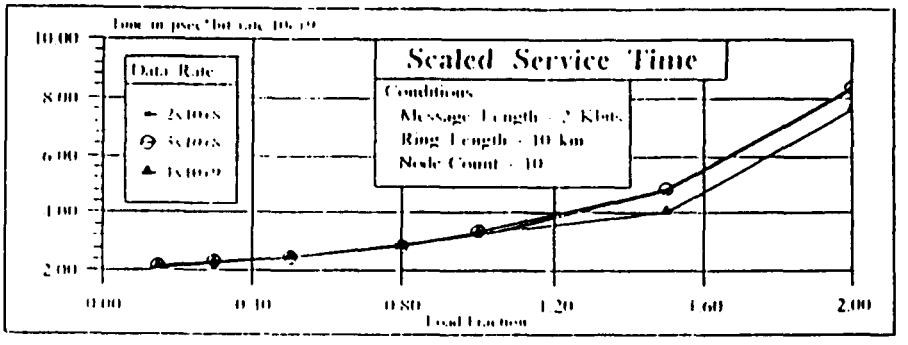


Figure 9b: Scaled Service Time for Various Data Rates

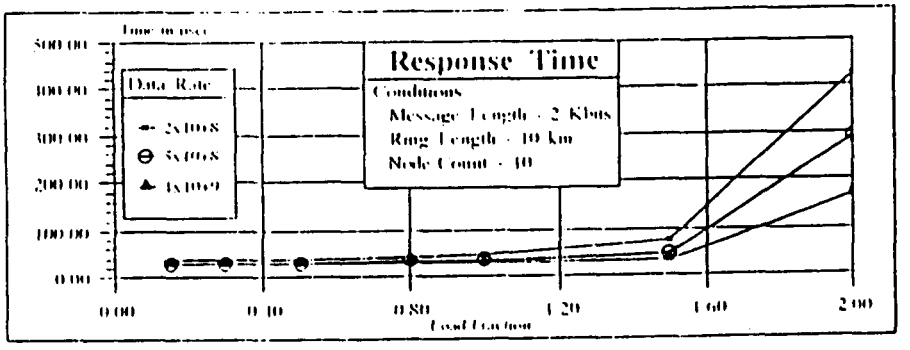


Figure 9c: Response Time for Various Data Rates

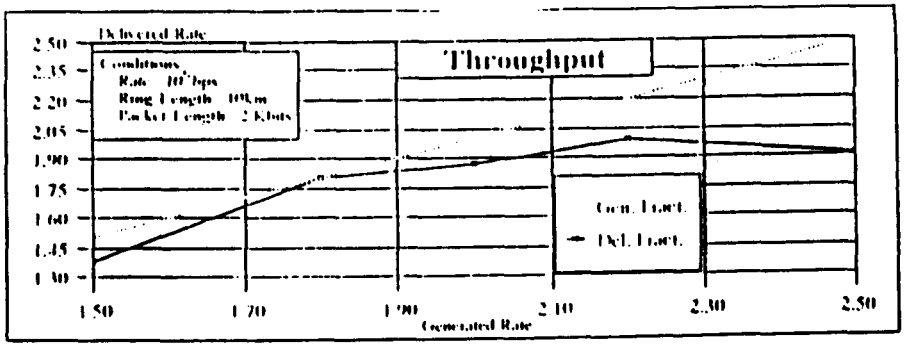


Figure 10: Throughput at High Load Fractions

A set of simulator runs were made to document the performance for network data rates ranging from 2.0 x 10+8 to 1.0 x 10+9. Network conditions are 10 nodes, 10 km and 2 Kbit messages. Figures 9a and 9b show scaled wait and service times, respectively. The scaling has been used to remove the effect of bit rate to demonstrate that the operational characteristics of the CSMA/RN system are independent of bit rate over the range of high data rate networks. Figure 9c shows response time as unscaled to illustrate the advantage available in CSMA/RN as bit rate increases. At low load fractions, most of the response time is caused by network travel time which is independent of bit rate; the secondary factor being service time to get the message on the network. At high load fractions, delay due to wait time tend to predominate, so the higher bit rate for CSMA/RN provides a definite improvement. However, the network performance is very adequate for all conditions considered.

The ability of an access protocol to maintain good throughput under saturated conditions is critical. Runs, shown in Figure 10, were made to examine throughput for loads between 1.5 and 2.5 load fraction. We see that bits delivered is maintained up to the ring saturation limit and remains approximately at the maximum condition as input load is increased further.

The simulation results, Figures 2 - 10, demonstrate very acceptable performance; a high data rate network using CSMA/RN access protocol operates effectively over a wide range of LAN and MAN conditions.

In developing the simulation, the question arose as to whether it was necessary to model the ring at the bit level, i.e., to be able to account for the condition of every bit in the network, or whether larger blocks, at least for modelling performance studies, could be considered inseparable. This lead to the postulation that one can scale the simulation results by treating each bit as a block in an "Up-sized" ring. Thus, a bit in a 10 km, 10 node ring with 2 Kbits messages would scale to represent 10 bits in 100 km, 10node ring with 20 Kbit messages. Scaling is equivalent to the network parameter, a , the round trip propagation time measured in message units. The question is whether the statistical performance of the ring would be affected by the separation of the block into bits, where in the scaled model, a block would be inseparable. It would seem unlikely that block size effects this small would have any appreciable influence on nominal ring operations.

To verify the scaling capability of CSMA/RN, a series of 4 runs were made. The 10 node, 10 km, 2 Kbit rings was compared to a 100 km, 20 Kbit; a 1000 km, 200 Kbit; and a 10000 km, 2 Mbit rings. Three performance factors were considered, the normalized service time, the message fracture

ratio and a histogram of empty packet lengths available to nodes when they place a message on the ring. The histogram counts empty block in 11 intervals related to message size. Of these the histogram is considered the best measure of whether bits can be used to represent blocks. If nodes see empty blocks in the ratio equivalent to the scaled bit size then the scaling factor is a very acceptable means to extrapolate CSMA/RN performance.

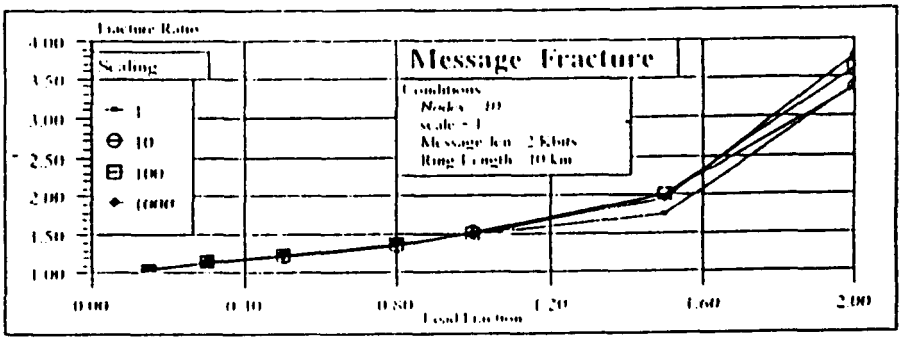


Figure 11a: Message Fracture for Scaled Runs

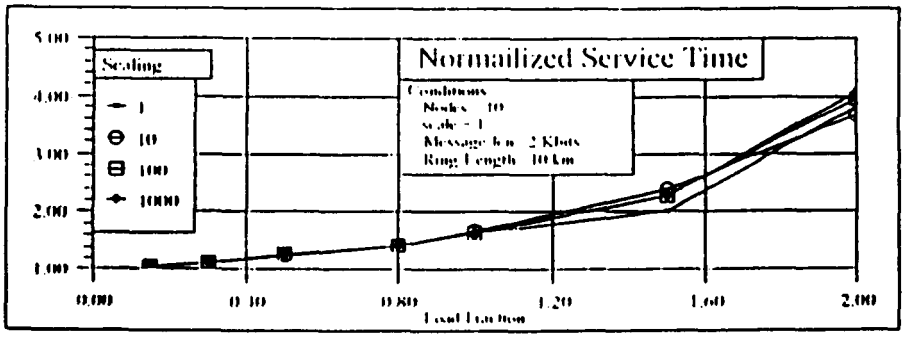


Figure 11b: Normalized Service Time for Scaled Runs

Figure 11a presents message fracture ratios for the 4 runs, above, for load fractions from 15% - 200%; Figure 11b presents the normalized service time based upon message length. Both figures illustrate that scaled conditions are nearly identical. Table 1 presents data based upon the histograms for empty packets. The histogram groups empty packets into the 11 intervals related to message size; i.e., the empty blocks are separated into equal size groups: the first group is packets which are <10% of message size; the second group, packets between 10% - 20% of message size, etc.; the last group, packets >100% of message size. The results are presented as the percentage of packets in a group to the total number of empty packets arriving at the nodes.

The four sets of data in Table 1 are the four run conditions stated above. If we compare identical entries for the four conditions, we see very little difference at the higher load fractions and no difference at lower loads. Thus, for each condition the ring scales nearly identical, i.e., a bit in the lowest size ring is equivalent to a block of 1000 bits in the largest ring. As a result we are very confident CSMA/RM results can be effectively scaled from the bit to the block level over a wide range of conditions.

Note that the scaling studies for CSMA/RN modelled a wide area network (WAN) with a gigabit rate and megabit message transfers. Performance of this network is excellent; it handles up to 2 gigabits/sec of data. Access and service times are excellent and since travel time is based upon the speed of light in the medium, the response time is strictly a function of separation distance between communicating nodes. In fact, it appears that CSMA/RN operational features provide a asynchronous data WAN which is as good as can be expected and should be a suitable access protocol for a National Research and Education Network [21].

7 Conclusions

CSMA/RN operates under conditions where a ring can contain a number of messages simultaneously. It is based upon TTattempt and truncate" for a node transmitting if an incoming carrier is detected on the ring. In this respect it differs from other access protocols which defer or abort transmission when a collision can occur. The results demonstrate that CSMA/RN is a viable protocol for a wide range of ring parameters and conditions. Service and wait times are excellent for a large range of load conditions and a simple analytical model is available to estimate operations. Message fracture does

not appear to be a serious a problem for rings which can contain an fairly large number of messages, i.e., where $a >> 1$. Throughput remains high under overload conditions A scaling parameter exists based upon a which allows the estimation of ring performance for WANs. Here, CSMA/RN performance is excellent access and suitable for a future national network system.

To date, CSMA/RN studies have been limited to simple asynchronous data operational conditions. Additional study is required to document its performance for messages with variable lengths, for non-uniform load conditions, for conditions where ring domination by a few nodes can occur, and for large node count conditions where message fracture is most likely. Protocol procedures must be developed and studies must be done for CSMA/RN to effectively handle integrated traffic, i.e., synchronous traffic consisting of voice and video data in conjunction with asynchronous messages.

References

[1] Skov, M.: TTImplementation of Physical and Media Access Protocols for High Speed Networks," IEEE Comm. Magazine; June 1989; pp 45-53.

[2] Henry, P. S.: TTHigh Capacity Lightwave Local Area Networks," IEEE Comm. Magazine; Oct 89; pp 20-26.

[3] Wagner, S. S.; Kobrinski, H.: TTWDM Applications in Broadband Telecommunications Networks," IEEE Comm. Magazine; March 89; pp 22-30.

[4] Karol, M. J.: TTOptical Interconnection Using Shuffle Net Multihop Networks in Multi-Connect Ring Topologies," ACM 0-89791-279-9/88/008/0025.

[5] Dykeman, D.; Bux, W: TTAnalysis and Tuning of the FDDI Media Access Control Protocol," Jour. on Selected Areas in Communication ; Vol 6, No 6; July 1988; pp 997-1010.

[6] Tobaji, F.A.; Fine, M.: TTPerformance of Unidirectional Broadcast Local Area Networks: Expressnet and Fastnet," IEEE Jour. on Selected Areas in Communication; Vol SAC-1; No 5; Nov 1983; pp 913-925.

[7] Newman, R.M.; Budrikis, Z.L.; Hullett; J.L.: TTThe QPSX Man," IEEE Communications Magazine; Vol 26, No 4; April 1988; pp 20-28.

[8] IEEE Computer Society; Draft of Proposed IEEE Standard 802.6 Distributed Queue Dual Bus Metropolitan Area Network (MAN); Draft D.O.; June 1988

[9] Maly, K; Zhang, L.: Game, D.: TTFairness Problems in High-Speed Networks," Old Dominion University, Computer Science Dept. TR- 90-15; Mar. 1990.

[10] Zafirovic-Vukotic, M; Niemegeers, I.G.; Valk, D.S.: TTPerformance Analysis of Slotted Ring Protocols in HSLAN's," Jour. on Selected Areas in Communications; Vol 6; No 6; July 1988; pp 1011-1023.

[11] Suda T., et. al.; TTTree LANs with Collision Avoidance: Protocol, Switch Architecture and Simulated Performance"; ACM 0-89791-279-9/88/008/0155

[12] Liu, M.T.: TTDistributed Loop Computer Networks," in Advances in Computers Vol 17; Yovits, M.C.(editor); Academic Press; NY; 1978; pp 163-221.

[13] Hilal, W.; Liu, M.T.: TTAnalysis and Simulation of the Register-Insertion Protocol," Proc. of Computer Networking Symposium; Dec. 10, 1982; pp 91-100.

[14] Bux, W.: TTLocal Area Subnetworks: A Performance Comparison," IEEE Transactions on Communications; Vol. Com-29; No. 10; Oct. 1981; pp. 1465-1473.

[15] Bhargava, A; Kurose, J.F.; Towsley,D: TTA Hybrid Media Access Protocol for High-Speed Ring Networks," IEEE Jour. on Selected Areas in Communications; Vol. 6; No.6; July 1988; pp 924-933.

[16] Chlamtac, I; Ganz,A.: TTA Multibus Train Communication (AMTRAC) Architecture for High-Speed Fiber Optic Networks," IEEE Jour. on Selected Areas in Communications; Vol. 6; No.6; July 1988; pp 903-912.

[17] Casey, L: TTChannel Allocation in FDDI II," Presented to FDDI II Ad Hoc Working Party, Denver; April 1986.

[18] Liu, M.T.; Hilal, W.; Groomes, B.H.: TTPerformance Evaluation of Channel Access Protocols for Local Computer Networks," Proc. Computer Networks ; Compcon '82; Sept. 20-23,1983; pp 417-426.

[19] Rubin, I.: TTAn Approximate Time-Delay Analysis for Packet-Switching Communications Networks," IEEE Trans. on Communications; Vol. Com-24; No 2; Feb. 1976; pp 210-221.

[20] Jaiswal, N.K.: *Priority Queues*; Academic Press; NY; 1968.

[21] Wintsch, S.: TTToward a National Research and Education Network," MOSAIC; Vol 20; No. 4; Winter 1989; pp 32-42.