UNIVERSITY OF ALBERTA

MAC Level and Routing Protocols for High-Speed Networks

BY ©

Cesur Baransel

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

DEPARTMENT OF COMPUTING SCIENCE

Edmonton, Alberta
Spring 1994

National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

ISBN 0-612-11152-0

November 30, 1993

Faculty of Graduate Studies
University of Alberta

Dear Sir/Madam,

As the co author of the following papers,

1. C. Baransel, W. Dobosiewicz, P. Gburzynski, "**CBRMA++ : On How to Increase the Performance of a MAC Protocol Without a Second Headend Station**", Silicon Valley Networking Conference, April 1992, California, USA.
2. C. Baransel, W. Dobosiewicz, P. Gburzynski, "**CBRMA++/SR: On the Design of a MAN/WAN MAC Protocol for High-Speed Networks**", accepted for publication in the special issue (Networks in the Metropolitan Area) of *IEEE Journal on Selected Areas in Communications*.
3. C. Baransel, W. Dobosiewicz, P. Gburzynski, "**Routing in Multihop Packet Switching Networks: Gbps Challenge**", submitted for publication to *IEEE Network Magazine*.

I hereby give permission to Mr. Cesur Baransel to include their content into his Ph.D. thesis.

Sincerely,

Assoc. Prof. Pawel Gburzynksi
Department of Computing Science
University of Alberta

November 30, 1993

Faculty of Graduate Studies
University of Alberta

Dear Sir/Madam,

As the co-author of the following papers,

1. C. Baransel, W. Dobosiewicz, "**CBRMA (Cyclic Balanced Reservation Multiple Access) MAC Protocol**", The Sixth International Symposium on Computer and Information Sciences–ISCIS VI, October 1991, Antalya, Turkey.
2. C. Baransel, W. Dobosiewicz, P. Gburzynski, "**CBRMA++ : On How to Increase the Performance of a MAC Protocol Without a Second Headend Station**", Silicon Valley Networking Conference, April 1992, California, USA.
3. C. Baransel, W. Dobosiewicz, P. Gburzynski, "**CBRMA++/SR: On the Design of a MAN/WAN MAC Protocol for High-Speed Networks**", accepted for publication in the special issue (Networks in the Metropolitan Area) of *IEEE Journal on Selected Areas in Communications*.
4. C. Baransel, W. Dobosiewicz, P. Gburzynski, "**Routing in Multihop Packet Switching Networks: Gbps Challenge**", submitted for publication to *IEEE Network Magazine*.

I hereby give permission to Mr. Cesur Baransel to include their content into his Ph.D. thesis.

Sincerely,

Assoc. Prof. Wlodek Dobosiewicz
Department of Computing Science
University of Alberta

UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR: Cesur Baransel

TITLE OF THESIS: MAC Level and Routing Protocols for High-Speed Networks

DEGREE: Doctor of Philosophy

YEAR THIS DEGREE GRANTED: 1994

(Signed) . . . .

Cesur Baransel
Kolejtepe Mahallesi,
Mühendis Evleri Sokak, 5/3
Gaziantep, Türkiye

Date: Dec.17. 1993. .

UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **MAC Level and Routing Protocols for High Speed Networks** submitted by Cesur Baransel in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Wlodek DOBOSIEWICZ (Co Supervisor)

Pawel GBURZYNSKI (Co Supervisor)

Ian F. AKYILDIZ (External)

Janelle J. HARMS (Examiner)

Wayne D. GROVER (Examiner)

Piotr RUDNICKI (Examiner)

Jim HOOVER (Chair)

Date: Nov 26, 1993

Abstract

This thesis discusses the design of routing and MAC-level protocols for high-speed (Gbps range) MANs and WANs. The effects of increased transmission rates are investigated and consequently, a set of requirements are stated regarding the design of MAC-level and routing protocols for high-speed networks. Two new MAC level protocols, namely *Cyclic Balanced Reservation Multiple Access ++ with Slot Reuse* (CBRMA++/SR) and *Slot Pre/Reuse* (SP/R), and one new routing protocol, *Compass Routing* are proposed. The suitability of these protocols for a high-speed networking environment is discussed and some simulation results are presented.

Contents

[1] Versions of this chapter have been accepted for publication/published in [BaD91, BDG92, BDGa, BDGb].

[2] A version of this chapter has been submitted for publication to *IEEE Network Magazine*.

List of Figures

Chapter 1

Introduction

Modern computer network architectures are designed in layers. Besides the merits of functional modularity, a layered architecture with clearly defined functionalities and non-overlapping boundaries is aimed at providing the necessary infrastructure for international communication standards to emerge. The most notable effort to that end is *The Reference Model for Open System Interconnection (*OSI*)* which was approved by ISO and CCITT in 1983 [ISO83]. This model provides a seven layer skeletal architecture which we will also use as a point of reference. According to this model, each layer is conceived as a black box or a module and the modules at a given layer are called *peers*. The communication is carried out over a *subnet* which conceptionally acts as a giant switch that provides a reliable and error-free transmission link between the involved parties. Since the subnet is a shared resource, its fair and effective management is essential. In the ISO model, this task is assigned to the ***network layer*** and is carried out via ***routing protocols*** and ***flow control mechanisms*** for point-to-point networks. To serve the same purpose, *Medium Access Control (*MAC*)* protocols are developed for multiaccess communication. Due to the recent developments in the transmission technology, most notably with the advent of the optical fiber as a transmission medium, new challenges presented themselves in the subnet design. New MAC-level protocols [IEE90, DEC90], switch architectures and routing protocols [OSM90] have been developed to cope with these challenges.

This document consists of two parts that investigate some of the issues involved in the design of MAC–level protocols and routing mechanisms for fiber optic networks. The first part is devoted to the MAC–level protocols in which two new protocols, CBRMA++/SR and SP/R, are presented. The second part addresses the routing protocol design. A discussion of the general design issues is followed by the introduction of some contemporary designs, eventually leading to the proposal of a new mechanism.

All simulations in this thesis have been carried out using SMURPH[1]. SMURPH is an object oriented programming environment based on C++ for specifying communication protocols and modeling communication networks. By a communication network we understand a configuration of ***stations*** interconnected via *channels*, running a collection of concurrent communicating processes. The distributed algorithm realized by those processes is called the communication protocol.

Unlike other protocol specification systems, e.g., ESTELLE [BuD87, EST87], LOTOS [BoB87, LOB88], or PROMELA [Hol91], SMURPH has a built-in notion of time. Thus, it can naturally and easily express protocol operations and physical phenomena occurring at the medium access control (MAC) level. Besides the specification language, the system provides a virtual environment for executing protocols. Thus, SMURPH specifications are directly executable. This virtual environment is based on an event-driven, discrete-time simulator which is hidden from the user. The user has an impression of running the protocol in a realistic environment which carefully reflects all relevant physical phenomena occurring in a real network, e.g., limited accuracy of independent

[1] SMURPH is an acronym for a *System for Modeling Unslotted Real-time PHenomena.*

clocks, race conditions, faulty channels. Its most typical application is to investigate the performance of a new MAC level protocol, comparing it to a number of other protocols in the same class. However, compared to other network simulators and evaluators (e.g. COMNET [MiS93], GILDA [PNC93], NETSIM [JuL93]), SMURPH has a number of distinct features which can be stressed in the following points:

- Protocols in SMURPH are fully specified. In principle, a SMURPH specification can be "compiled into silicon," i.e., made completely functional in a mechanical way.
- In SMURPH, networks and protocols are *emulated* rather than simulated. Thus it makes sense to use SMURPH for protocol verification, e.g., conformance testing.
- SMURPH doesn't purport to be a "no-programming" system and we do not perceive it as a disadvantage. Intentionally, SMURPH is a protocol prototyping environment and with the current state of the art in program synthesis it is extremely difficult to produce novel protocols by moving icons on the screen. On the other hand, owing to the object-oriented nature of SMURPH specifications, it is natural and easy to build libraries of protocols and their components.

SMURPH and its predecessor LANSF [Gbr91] have been used to investigate the performance and correctness of a number of protocols for local and metropolitan area networks (e.g., [Ber89, GR89a, GR89b]). The simulation methodology of SMURPH is extensively defined in [GbM89, DoG93, Gbu94]. The package (together with detailed documentation) is freely available to the research community via ***anonymous ftp*** from `menaik.cs.ualberta.ca` (129.128.4.241). The present version of SMURPH runs under UNIX[2] on a variety of equipment. There also exists a version of SMURPH for Apple Macintosh.[3]

[2] UNIX is a trademark of AT&T Bell Labs.

[3] Macintosh is a trademark of Apple Computer, Inc.

Bibliography

[Ber89] B.R. Bertan, "Simulation of MAC Layer Queuing and Priority Strategies of CEBus", *IEEE Transactions on Consumer Electronics*, 35 (August 1989), 557 563.

[BoB87] T. Bolognesi, E. Brinksma, "Introduction to the ISO Specification Language LOTOS", *Computer Networks and ISDN Systems*, 14, 1987, 25 59.

[BuD87] S. Budkowski, P. Dembinski, "An Introduction to ESTELLE a Specification Language for Distributed Systems", *Computer Networks and ISDN Systems*, 14, 1987, 3 23.

[DEC90] *Fiber Optic Data Interface System Level Description*, Digital Equipment Corporation, June 1990.

[DoG93] W. Dobosiewicz, P. Gburzynski, "A C++ Environment for Modeling Communication Systems", Proc. of the 26th Annual Simulation Symposium, SCS + IEEE + ACM/SIGSIM, 1993, 196–205.

[EST87] *User Guide for the NBS Prototype Compiler for Estelle*, U.S. Dept. of Commerce, National Bureau of Standards, Report No. ICST/SNA - 87/3, October 1987.

[GbM89] P. Gburzynski, J. Maitan, "An Object-oriented Configurable Simulator for Low-level Communication protocols", TR90-8, University of Alberta, 1989.

[GbR91] P. Gburzyński, P. Rudnicki, "LANSF: A Protocol Modelling Environment and its Implementation", *Software Practice and Experience*, 21, 1 (January), 1991, 51 76.

[Gbu94] P. Gburzynski, *Medium Access Protocols for Local and Metropolitan Area Networks: Specification and Simulation*, Prentice Hall, 1994?.

[GR89a] P. Gburzyński, P. Rudnicki, "A Note on the Performance of ENET II", *IEEE Journal on Selected Areas in Communications*, 7 (April 89), 424 427.

[GR89b] P. Gburzyński, P. Rudnicki, "On Executable Specifications, Validation, and Testing of MAC-level Protocols", Proc. of the 9th IFIP WG 6.1 Int. Symposium on Protocol Specification, Testing, and Verification, June 1989, Enschede, The Netherlands, 1990, 261–273.

[Hol91] G.J. Holzmann, *Design and Validation of Computer Protocols*, Prentice Hall, 1991.

[IEE90] IEEE, "Std 802.6—1990, IEEE Standards for Local and Metropolitan Area Networks: Distributed Queue Dual Bus(DQDB) of a Metropolitan Area Network (MAN)", July 1991.

[ISO83] *ISO International Standard 7498, Information Processing Systems — Open System Interconnection — Basic Reference Model*, Geneva, October 1983.

[JuL93] J.R. Jump, S. Lakshmanamurthy, "NETSIM: A General-Purpose Interconnection Network Simulator", Proc. of MASCOTS'93, San Diego, California, 1993, 121–125.

[LOB88] L. Logrippo, A. Obaid, J.P. Briand, M.C. Fehri, "An Interpreter for LOTOS, A Specification Language for Distributed Systems", *Software Practice and Experience*, 18, 4 (April 1988), 365–385.

[MiS93] R. Mills, S. Skinner, "COMNET III: The New Enterprise-Wide Performance Analysis Tool", Proc. of MASCOTS'93, San Diego, California, 1993, 349–350.

[OSM90] Y. Oie, T. Suda, M. Murata, D. Kolson, H. Miyahara, "Survey of Switching Techniques in High Speed Networks and Their Performance", INFOCOM'90, 1242–1251.

[PNC93] C.C. Palmer, M. Naghshineh, J.S.C. Chen, "The GILDA LAN Design Tool", Proc. of MASCOTS'93, San Diego, California, 1993, 353–354.

Part I

MAC–Level Protocols for High–Speed MANs

Chapter 2

CBRMA++/SR[1]

2.1 Introduction

The design of the MAC-level protocol is the most crucial phase in any network design since the decisions made at this level will determine the major functional characteristics of the network and set the upper limits of its capabilities in general. Since the activities of the upper layers will be more and more carried out by software as one goes upward in the network protocol hierarchy, any attempt to provide a functionality that counteracts its limitations will be more and more costly, eventually bordering on infeasibiltiy. Especially with the advent of optical fiber as a transmission medium, more strict constraints on message/packet/slot processing times are imposed at the intermediate nodes thereby providing a new challange to MAC-level protocol designers.

In this thesis, we consider optical fiber as the basic transmission medium and 1 Gbps or more as the basic transmission rate. Consequently, a MAC-level protocol designed to operate in this environment must satisfy certain requirements:

1. It must be simple enough to be implemented directly in hardware, so that it can exploit the bandwidth capacity offered by optical fiber.

2. It must be fair, i.e. the throughput of a station must be independent of its location within the network. Furthermore, a bursty station should not be served in detriment of the others and the protocol should not allow a station to usurp the available bandwidth capacity inadvertently. This does not mean that the bursty stations should be assigned to a lower priority since every station is entitled to use a reasonable portion of the available time-bandwidth product. However, a bursty station should not be granted with additional bandwidth which is stolen from the moderately or lightly loaded users. Also, in the presence of multiple bursty stations, available idle bandwidth should be distributed evenly among them.

3. The protocol should provide concurrent access to the transmission medium. This item is related to the well-known **a-parameter** [Sta84]. Any MAC protocol which is sensitive to this parameter limits its usefulness to networks up to a certain size and rate.

4. The protocol must be flexible enough to satisfy heterogenous traffic demands, such as high-priority, synchronous and asynchronous traffic.

5. It must be predictable since unpredictability requires more resources to be allocated at the stations by the users to be able to cope with the unexpected and increases the complexity of the protocol.

[1] Versions of this chapter have been accepted for publication/published in [BaD91, BDG92, BDGa, BDGb].

6. The overall structure should be robust. In particular, the malfunction of a switching element or a severed link should not take the network down with it. In other words, the effect of the malfunctions should be kept as local as possible and/or the network should be able to "heal" itself so that the integrity of the network is preserved.

7. In the presence of rich connectivity, the protocol should be able to make effective use of alternative routes. Therefore, the means of providing flow control and load distribution at least the formation of an infrastructure to facilitate the provision of these services by upper layer protocols — should be considered.

2.2 Alternative Approaches

By definition, MAC protocols are required when a single carrier is shared by multiple nodes. In other words, a MAC protocol is essentially an arbitration mechanism that is responsible for the resolution of the contentions by the nodes for a shared transmission medium. Designing a MAC protocol requires information regarding the topology of the network, the type of the carrier and the signalling mechanism. Some important implications of these factors can be stated with respect to MAC protocol design: different topologies are suitable for different networks, in terms of geographical size and the number of nodes involved. They also have different degrees of connectivity and some are designed with special routing schemes in mind. The carrier type is also crucial since different media have different transmission and attenuation rates. Furthermore, transmission on optical fiber is inherently unidirectional which is not the case for copper links. As for the signalling mechanism, the major issue is the distinction between baseband transmission and broadband transmission. In case of the latter, the MAC protocol should be able to handle multiple channels.

The type of applications that the network is supposed to support is also important. If a certain traffic pattern and load can be assumed, the MAC protocol can be tailored accordingly to provide better performance.

The previous remarks are included to clearly define the environment for which our protocol is designed. Furthermore, the obvious approach to demonstrate the "betterness" of a new protocol is to compare it with the ones in existence. In [DoG91], it is argued that such a comparison is only meaningful if a particular topology is chosen[2]. The bottom line of the argument presented in that paper is that <*topology,protocol*> pairs are the meaningful members of the universal set of the MAC protocols, not the protocols alone. To do otherwise is just another attempt to compare the proverbial apples with oranges. By the same token, we will compare our protocol with the recently established IEEE standard DQDB protocol for the following reasons: both protocols are designed for high–speed MANs/WANs; the set of the possible applications has a wide range and is not restricted to data transmission; the transmission discipline is that of the slotted bus and the tasks performed by the so-called headend stations are similar.

Before going into the details of our protocol, we will first evaluate DQDB according to the aforementioned criteria. However, it must be noted that requirements 6 and 7 are excluded from these discussions. The reason for this is quite obvious since on a bus topology the issues related to multiple paths between station pairs are meaningless and a severed link effectively divides the network into two, shattering its integrity. Furthermore, it is not our intention in this thesis to address the problems regarding the handling of two or more smaller networks after such a mishap.

2.3 DQDB

The *Distributed-Queue-Dual-Bus* (DQDB) medium access control (MAC) level protocol has been accepted as the IEEE 802.6 standard for high-speed metropolitan area networks (MANs). DQDB is

[2] Topology in this context implies networks with identical physical layer characteristics as well as connectivity.

extensively defined in [IEE90], so no attempt will be made to describe it here. Although this protocol has properties which make it appealing for many high-speed network applications[3], it satisfies condition 3 and (arguably) 1, but fails to satisfy the other conditions:

1. DQDB violates condition 2, at least to some extent[4]. The BWB (Bandwidth Balancing) mechanism, which has been introduced as a countermeasure, is proven to be effective but only after a rather long transition time [As90a, As90b], a fact that renders it almost useless in case of short-lived bursts. DQDB also permits a single heavy-user to dominate the overall throughput, consequently increasing the access delays of the low-traffic users significantly. [As90a, As90b, AWZ90].

2. DQDB does not satisfy condition 4. DQDB makes ineffective priority assignments; it is observed that the priority mechanism is not effective under heavy load when the nodes generating low-priority traffic lie between the headend and a high-priority node [AWZ90]. Furthermore, the protocol may even exhibit the so-called inverse priority behavior [As90a, AWZ90].

3. DQDB violates condition 5. Its behavior is unpredictable in the sense that the final capacity distribution depends on the network conditions when the overload occurs. In other words, starting from different initial conditions, one can arrive at different states [PhB92].

Numerous variations of the DQDB protocol have been proposed, some of them satisfying conditions 2 and 4 (e.g. [LAT92]). We will not discuss these variations here restricting ourselves to the standard DQDB.

2.4 CBRMA

Our aim is to design a scheme which can satisfy the performance requirements stated in section 1. In the rest of the chapter, we will argue that *Cyclic-Balanced Reservation Multiple Access ++ with Slot Reuse* (CBRMA++/SR) MAC protocol has the necessary qualities to be so. The design of the protocol is a three-step process and each step has been described in [BaD91, BDG92, BDGa]. As the first step towards the aforementioned goal, we introduce CBRMA [BaD91], which we claim satisfies the basic requirements and then improve it twice [BDG92, BDGa]. The first improvement involves getting the throughput of dual bus topology on a folded bus, therefore, the elimination of the second headend station in the configuration. In [BDG92], we explained how this can be done and named the second variation of our protocol CBRMA++. Later we augmented CBRMA++ with an efficient slot reuse mechanism and in [BDGa] introduced CBRMA++/SR. The slot management of CBRMA++/SR is unique since beside slot reuse, it also provides the upper layer protocols with helpful information about future incoming traffic profiles. In the remainder of this section, we will first state the important differences between our protocol and CRMA — a protocol previously introduced by IBM — to prevent possible confusion. Later, a basic description of the protocol along with the code of the basic algorithms, an example cycle and some simulation results will be provided. The next section is devoted to the discussion related to the elimination of the second headend station and to the details of the slot reuse mechanism.

2.4.1 Comparison with CRMA

Although the name of the protocol is somewhat akin to the *Cyclic Reservation Multiple Access* (CRMA) protocol developed at IBM, only the idea of cyclic reservation is similar and even it is

[3] Such as being able to sustain an aggregate transmission rate near the total bus capacity independently of the network's size and transmission rate.

[4] When oversaturated DQDB is patently unfair. At more realistic traffic loads, the significance of the variation in the throughput rates of the transport layer protocol due to the location of the station becomes arguable.

handled in a different way. CRMA requires more complex structures both at the nodes and at the headend station. The reservations made by the nodes are not certain and need to be confirmed by the headend to be valid. They can be rejected and in such a case a retry is necessary by the nodes[5]. Consequently, nodes have to maintain three different message queues, namely *Confirmed Reservation Queue* (CRQ), *Tentative Reservation Queue* (TRQ) and *Entry/Reentry Queue* (ERQ). The headend station also maintains two queues called *Global Reservation Queue* (GRQ) and *Elasticity Buffer* (EB), which is also a queue despite its name. Multiple reservation slots co-exist on the bus at a given instant and the number of control slots is six (*Reserve, Confirm, Start, Reject, unused and Noop*). Since a high generation rate of reservation slots can result in considerable access delays, a backpressure mechanism is developed in order to reduce the worst-case access delay. The details of this protocol can be found in [MNW90, Na90a, Na90b, TrD90].

Our protocol differs from CRMA in a number of ways:

1. There can be only one reservation slot on the bus at a given instant and it is read/updated by a node twice, once on the forward bus and once on the backward bus.

2. There are no confirm and reject slots, therefore no backpressure mechanism either.

3. Message queues at the nodes are simpler and the headend station acts only as a slotter and maintains no queues.

4. Reservations are final and are not subject to the further approval of the headend, removing the decision-making responsibility from headend and distributing it amongst the nodes.

5. Every station can get assertive information (not just a close approximation) about the global traffic profile by simply inspecting the contents of the reservation slot. The information at the disposal of the MAC and upper layer protocols cover the traffic load and available bandwidth for the next cycle with utmost certainty.

6. The coherency of the basic mechanism makes a powerful slot reuse mechanism possible a mechanism that can not be applied to CRMA directly because of the uncertainties involved in the reservation process.

2.4.2 Basic Algorithms and Data Structures Used by CBRMA

CBRMA is a slot reservation scheme suitable for both the folded bus and the dual bus topologies (figures 2.1,2.2). The protocol will be explained for the folded bus configuration. Its counterpart for the dual bus requires only the duplication of the data structures and the protocol processes along with some minor additions both at the headends and the nodes. That issue will be addressed at the end of the subsection.

In CBRMA, the bandwidth allocation process is organized in cycles. At the beginning of every cycle, each station is granted the same number of slots. This default number is called **fair share**. A **reservation cycle** begins at the headend with the headend station issuing a *reservation slot* which passes by each station twice, once on the forward bus and once on the backward bus (figure 2.2).

Since the traffic loads of stations may vary, a station can either be content with its fair share (or a part of it) or ask for more. If a station uses only a portion of its fair share, the remaining slots are recorded as ***unused.*** These are gathered together and distributed evenly among users with heavy slot demands. The distribution of unused slots is done while the reservation slot is passing by the stations on the backward bus (**balancing cycle**).

A reservation slot has five subfields:

1. Cycle number (C).

[5] A reservation slot contains the reservations made by a number of nodes and when a reject slot is issued by the headend station the outcome is the invalidation of the reservation slots that are present on the bus altogether.

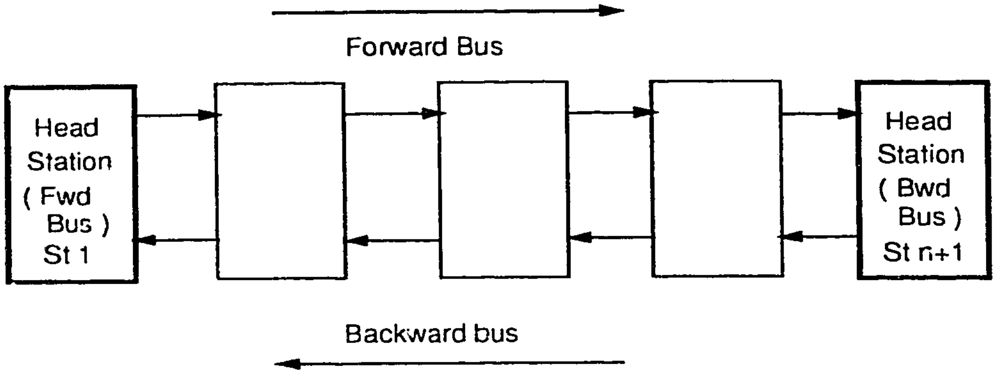

Forward Bus
Head Station (Fwd Bus) St 1
Head Station (Bwd Bus) St n+1
Backward bus

Figure 2.1: Dual Bus Topology.

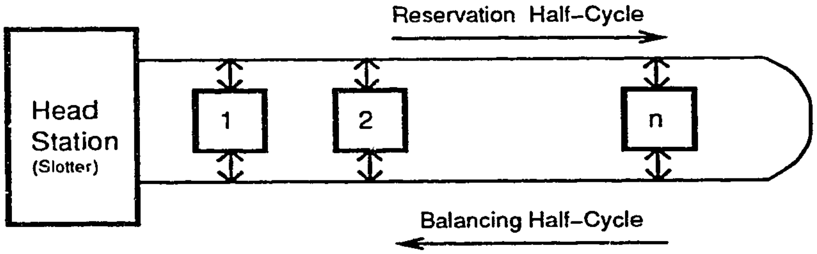


Figure 2.2: Half-Cycles of CBRMA (Folded Bus Topology).

2. Number of slots reserved so far (R).

3. Number of stations (so far) that need more slots than their fair share (ES).

4. Total number of extra slots requested by ES stations so far (ER).

5. Total number of slots left unused by lightly-loaded stations so far (U).

At each node, the necessary data structure is composed of a single message queue and four counter variables named *CycleStartCounter, XMitCounter, QueueSize and ExtraRequestCounter.* CycleStartCounter is used as a countdown counter and indicates the number of slots that the node is supposed to let go by before transmitting. XMitCounter contains the number of slots that the node will use in the next cycle. QueueSize is the number of packets backlogged at the node. ExtraRequestCounter is set to the number of slots that the node can use if given the chance.

2.4.3 Reservation Algorithm

Upon detecting the reservation slot on the forward bus, a node executes the following algorithm. In the algorithm, the fields that belong to the reservation slot are marked with the prefix "*Slot*→" to distinguish them from the local variables. As can be seen, there are three *if-statements* that correspond to three possible cases:

1. If the node needs more than its fair share, it claims its fair share and then requests more to transmit the rest of the queue by incrementing the *ER* field of the slot.

2. If its current need is exactly equal to the fairshare, it simply claims it and sets the local counters to appropriate values.

3. If the node needs less than its fair share, it marks the remaining slots as unused by incrementing the *U* field of the slot accordingly.

- Reservation Algorithm.

```
ExtraRequestCounter = 0 ;
CycleStartCounter = Slot->R + 1 ;

if (QueueSize > FairShare) {
    XMitCounter = FairShare ;
    ExtraRequestCounter = QueueSize - FairShare ;
    Slot->ES ++ ;
    Slot->ER += ExtraRequestCounter ;
    Slot->R  += FairShare ;
    exit() ;
}

if (QueueSize == FairShare) {
    XMitCounter = FairShare ;
    Slot->R  += FairShare ;
    exit() ;
}

if (QueueSize < FairShare) {
    XMitCounter = QueueSize ;
    Slot->R += QueueSize ;
    Slot->U += (FairShare - XMitCounter) ;
    exit() ;
}
```

2.4.4 Balancing Algorithm

As the reservation slot turns the fold and starts to propagate on the reverse bus, the balancing cycle begins. The reservation slot is handled by the nodes according to the following algorithm.

- Balancing Algorithm.

```
if (Slot->U == 0) exit() ;

if (ExtraRequestCounter == 0) {
    CycleStartCounter += Slot->U ;
    QueueSize -= XMitCounter ;
    exit() ;
}

if (Slot->U >= Slot->ER) {
    Slot->U -= ExtraRequestCounter ;
    Slot->R += ExtraRequestCounter ;
    Slot->ES -- ;
    Slot->ER -= ExtraRequestCounter ;
    CycleStartCounter += Slot->U ;
    XMitCounter  += ExtraRequestCounter ;
    QueueSize -= XMitCounter ;
    exit() ;
}

else {
    int othersneed, difference, fair, share ;

    othersneed = Slot->ER - ExtraRequestCounter ;
    fair = floor (Slot->U / Slot->ES) ;
    difference = Slot->U - othersneed ;

    if (difference > fair)
         share = difference ;
    else share = fair ;

    if (share > ExtraRequestCounter)
        share = ExtraRequestCounter ;

    Slot->U -= share ;
    Slot->R += share ;

    XMitCounter += share ;
    CycleStartCounter += Slot->U ;

    Slot->ES -- ;
    Slot->ER -= ExtraRequestCounter ;
    QueueSize -= XMitCounter ;
    exit() ;
}
```

As can be seen, in the absence of any unused slot, there is nothing to do, since the balancing algorithm deals with the fair distribution of the unused slots amongst users operating under heavy traffic load. Otherwise, the nodes that are not requesting extra slots simply adjust their counters to ensure the slot contiguity, by executing the body of the second if statement. The third or fourth block of statements is executed by nodes that placed a request for extra slots in the reservation phase. The third block deals with the case where the number of unused slots is greater than or equal to the requested amount. In this case there is no problem and all requests can be fully met. Otherwise, there are calculations to be performed to ensure fairness[6].

The reservation slot for $cycle_{i+1}$ is followed by the empty slots whose allocation is negotiated in $cycle_i$. Stations begin to decrement their *CycleStartConters* by 1 for every slot that passes by on the forward bus after detecting the reservation slot on the forward bus. The station that has its *CycleStartCounter* decremented down to zero begins to transmit using the next slot and the number of slots allocated to the station indicated by the value of its *XMitCounter*.

By the basic definition of the protocol, the following points are clear: firstly, a station must make a reservation before starting to transmit. Therefore, even under light traffic conditions zero medium access delay is not possible. Secondly, since it is not possible to perform arithmetic operations on a field of reservation slot without storing it at least partially, a delay is inevitable. The amount of the

[6] In which a division operation is necessary. In the absence of a fast division circuitry, a simple look-up table that can be accessed on *(how many slots are available, how many stations are contending for them)* basis can be used to alleviate this problem.

delay is totally dependent upon the speed of the arithmetic circuitry at the disposal of the node, but it is clear that the whole traffic on the bus will be delayed by the same amount. In the simulations, we used a shorter bus length for DQDB since the delay can be regarded as increasing the distance between neighbouring stations.

As for the operation on the dual bus, two minor additions are necessary to the protocol: in order to complete the reservation cycle, a headend station should repeat the reservation slot issued by the other headend station on the outgoing bus. Therefore a mechanism is necessary at the headends to differentiate between its own reservation slot and the reservation slot of the other headend — a problem that can be solved at the expense of a single bit in the reservation slot structure. The other addition involves a decision mechanism at the nodes which is necessary to select one of the available two buses since the nodes can transmit on both of them. In simulations, nodes use the forward bus for transmitting to downstream nodes and the backward bus for transmitting to upstream nodes, consequently minimizing the distance between station pairs and maximizing the throughput.

2.4.5 An Example Cycle

Assume that a sample network consists of 5 stations. The bus length is 30 slots and the fair share is 6 slots per station. Suppose that at the beginning of a reservation cycle i, the following reservation requests are outstanding.

Station number	*The length of message queue*	*Unused (-) Extra (+)*
1	8	2
2	2	-4
3	5	-1
4	7	1
5	8	2

The negative numbers in the last column denote the number slots that are left unused by a station out of its fairshare. The positive numbers indicates extra slot requests. The content of the reservation slot for this configuration, as it passes by each station, is depicted in figure 2.3.

Another example in which the available bandwidth and the slot requests are not aligned so nicely will be given in the improvements section. This example also shows how the forward and backward buses are controlled by a single headend station.

2.4.6 Performance of CBRMA

In figure 2.4 and 2.5 some simulation results are provided. As the other simulations in this chapter, they have been obtained for a network composed of 32 stations placed in equidistant intervals on a folded or dual bus. The transmission rate is 1 Gbps and the bus length is 68250 bits[7] or 160 slots. Considering the 5 ns/m propagation rate of optical waveguides, this corresponds to a bus length of roughly 13650 m. The fair share is set to 5 slots per cycle. Each slot is 416 bits long and two consecutive slots inserted into the transmission link are separated by 2-bit interslot gaps. Slots have 32-bit headers and 384-bit segment payloads. The throughput rates are calculated in terms of segment payload as opposed to slot length. Therefore, without slot reuse, the maximum throughput

[7] As indicated earlier (see section 4.4), we used a shorter bus length for DQDB in the simulations. The bus length is shortened by 6850 bits (from 68250 to 61400) giving approximately 214-bit excess transmission time delay per station for CBRMA.

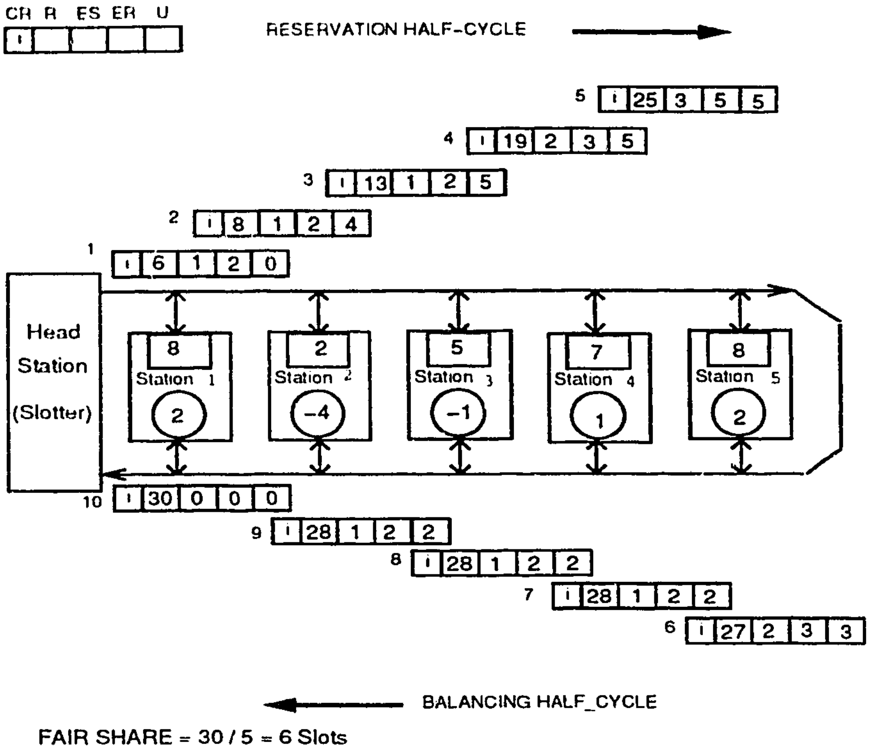


Figure 2.3: An example cycle

rate[8] per bus is calculated to be

$$r = 384/(416+2) = 0.919$$

Stations operate under asymptotic conditions[9] and the likelihood of a given source–destination pair is the same for all pairs of stations where source $\neq$ destination. Each message is exactly one slot long and therefore no bandwidth is wasted due to internal fragmentation. A simulation is terminated when 250000 slots are received globally.

Figures 2.4 and 2.5 show the throughput per station of CBRMA and DQDB on a dual bus under asymptotic conditions. In figure 2.4, BWB–MOD[10] value is 0 whereas it is set to 1 in figure 2.5. Note that the throughput rates shown in the figures are the total of the two buses. Since S_1 and S_n can only use one bus, their throughput appear to be low. In fact, the throughput of S_n is much higher than that of any other station on the reverse bus. The same argument also applies to S_1 for the throughput on the forward bus.

[8] For a given cycle, the overhead due to reservation slots is considered to be negligible since only 1 reservation slot is used for every 160 data slots.

[9] That is to say, each node is trying to seize all medium capacity on either bus.

[10] Stands for *bandwidth balancing modulus*. Every node must let one empty slot pass by for every BWB–MOD slots taken.

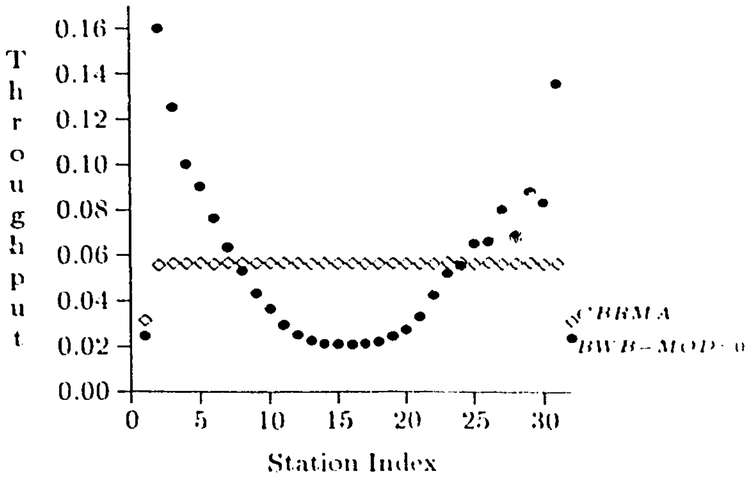


Figure 2.4: CBRMA vs. DQDB with BWB-MOD=0 on dual bus.

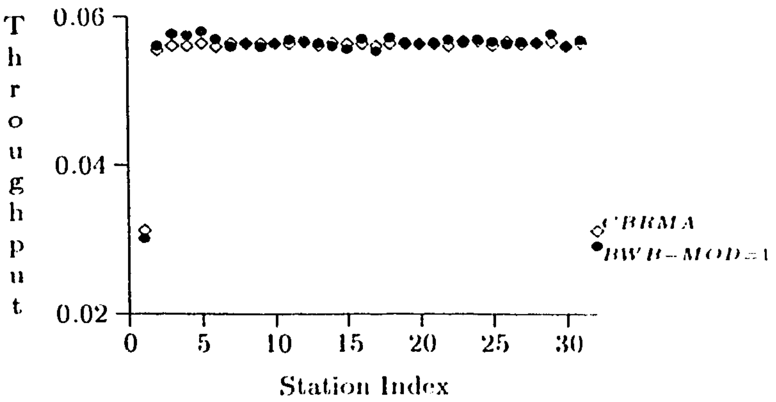


Figure 2.5: CBRMA vs. DQDB with BWB-MOD=1 on dual bus.

2.5 Improvements

2.5.1 CBRMA++: Elimination of the Second Headend Station

In a folded bus configuration, S_1 will only use the forward bus for transmission since all the other stations are located downstream with respect to its position, and consequently are reachable on the forward bus. Therefore every slot filled by S_1 can be regarded as read and emptied on the forward bus by the time it reaches the folding point and therefore, there is no reason for not reutilizing such slots after they turn the fold. If the same condition can be guaranteed for the slots that are transmitted by the other stations, it becomes possible to use a single slot twice, first on the forward bus and later on the backward bus. This is the main idea behind the CBRMA++ protocol and it can also be regarded as a simple slot reuse mechanism. In the remainder of this subsection, we will explain how this can be done and provide the related simulation results.

A very simple mechanism is sufficient to guarantee the aforementioned condition: in CBRMA++, we assume that stations maintain two different message queues; one for the messages that are to be transmitted to downstream nodes (queue-F) and one for the messages that are destined to upstream nodes (queue-B). The messages that come from queue-F will be transmitted on the forward bus while the messages that reside in queue-B will use the backward bus. Therefore it is guaranteed that the slots transmitted on the forward bus will reach their destination before the fold. This being the case, it is necessary for a station to make separate reservations for each bus. Therefore, the number of counter fields in the reservation slot must be doubled, each set associated with the respective bus. In consequence, it becomes possible for a single slot to carry two segments in a cycle. Therefore the global performance of the network will be doubled under asymptotic conditions (i.e., when each station has a continuous supply of messages going in both directions). It is also obvious that the data structures and the processes are the same as for the dual bus variant of CBRMA. One difference is the lack of the second headend station.

On the other hand, under normal conditions, we expect the number of messages in each queue to be different. To begin with, it is clear that S_1 will only use the forward bus (since there is no upstream node to it) and S_n will always use the backward bus for a similar reason. Also at S_2 queue F will probably be longer than queue B. However, this does not degrade the performance of the protocol, since under uniform load, for each station that wants to use the forward bus more frequently, there will be another station looking for extra free slots on the backward bus. The reservation scheme is capable of effectively distributing the bus capacity amongst the stations, so that, for example, S_2 will be able to gain more bandwidth on the forward bus in exchange for the slots it left unused out of its fair share on the backward bus. Figure 2.6 illustrates one additional mechanism of the protocol. Note that in this figure there is no irregularity regarding the throughput of S_1 and S_{32}. Because S_1 never uses its fair share on the backward bus and the same is true for S_{32} when its fair share on forward bus is concerned, we can transfer the fair share of S_{32} on forward bus to S_1 simply by doubling the fair share of S_1 and setting the fair share of S_{32} to zero. Carrying out the same operation for the backward bus will produce the desired effect. This approach can also be used whenever it is necessary to allocate excessive bandwidth to special stations, such as file servers.

In the remainder of this subsection we will present simulation results to show that the performance of CBRMA++ on a folded bus is equal to the performance of CBRMA and DQDB on a dual bus. Figure 2.6 depicts the throughput of CBRMA++ vs. CBRMA and DQDB on a folded bus under asymptotic uniform traffic conditions. As can be seen from the figure, the thioughput rate of CBRMA++ is approximately twice as high when compared to that of CBRMA and DQDB. Actually it is equal to the throughput of these two schemes on dual bus. Figure 2.7 depicts the performance of CBRMA++ on folded bus vs. the performance of CBRMA and DQDB on dual bus.

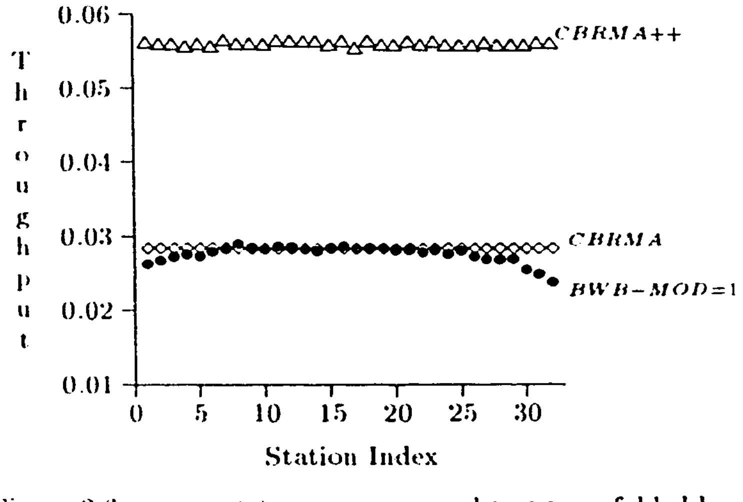


Figure 2.6: CBRMA++ vs. CBRMA and DQDB on folded bus.

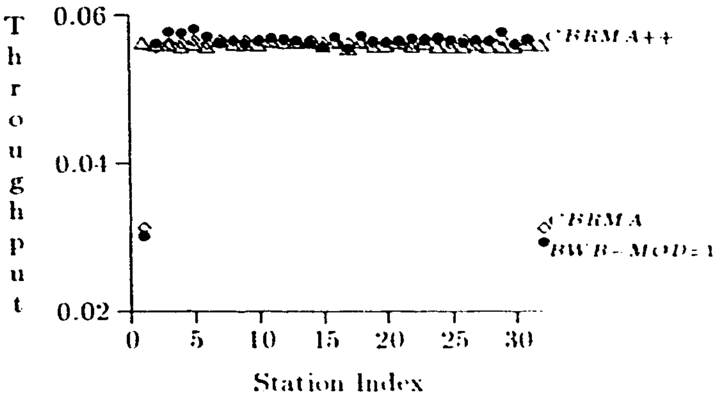


Figure 2.7: CBRMA++ on folded bus vs. CBRMA and DQDB on dual bus.

2.5.2 An Example Cycle – 2

Consider a network with 8 stations and a bus length of 80 slots so that the fair share is 10 slots per station; assume also that at the beginning of reservation cycle *i*, the following reservation requests are valid (figure 2.8) :

Station number	*The length of queue-F*	*The length of queue-B*
1	15	-
2	5	15
3	25	-
4	5	20
5	-	5
6	5	30
7	25	-
8	-	20

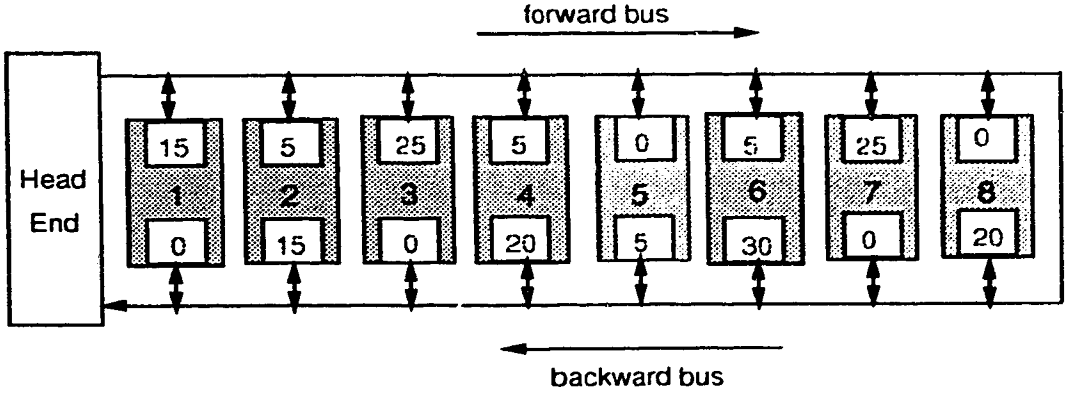


Figure 2.8: Outstanding reservation requests in cycle *i*.

Reservation Half Cycle

The headend station starts the reservation cycle by sending a reservation slot whose contents will be denoted as $< C, R_f, ES_f, ER_f, U_f, R_b, ES_b, ER_b, U_b >$ from now on. At the beginning, the slot contains *<i,0,0,0,0,0,0,0,0>* and the stations update it in the following manner:

- Station 1: The first station has messages for the forward bus only. So it claims its fair share first, which is 10. Then it states its willingness to use 5 additional slots, if it is given the chance. At this time, the number of stations that require extra slots on the forward bus is 1. Since station 1 has nothing to transmit on the backward bus, it simply puts its fair share into common use by incrementing U_b by 10.

  (Station 1 updates the reservation slot contents to *<i,10,1,5,0,0,0,0,10>*)

- Station 2: This station has messages for both upstream and downstream nodes. Its need on the forward bus is less than its fair share while the station can use 5 more slots on the backward bus.

  (Station 2 updates the reservation slot contents to *<i,15,1,5,5;10,1,5,10>*)

  Following the same principles the other stations make their reservations as follows:

- Station 3: *<i,25,2,20,5,10,1,5,20>*
- Station 4: *<i,30,2,20,10,20,2,15,20>*
- Station 5: *<i,30,2,20,20,25,2,15,25>*
- Station 6: *<i,35,2,20,25,35,3,35,25>*
- Station 7: *<i,45,3,35,25,35,3,35,35>*
- Station 8: *<i,45,3,35,35,45,4,45,35>*

When the reservation slot reaches the fold, the reservation scenario of the forward bus is as good as it can be, since the number of extra slot requests and the number of available slots are exactly the same. However, on the backward bus, we are 10 slots short to meet all demands. So, there has to be a compromise and the protocol will make sure in the balancing half-cycle that it will be a fair one.

During the reservation cycle, the *CycleStartCounters* and *XMitCounters* are set as follows:

Counters for FORWARD BUS

Station number	*CycleStart Counter*	*XMit Counter*
1	1	10
2	11	5
3	16	10
4	26	5
5	-	-
6	31	5
7	36	10
8	-	-

Counters for BACKWARD BUS		
Station number	*CycleStart Counter*	*XMit Counter*
1	-	-
2	1	10
3	-	-
4	11	10
5	21	5
6	26	10
7	-	-
8	36	10

Balancing Half-Cycle

At this moment, the reservation slot passes the fold and begins to propagate towards the headend station, thereby commencing the balancing half-cycle. If the number of unused slots is greater than or equal to the number of extra slot requests for a bus, there is no problem since all of them can be met. Otherwise each station computes its fair share of the unused slots and compares it with the difference between the unused slot count and the remaining extra slot requests, if this value is positive. Trying to maximize its gain the station chooses the maximum of these two values and updates the reservation slot accordingly:

- Station 8: This station knows that it has asked for 10 more slots and now realizes that only 35 slots are available for 4 stations (including itself) that together now asked for 45 slots. Following the algorithm, the station computes its fair share first: $\mathcal{F} = \lfloor 35/4 \rfloor = 8$. It also computes that the remaining 3 stations are asking for a total of 35 slots. Since ($35 - 35 = 0$), it chooses to settle with its fair share and claims 8 of the unused slots.

  (Station 8 updates the reservation slot contents to <*i,45,3,35,35,53,3,35,27*>)

- Station 7: This station finds out that its extra slot request can be met entirely.

  (Station 7 updates the reservation slot contents to <*i,60,2,20,20,53,3,35,27*>)

- Station 6: This station has no outstanding slot requests for the forward bus while it needs 20 extra slots in the reverse direction. Its fair share is $\mathcal{F} = \lfloor 27/3 \rfloor = 9$. On the other hand, the remaining two stations are asking for only ($35 - 20 = 15$) slots. If the station chooses to claim only 9 slots that will cause $27 - 9 - 15 = 3$ slots to be wasted. So, claiming 12 instead of 9 slots, station 6 updates the reservation slot contents as follows:

  <*i,60,2,20,20,65,2,15,15*>)

  Following the same principles the other stations make their reservations as follows:

- Station 5: Having no extra slot requests, it does not update the slot.
- Station 4: <*i,60,2,20,20,75,1,5,5*>
- Station 3: <*i,75,1,5,5,75,1,5,5*>
- Station 2: <*i,75,1,5,5,80,0,0,0*>
- Station 1: <*i,80,0,0,0,80,0,0,0*>

During the balancing cycle, the values of counters are finalized as follows;

Counters for FORWARD BUS

Station number	*CycleStart Counter*	*XMit Counter*
8	-	-
7	56	25
6	51	5
5	-	-
4	46	5
3	21	25
2	16	5
1	1	15

Counters for BACKWARD BUS

Station number	*CycleStart Counter*	*XMit Counter*
8	63	18
7	-	-
6	41	22
5	36	5
4	16	20
3	-	-
2	1	15
1	-	-

2.5.3 CBRMA++/SR: An Efficient Slot Reuse Mechanism

A common deficiency of DQDB, CRMA and CBRMA is that once a slot is filled by a segment, it will have to propagate to the end of the bus. The use of erasure nodes is suggested in [Rod90] to overcome this problem for DQDB. In this section we will propose a method to serve the same purpose for CBRMA++.

Although a slot is used twice in CBRMA++, it is clear that the performance of the protocol can be improved further by using a slot more than once on the forward and/or backward bus. The improvement in the throughput rate will be more pronounced when the number of stations is large. Figure 2.9 depicts a situation of interest for the following description of slot reuse (SR) mechanism. At this moment, the negotiations for the bandwidth allocation for $cycle_i$ was completed (i.e., the reservation slot of $cycle_i$ has reached the headend station). Therefore, each station knows how many slots it will use and where their payloads will be transmitted. It also has complete information regarding the number of back-logged packets in the local queue and their destinations, in other words, the information related to its future slot request for $cycle_{i+1}$. However, the station does not know for how many slots it will be the destination in $cycle_i$. If this information were also available, the station would be able to use the slots that it will be sent for transmission and to resize its request for $cycle_{i+1}$.

The SR mechanism works as follows: as soon as the reservation slot reaches the headend station, the headend station begins to transmit the **Reception Vector** (RV) on the forward bus with its contents set to all zeroes. RV has a counter field for every station and therefore the overhead of

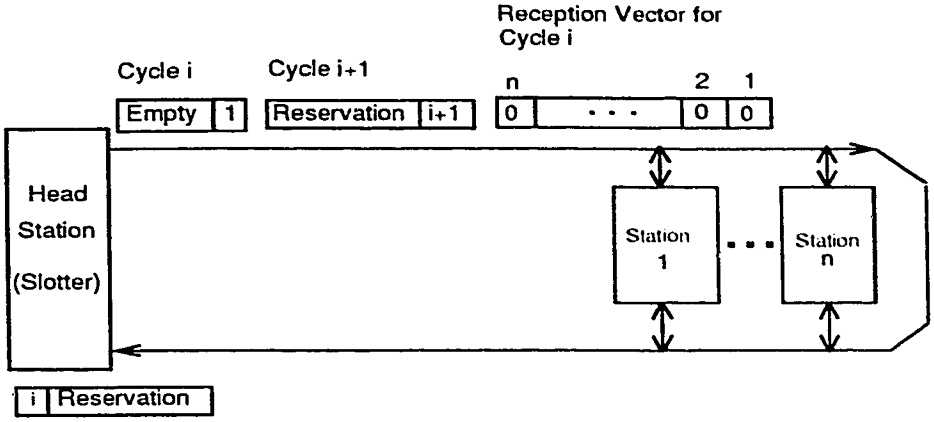


Figure 2.9: Slot reuse mechanism

SR is one counter field per station[11]. RV(i) contains the number of slots to be destined for S_i in $cycle_i$. Consequently, S_1 sets the corresponding members of RV according to the receiver addresses of the packets scheduled for transmission in $cycle_i$. When RV arrives at S_2, the station inspects RV(2) to obtain the number of slots it will receive, increments the local count of the slots scheduled for transmission by that number and increments the appropriate elements of RV accordingly. This operation will be repeated by every station until S_n is reached. S_n sets RV contents to all zeroes and the same operation is started anew for the backward bus. Note that RV is reused and there is no need for a second RV.

Suppose that S_1 was granted 6 slots in $cycle_i$ and it will use them to transmit 3 slots to S_2, 1 slot to S_5 and 2 slots to S_8. To inform S_2, S_5 and S_8 about the number of slots they will receive, S_1 sets the second, fifth and eighth elements of the vector to 3, 1 and 2, respectively. Other stations may increment these values if they happen to have packets addressed to the same destinations.

When RV reaches S_2, the station will be informed about the future transmission of 3 slots from S_1 in $cycle_i$. Assuming it has been provided with 5 slots during the preceeding balancing phase, the effective bandwidth allocated to S_2 in $cycle_i$ becomes 8 slots. Using this information, S_2 is able to update RV not only for 5 but also for the additional 3 slots. Therefore, the 3 slots reused by S_2 will also be reused by the downstream recipients of their payloads.

If a station is not able to use all of the reusable slots for transmission it passes the excessive slots to its immediate successor by simply changing their destination address and indicating the invalidity of information that they carry by flipping the empty/full bits.

In order to illustrate the performance of the SR mechanism, we provide three simulation results. The results are drawn against the performance of DQDB in the dual bus configuration[12] to provide a comparison basis. Figure 2.10 shows the results obtained under asymptotic conditions. In this case, the traffic pattern is artificial so that every station has as many packets as it can use on either bus. Therefore, figure 2.10 gives the total bandwidth available to stations. But this bandwidth is not the same for forward and backward bus. Figure 2.11 shows the available bandwidth on the forward and backward bus, separately.

The global throughput of DQDB in figures 2.10 and 2.11 is 1.76. CBRMA++ scores slightly better, since there is no loss due to the bandwidth balancing mechanism. The global throughput of

[11] Current slot length is the same as that of DQDB and current counter field length is 8 bits, so that a single slot contains 48 counter fields. If a network contains more than 48 stations, additional slots are used to accommodate RV. The number of slots required by RV is equal to $\lceil n/48 \rceil$, where n denotes the number of stations in the network.

[12] Note that the performance of DQDB on the folded bus will be half of what is shown in the figure, assuming that slots are reused only at the (single) headend station.

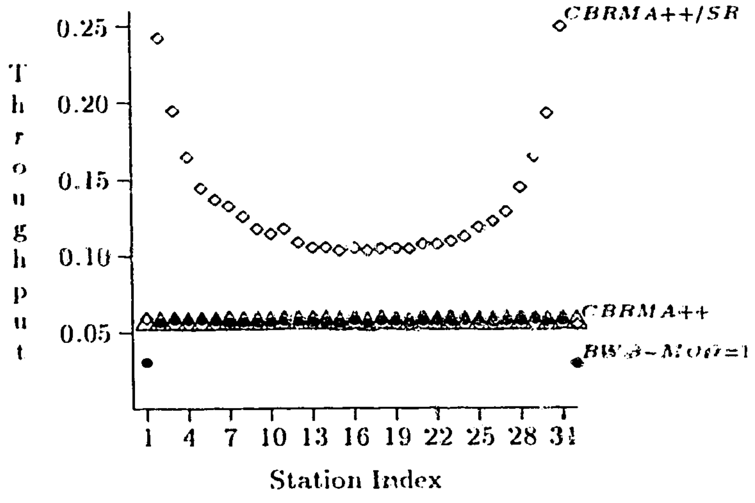


Figure 2.10: Performance under asymptotic conditions. Available bandwidth on the forward and backward buses are shown together.

CBRMA++/SR is 4.09. Under uniform load, this figure drops to 3.65, as can be seen in figure 2.12. The reason for the difference is quite obvious, since the stations closer to the ends of the bus do not have sufficiently many messages queued to make full use of the available bandwidth.

A number of methods for the implementation of erasure nodes for DQDB are suggested and analyzed in [Rod90, GaL91]. However, these schemes are not as effective as the SR mechanism of CBRMA++/SR, since the station that clears the slot and the one that uses it are different stations. In other words, the slot that is cleared by S_i can be reused by S_{i+1} (or S_{i-1} depending on the direction of transmission) at the earliest. In our protocol, a station can reuse a slot as soon it completes the inspection of the destination address field.

As can be seen in the figures, the SR mechanism disturbs the fairness of the protocol. However, it is clear that no compromise is made out of the fair share. The irregularity stems from two factors: first, some stations are destinations for more slots than others. Second, stations located closer to the ends of the bus will find more empty slots at their disposal on forward or backward bus, depending on their location. This explains why the throughput of S_{31} on the forward bus and the throughput of S_2 on the backward bus are so high under asymptotic conditions. On the other hand, under more realistic traffic patterns, the first factor will be irrelevant, since most likely the stations will not be able to fill all available empty slots, as can be seen in figure 2.12. As for the influence of the second factor, it is not strong enough to cause large variations in the throughput rates of the stations, since the bandwidth allocation mechanism does not allow a station to usurp the bus while the others are also contending for it. Therefore, under moderate or heavy load which is more or less uniform across the network, the effect of SR will follow the same pattern as bandwidth demand (i.e., gradually tapers off down to zero as one progresses downstream on a bus).

2.6 Conclusions

In this chapter, we presented a new protocol called CBRMA++/SR which can sustain a high aggregate data transmission rate while not compromising fairness. The good performance of the protocol is due to the preventive approach employed in the design. This is in contrast to CRMA which allows the occurence of irregularities in the network and then provides countermeasures[13] to compensate for

[13] The backpressure mechanism.

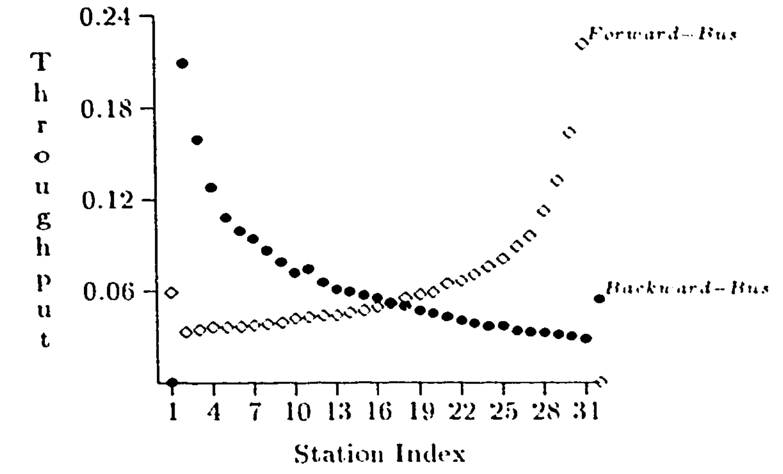


Figure 2.11: Performance under asymptotic conditions. Available bandwidth on the forward and backward buses are shown separately.

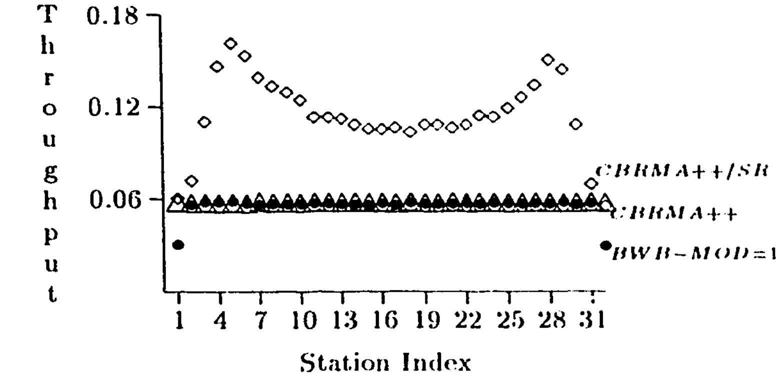


Figure 2.12: Performance under uniform traffic.

the undesired side-effects. The latter approach is bound to be less effective since both the detection of the irregularity and the application of the countermeasure take time which is not negligible under high transmission rates.

A remark is in order regarding the medium access delay. In our protocol, the delay between the beginning of the two consecutive transmission sessions[14] is always equal to the round trip delay of the bus when there is no SR mechanism in action. This is so, regardless of the global load on the network. However, the available bandwidth can change from the fair share up to the capacity of the whole cycle. In DQDB, shorter access delays are possible under light load. On the other hand, the SR mechanism has also the effect of reducing the access delays. The rate of the improvement is directly related to the number of slots in which stations act as the receiver.

When handling multislot packets, DQDB cannot ensure that the slots forming the packet will not arrive interleaved with slots coming to the same receiver from other senders. The protocol has to keep a separate queue for each sender or the slots have to be reordered before being passed to the upper layer protocols. This may add to the complexity of the protocol design and certainly will decrease the performance, especially at high transmission rates. Besides ensuring the slot contiguity,

[14] In which a number of slots are transmitted instead of one. Since the slot contiguity is guaranteed, using the delays between the transmissions of two slots can be misleading in the determination of the medium access delays.

CBRMA++/SR provides the upper layer protocols with the number of slots to be transmitted and received, in advance. We believe that such information can prove to be quite useful in the design of more efficient higher layer protocols.

Bibliography

[As90a] H.R. van As, "Major Performance Characteristics of the DQDB MAC Protocol", SBT/IEEE International Telecommunications Symposium, September 1990, 113–120.

[As90b] H.R. van As, "Performance Evaluation of Bandwidth Balancing in the DQDB MAC Protocol", Eighth Annual European Fibre Optic and Local Area Networks Conference, EFOC/LAN 90, June 1990, 231–239.

[AWZ90] H.R. van As, J.W. Wong, P. Zafiropulo, "Fairness, Priority and Predictability of the DQDB MAC Protocol under Heavy Load", Int. Zurich Seminar on Digital Communications, March 1990, 410–417.

[BaD91] C. Baransel, W. Dobosiewicz, "CBRMA (Cyclic Balanced Reservation Multiple Access) MAC Protocol", *The Sixth International Symposium on Computer and Information Sciences-ISCIS VI*, October 1991, Antalya, Türkiye.

[BDG92] C. Baransel, W. Dobosiewicz, P. Gburzynski,"CBRMA++ : On How to Increase the Performance of a MAC Protocol Without a Second Headend Station", *Silicon Valley Networking Conference*, April 1992, California, USA.

[BDGa] C. Baransel, W. Dobosiewicz, P. Gburzynski, "CBRMA++/SR: A High-Speed MAN/WAN MAC Protocol with An Efficient Slot Reuse Mechanism", accepted for publication in the proceedings of *The Eight International Symposium on Computer and Information Sciences-ISCIS VIII*, November 1993, Istanbul, Türkiye.

[BDGb] C. Baransel, W. Dobosiewicz, P. Gburzynski, "CBRMA++/SR: On the Design of a MAN/WAN MAC Protocol for High–Speed Networks", *to appear in the special issue (Networks in the Metropolitan Area) of IEEE Journal on Selected Areas in Communications*, 1993.

[CGL91] M. Conti, E. Gregori and L. Lenzini, "A Methodological Approach to an Extensive Analysis of DQDB Performance and Fairness", *IEEE Journal on Selected Areas in Communications*, 9, 1 (January 1991), 76–87.

[DoG91] W. Dobosiewicz, P. Gburzynski,"The Topology Component of Protocol Performance", in Proc. Conference on Local Computer Networks, Minneapolis, October 1991, 582–588.

[GaL91] M. W. Garret, S. Q. Li, "A Study of Slot Reuse in Dual Bus Multiple Access Networks", *IEEE Journal on Selected Areas in Communications*, 9, 2 (February 1991), 258–256.

[IEE90] IEEE, "Std 802.6—1990, IEEE Standards for Local and Metropolitan Area Networks: Distributed Queue Dual Bus(DQDB) of a Metropolitan Area Network (MAN)", July 1991.

[LAT92] J. Liebeherr, I.F. Akyildiz, A.N. Tantawi, "An Effective Scheme for Pre-Emptive Priorities in Dual Bus Metropolitan Area Networks", SIGCOMM'92.

[MNW90] H.R. Muller, M.M. Nassehi, J.W. Wong, E. Zurfluh, W. Bux and P. Zafiropulo, "DQMA and CRMA: New Access Schemes for Gbit/s LANs and MANs" INFOCOM'90, 185–191.

[Na90a] M.M. Nassehi, "CRMA: An Access Scheme for High-Speed LANs and MANs", ICC'90, 1697–1702.

[Na90b] M.M. Nassehi, "Cyclic Reservation Multiple-Access Scheme for Gbit/s LANs and MANs based on Dual-Bus Configuration", Eighth Annual European Fibre Optic and Local Area Networks Conference, EFOC/LAN 90, June 1990, 246–251.

[PhB92] V.P.T. Phung, R. Breault, "On the Unpredictable Behavior of DQDB", *Computer Networks and ISDN Systems*, 24, 145–152, 1992.

[Rod90] M. A. Rodrigues, "Erasure Nodes: Performance Improvements for the IEEE 802.6 MAN", INFOCOM'90, 636–643.

[Sta84] W. Stallings, "Local Network Performance", *IEEE Communications Magazine*, 22, 2 (February 1984), 27–36.

[TrD90] P. Tran-Gia, R. Dittmann, "Performance Analysis of the CRMA-Protocol in High-Speed Networks", Univ. of Würzburg, Institute of Computer Science Research Report Series, Report No. 23, December 1990.

Chapter 3

SP/R

3.1 Introduction

In the development of a MAC protocol based on slotted access for high speed MANs and WANS, the possible approaches range from very simple *create-the-slots-and-do-nothing-else* to *control-everything*. In the first case, the hardware requirements and the intelligence of the MAC protocol can be kept to a minimum at the expense of unpredictability, unfairness and nonflexibility. In the latter case, a more intelligent MAC protocol, more complex and therefore more costly hardware and nonnegligible amount of bookkeeping are required to provide a better service. Furthermore, even the availability of these resources may not be able to guarantee the quality of service of the MAC protocol. Despite its considerable complexity, the criticism aimed towards the DQDB protocol is a well-known manifestation of this fact [CGL91, MNW90, As90a, As90b, AWZ90, Rod90, GaL91].

In the first chapter we proposed a series of protocols, each one augmenting the previous version in a certain way. Although the performance of each of these protocols is superior to that of DQDB as far as fairness, predictability and flexibility issues are concerned, they are by no means less complicated than DQDB. However, we believe that considerably simpler protocols possessing comparable performance characteristics are possible and the protocol proposed in this chapter is one of them.

3.2 Basic Approach

The protocol that we have in mind is suitable for the dual bus configuration (figure 2.1). The transmission medium is optical fiber; therefore, transmission is unidirectional on both buses[1] and each station has the ability to act as an erasure node. The SP/R–MAC protocol is intended as an alternative to DQDB with slot reuse mechanism. Our aim is to develop a MAC protocol which can efficiently exploit the erasure node mechanism while preserving fairness in bandwidth allocation on a dual bus network.

One way of achieving fairness (while maintaining a high aggregate throughput rate) in a MAC protocol is to circulate a certain amount of information regarding the bandwidth requirements of the stations and regulate the bandwidth allocation according to this information. This information can be processed either in a distributed fashion or by a central authority. The DQDB protocol is based on distributed control whereas IBM's CRMA protocol adopts the centralized approach. In the distributed case, temporary unfair treatment is very likely to occur since there is no way to convey this information to all involved stations instantaneously. The centralized approach is more suitable if fairness is the only issue of concern; and it was also our approach in developing CBRMA++/SR. However, a certain time is required for the bandwidth allocation requests to reach the central control

[1] In the rest of the chapter we will use subscripts F and B to refer to forward and backward buses respectively.

and consequently zero medium access delay may not be possible even under light or moderate traffic loads since the bandwidth is allocated upon demand and in a centralized fashion.

Another approach is to begin with a deterministic bandwidth allocation policy (a function which is carried out by the headend stations) and let the stations put the unused portion of their allocated bandwidth into common use. The merit of this approach is that there will be no way for a station to dominate the bus and get excessive service to the detriment of others. On the other hand, some measures are required to prevent slot waste. In this protocol, we choose not to circulate any status or request information within the network, but instead, adopted a slot pre/reuse mechanism as an integral part of the network rather than an option. As a result, it can be said that SP/R–MAC protocol is based both on an explicit bandwidth allocation policy and controlled slot reuse. In developing this protocol, our approach was to assign a more active role to this particular mechanism along with the new **ownership** concept so that a more responsive infrastructure can be established.

At this point we would like to differentiate between three slot reuse mechanisms: the slot can be used before it reaches its intended user (pre–use); the slot can be reused by the destination station upon receiving a message from an upstream station (receiver reuse), or the slot is cleared by the destination station and can be used by its immediate downstream neighbour at the earliest (destination release). In the first case , the slot should be cleared before it reaches its intended user or it should carry a message to the user itself. Our protocol makes use of slot pre-use and receiver reuse.

The performance of the presented protocols will be illustrated by simulation results. The simulation model assumes a network composed of 32 stations placed in equidistant intervals on a dual bus. The transmission rate is 1 Gbps and the length of either forward or backward bus is 30690 bits or approximately 73 slots. Considering the 5 ns/m propagation rate of optical waveguides, this corresponds to a bus length of roughly 6138 m. Each slot is 416 bits long and two consecutive slots on the transmission link are separated by a 2–bit interslot gap. Slots have 32–bit headers and 384–bit segment payloads.

The rest of the chapter is organized as follows: first we will obtain the specific values for the achievable throughput rates in our network via simulation. For the related mathematical analysis the interested reader is referred to [GaL91, BaM92]. Then the protocol will be described in detail. The chapter will conclude with the discussion of the simulation results.

3.3 The Maximum Throughput Rates Under Asymptotic and Uniform Traffic Patterns

In our network each station maintains two separate message queues, one for the forward bus and one for the backward bus. The throughput rates are calculated using the segment payload as opposed to slot length. Therefore, without the slot pre/reuse, the maximum throughput rate per bus is calculated to be 0.919, yielding a global throughput rate of 1.838.

When receiver reuse is employed, the throughput improves considerably: in figure 3.1[2], the throughput on the forward bus is 3.65[3]. The headend stations assign slots to stations on a round robin basis and send them full[4]. Compared to 0.919, the global improvement is approximately 4-fold. Note that this scheme provides the upper limit for throughput rates under asymptotic load[5] since each slot is pre-used by the headend and receiver reuse is employed afterwards.

[2] In this figure, the throughput rates on the forward and the backward buses are drawn separately along with the totals.

[3] And the combined throughput equals 7.3, which is practically the same as the maximum achievable on a dual–counterrotating ring (e.g., 8 × 0.919 = 7.35 in Metaring).

[4] The throughput of headend stations which is 0.919 is not shown in the figure to save space.

[5] When slots are allocated evenly across the stations.

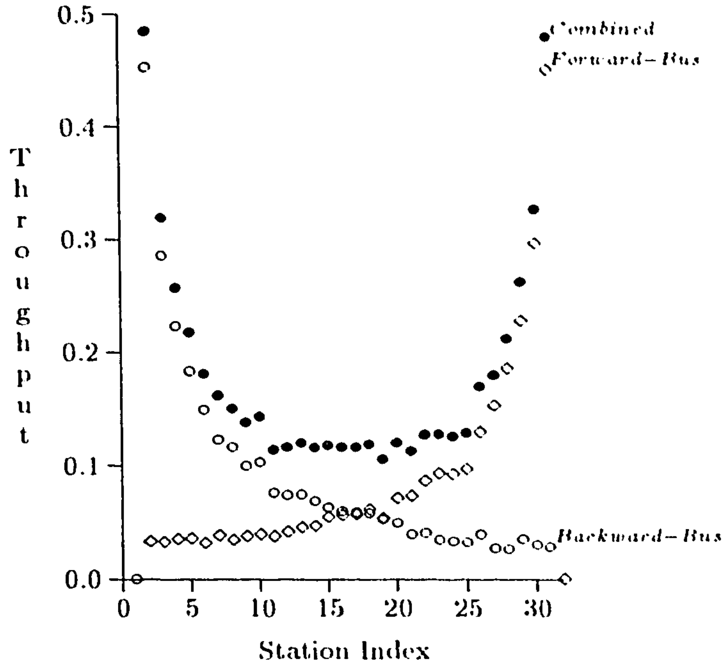


Figure 3.1: Throughput rates under asymptotic conditions.

Figure 3.2 gives the related average medium access delays[6]. The medium access delay is defined as the time elapsed from the placement of a packet in the transmission buffer to the end of its transmission, including the transmission time in units of slots. The minimum access delay is the time to transmit a single slot. The combined average medium access delays are calculated according to the following formula,

$$d_T = \frac{t_F \times d_F + t_B \times d_B}{t_F + t_B}$$

in which d stands for the average access delay and t represents the number of slots transmitted on the related bus. It is obvious that for the stations located closer to the headend stations, the addition of the slot reuse mechanism does not reduce the medium access delays considerably. However, the positive effect becomes quite obvious as one moves downstream on a bus. In our protocol, we use a simple control mechanism to decrease the medium access delays without detriment to the downstream stations.

When the average amount of generated traffic is the same across the network and the traffic is uniformly distributed, the load offered by any station on either bus is proportional to the number of destination stations downstream. Let us assume that a station transmits to each of the other stations with equal probability. Also assume that there are n stations in the network and each message fits exactly into a single slot such that there are no multi slot messages and no bandwidth is wasted due to internal fragmentation. Consequently, the probability of S_i selecting a particular S_j as the destination is given as $p(t_{ij}) = 1/(n-1)$. In this case, at S_i, a newly generated packet will be transmitted on the forward or backward bus according to the following probabilities:

$$p_F(t_i) = \frac{n-i}{n-1} \qquad (3.1)$$

[6] The average medium access delays on the backward bus are symmetric to those on the forward bus. They are not shown to preserve the clarity of the figure but can easily be surmised from the overall delays.

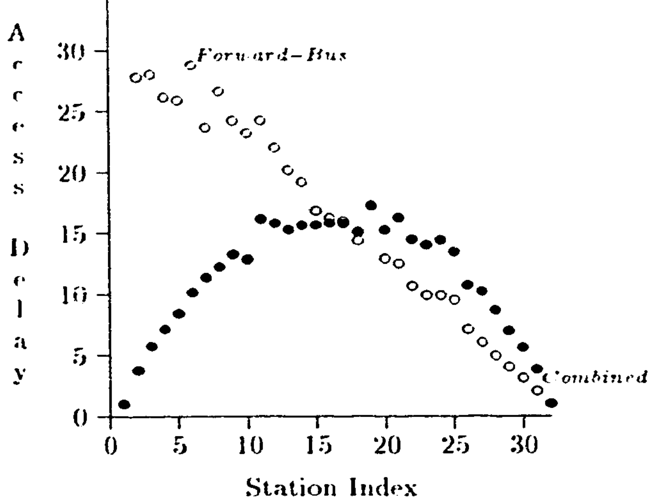


Figure 3.2: Medium access delays under asymptotic conditions.

$$p_B(t_i) = \frac{i-1}{n-1} \tag{3.2}$$

Carrying out the same argument for the probability of reception, we will end up with the following equations:

$$p_F(r_i) = \frac{i-1}{n-1} \tag{3.3}$$

$$p_B(r_i) = \frac{n-i}{n-1} \tag{3.4}$$

These equations state that, as a station gets closer to the headend on either bus, its demand for bandwidth on that particular bus is expected to be getting higher when compared to the demands of the other stations located downstream to it. Furthermore, if receiver reuse is allowed, the number of slots that can be received decrease as the station gets closer to the headend on either bus. In other words, as one moves downstream on a bus, the probability of slot reception increases and the probability of the transmission decreases by the same factor. Our simulator captures this aspect by decreasing the message arrival rate as one proceeds downstream on either bus. Due to this fact, the global throughput rate decreases when we change the offered load pattern from asymptotic to heavy uniform load, as seen in figure 3.3.

In this case, the overall message generation rate is still high enough to allow the headend stations continuously to send full slots. Nevertheless, only the stations located in the central region of the network were able to attain the throughput rates comparable to those depicted in figure 3.1. This is due to the fact that the difference between the lengths of the message queues for the forward and the backward buses is not negligible at every station. Consequently, not every station may be able to use one bus as efficiently as it does the other. Furthermore, significant capacity is wasted as indicated by the decrease in the global throughput from 7.30 to 3.03.

3.4 The Protocol

In this section, we present the basic form of our new protocol. The mechanism is based on the following observations:

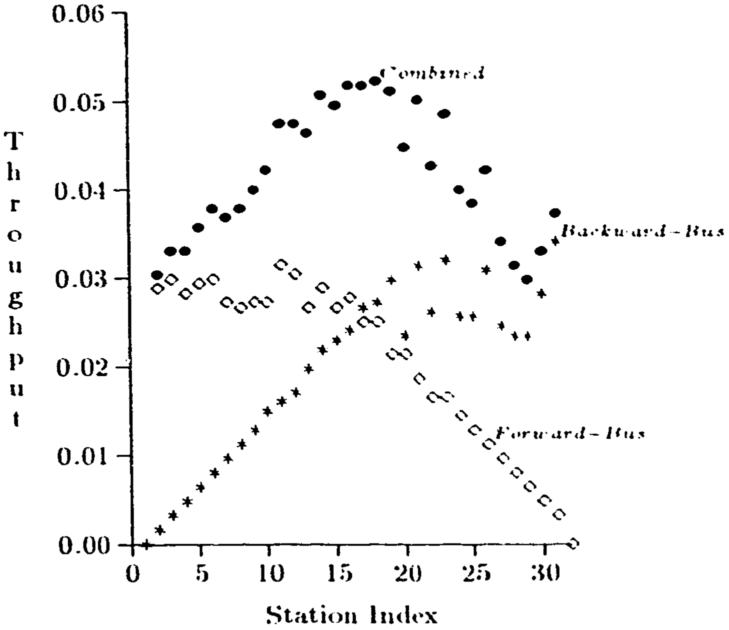


Figure 3.3: Throughput under uniformly distributed heavy traffic.

1. The lengths of the queues in the nodes that hold the messages to be transmitted on the forward and the backward bus are dependent upon the location of the the nodes on the network and are proportional to the the number of receiver stations located downstream. This is an important principle of the bandwidth allocation policy in our protocol.
2. An uncontrolled slot reuse mechanism has a limited effect on the throughput rates and medium access delays of the nodes located closer to the headend stations. Therefore, to achieve fairness, the positive effect of the slot reuse mechanism should be provided to these stations by explicit control in bandwidth allocation.
3. A MAC protocol should provide the necessary flexibility to handle heterogeneous traffic demands without sacrificing fairness. Therefore, some measures are required to break the rigidness of a possible deterministic bandwidth allocation mechanism when it is appropriate.

At this point, it is time to state our understanding of *fairness*. Loosely speaking, it can be understood as dividing the available bandwidth amongst the active users evenly. Therefore, under asymptotic conditions equal medium access delays and throughput rates indicate a fair protocol. On the other hand, a mechanism that can distribute the available global bandwidth[7] in a fair manner without further consideration does not necessarily yield comparable throughput rates across the network in a dual bus configuration since an additional decision step is required to arbitrate this fair share of bandwidth assigned to a station on a forward and backward bus basis (for example, it does not make much sense to allocate the same amount of bandwidth to S_1 and S_n on the forward bus). Therefore, the fair global share of a station should not be divided evenly on both buses, since this approach actually leads to an inherently unfair MAC protocol: assume that the same bandwidth is assigned to S_2 and S_m on the forward and backward bus, where S_m is located in the middle point of the network. Under normal conditions S_m will be able to make good use of both buses while S_2 will be able to do so on the forward bus only. To achieve a comparable throughput, S_2 needs to

[7] In this context, the global bandwidth is the sum of the bandwidth available on two buses.

use the backward bus as much as the forward bus which is not likely under realistic traffic patterns. As a result, for the dual bus configuration, the location of the station is an important parameter of the bandwidth allocation process since it is not only how much is allocated but also where it is allocated.

To devise a mechanism to regulate the bandwidth allocation, equations 3.1–3.4 are re–examined. For the station located in the middle, it is easy to see:

$$p_F(r_i) = p_F(t_i)$$

In other words, within a sufficiently long interval of time, the average number of slots to be transmitted on the forward bus is expected to be equal to the number of slots to be received on the same bus. On the other hand, as we move closer to the headend, the stations will be needing more slots than they can expect to receive. Our approach is to allocate some extra slots explicitly to the stations. In the ultimate case, all the slots possibly needed by S_1 should be reserved explicitly since it will not receive anything on the forward bus. The same argument also applies to S_n as for the transmission on the backward bus. In order to facilitate the explicit slot allocation, it is necessary to add a field to the conventional slot structure. Therefore, in our scheme, the slot header has four fields; namely **Sender Addresss, Receiver Address, Busy/Empty bit** and **Owner Id**. Headend stations act both as ordinary stations and as slotters/arbiters. It is the responsibility of the headend station to assign the appropriate station ids to the **Owner Id** field of each slot. This field will be updated by other stations according to the rules that govern slot reuse. The scheme will be explained for the forward bus only since the operation on the backward bus is symmetrical.

3.4.1 Bandwidth Allocation at Headends

Assuming there are n stations in the network and $m = \lfloor (n+1)/2 \rfloor$, S_1 first creates m slots for its own exclusive use. Then, it creates $(m-1)$ slots and assigns them to S_2. This process continues until a single slot is sent to S_m and then it starts anew from the beginning. The bandwidth allocation algorithm is given below.

•Bandwidth Allocation Algorithm, Version 1

```
int m = floor((n+1)/2) + 1 ;
while (1) {
       for (i=1, i<=m, i++) {
            for (j=0, j<m-i, j++) {
                 create_slot() ;
                 Slot->OwnerId = i ;
                 transmit_slot() ;
            }
       }
}
```

In the algorithm, the body of the *while* statement can be considered as a *cycle*. In each cycle $(m^2/2 + m/2)$ slots are issued, the first m belonging to S_1. When m is large, the message queues at the stations can grow to be rather long. Therefore, instead of allocating a continuous chunk of bandwith to one station, allocation of smaller portions in an interleaved manner may be desirable to prevent long queues under medium and heavy traffic and to reduce medium access delays[8]. In this case, the algorithm should be changed into the following form.

•Bandwidth Allocation Algorithm, Version 2

[8] Especially for the stations transmitting small amounts of information and waiting for a response prior to further transmissions.

```
int m = floor((n+1)/2) + 1 ;
while (1) {
       for (i=m, i>0, i--) {
            for (j=1, j<=i, j++) {
                 create_slot() ;
                 Slot->OwnerId = j ;
                 transmit_slot() ;
            }
       }
}
```

At the beginning of a cycle, S_1 assigns a single slot to every station beginning from itself up to S_m. Then it sends a single slot to every station up to S_{m-1}, this time excluding S_m. In the next step S_{m-1} will be excluded from the list of the recipients. The algorithm proceeds in this manner until S_1 is the only station remaining in the list and then starts again. Note that the number of slots sent to the stations in a cycle is the same for both algorithms. On the other hand, for its aforementioned advantages, we suggest the second form, especially for large networks.

Allocating the the bandwidth explicitly for the first half of the bus and doing nothing else does not mean that remaining stations are exposed to starvation. Statistically, they will be protected from starvation by the the slot reuse mechanism more and more as one moves toward the end of the bus. Furthermore, an additional measure to prevent a single station from usurping the bus will be introduced shortly.

3.4.2 Medium Access and Slot Pre/Reuse at Stations

Adjusting the bandwidth allocation pattern has its cost, since some slots travel empty on the bus until they reach their "owners". The slot reuse mechanism becomes active only after that. However, some of the lost bandwidth can be regained by slot pre–use. Consider a slot owned by S_{14} propagating on the forward bus. There is no reason to prevent S_2 from using this slot to transmit a packet to, say, S_{10}. That is to say, if the owner of a slot is located downstream relative to the destination of a packet waiting for transmisssion in the buffer of the station that it is passing by, the station is allowed to use (pre–use) the slot. Also, if the owner of the slot chooses not to use the slot, the first downstream station that wishes to use it, is allowed to do so. However, the access algorithm is still not complete. As an example consider the following scenario: assume that on the forward bus the stations from S_1 to S_m have nothing to transmit so that all slots arrive to S_{m+1} empty. If S_{m+1} uses all of them to transmit to S_n, the stations in between will all starve. Although the possibility of such a case is quite small especially for large networks, nevertheless, it is not zero and this requires a preventive measure. The solution is to force the stations to update the **Owner Id** field of the slots that they can potentially use but leave unused because they have no message to transmit. The purpose of this operation is to disperse the unused bandwith capacity within the network as evenly as possible. In order to do so, each station maintains a counter, increments it by 1 for every unused slot and copies its value into the **Owner Id** field of the slot. For S_i, the value of this counter for the forward bus starts with $(i+1)$ and increases until $(n-1)$, then wraps back to $(i+1)$. The operation for the backward bus is similar.

The access algorithm is given below:

•Medium Access Algorithm

```
for (every_incoming_slot) {
    if (Slot->OwnerId == Station->Id)
        grab_slot() ;
    else
```

```
        if (Slot->Empty)
            if ((Slot->OwnerId >= Buffer.Receiver)
                 || (Slot->OwnerId < Station->Id))
                grab_slot() ;
            else
                repeat_slot_on_outgoing_link() ;
        else
            if (Slot->Receiver == Station->Id)
                grab_slot() ;
            else
                repeat_slot_on_outgoing_link() ;
}

function grab_slot() {
   if (there_is_message)
       use_slot_to_transmit() ;
   else {
       update_Owner_Id () ;
       transmit_empty_slot () ;
   }
}
```

The throughput rates and medium access delays of SP/R-MAC protocol are given in figure 3.4 and 3.5 respectively. The offered load is still asymptotic but headend stations are allowed to fill only the slots explicity allocated to them. Consequently, their throughput drops from the previous 0.919 to 0.117. On the other hand, their immediate successors enjoy over two-fold increase in throughput rates on their favorite buses, i.e the buses they will most likely use. The throughput rate per bus slightly decreases from 3.65 to 3.50. This is the cost of adjusting the bandwidth allocation in our protocol against the uncontrolled slot pre/reuse mechanism.

3.5 Simulation Results

In this section, some simulation results that are obtained under more realistic traffic patterns will be reported. Each node still maintains two different queues but they are no longer equally loaded. Instead, their mean message interarrival times are set to conform to their respective transmission probabilities. No restriction is imposed on the length of the queues. Figure 3.6 gives combined throughput rates of both buses under three different workloads. The combined medium access delays are depicted in figure 3.7. For the medium-load, heavy-load and overload cases the global throughput rates are 1.72, 3.42 and 4.16, respectively. Note that for the overload case, the throughput rates are not the same at all stations since the stations begin to use their "less-favorite" buses more heavily due to the availability of more messages in their queues. This outcome can also be predicted from the results obtained under asymptotic load. However the important point is that there is no decrease in the throughput rates of the middle stations when the load profile changes from heavy-load to overload — the region in which the network is most heavily congested.

Figures 3.8–3.11 illustrate the behaviour of the protocol in the presence of continuously bursty (i.e., operating under asymptotic conditions) stations. In figures 3.8–3.10, a single bursty station is placed in three different locations on the bus. Note that the bandwidth available to the bursty station increases as the intensity of the uniform background traffic decreases. In figure 3.11, three bursty stations are placed together. The protocol is quite effective in guarding the less heavily loaded stations against the bursty ones. Most importantly the protocol does not suffer from the following deficiencies of the slot reuse mechanisms observed for DQDB (see [GaL91]):

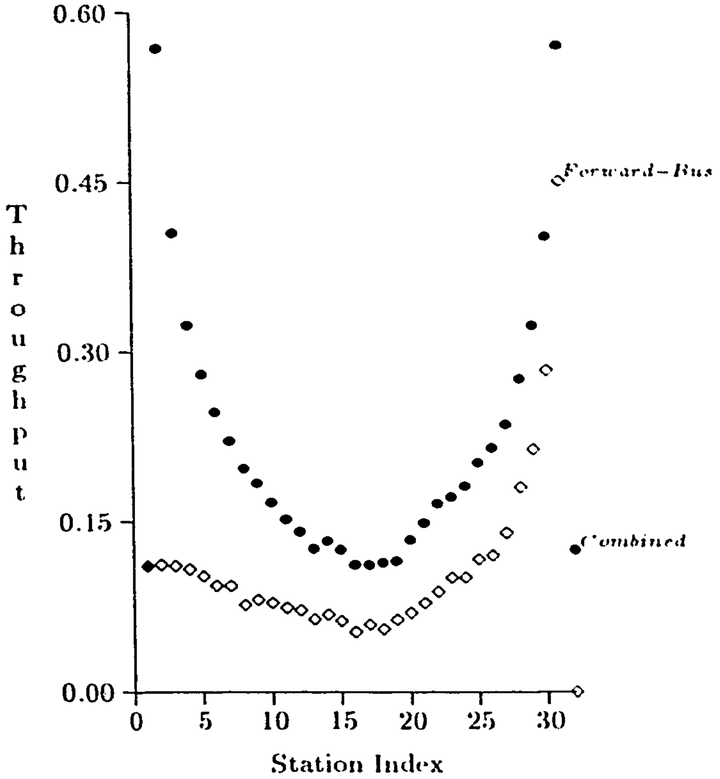


Figure 3.4: SP/R MAC protocol, throughput rates under asymptotic conditions.

1. In order to benefit from the slot reuse, it is necessary to employ the standby state which was dropped by the 802.6 committee because of its contribution to the unfairness of the protocol.

2. Not all stations can benefit from the slot reuse equally. When the number of erasure nodes is limited to a small number, the unfairness of the protocol is compounded since the average medium access delays observed at the nodes located prior to the erasure nodes can be up to three times higher than for their immediate successors. Specifically, the station just preceding the first erasure node gets the worst service.

3. To improve the throughput, it is necessary for the stations to inform others about its reuse of a slot by sending a "negative request". NREQ causes the withdrawal of the station's outstanding request from the global request queue. As a result, the counters of all upstream stations are decreased thereby decreasing their obligation to yield free slots to others. Since the downstream stations are less likely to benefit from the NREQ, the upstream stations send early at the expense of those located further down the bus.

and also;

1. Slot pre-use is not employed in DQDB, therefore the slots should travel empty until they reach their first user. The bandwidth waste can be substantial especially as the bus length increases.

2. The BWB mechanism of the DQDB protocol is known to be ineffective and slow to adapt in the presence of short-lived bursts [As90a, As90b]. The erasure nodes are introduced on top of it and because of its location-dependent characteristics they may contribute to the unpredictability of the protocol.

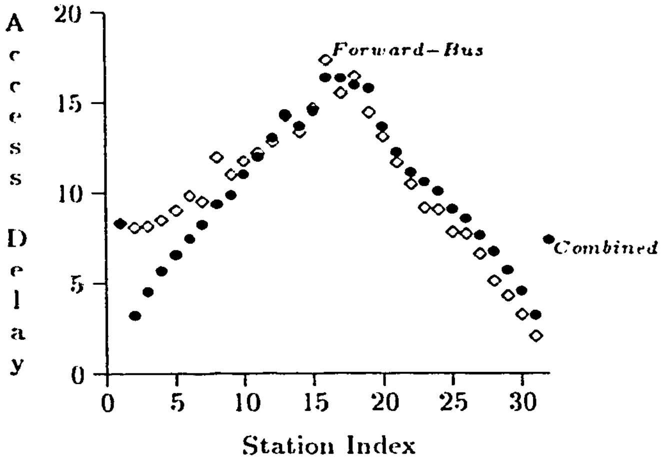


Figure 3.5: SP/R MAC protocol, medium access delays under asymptotic conditions.

3.6 Implementation Notes

In SP/R, each node needs to inspect the *Owner Id* and *Receiver Address* fields of every slot before deciding to use the slot or to re-transmit it. Therefore, the transmission will be delayed at every station by the time required to receive these two fields completely. In our simulations, the transmission rate is 1 Gbps and each field is 8-bit long. Consequently, the delay is 16 *ns* per station[9].

The effectiveness of the slot reuse mechanism can be improved at the nodes by changing the way the message queues are managed. In our simulations, we used the simple FIFO method. On the other hand, a more efficient but also more complex alternative is available. Suppose that an empty slot owned by S_{28} is passing by S_{10}. If the message at the start of the queue is to be sent to S_{30}, S_{10} will not be able to use this slot. However if it maintains some additional information about other messages in the queue, the station may be able to transmit selecting a message addressed to a station preceeding S_{28}. Because of the additional complexity, we suggest this improvement as an optional feature of the protocol, perhaps more suitable for the nodes located at the central region of the network so that thay can benefit more from slot reuse. For the time being, this variation is not included in our simulations.

3.7 Conclusions

In this chapter, we proposed a new MAC protocol with a bandwidth allocation scheme based on explicit bandwidth allocation and controlled slot reuse. In this protocol, no request and/or status information is circulated within the network. Since any attempt to provide every station with the exact information requires a delay, by the time this information reaches the farthest station it will be obsolete because of the highly dynamic nature of the network activity. This fact will be more pronounced as the network size and/or transmission rate is increased. In this context, it seems reasonable to start with a sound approximation in bandwidth allocation and handle the deviations probabilistically rather than trying to handle each case explicitly.

Due to the *ownership* mechanism, there is no way for a station to dominate a bus as long as there is competition for it. On the other hand, bandwidth waste is prevented by the rules that control slot reuse. The integrity of our approach stems from the fact that the slot-reuse capability is designed

[9] Note that, if value of 0 in *Receiver Address* field is interpreted as the indication of an empty slot it is possible to dismiss the*Busy/Empty Bit* from the slot structure — as we chose to do. Otherwise, the slot delay will be 17 *ns* per station.

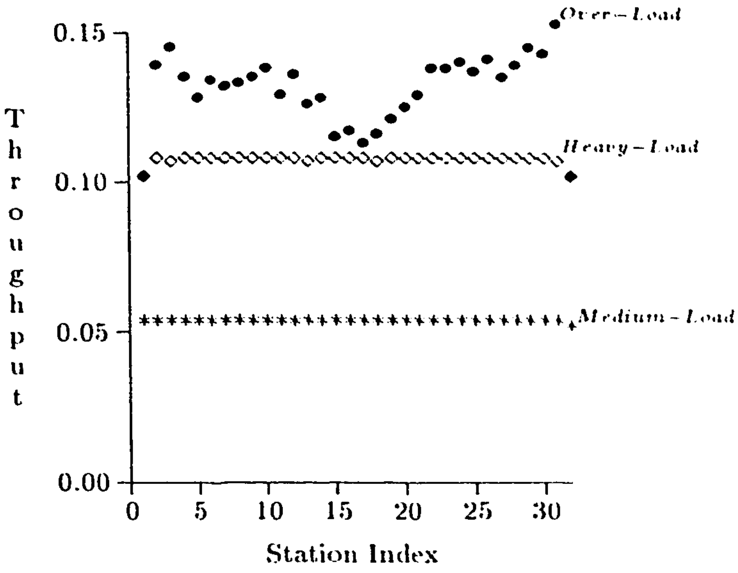


Figure 3.6: SP/R MAC protocol, throughput rates under uniformly distributed traffic with various workloads.

to contribute actively to the fairness of the protocol rather than being employed as an option aimed solely to boost the throughput rates. Furthermore, the protocol offers zero medium access delay under light or moderate load.

We also compared our protocol with Metaring [CiO90]. Even though the study was not an extensive one, the results turned out to be very encouraging. The maximum throughput rate of SP/R–MAC is practically the same as Metaring with large k, say 50 or more. When k is set to 10 or less, the performance of Metaring deteriorates rapidly. Furthermore, the ratio of maximum average acces delay to minimum average acces delay observed across the stations can be as large as 10000 even under light load. As for medium and heavy load, Metaring cannot handle the traffic in real time. This being the case, our protocol seems to be able to offer a very high throughput rate without exposing the stations to the danger of starvation.

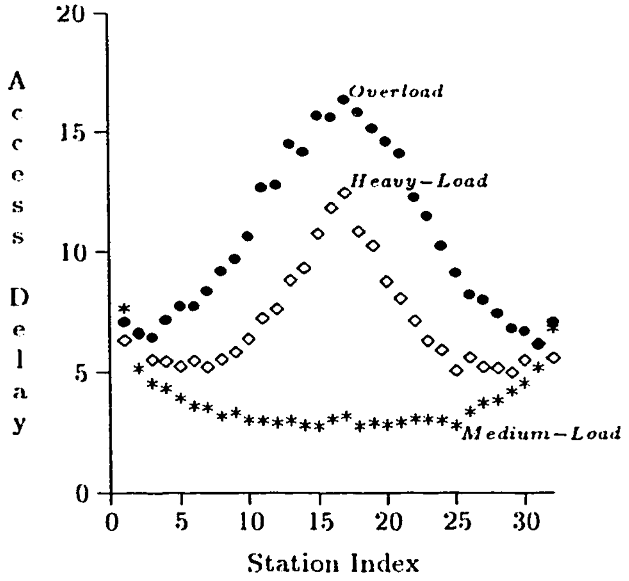


Figure 3.7: SP/R MAC protocol, average medium access delays under uniformly distributed traffic with various workloads.

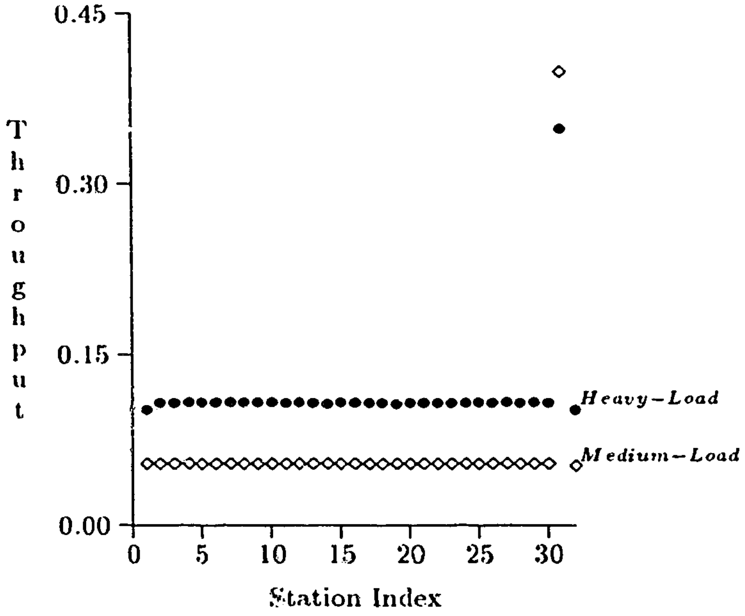


Figure 3.8: SP/R MAC protocol, S_{31} is bursting against medium and heavy-load uniform background traffic.

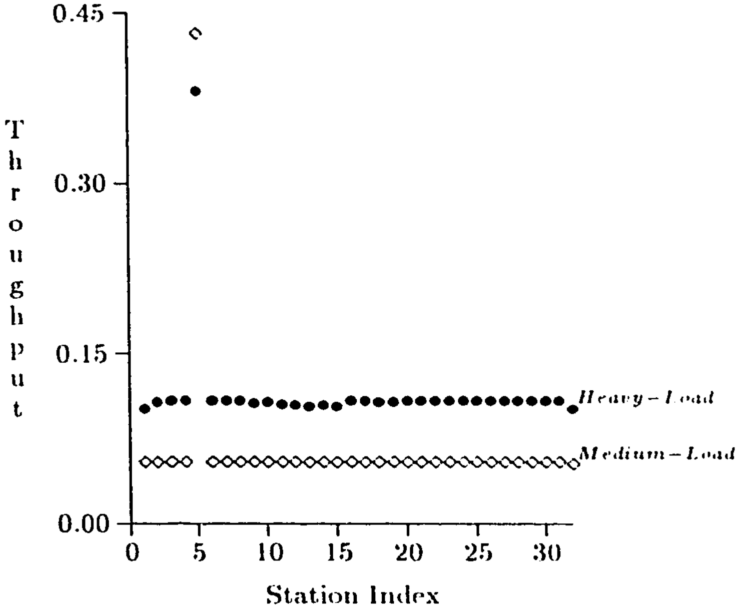


Figure 3.9: SP/R MAC protocol, S_5 is bursting against medium and heavy-load uniform background traffic.

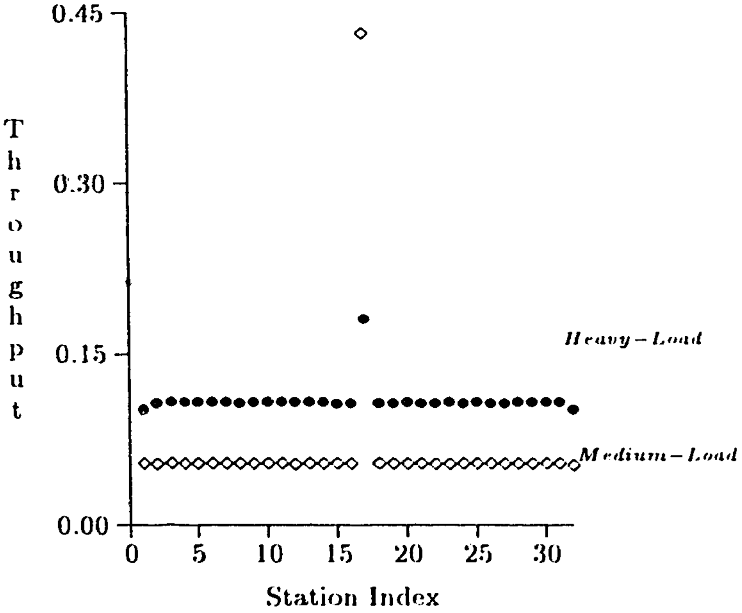


Figure 3.10: SP/R MAC protocol, S_{17} is bursting against medium and heavy-load uniform background traffic.

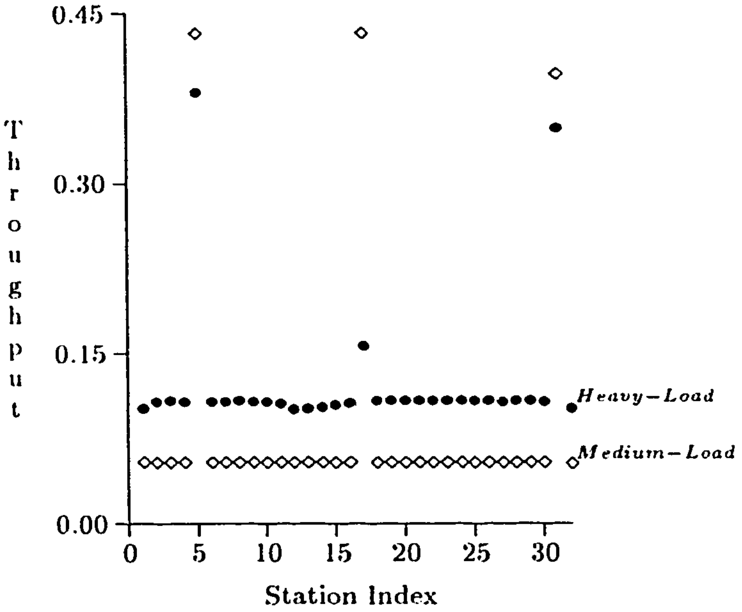


Figure 3.11: SP/R MAC protocol, S_5, S_{17} and S_{31} are bursting against medium and heavy-load uniform background traffic.

Bibliography

[As90a] H.R. van As, "Major Performance Characteristics of the DQDB MAC Protocol", SBT/IEEE International Telecommunications Symposium, September 1990, 113–120.

[As90b] H.R. van As, "Performance Evaluation of Bandwidth Balancing in the DQDB MAC Protocol", Eighth Annual European Fibre Optic and Local Area Networks Conference, EFOC/LAN 90, June 1990, 231–239.

[AWZ90] H.R. van As, J.W. Wong, P. Zafiropulo, "Fairness, Priority and Predictability of the DQDB MAC Protocol under Heavy Load", Int. Zurich Seminar on Digital Communications, March 1990, 410–417.

[BaM92] S. Banerjee, B. Mukherjee, "Incorporating Continaation of Message Information, Slot Reuse and Fairness in DQDB Networks", *Computer Networks and ISDN Systems*, 24, 1992, 153–169.

[CGL91] M. Conti, E. Gregori and L. Lenzini, "A Methodological Approach to an Extensive Analysis of DQDB Performance and Fairness", *IEEE Journal on Selected Areas in Communications*, 9, 1 (January 1991), 76–87.

[CiO90] I. Cidon and Y. Ofek, "A Full-duplex Ring with Fairness and Spatial Reuse", INFOCOM'90, 969–981.

[GaL91] M. W. Garret, S. Q. Li, "A Study of Slot Reuse in Dual Bus Multiple Access Networks", *IEEE Journal on Selected Areas in Communications*, 9, 2 (February 1991), 258–256.

[MNW90] H.R. Muller, M.M. Nassehi, J.W. Wong, E. Zurfluh, W. Bux and P. Zafiropulo, "DQMA and CRMA: New Access Schemes for Gbit/s LANs and MANs" INFOCOM'90, 185–191.

Chapter 4

CBRMA++/SR and SP/R: A Simulation Study

4.1 CBRMA++/SR and SP/R: A Simulation Study

In this chapter, the performance of CBRMA++/SR and SP/R will be examined further via simulation. In the performance results presented so far, the assumption of uniform traffic distribution was explicit. This implies a symmetry between senders and receivers which is not necessarily realistic. In real-life applications, existence of special nodes may result in skewed traffic distributions rather than uniform. The basic purpose of this chapter is to investigate the behavior of CBRMA++/SR and SP/R under skewed traffic patterns.

The aforementioned symmetry is directly related to the amount of traffic addressed to a given receiver station out of the traffic generated by a given sender. For an $\mathcal{N}$ node network, the ratio is assumed to be $1/(\mathcal{N}-1)$. In other words, every station is equally likely to be chosen as the sender or receiver of a newly generated packet. If the average amount of traffic generated by each node is the same, unidirectionality of the transmission implies gradually decreasing bandwidth requirements and throughput rates from upstream to downstream nodes on a bus. On the other hand, the aforementioned symmetry can be broken when some special nodes are introduced into the network structure. From now on, we call these nodes *servers* and others *clients*. Since servers form points of interest in the network, some locality in the selection of sender and receiver addresses for a newly generated packet can be expected. This locality can disturb the uniform message distribution in a number of ways:

1. In response to a small amount of data received from a station, a server sends a considerably larger amount of data (e.g., a database server responding to a database search command). In this case, a server acts as a bursty source.
2. In response to a large amount of data received from a station, a server sends a smaller amount of data (e.g., a printing server, acknowledging the successful printing of a large file). In this case, a the server acts as a bursty sink.
3. A station may behave like a combination of (1) and (2), as does a file server.
4. Although the amount of traffic generated by each station can be comparable, the server stations can be referenced more heavily than the other stations. In this case, a server still acts a sink, but its bandwidth requirement is not necessarly smaller than any other station's.

For the simulation studies of this chapter, the network topology remains unchanged. There are 32 stations which are placed at equidistant intervals on the bus. Each station can send and receive on both buses. Amongst many possible cases, we choose to work on the following scenarios:

1. 4 out of 32 stations are assumed to be servers. Their placement is of interest and we will consider three possible cases. In placement pattern 1 (p_1), stations 1, 2, 3, 4 in placement pattern-2 (p_2), stations 15, 16, 17, 18 and in placement pattern 3 (p_3) stations 1, 9, 17, 25 are assumed to be servers. p_1 clusters the servers at the beginning of the forward bus, p_2 clusters them in the middle while p_3 places them at regular intervals on the bus. Note that, in terms of regularity a better placement for p_3 would be 6, 13, 20, 27. However, the arrangement chosen is expected to illustrate the differences of two protocols better.

2. Under uniform traffic pattern, and assuming the traffic load offered by the stations to be comparable, in our network, approximately $4/32 = 12.5\%$ of the traffic is sent/received by the servers. The effect of different ratios on the performance is of interest and will be investigated as follows:

   (a) The average amount of traffic generated by each node is the same. However, clients select a server as the receiver with the probability of .25, .50 or .75 instead of .125. This value will be denoted by α. All servers are equally likely to be selected.

   (b) The ratio of global traffic generated by servers can be changed from 1/8 to 3/8, 5/8 and 7/8. This value will be denoted by β.

   (c) The combined effect of different values of α and β on performance is investigated.

In the simulations, the packet length is constant and a packet fits exactly into a single slot without causing any internal fragmentation. Packet interarrival times are exponentially distributed with a mean normalized to the transmission time of a single slot.

To begin with, we find a traffic load heavy enough to yield so called *saturation throughput* under uniform traffic pattern. Saturation throughput indicates the level of load that the network can handle without causing long queues to build up at *any* station. Then, without modifying this global traffic load, we will change the aforementioned parameters. The effect of each different parameter is to distribute the global load in a different pattern between sender-receiver pairs. Consequently, the flexibility of the protocol to cope with heterogenous traffic patterns can be observed. Ideally, the location of the nodes and the amount of traffic they generate should not effect the performance, since the capacity of the network is not exceeded. Otherwise, the implication of location dependence becomes obvious. Due to slot pre/reuse mechanisms employed by the protocols, effects of location dependence are most likely to emerge. However, the reactions of two protocols may not necessarily be the same.

Figures 4.1 and 4.2 give the throughput of stations under CBRMA++/SR and SP/R protocol respectively for uniformly distributed traffic. The global throughput of CBRMA++/SR with the message interarrival time of 8 slots is 2.71 and of SP/R is 3.49. At this rate, the queues begin to grow steadily in CBRMA++/SR. The growth rate is monotone decreasing from upstream to downstream on each bus and tapers off down to zero towards the middle of the bus. In SP/R only headends suffer from the same effect although at a much slower rate (1/10 compared to that of CBRMA++/SR).

When mean message interarrvial time is increased to 12 and 16 slots, both protocols yield approximately the same global throughput, 2.26 and 1.91 respectively. In both cases, there is no queue build up across the stations and medium access delays also become comparable. To form a basis for comparison, we choose 12-slot mean message interarrival rate (per station) as a close approximation to a fairly heavy load which the network can handle.

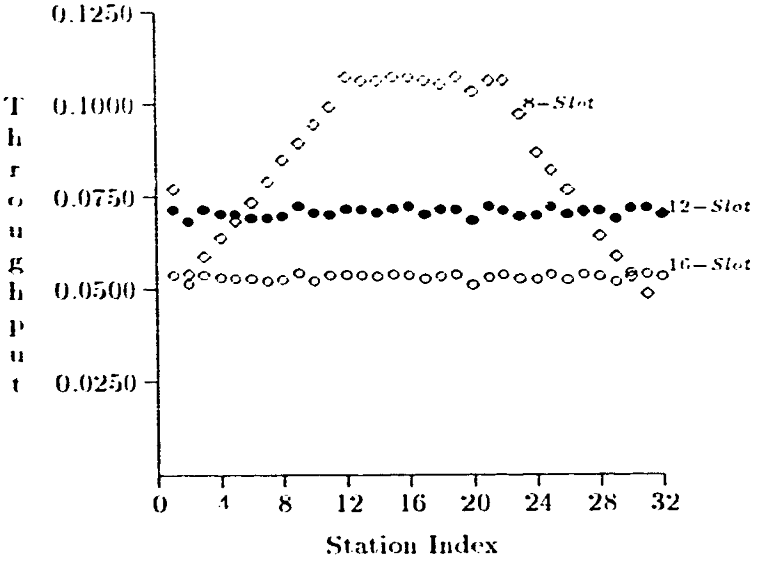


Figure 4.1: CBRMA++/SR

When they act as sources/sinks, the location of servers become increasingly important in terms of average hop count[1] of the network ($\bar{h}$), and therefore, the overall throughput. When $\bar{h}$ gets larger, a slot needs to stay in the network longer. Consequently, available bandwidth decreases along with slot reuse. For bus topology, $\bar{h}$ is not the same across the network: it is approximately twice as great at headends compared to that of a station located in the middle of the bus. In our network, the related values of $\bar{h}$ (in hops) for the server group are calculated as follows:

1. For p_1, $16, 15.03, 14.13, 13.29$ for stations $1, 2, 3, 4$ respectively, giving an average of 14.61.

2. For p_2, $8.32, 8.25, 8.25, 8.32$ for stations $15, 16, 17, 18$ respectively, giving an average of 8.28.

3. For p_3, $16, 10.06, 8.25, 10.58$ for stations $1, 9, 17, 25$ respectively, giving an average of 11.22.

Therefore, all else being equal, p_2 is expected to give better results as the value of α and/or β is increased.

Figures 4.3–4.11 demonstrate the interaction between the location of servers and different values of α. The mean message interarrival time is 8 slots for all stations. Servers are equally likely (amongst their group) to be addressed by other stations. Servers address the other stations with equal probability, regardless of the value of α. When the likelihood of servers being selected as the receivers by other stations (i.e., value of α) is increased, bandwidth requirements of the stations on the reverse bus increase. That is to say, global load is no longer equally divided between two buses.

Figures 4.3–4.5 give the throughput of stations in which servers located according to p_1 for $\alpha = .25$, $\alpha = .50$ and $\alpha = .75$, respectively[2] where $\beta = 1/8$.

For $\alpha = .25$, the total throughput rates across the stations are not different from those given in figure 4.1 and 4.2 for CBRMA++/SR and SP/R, respectively. However, their throughput on the forward bus decrease. The difference is compensated by the increase of the throughput on the reverse bus. This shows that our estimation for the saturation throuhput rate was slightly pessimistic, since there is still some bandwith available on the backward bus. However, as more and more traffic

[1] Refer to chapter 5 for details.

[2] In the following figures, ○ denotes the values associated with CBRMA++/SR and ● denotes the values associated with SP/R invariably.

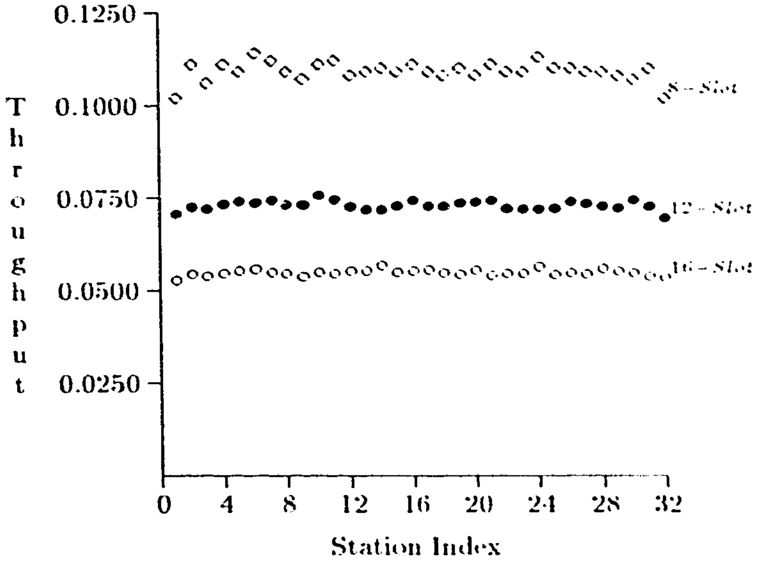


Figure 4.2: SP/R

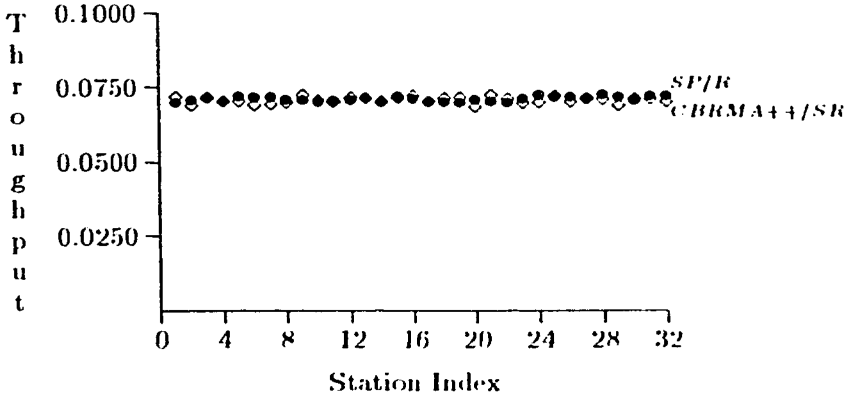


Figure 4.3: Servers:1,2,3,4, α = .25.

is transferred from the forward to backward bus (i.e., the value of α is increased), the adverse effects become observable, as seen in figures 4.4 and 4.5. The gradual decrease of throughput of CBRMA++/SR is partly the outcome of the decreased traffic activity on the forward bus. The other reason is the increasing throughput rates on the reverse bus from station 32 to 1, due to slot reuse. The results reveal the absence of any fairness mechanism regarding the slot reuse. Otherwise, stations are served at least as much as their fair shares and there is no station exposed to the danger of starvation.

SP/R behaves quite differently. Since the stations on the lower half of the reverse bus have no explicitly allocated slots, their throughput decreases rapidly so that, for stations 15 5, the throughput on both buses become practically the same. From station 15 toward station 5 the available bandwidth gets slightly larger, due to the increased probabilty of being addressed on the reverse bus, but not significantly (figure 4.5). On the other hand, throughput of stations 18 32 seem to be uneffected, even for α = .75. These stations have the same throughput on the forward bus for both protocols and therefore, the difference is due to their performance on the backward bus. As the demand for bandwidth increases, CBRMA++/SR distributes the available bandwidth more fairly

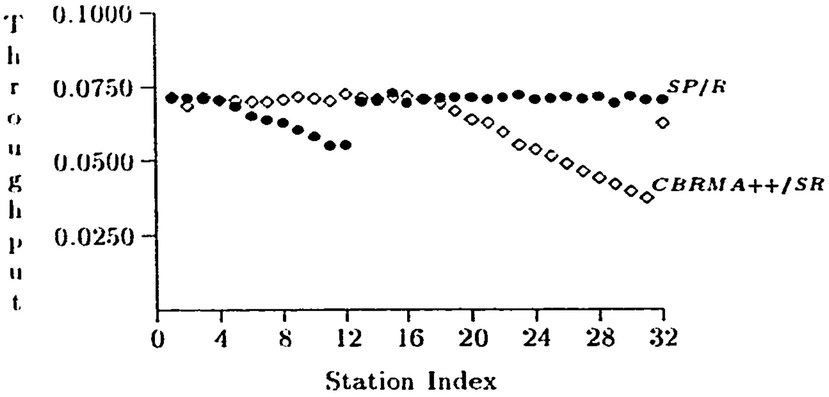


Figure 4.4: Servers:1,2,3,4, α = .50.

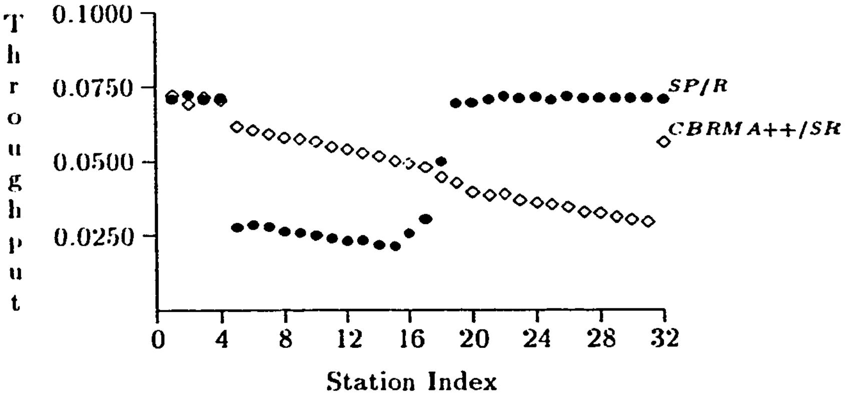


Figure 4.5: Servers:1,2,3,4, α = .75.

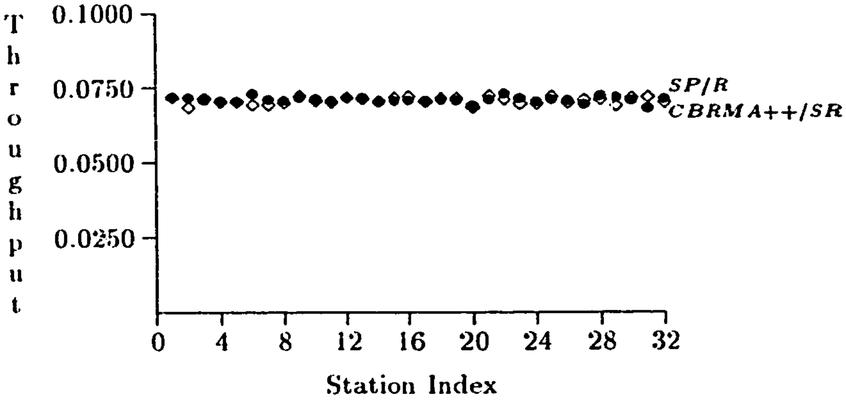


Figure 4.6: Servers:15,16,17,18, α = .25.

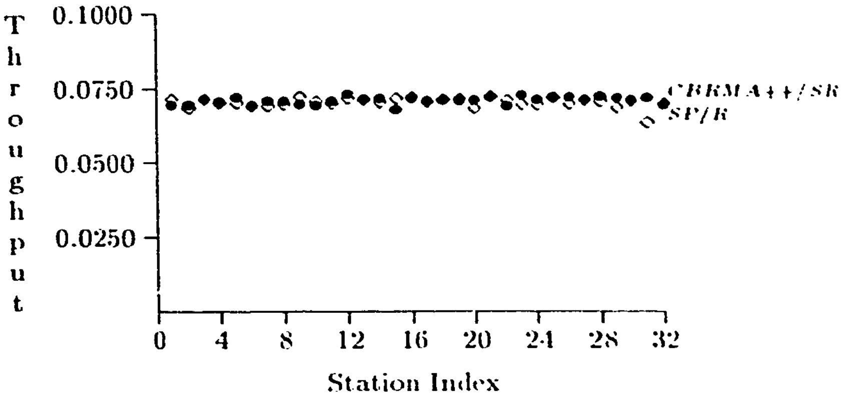


Figure 4.7: Servers:15,16,17,18, α = .50.

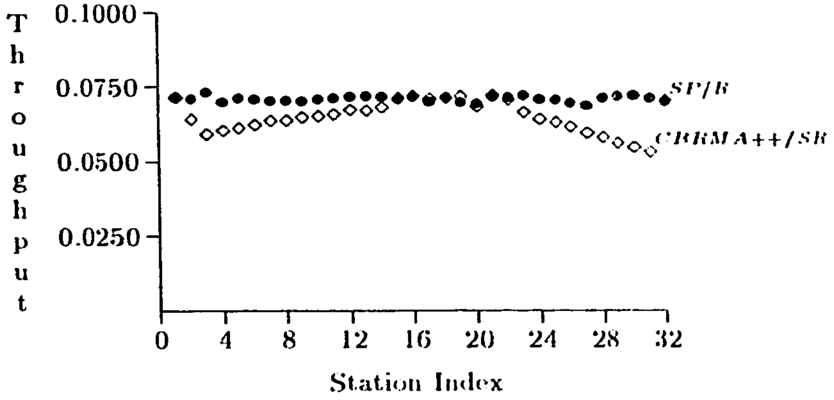


Figure 4.8: Servers:15,16,17,18, α = .75.

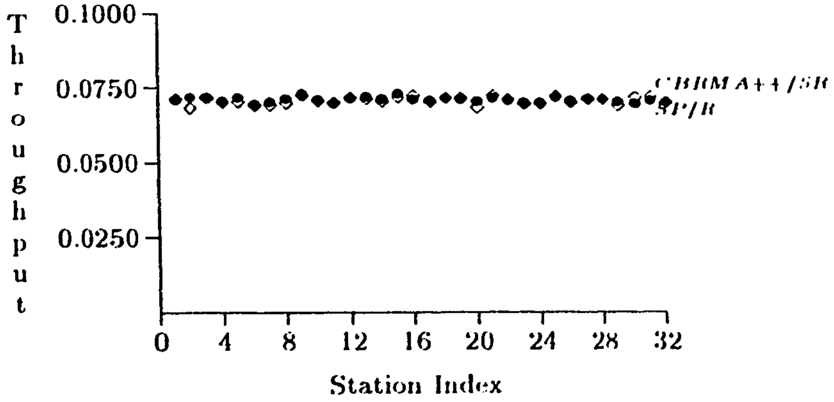


Figure 4.9: Servers:1,9,17,25, α = .25.

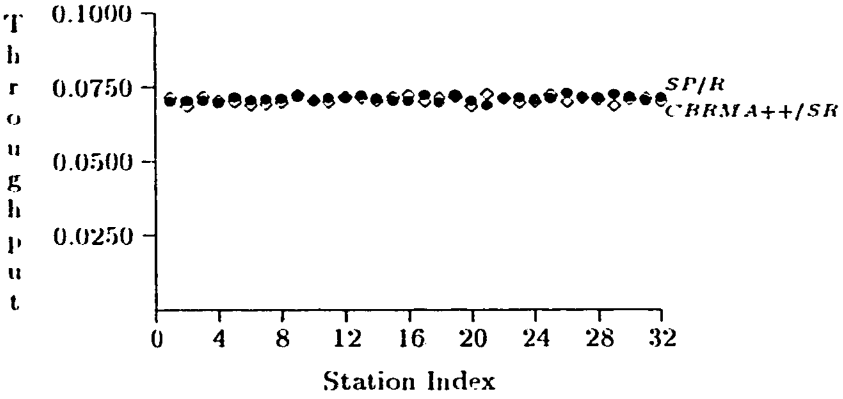


Figure 4.10: Servers:1,9,17,25, $\alpha = .50$.

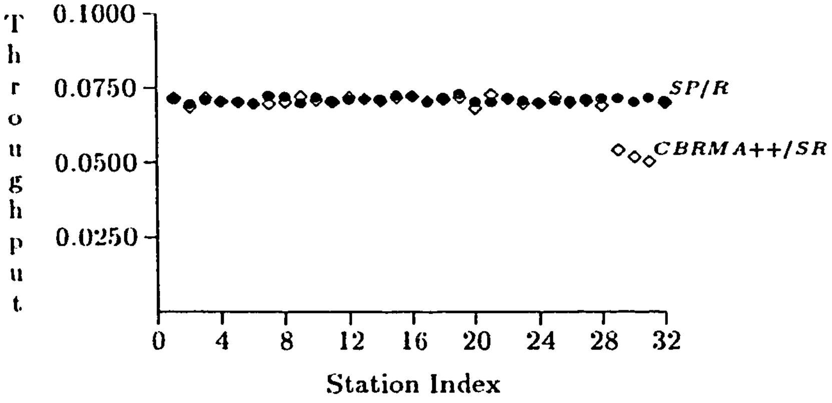


Figure 4.11: Servers:1,9,17,25, $\alpha = .75$.

compared to SP/R. On the other hand, SP/R provides a better service to a smaller number of stations. Note that, when α is changed from .50 to .75, the quality of the service provided to stations 8–17 deteriorates rapidly. In other words, stations 18–32 are served at their expense. Consequently, SP/R is more sensitive to the placement of the stations that generate and/or absorb a large portion of the network traffic.

For $\alpha = .50$ and $\alpha = .75$, the amount of offered traffic on the backward bus exceeds the available capacity and consequently, queues begin to grow at the stations. Other than the extra banwidth requests, the increased average internodal distance also compounds the problem. However, the location of the long queues are not the same. For $\alpha = .50$, in CBRMA++/SR queue lengths decrease significantly from station 31 to 22. Station 32 is not affected due to its double fair share. In SP/R, increasing queue lengths are observable for stations 14–5. For $\alpha = .75$, the adverse effect turns out to be global and there is no unaffected station.

When servers are located according to p_2 (figures 4.6–4.8) or according to p_3 (figures 4.9–4.11), both protocols perform considerably better. For $\alpha = .50$, long queues completely disappear at the stations. For $\alpha = .75$, SP/R performs better. In CBRMA++/SR, the effect of the unfair slot reuse becomes more observable while SP/R takes full adventage of slot preuse. It is also interesting that the behaviour of CBRMA++/SR is not identical on forward and backward buses for the case of p_3 and $\alpha = .75$. Stations 31–26 have lower throughput rates for two reasons: firstly, the amount of traffic sent on the forward bus by stations 2–8 and 32–26 is not the same. The latter group hardly uses the forward bus since there is no other server located after station 25. Secondly, the number of slots available for slot reuse is not significant since the total bandwidth requirement is larger on the backward bus. For example, server 1 is addressed by 31 stations on the backward bus while server 25 is addressed by 24 stations on the forward bus. Consequently, for the case of p_3 and $\alpha = .75$, queue lengths steadily increase at stations 31–26 on the backward bus. Station 32 is not effected due to its double fair share. The results presented in figures 4.3–4.11 are summarized in the following table.

		CBRMA++/SR			*SP/R*		
p	α	*Total*	*Forward*	*Backward*	*Total*	*Forward*	*Backward*
1	0.25	2.25	1.01	1.24	2.27	1.01	1.25
	0.50	2.00	0.76	1.24*	2.18	0.76	1.42*
	0.75	1.55	0.51	1.04*	1.65	0.51	1.14*
2	0.25	2.26	1.13	1.13	2.27	1.135	1.135
	0.50	2.25	1.13	1.12	2.27	1.13	1.14
	0.75	2.08	1.04*	1.04*	2.27	1.14	1.13
3	0.25	2.26	1.10	1.16	2.27	1.11	1.16
	0.50	2.25	1.03	1.22	2.27	1.04	1.23
	0.75	2.20	0.976	1.22*	2.27	0.98	1.29

(* indicates the existence of at least one station with steadily growing packet queue on the relevant bus.)

Figures 4.12–4.20 show the interactions between different server placement patterns and different values of β. The mean message interarrival times are no longer the same for clients and servers. Servers generate 37.5%, 62.5% and 87.5% of the global traffic for the cases of $\beta = 3/8$, $\beta = 5/8$ and $\beta = 7/8$, respectively. Servers and clients are equally loaded amongst themselves. Consequently, the mean message interarrival times are 4, 2.4, 1.7 for servers and 16.8, 28, 84 for clients for the cases of $\beta = 3/8$, $\beta = 5/8$ and $\beta = 7/8$, respectively. In these scenarios, $\alpha = 1/8$, every station is equally likely to be addressed and therefore there is no station that acts as a sink for a considerable portion of the network traffic.

In figures 4.12–4.14, servers are clustered at the beginning of the forward bus. This placement illustrates a weakness of SP/R. In SP/R, station 1 cannot use all the available bandwidth even when it is the only active station, since slot preuse is not allowed at headends. In CBRMA++/SR, this

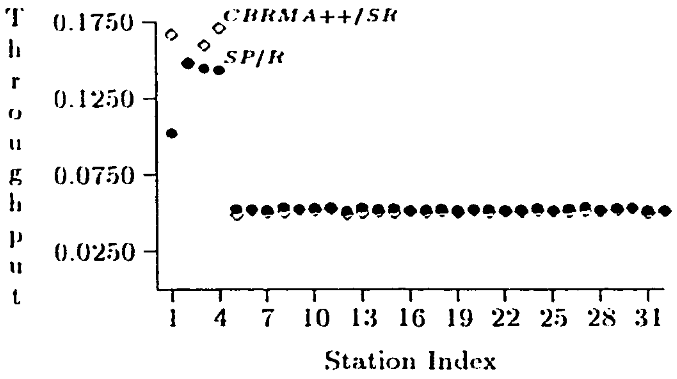


Figure 4.12: Servers:1,2,3,4, $\beta = 3/8$.

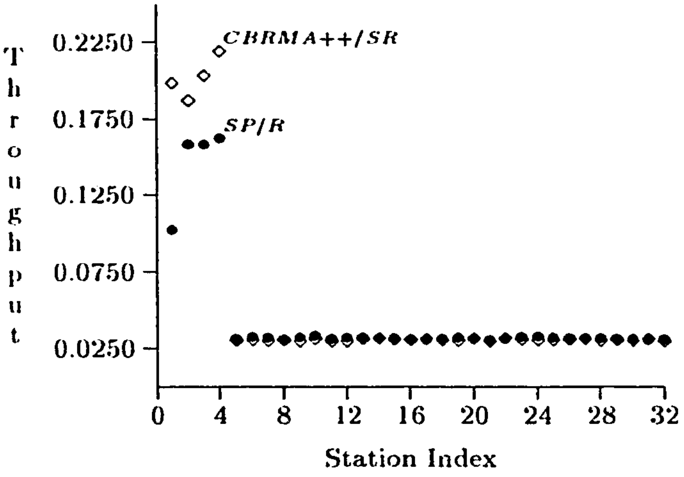


Figure 4.13: Servers:1,2,3,4, $\beta = 5/8$.

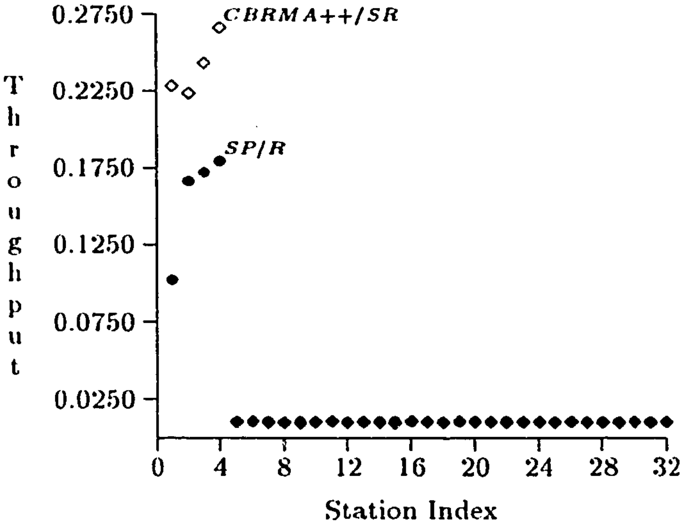


Figure 4.14: Servers:1,2,3,4, $\beta = 7/8$.

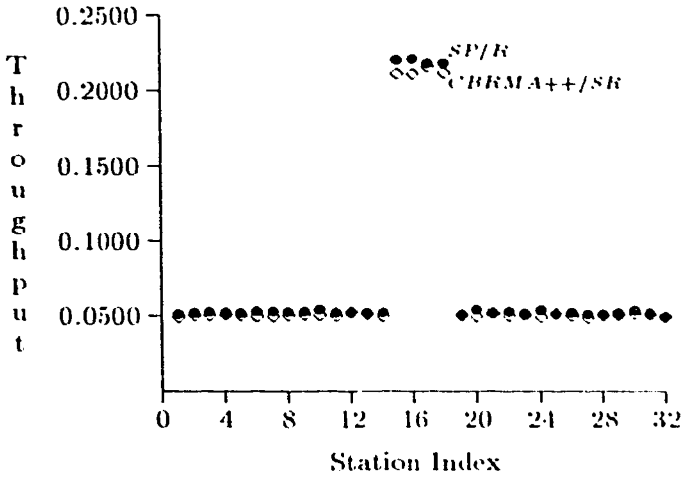


Figure 4.15: Servers:15,16,17,18, $\beta = 3/8$.

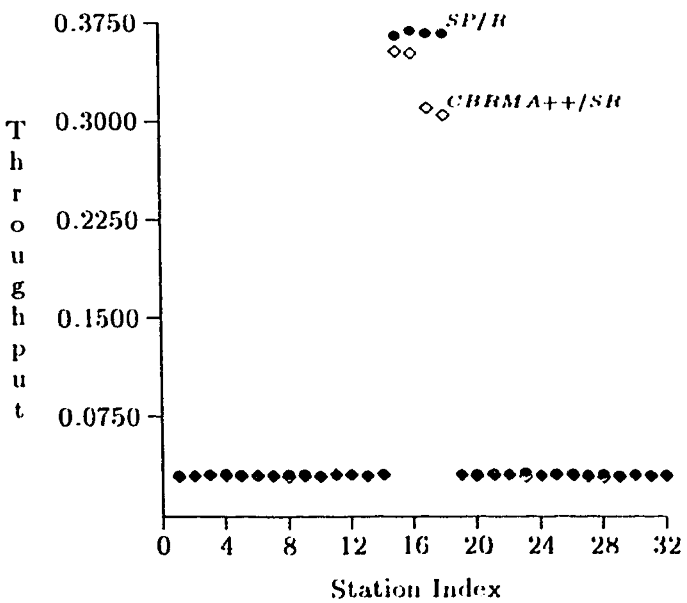


Figure 4.16: Servers:15,16,17,18, $\beta = 5/8$.

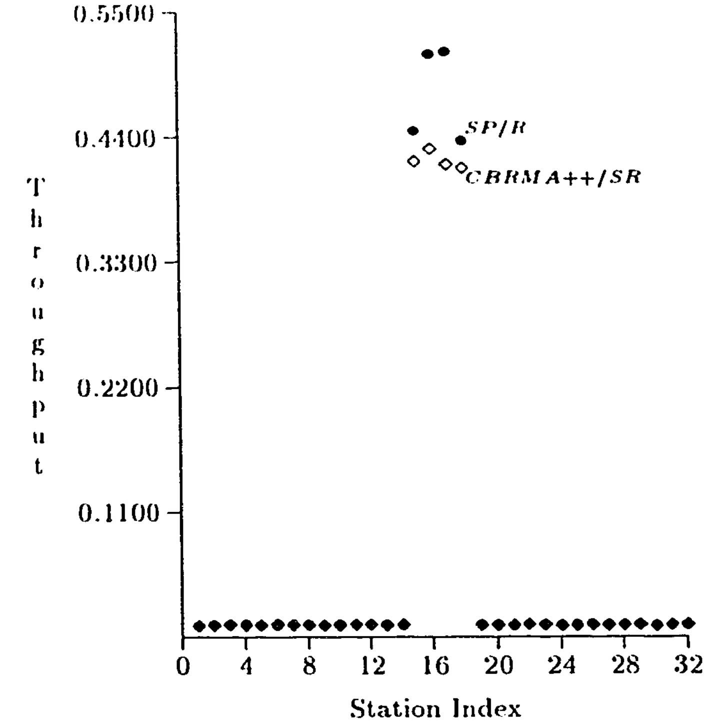


Figure 4.17: Servers:15,16,17,18, $\beta = 7/8$.

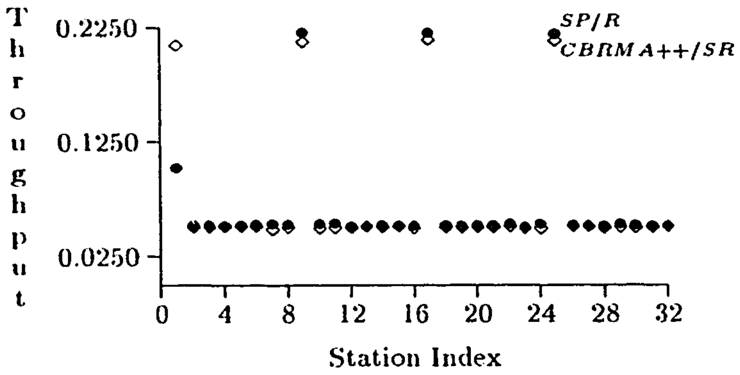


Figure 4.18: Servers:1,9,17,25, $\beta = 3/8$.

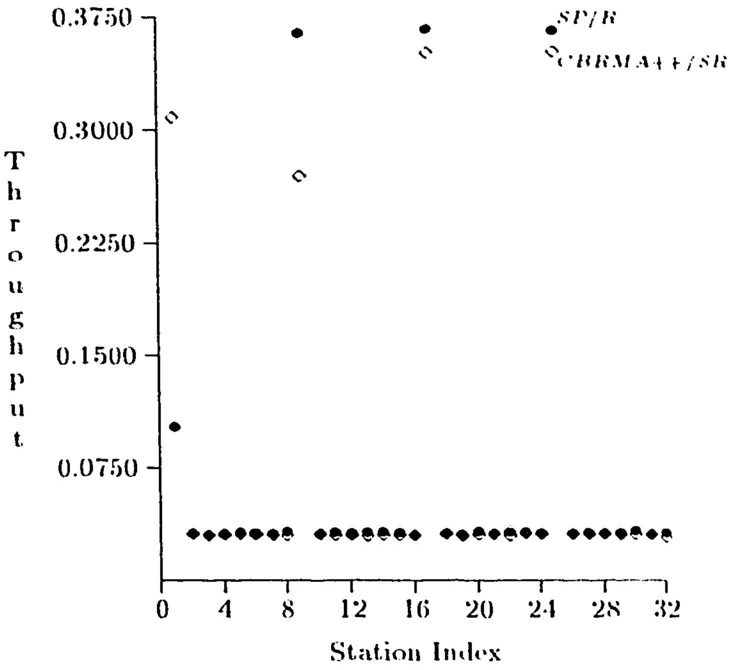


Figure 4.19: Servers:1,9,17,25, $\beta = 5/8$.

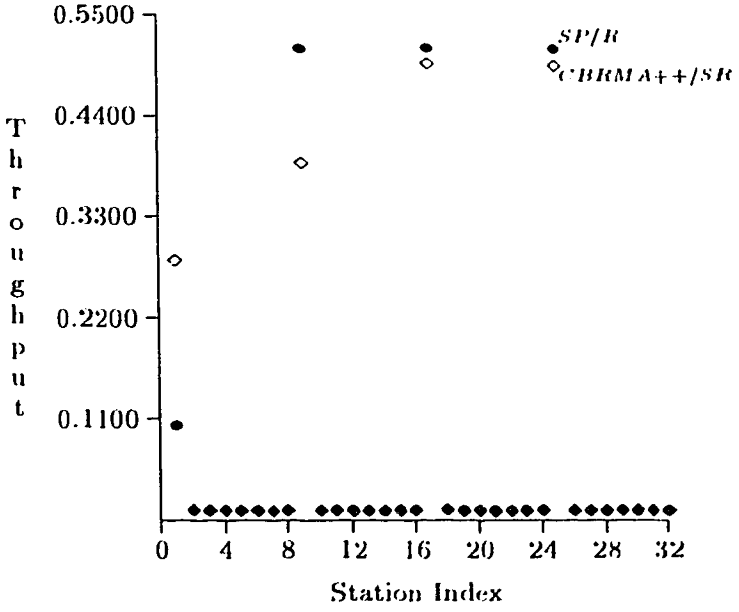


Figure 4.20: Servers:1,9,17,25, $\beta = 7/8$.

problem is solved in the balancing half-cycle. Although, global performance of CBRMA++/SR is better, queues steadily grow longer under both protocols. For $\beta = 7/8$, the growth is almost twice as fast compared to that of CBRMA++/SR at station 1.

When servers are clustered in the middle (figures 4.15–4.17) or placed at equidistant intervals (figures 4.18 4.20), both protocols yield better performance results. In CBRMA++/SR, there are no unstable queues at the servers for $\beta = 3/8$. In SP/R, only at station 1, the packet queue on the forward bus grows longer under p_3. For higher values of β, fast growing queues are constantly observed for both protocols. In CBRMA++/SR:

1. at stations 17 (on forward bus) and 18 (on both buses) for $\beta = 5/8$ and p_2,
2. at stations 15 (on forward bus), 16 and 17 (on both buses), 18 (on backward bus) for $\beta = 7/8$ and p_2,
3. at stations 1 and 9 (on forward bus) for $\beta = 5/8$ or $\beta = 7/8$ and p_3.

In SP/R:

1. at stations 15 (on backward bus) and 18 (on forward bus) for $\beta = 7/8$ and p_2,
2. at station 1 for $\beta = 3/8$, $\beta = 5/8$, $\beta = 7/8$ and p_3.

When servers are located in the middle, long queues are not observed under SP/R, contrary to the performance of CBRMA++/SR for $\beta = 5/8$. The location of trouble spots for $\beta = 7/8$ and p_2 is obvious: station 13 being the last server on the forward bus gets the worst service while the same is true for station 15 on the backward bus.

The behaviour of CBRMA++/SR is more complicated. For p_1, the performance difference between servers is only related to the number of reused slots. Obviously, this number increases from station 1 to 4 and so are the throughput rates. When servers are located in the middle, stations 17 and 18 achieve lower throughput rates compared to that of stations 15 and 16 (figure 4.16). The reason is the behaviour of the protocol during balancing half-cycle. Consider the following example.

Assume that, stations $2, 4, 15, 16, 17, 18$ requested $1, 1, 30, 30, 30, 30$ extra slots for transmission on the forward bus during reservation. There are 112 slots requested by 6 stations in addition to their fair share. Also assume that 79 slots are marked as unused. In the balancing process, the stations act as follows:

1. *Station 18*: Using the variable names from the description of the algorithm, *fair* $= \lfloor 79/6 \rfloor = 13$, ***othersneed*** $= 92$, ***difference*** $= 79 - 92$ and therefore negative. Consequently, station 18 claims 13 slots leaving 66 unused slots.
2. *Station 17*: ***fair*** $= \lfloor 66/5 \rfloor = 13$, ***othersneed*** $= 62$, ***difference*** $= 66 - 62$ and therefore less than ***fair***. Consequently, station 17 claims 13 slots leaving 53 unused slots.
3. *Station 16*: ***fair*** $= \lfloor 53/4 \rfloor = 13$, ***othersneed*** $= 32$, ***difference*** $= 53 - 32 = 21$ and therefore greater than ***fair***. Consequently, station 16 claims 21 slots leaving 32 unused slots.

At this step, the number of unused slots and the extra slot requests become equal. At the end, the unused 79 slots are allocated as $1, 1, 30, 21, 13, 13$ amongs the stations $2, 4, 15, 16, 17, 18$, respectively. When the value of β is increased to 7/8, small extra bandwidth requests of the client stations disappear and so do the differences amongst the throughput rates of the servers (figure 4.17).

In figures 4.19 and 4.20, the performance of station 9 deserves some comment, since it is the lowest amongst servers in figure 4.19 and becomes higher than station 1 but remains lower than the other two servers in figure 4.20. In the scenarios that these figures illustrate, there are fast growing queues at stations 1 and 9 on forward bus. The reason for this occurence is that: although the number of slots allocated to these two stations is greater on the forward bus, the number of packets

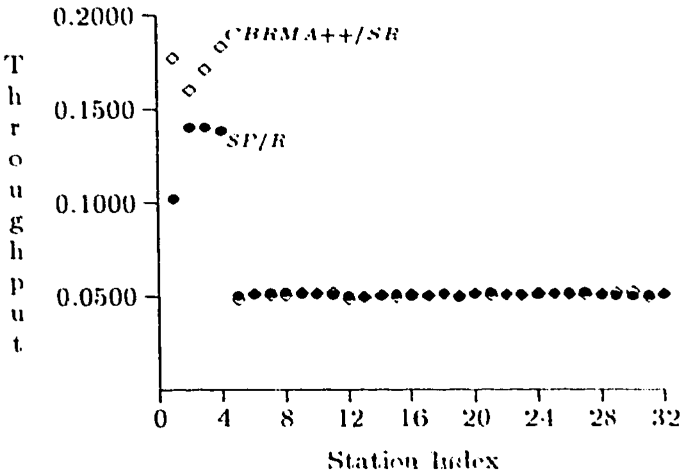


Figure 4.21: Servers:1,2,3,4, $\alpha = .25$, $\beta = 3/8$.

that need to be transmitted on the forward bus exceeds the allocated capacity. Station 1 benefits more from the imbalance in the distribution of unused slots compared to station 9, as mentioned in the preceeding example. Compounded with its double fair share, station 1 achieves a higher throughput rate than station 9. As servers become more loaded, this unfairness disappears and station 1 is only granted its double share. Consequently, more bandwidth becomes available to the other three servers on the forward bus. Moreover, station 9 makes better use of the backward bus due to increased message generation rate. Obviously, the packet queue on the forward bus grows much faster. The results presented in figures 4.12–4.20 are summarized in the following table.

		$\mathcal{CBRMA}++/\mathcal{SR}$			$\mathcal{SP}/\mathcal{R}$		
p	β	*Total*	*Forward*	*Backward*	*Total*	*Forward*	*Backward*
1	3/8	2.06	1.22*	0.839	1.99	1.11*	0.88
	5/8	1.65	1.11*	0.548	1.45	0.865*	0.59
	7/8	1.24	0.987*	0.256	0.91	0.607*	0.303
2	3/8	2.26	1.13	1.13	2.34	1.17	1.17
	5/8	2.17	1.07*	1.10*	2.34	1.17	1.17
	7/8	1.96	0.98*	0.976*	2.20	1.10*	1.10*
3	3/8	2.26	1.20	1.06	2.22	1.13*	1.09
	5/8	2.13	1.15*	0.98	2.08	1.05*	1.03
	7/8	1.94	1.03*	0.91	1.93	0.985*	0.947

For a given α, β pair, the graph is quite similar to the graphs parametric in β. For example, compare the figures 4.3 and 4.12 with figure 4.21. As β is increased, the amount of traffic generated by clients decreases and the effect of α becomes less important on the behaviour of the network. Since the traffic generated by servers is uniformly distributed, unfairness due to the high values of α tends to become less severe. For example, compare figures 4.5 and 4.12 with figure 4.22. Conversely, higher values of α contributes to the number of slots available for reuse at servers. Since each server is equally likely to be addressed by the clients, the fairness in slot reuse is improved. This effect is more noticable in SP/R. For example, compare figures 4.8 and 4.17 with figure 4.23. The results are summarized in the following table.

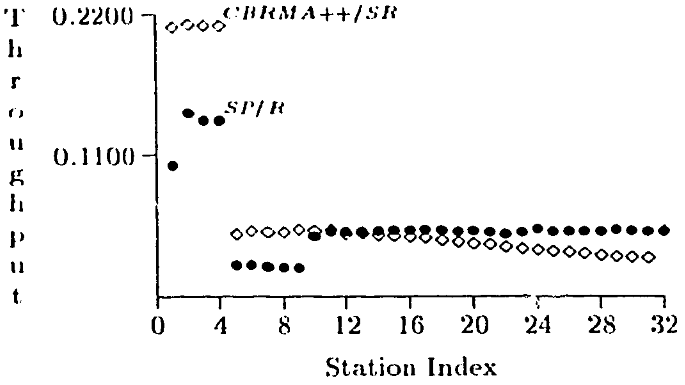


Figure 4.22: Servers:1,2,3,4, $\alpha = .75$, $\beta = 3/8$.

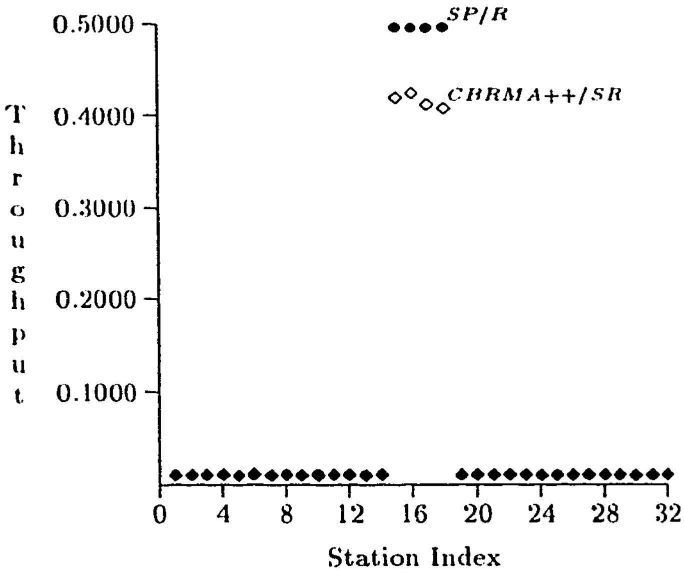


Figure 4.23: Servers:15,16,17,18, $\alpha = .75$, $\beta = 7/8$.

			CBRMA++/SR			*SP/R*		
p	α	β	*Total*	*Forward*	*Backward*	*Total*	*Forward*	*Backward*
1	0.25	3/8	2.10	1.18*	0.92	1.94	1.01*	0.93
		5/8	1.68	1.08*	0.60	1.40	0.80*	0.60
		7/8	1.25	0.98*	0.27	0.86	0.59*	0.27
	0.50	3/8	2.20	1.10*	1.10	1.94	0.83*	1.11
		5/8	1.73	1.03*	0.70	1.40	0.70*	0.70
		7/8	1.27	0.97*	0.30	0.85	0.54*	0.31
	0.75	3/8	2.03	0.98	1.05*	1.79	0.66*	1.13*
		5/8	1.79	0.98*	0.81	1.39	0.58*	0.81
		7/8	1.29	0.95*	0.34	0.85	0.51*	0.34
2	0.25	3/8	2.26	1.13	1.13	2.27	1.13	1.14
		5/8	2.21	1.09*	1.12	2.27	1.13	1.14
		7/8	1.95	0.98*	0.97*	2.20	1.10*	1.10*
	0.50	3/8	2.26	1.13	1.13	2.27	1.13	1.14
		5/8	2.26	1.13	1.13	2.27	1.13	1.14
		7/8	1.95	0.975*	0.974*	2.24	1.12*	1.12+
	0.75	3/8	2.26	1.13	1.13	2.26	1.13	1.13
		5/8	2.26	1.13	1.13	2.27	1.13	1.14
		7/8	1.95	0.974*	0.973*	2.27	1.13	1.14
3	0.25	3/8	2.26	1.18	1.08	2.16	1.08*	1.08
		5/8	2.14	1.14*	1.00	2.02	1.02*	1.00
		7/8	1.95	1.03*	0.92	1.87	0.95*	0.92
	0.50	3/8	2.26	1.14	1.12	2.16	1.03*	1.13
		5/8	2.17	1.15*	1.02	2.02	0.99*	1.03
		7/8	1.95	1.03*	0.92	1.88	0.95*	0.93
	0.75	3/8	2.26	1.10	1.16	2.16	0.98*	1.17
		5/8	2.19	1.14*	1.05	2.02	0.97*	1.05
		7/8	1.96	1.03*	0.93	1.87	0.94*	0.93

4.2 Conclusions

In this chapter, some skewed traffic patterns are introduced and behaviours of CBRMA++/SR and SP/R are demonstrated under these traffic patterns. In general, CBRMA++/SR has a more fair mechanism for the distribution of available bandwidth compared to SP/R. SP/R has a greedier approach and may sacrifice fairness for efficiency. In this regard, its behaviour is similar to that of Metaring and therefore needs to be flow-controlled when network traffic is highly skewed and the locations of bursty stations are not known. Otherwise, both protocols have mechanisms to provide server stations with extra bandwidth, if need be. This can be done by setting *fair share* in CBRMA++/SR and by setting *Owner Id* fields of slots in SP/R.

Part II

Routing Protocols for High–Speed WANs

Chapter 5

Routing in Multihop Packet Switching Networks: Gbps Challenge[1]

A packet-switching WAN is not just a bigger packet switching LAN and the *modus operandi* of the latter is not necessarily applicable to the former. A similar distinction exists also between high speed packet-switching WANs/MANs and their slower counterparts. Consequently, there is the need for new routing protocols and congestion and flow control mechanisms. This chapter is a survey aimed to discuss the aforementioned "differences" in relation to the design of routing protocols for high speed, multihop, packet-switching networks. However, such a discussion cannot be conducted in isolation since, in the design of a routing mechanism, three main building blocks can be distinguished, not only as independent entities but also as closely interacting components. Therefore, a successful design should carefully consider the impact of one entity on the others, as well as their individual characteristics that will eventually determine the quantity and the quality of the service that the network can provide to the upper layer protocols. These entities are:

1. Routing protocol.
2. Congestion control mechanisms that can be effectively incorporated into the routing protocol.
3. Network topology.

Due to this coupling, this chapter is organized as follows: first, routing protocols and congestion control mechanisms, that are in practice as of today, are discussed briefly. After providing some basic definitions related to network topology, the challenges posed by the Gbps networks are investigated. The last section is devoted to case studies in which a number of contemporary design proposals are presented. The scope is restricted to multihop packet-switching networks. The term *multihop* refers to networks in which traffic is routed by intermediate nodes. Furthermore, transmission is assumed to be slotted with the exception of two flooding networks, namely, Arbnet+ and Noahnet.

5.1 Routing Protocols

In a multihop, packet-switching network, the function of the routing algorithm is to guide the packets through the communication subnet to their correct destinations. Different taxonomies of routing algorithms are possible: ***static*** vs. ***adaptive***, ***centralized*** vs. ***distributed***, etc. Here we prefer to classify

[1] A version of this chapter has been submitted for publication to *IEEE Network Magazine*.

them into two groups, namely, ***table-based-routing*** and ***self-routing***[2]. The first one covers most of the traditional approaches which have been applied to many slow networks, including ***shortest path routing*** (shortest paths can be computed using different algorithms e.g., Bellman-Ford, Dijkstra, Floyd-Warshall) and ***optimal routing*** (feasible direction methods, projection methods). Aside from other problems (convergence delay which is proportional to the network diameter, susceptibilty to oscillations etc.), these algorithms are computationally expensive and require a lot of bookeeping and periodic transmission of some status information amongst the nodes. In the case of self-routing, the routing decision is solely made upon some routing information extracted from the packet's header which is generally the destination address itself. Most of the multiprocessor interconnection networks use this scheme (e.g., shuffle-exchange networks, hypercube, data manipulator networks, Benes networks, Clos networks). Another well-known example that can be included in this category is ***flooding***. This classification is aimed to group the routing algorithms according to their packet switching speed[3]. Next we will briefly discuss some of these routing algorithms.

5.1.1 Table-Based Routing

In this category, a routing table is consulted to select the outgoing link upon which an incoming packet is to be forwarded. Although the location of these tables, the way they are maintained and the information contained within them can differ from one implementation to another, there are some common characteristics they all share:

1. In most applications, the routing table contains an entry for every destination indicating the proper output link. Therefore, the table size increases with the network size and can be very large for a network with many nodes.
2. For practical reasons (e.g., to be able to cope with congestion and faulty links), the table entries need to be updated. Therefore, some network capacity should be periodically allocated to the extra traffic that disseminates status information, reducing the capacity available to the users.

Because of these commonalities, we choose to group shortest-path and optimal routing algorithms together. Now we proceed to discuss them further.

Shortest-Path Routing

The basic idea is to have a routing table that indicates the output link that lies along the path with the minimum distance or minimum hop count and/or relatively uncongested links for every destination address. If the minimization operation is based on a dynamic criteria (such as link lengths adjusted according to the prevailing congestion level upon them), then the routing algorithm becomes more adaptive to the everchanging traffic conditions. This mechanism requires global topological knowledge, i.e., a list of all nodes in the network and their interconnections as well as a cost for each link [Sch87]. In practice, both centralized (for Tymnet) and distributed (for Arpanet) versions of it are implemented. Tymnet uses virtual circuits and the shortest path calculations are performed by a special node called ***supervisor***. Supervisor also decides upon the paths to be used by a virtual circuit. The intermediate nodes are informed about the path by a ***needle*** packet that travels from the origin to the destination threading the virtual circuit as it moves through the network with user data trailing behind. The shortest-path calculation algorithm used by the supervisor is a modified form of the Floyd's algorithm [Sch87].

[2] Also known as header-routing

[3] Another important factor regarding the switching speed is the existence of buffers. When there are no buffers, the use of photonic switches becomes possible and E/O and O/E (optic-to-electronic) conversions can be avoided. However, the photonic switching technology is still in its infancy and for the time being is not a practical alternative to its electronic counterpart. Its current status discussed in [JaM93].

Arpanet employs a distributed approach in which every node maintains its global database and carries out the shortest path calculations taking itself as the root. The original algorithm, based on Bellman–Ford method, was implemented in 1969[4]. It has been modified twice since then due to the problems caused by oscillations, in 1979 and 1987. The latest modification was warranted by the increased traffic load which once again lead to severe oscillations. The newest algorithm is still prone to oscillations, but not nearly as much as the first [BeG92]. The details of the aforementioned algorithms can be found in [CLR90].

The main shortcomings of the shortest–path algorithms are the use of only one path per source destination pair and their poor adaptability to abrupt traffic shifts which is limited by their susceptibility to oscillations [BeG92].

Optimal Routing

Optimal routing is based on the theory of optimal multicommodity flows. With an appropriate[5] cost function, it is possible to express the routing problem as an optimization problem. The typical cost functions are related to link capacities and the amount of traffic carried by each link which is also termed as a *flow*. Although the basic premise, i.e., achieving good routing solely by optimizing the average levels of link traffic, is somewhat strong, theoretically more effective alternatives (such as the ones that take queue lengths into consideration as well) are impractical due to the overhead and delays involved in the transfer/exchange of the queue length information amongst the nodes [BeG87].

For example, optimal routing is used in CODEX network with the following cost function[6]:

$$\mathcal{Z}_{ij}(\mathcal{F}_{ij}) = \frac{\mathcal{F}_{ij}}{\mathcal{C}_{ij} - \mathcal{F}_{ij}} + d_{ij}\mathcal{F}_{ij} \tag{5.1}$$

where $\mathcal{Z}_{ij}$ is the cost function of each *link(i,j)*, $\mathcal{C}_{ij}$ is the link capacity, $\mathcal{F}_{ij}$ is the data rate of the link and d_{ij} is the processing and propagation delay. CODEX network uses virtual circuits for user traffic and datagrams for its own system messages. Every node monitors some parameters of its adjacent links and periodically broadcasts them to all other nodes. The given formula is valid when all the links are of the same priority. For multilevel priorities and other details see [BeG87] and the references therein.

5.1.2 Self–Routing

Routing in Multiprocessor Interconnection Networks

The vast majority of the designs for implementing ATM switches is based on interconnection networks. Interconnection networks can be constructed from a single stage of switches or multiple stages of switches. In a single–stage network[7], packets may have to be passed through the switches several times before reaching their final destinations. In a multi stage network, generally one pass through the multiple stages of switches is sufficient to transfer the packets to their final destinations

[4] In this version, the nodes exchanged their estimated shortest distances to every destination in every 625 msec.

[5] Providing that a function is sufficiently differentiable, it can be expanded as a Taylor series. If the first derivatives exist, then at local minima the Jacobian gradient vector has all elements zero. If the second derivatives exist, the Hessian is positive definite at the minimum. For convex functions the local minima are also global. The gradient methods are based on the Taylor series expansion. Optimization methods which use only Jacobian gradient vector are termed *first order methods*. If the optimization method utilizes the second dervatives as well, it is termed as *a second order method*. The *steepest descent method* uses Jacobian gradient to determine a suitable direction of movement and is the fundamental first order method. All in all, the appropratie choice of the cost function greatly simplifies the optimization process. For details, see [AdD74, Tah82, PSU88].

[6] The cost function requires $\mathcal{C}_{ij} > \mathcal{F}_{ij}$.

[7] Also called recirculating network. The number of recirculations depens on the connectivity. In general, the higher is the connectivity the fewer recirculations.

[SiH88]. A survey of switching techniques in high-speed networks can be found in [OSM90]. A more comprehensive reference is [Hui90].

Interconnection networks is a well developed area and, with many books [Bae80, HwB84, Sto87, Sie90] and surveys papers [Fen81, ReG87, YaA87, SiH88] available, we will not attempt another introduction here. For the discussion of hypercube and shuffle-like networks refer to section 6.1 and 6.2, respectively.

Flooding Networks

Some networks use a flooding protocol to route messages. Flooding in its purest sense means that an incoming packet is to be forwarded on every outgoing link except the one it arrived on [BeG87]. The outstanding qualities of flooding can be summarized as follows:

1. The approach is highly robust in case of link failures.
2. Error recovery at the destination is simplified by the availability of extra copies of the same packet.
3. If the network is richly connected, flooding has the property of making excellent use of alternative routes.
4. Network modification can be made on a live network.
5. The algorithm is suitable not only for regular but also for arbitrary topologies, therefore enabling the network to be highly expandable.
6. Flooding always chooses the shortest path (since it chooses every possible path in parallel).
7. It is simple to implement and introduces less processing overhead than any other routing scheme.
8. No routing tables are necessary and no bookkeeping is required.
9. The approach is quite suitable to be implemented directly in hardware.

The most important weakness of flooding is that packets may loop and, as a result, unlimited copies of a single packet can be produced. Therefore some countermeasures[8] to choke this process are necessary for the approach to be useful. In general, flooding is considered to be more useful in broadcasting rather than one to one communication. Arpanet uses flooding to broadcast periodic status information to the nodes. In the case studies section (section 6), we will introduce some of the designs that use flooding.

5.2 Congestion Control

Congestion is the network state where either because of mismanagement (e.g., improper access and routing), excessive requests, or faults, the demand for resources exceeds the available capacity [KAS91]. In that case, the queue sizes at bottleneck nodes grow indefinitely and eventually exceed the available buffer space. Consequently, some packets will have to be discarded and later retransmitted thereby wasting communication resources. It is thus necessary to prevent excess traffic from entering the network. Also, due to speed disparity and/or temporary lack of resources, there are moments when the receiver can not accept the incoming flow. Should this be the case, the sender must be

[8] Such a mechanism may require some bookkeeping; e.g., a node records the id of the packets that it has relayed, so as not to resend a copy of a packet which loops back.

made aware of the situation as soon as possible and either adjust its speed or abstain from any further transmission which is bound to be rejected.

Congestion control is a dynamic problem and cannot be solved with static mechanisms alone. It is also a difficult problem to solve due to a set of requirements associated with a possible solution, which are stated in [Jai90] as follows:

1. The scheme must have a low overhead and should not increase traffic during congestion.

2. The scheme must be fair so that during the congestion the available resources are allocated fairly[9].

3. The scheme must be responsive. Due to the highly dynamic nature of the network, the resource availability profile changes very rapidly. The congestion control has to be agile enough so that the demand curve can follow the capacity curve very closely.

4. The congestion control scheme must be very robust so that it can function properly under unfavorable conditions.

5. The scheme must be socially optimal. That is the total network performance should be maximized regarding the network as a whole rather than considering each user in isolation.

Congestion control approaches can be broadly divided into two groups, namely, *preventive* and *reactive*. Its design is affected by many other factors, including the connection mechanism of the network (i.e., connectionless vs. connection-oriented) and the flow control policy employed at the transport layer (i.e., window-based vs. rate-based) [Jai90]. Traditional window-based control algorithms are reactive in nature and have been used in a number of slow networks, ARPANET, TYMNET, SNA and CODEX to name a few. They are basically closed feedback control methods of flow control and require the destination to adjust the window size[10] of the sender (thus the number of unacknowledged or outstanding packets) by sending feedback signals. The applicability of window based flow control schemes to high-speed networks is adressed in a number of references [Kle92, BCS90]. The problems associated with this approach mostly stem from the dominating propagation delays across the network which renders the feedback information obsolete, thus useless. Window based schemes are also very slow to adapt to changing load patterns and they are only effective for congestions that last for several round trip delays.

In high-speed networks, flow control mechanisms have to be more of a preventive type rather than being reactive. The majority of the contemporary designs or proposals[11] have a structure that exercises flow control at two levels to prevent congestion. First, there is a check at the *call acceptance level* to determine whether or not a new flow can be accommodated within the network considering the present load. At this stage the aim is to avoid long term congestion and therefore, the client is requested to submit some kind of indicator value(s) regarding the extent and quality of the service it demands from the network. The typical examples are the declaration of the peak rate, minimum throughput demanded in case of a congestion or some parameters related to bursty traffic specification (e.g., peak cell rate, average cell rate and maximum burst size[12]). Regardless of the specific details of the individual designs, the bottom line is the necessity for the user to have a "contract" with the network before being able to proceed with the transmission. After the call/session/flow is accepted by the network (or, the virtual path is allocated in ATM terminology), the responsibility of monitoring the user's adherence to the parameters that it declared is carried out at the *cell level*. This task is far from being trivial due to the statistical nature of the flow and its design can be quite tricky.

[9] It is also quite properly pointed out that *fairness* is a vague concept and no widely accepted definition of it exists.

[10] The number of packets that can be outstanding in the network at a time; generally counted on a session basis.

[11] Note that most of them are designed to function in an ATM network.

[12] The interested reader is referred to [Tur92] for a discussion of these methods.

The current trend in preventive flow control seems to be towards rate-based mechanisms [Jai90]. A well-known rate-based control scheme is the *leaky-bucket scheme* [Tur86]. It is a mechanism for policing the negotiated rate rather than being a mechanism for lossless control. In this scheme, negotiated transmission rate is translated to the bucket size. Nodes put their packets into corresponding buckets which are allocated on a session basis. Buckets "drip" periodically and transmit a packet. Packets arriving to a full bucket are discarded. Leaky bucket is basically an admission policy which exercises the control at the point of network entrance.

Various forms of the leaky bucket scheme are proposed in the literature. One particular version [BCS90] works as follows. After the call set-up negotiation between the network and the session initiator, the packets leaving the transmitter can be marked by two colors, either green or red. The green packets are transmitted at the rate guaranteed at the time of call set-up. On the other hand, the red ones represent the rate in excess and are handled differently at the intermediate nodes, according to the traffic conditions. The basic idea is to convey the more important or delay sensitive data using green packets.

Virtual Clock is another rate-based control algorithm and was designed as a part of *Flow Network* [Zha91]. This network provides users with guaranteed performance by requiring explicit resource reservation. The user's data transmission demand is termed as a *flow* and the network reservation control determines how much share of the resources each flow may take on the *average*. It also ensures that no congestion will occur if every flow transmits according to its reserved average throughput rate. In case of violations, the most offending flows will receive the worst service and their packets are either placed at the end of the service queues or get dropped.

Not all congestion control algorithms are based on a session or flow basis, some of them only operate at the packet level and completely devoid of the flow concept. There are very good reasons behind this approach which become more agreeable when the network operates at much higher rates and/or the intermediate node are deprived of large buffers. Although the discussion of the cases which are of interest here (such as congestion control in ShuffleNet and MSN) will be postponed until the case studies, one design will be mentioned, which is described in [KAS91]. This is a neural arbiter and makes use of a two-layer *Adaptive Resonance Theory* based neural network and a *fuzzy-logic* based pre-processor to maintain distributed cut-through links between communicating users. At each intermediate node, the packet is either dropped or allowed to participate in the competition according to a weighted-sum criteria which is based on the information extracted from its header. Although the scope of the design is rather insignificant (4x4 MSN), it indicates another line of approach to this complicated problem. Another neural network based solution for the integration of ATM call admission and link capacity control can be found in [Hir91].

The speed disparity between senders and receiver places additional demands on flow and congestion control mechanisms. One particular example is the structures that interconnects networks with different transmission rate and capacity. In [WoS89], the bottleneck situation at a gateway that connects a lower-speed LAN to a high-speed MAN is discussed and a flow control scheme is proposed. A study of packet loss in high-speed networks interconnecting conventional local area networks can be found in [GMW92].

5.3 Buffering Policies: Deflection vs. Store–and–Forward

While a routing decision is being made at an intermediate node, it is possible for more than one incoming packet to opt for the same outgoing link. In such a case, the packets that lost in the contention can either be buffered or relayed on non-optimal links, i.e. *deflected.* Deflection routing is suitable for networks with limited or non-existent buffer space. The rationale of eliminating large buffers in high-speed networks can be stated as follows:

1. Networks with practically infinite-memory switches are as susceptible to congestion as networks with low-memory switches. In the former case, the queuing delays can get so long that

by the time the packets come out of the switch, most of them may have been already retransmitted by the higher layers due to timeouts. In fact, too much memory is more harmful than too little memory, since the packets or their retransmissions have to be dropped after they have consumed precious network resources [Jai90].

2. Due to the nature of real-time applications which are characterized by stringent delay requirements, long buffers should be avoided, at least for this particular group of clients.

3. Elimination of buffers can speed up switching significantly so that the process can follow the link speed as closely as possible. Particularly, all electronic components can be removed from the switch and replaced with their optic equivalents that can operate at the link speed by avoiding E/O and O/E conversions. In this case, all control operations also need to be performed optically. One of the major challenges for the designers of photonic switching systems involves contention resolution. Although internal contention can be avoided by using nonblocking switches, the problem of the output contention still remains. The major problem is the lack of optical equivalent of electronic buffer memories. Contention resolution without resorting to E/O conversions seems to be very difficult at the current level of technology unless cumbersome optical delay lines are used [JaM93].

It is obvious that deflection causes some packets to traverse longer paths. Furthermore, unless some countermeasures are taken, it is possible for a packet to travel indefinitely. In general, it can be said that if the probability of deflection at every intermediate node is equal to 1/2 or greater, deflection routing is not a good alternative. The problem can be examined in conjuction with the so called *Gambler's Ruin* problem. Suppose that at a given instant, a packet is at half way distance to its destination and still has $d/2$ hops to cover. In other words, it has some money ($d/2$, the distance it has already covered) and it needs as much more to finish the game (to reach its destination) as opposed to going backrupt (before being pushed d hops away from its destination again, or practically going back to the position where it started). At every node it rolls a dice (competes with other packets for a particular link) and loses with a certain probability (p_d, probability of deflection). If it wins (and is not deflected), it gets closer to the destination by one hop. Otherwise, the remaining distance increases by the penalty of deflection. In general, the deflection penalty is equal to or greater than 2. For example, in ShuffleNet, an undeflected packet can gain only one step in the right direction while losing k in case of deflection. Consequently, the necessity of some countermeasures is obvious. In practice, p_d can be decreased in a number of ways:

1. The load offered to network is kept under its possible maximum so that contentions do not occur so often.

2. A network topology that can offer multiple shortest paths is used.

3. In routing, priority is given to the packets closer to their destinations.

4. In routing, priority can be given to the packets that have been previously deflected.

5. In routing, priority can be given to the packets that have been in the network for the longest time.

6. The packets that lost the contention are buffered and therefore given another chance in the next round rather than being immediately diverted from its course.

7. Packets that exceed a certain hop count limit are removed from the network therefore reducing the number of packets competing for links.

An effective solution may exercise a combination of these options to optimize its performance. In the case studies section, we will also address this particular aspect for the networks discussed there.

Undeniably, deflection routing can cause some decrease in the performance of the network compared to its store-and-forward counterpart, all else being equal. In [AcS92] a comparative analysis is presented for ShuffleNet[13] with $p = 2$. The study shows that the achievable aggregate capacity degrades as the number of nodes increases under deflection routing, but even for networks with several thousand nodes getting no worse than 25% of that for store–and–forward. Therefore, for an order-of-magnitude capacity advantage, a link speed–up factor of 25 to 50 will be needed. Problems arising from deflection routing in access and congestion control are discussed in [Max90].

5.4 Topology

Topology defines the connectional properties and spatial relationships amongst the nodes of a network. Topological properties of a given network can be examined separately from the routing and congestion control mechanisms and can provide clues regarding the suitability of different choices in the design of the routing scheme. They are also directly related to the maximum throughput and fault tolerance of the network.

Networks can be grouped into two broad categories according to their topologies, namely, point–to–point networks and broadcast networks[14]. Broadcast networks employ a broadcast channel for interconnection so that each node can transmit using this channel and every transmission can be received by all the nodes in the network. In point–to–point networks, nodes are connected to each other rather than a broadcast channel. Due to the cost of full connectivity[15], in practice a node is connected directly to a subset of nodes. Connections are made in such a way that there is at least a path between any given two nodes. A *path* is simply a collection of node–to–node links connecting a given source to a given destination. In this structure, a packet needs to be relayed from node to node to reach its destination. If a given node has more than one outgoing link, a routing decision needs to be made by the node to select the outgoing link upon which the incoming transient packet will be relayed. The number of nodes that performs a routing task on a given path is defined as the *hop count of the path*[16]. For efficiency reasons, a node tries to relay packets along a path of minimum delay (shortest path) to their destinations. The determination of the shortest path is related to the costs assigned to links. If links are of equal capacity and equal length, and the links costs are determined solely by these two factors[17], the shortest path also defines a path with the minimum hop count.

The topological properties of a network that are of interest here are defined as follows:

1. *Network Size* ($\mathcal{N}$) : Simply defined as the number of the nodes in the network.

2. *Diameter* ($\mathcal{D}$) : The diameter of a network is the maximum of all shortest path distances in the network.

$$\mathcal{D} = \max\{\pi_{ij}\} \qquad 1 \leq i, j \leq \mathcal{N} \tag{5.2}$$

   where π_{ij} stands for the shortest path distance between nodes S_i and S_j (this distance is measured in hops). One important property of this parameter is its growth rate with the network size $\mathcal{N}$ (e.g., logarithmic vs. linear).

[13] The degree of connectivity p is defined in section 4.

[14] For efficiency reasons, the topology of very large networks may be organized into several hierarchical layers as in telephone networks. Consequently, hybrid topologies may form. This case will not be considered here.

[15] Connecting every node to all other nodes in the networks by a direct link.

[16] In some networks, the hop count of a path can be different than the number of nodes that the path passes through. One example is the networks in which a virtual topology is embedded into a physical topology. In such a network, an input port may be wired to an output port or a wavelength can be assigned to another directly. According to our definition, such a node does not contribute to the hop count of a path. Conversely, the same node can be visited by a packet on different wavelengths. In this case, each visit that requires another routing decision counts as a hop. The same argument also applies to the following calculation of the average hop count.

[17] Therefore not taking the congestion into account and making all link costs equal.

3. *Average Hop Count* ($\overline{h}$) : The average number of nodes along the shortest path between a pair of nodes in the network and is calculated as follows:

$$\overline{h} = \frac{1}{\mathcal{N}} \frac{\sum_{i=1}^{\mathcal{N}} \sum_{j=1}^{\mathcal{N}} \pi_{ij}}{\mathcal{N} - 1} \qquad (5.3)$$

   where $1 \leq i, j \leq \mathcal{N}$ and $i \neq j$.

4. *Degree of Connectivity* : Defines the number of links incident to and from a node. When these two quantities are the same for a node and across the network, the network topology is said to be regular. Such a network is also referred as *p-connected* where p is equal to the in/out degree of a node. For some interconnection patterns the degree of connectivity has to be increased as the network size increases.

5. *Deflection Penalty* : Defined with respect to the routing algorithm and gives the least upper bound on the number of hops that a single deflection adds to a packets delay. Sometimes it is also defined as the ***girth*** (the length of the shortest cycle, if any) of the graph that corresponds to the topology of the network.

The topological properties of a network are directly related to its throughput ($\mathcal{U}$). If every link assumed to be of equal capacity and each link traversal takes one unit of time, then $\overline{h}$ gives the ***sojurn time*** per slot — the mean amount of time the slots must stay in the network to reach their destination. In other words, $\overline{h}$ gives the average cost per slot transmission in terms of time and/or the number of links that must be traversed. If,

1. the network is symmetric,

2. all links are of equal length and capacity,

3. nodes have infinite buffer space,

4. the traffic is uniformly distributed and,

5. the packet generation rate is the same for all nodes,

the relationship between $\overline{h}$ and $\mathcal{U}$ can be expressed in closed form as follows:

$$\mathcal{U} = \frac{\textit{Total Number of Links in the Net}}{\overline{h}} \qquad (5.4)$$

This equation is used to approximate the throughput of various networks in the case studies section for the networks that satisfy the aforementioned criteria. The equation seems to favor the network topologies with lower $\overline{h}$ over the others in terms of performance. On the other hand, semirandom or random topologies with lower hop counts (Moore bound[18] is used for comparison) reportedly yielded poor throughput rates compared to the regular networks with higher values of $\overline{h}$ [Ro92a, Ro92b]. This is not suprising since the fairness and regularity of the topology, as well as the variance of h_i amongst the nodes have their own merits which should be carefully considered prior to a meaningful evaluation.

5.5 Gbps Networks and New Challenges

In this section, we will investigate the fundamental issues that render Gbps networks different from their slower counterparts and discuss their implications on the design of network topology and routing and congestion control mechanisms. Here, we choose to group these issues into two categories:

[18] See 5.6.2 for details.

(1) Processing Bottleneck: In a packet swiching network, the time required for a routing decision cannot be longer than a single packet's transmission time. Otherwise, the system becomes unstable since at a given node the utilization factor ($\rho = \lambda/\mu$) becomes greater than 1 and queue lengths grow indefinitely, where λ and μ denote arrival and service rates respectively. Considering 53 byte cell length[19] of ATM networks, at a 1 Gbps rate, a node has 424 ns to complete the following tasks:

1. Choosing the appropriate output links for every transient packet either by using the routing information contained in the packet header or by consulting a table, usually using the destination address as the index.
2. Resolving the contentions in such a way that the network performance is optimized.

If the routing process is aimed to satisfy as many packets as possible with their demands and to deflect the others in such a way that the total deflection penalty is also minimal, the problem turns out to be far from being trivial. In a node with the connectivity degree of p, p incoming packets can be assigned to p outgoing links in $p!$ ways if deflection routing is used. In the store–and–forward case, the number of possible assignments becomes as large as p^p. The following table illustrates the scope of the problem space.

p	*Deflection*	*S & F*
2	2	4
4	24	256
8	40320	16777216
16	2.09×10^{13}	1.84×10^{19}

The existence of multiple priority levels may also contribute to the problem rather than alleviating it. Similar problems are encountered in telephone networks and multiprocessor interconnection networks as well. The proposed solutions are heuristic in general. In this regard, the appeal of the structures with low connectivity degrees is obvious. Furthermore, software based solutions can be discarded from the list of feasible alternatives, since within the given time frame only a handful of instructions can be executed. This implies routing protocols that can be effectively embedded into hardware, which in turn implies the requirement of simplicity.

(2) Dominating Propagation Delays: In [Kle92], the effect of latency due to the speed of light is discussed with respect to parameter–a, where

$$a = \frac{\textit{Propagation Delay}}{\textit{Packet Transmission Time}} \tag{5.5}$$

On a link across the USA, the corresponding values of parameter–a are calculated under different transmission rates as follows:

	Capacity	Packet Length	Prop. Delay	
NETWORK	(Mbps)	(bits)	(microsec.)	Ratio a
Local Net	10.00	1000	5	0.05
WAN	0.05	1000	20000	1.00
Satellite	0.05	1000	250000	12.50
Fiber link	1000.00	1000	15000	15000.00

The value of a measures how many packets can be pumped into one end of the link before the first bit appears at the other end. The enormous variation in the value of this critical system

[19] Which is considered small in the literature.

parameter is the source of the problems encountered not only at the routing and flow control but at the application level.

In the remainder of this section, we will investigate the impacts of these factors on the design issues related to topology, routing and congestion control.

5.5.1 Impacts on Topological Design

Due to high installation and maintenance costs, it is reasonable to expect some optimization to happen during real-life implementations of MANs and especially WANs. Also taking into consideration the fact that propagation time on the links is the major contributor to packet delays, it is desirable to minimize the total cable length of the network. The corresponding geometric problem of connecting n given points in the plane with shortest set of lines is known as the *Euclidean minimum spanning tree problem* [Sed88]. One way to solve this problem is to build a complete graph with $n(n-1)/2$ undirected edges which are weighted according to the distance[20] between the corresponding points. Then a minimum spanning tree (MST) construction algorithm is applied. However, it is possible to do better than that when some additional concentrator points are allowed [Law76]. This problem is the so-called *Steiner Tree Problem* whose origin can be traced back to Fermat. The extra points are referred as the *Steiner points* and it is proved that for any n points to be spanned there exists a minimum length Steiner tree which contains no more than $(n-2)$ Steiner points. It is also conjectured that the mimimum spanning tree is no more than $(2/\sqrt{3})$ times as long as the minimum length Euclidean Steiner tree [SLL81]. However, the problem of building the minimum Steiner tree for a given collection of points is NP-complete [GaJ79]. It has been expressed in many forms and a catalog of its different formulations can be found in [GoM93]. The most efficient heuristic [Win87] which seems to produce good solutions is given in [SLL81].

When this sort of optimization is employed, regular network structures such as ShuffleNet can no longer be preserved. In order to retain the advantages of the regular interconnection networks, embedding a virtual topology into a given physical topology can be considered. One way of achieving this is what is known as the *wavelength division multiplexing* (WDM). WDM carves the bandwith offered by the optic fibre into a number of smaller bandwidth carriers and assigns them on a wavelength basis to stations. A wavelength can either be assigned to a transmitter-receiver pair (so that a point-to-point network is obtained) or can be shared by a group of transmitters and receivers. In the latter case it becomes possible to add a small number of nodes to structures that are defined for a certain number of nodes such as ShuffleNet or hypercube. The advocates of this approach claim that the topological changes can be carried out dynamically to trace the pattern of traffic load to maximize the network performance. Details of WDM can be found in [VWD91, MoG90, BFM90, LaA91, ZhA90, HIK88, Aca87, AKH87]. These studies, raise the following questions:

1. The embeddings of hypercube and ShuffleNet are discussed in [VWD91] and [BFM90], respectively. However, the efficient bit-controlled routing algorithms developed for these networks minimize the number of hops to be crossed and are not concerned with the lengths of the links. In other words, the dominating propagation delays are exluded from the routing procedure. Consequently, the minimization of hop counts can no longer guarantee the minimization of the delays. A remedy is suggested in [VWD91] by assigning the wavelengths in such a fashion that the physical lengths of the virtual connections are minimized. However, the researchers also indicate that the subproblems of network configuration, namely, (1) the mapping of the nodes in the physical topology to the nodes in the virtual topology, (2) the mapping of the edges in the physical topology to the edges in the virtual topology, (3) the allocation of the

[20] The metric in which this distance is measured is not necessarily Euclidean: the distance between two points (x_1, y_1) and (x_2, y_2) can be expressed either in Euclidean metric $\mathcal{L}_2$ as $(\sqrt{(x_1-x_2)^2+(y_1-y_2)^2})$ and in rectilinear metric $\mathcal{L}_1$ as $(|x_1-x_2|+|y_1-y_2|)$. Rectilinear metric is suggested as a more realistic alternative for urban environments [BFM90].

wavelengths to edges, are all expected to be NP–hard as the search space grows at least as fast as $N!$. Consequently, effective heuristics are in demand.

2. The other problem is related to the routing procedure. The necessity of hardwiring the routing algorithm into the switch fabric was stated previously. In [LaA91], the total network throughput of ShuffleNet is improved by modifying the topology so that the largest flow on any link is minimized. The throughput improvement ranges from 6.8% for the quasi–uniform traffic matrix to 24.3% for the ring type traffic. The employed flow deviation method requires the iterative computation of shortest–paths. However, it is not clear how a switch designed for ShuffleNet connectivity pattern is supposed to function on the modified topology. It seems that the cost of the gained flexibility is the reintroduction of large routing tables into the network.

5.5.2 Impacts on Routing Protocol Design

For large and slow networks, most existing or proposed routing methods use variants of shortest path or least cost algorithms to attain an optimal (or more favorable) delay-throughput curve [BeG87]. From a theoretical standpoint, the optimal routing problem can be viewed as an optimization problem. With a cost function minimized according to certain restrictions, optimal routing can be achieved[21]. The cost function assigns each link a certain cost value which is adjusted dynamically according to the congestion level on the link and/or the availability of the link itself. The optimization process requires the coordination amongst all the nodes of the subnet rather than just a pair of nodes as in the higher layer protocols. This coordination is achieved using a rather complex collection of algorithms that work more or less independently and yet support each other by exchanging services or information. However, for Gbps networks, this approach presents serious drawbacks. In [Sch80] six performance measures are proposed that can be used to compare the decentralized routing algorithms, the first and the most important one is being the *speed of response*:

> *This measure is obviously extremely important in dynamic environment since the speed of response must be faster than the rate of change of network topology. Otherwise convergence will not occur and the routing algorithm will be useless.*

Regarding flow based methods, in [BeG87] it is stated that;

> *Implicit in flow models is the assumption that the statistics of the traffic entering the network do not change over time. This is a reasonable hypothesis when these statistics change very slowly relative to the average time required to empty the queues in the network and when link flows are measured experimentally using time averages.*

Simply stated, the time interval between the changes is assumed to be long enough to propagate the related update information to the involved nodes, to calculate a new optimum state and to benefit from the reaction of the routing algorithm to the requirements of the new state. Today with the advent of the Gbps networks, these computationally very expensive and less agile algorithms cease to be feasible and their functionality has to be provided by less complex but much faster new alternatives. In the absence of timely global status information, local minimization techniques take precedence.

5.5.3 Impacts on Congestion Control

In [BCS90] the unsuitability of the conventional mechanisms based on end–to–end or hop–by–hop windowing schemes for controlling congestion within high–speed networks are discussed as follows:

[21] The optimal routing problem is shown to be mathematically equivalent to the problem of optimal flow control [BeG87]. Therefore, the optimality conditions and algorithms developed for optimal routing are applicable to the latter.

1. *Window based mechanisms typically rely on end to end exchange of control messages in order to regulate traffic flow. The control messages (sometimes with additional congestion information added by the intermediate nodes) are used as feedback by the source node to regulate its traffic. In high speed networks, the propagation delays across the network typically dominate. Thus the feedback is usually outdated and any action the source takes is too late to resolve buffer overflows and avoid congestion. This argues for mechanisms that do not heavily rely on network feedback.*
2. *It is also important that the congestion control mechanism operate at the speed of the communication link. For this reason, especially in the case of hop by hop window based mechanisms, computationally intensive control schemes are less desirable than simple schemes that can be easily implemented in high speed hardware.*
3. *The nature of the traffic also effects the design of the congestion control. While data traffic can usually be slowed down in order to cope with network congestion, it is likely that the real-time nature of the traffic will require some level of bandwidth guarantee. Real-time traffic (e.g., voice, video) has an intrinsic rate determined by the external factors that are outside the control of the network. Typically this rate can be estimated by the network prior to the establishement of the connection. The ability to slow down such sources is usually very limited. However the packet arrival process is stochastic implying that there is no guarantee that over short periods the resource will keep to the specified average rate. In addition, the initial estimate of the rate may be incorrect.*

The subject is also discussed in [Jai90] which draws attention to the basic principles of control theory by indicating that no scheme can solve the congestion problem that is shorter than its feedback delay and suggests a multi-level congestion control architecture for handling short term and long term congestions at different levels of the network hierarchy.

5.6 Case Studies

In this section, we will discuss some design proposals from the literature. Their common characteristics are the self-routing capability and the ability to function without requiring large buffers. The latter also implies suitability for deflection routing.

5.6.1 Hypercube

A hypercube consists of $\mathcal{N} = 2^n$ nodes that are numbered by n-bit binary numbers, from 0 to $2^n - 1$ and interconnected so that there is a link between two processors iff their binary representations differ by one and only one bit. Therefore the nodal connectivity degree is n which is also the diameter, since every hop can effect only a single bit in the address field and the source and destination addresses can differ by n bits at most. The distance between a pair of nodes is equal to the *Hamming distance* between their addresses expressed in binary format. The average hop distance is equal to the half of the diameter ($\overline{h} = \frac{(\mathcal{N}\log_2\mathcal{N})}{2(\mathcal{N}-1)} \approx \frac{n}{2}$). The penalty of deflection is 2. Other topological properties of the hypercube are discussed in [SaS88].

Hypercube is a richly connected structure. Although it is non-optimal in terms of diameter, it can deliver optimal performance when the traffic is uniformly distributed, even with very simple routing mechanisms. Routing in hypercube can be performed as follows: At every intermediate node an XOR operation is performed between the current node address and the destination address. The locations of 1 *s* indicate the connection functions that need to be executed thus the output links lying on the shortest path. The basic routing algorithm for hypercube is given in [Kat88] as follows where $\oplus$ operator stands for XOR operation:

```
Compute relativeaddr = currentaddr ⊕ destinationaddr.
Starting with the most significant bit of relativeaddr:
        let $i$ be the bit number of first 1 in relativeaddr.
Forward the packet on link $i$.
```

The performance of the hypercube under deflection routing is investigated in [Szy90, GrH92, Haj91] and the results show that the structure is very suitable for deflection routing. Between two nodes, *node*(i) and *node*(j) with a Hamming distance of $\mathcal{H}(i,j) < n$, there are $\mathcal{H}(i,j)$ node–disjoint paths of length $\mathcal{H}(i,j)$. Furthermore, n different node–disjoint paths whose lengths are less or equal to $(\mathcal{H}(i,j)+2)$ are available [SaS88]. Consequently, a packet suffers at most $O(\log n)$ deflections under uniform traffic, which is rather small [GrH92]. There are n unidirectional links in the network per node and a packet must traverse $n/2$ links on the average, limiting the throughput to 2 per node in the long run. In [GrH92], it is shown that even with the added burden due to deflections, throughput near 2 packets per node per slot can be sustained.

Since the number of nodes must be a power of 2, there are large gaps in the sizes of the system that can be built with the hypercubes. One solution to this problem can be found within WDM based structures. Another one is to use incomplete hypercubes which are defined for arbitrary number of nodes [Kat88]. In an incomplete hypercube, node connectivity rules remain intact as the link numbering given in the previous algorithm; i.e., *link*(i) connects two nodes whose numbers differ only at ith bit position. Note that same link number is obtained when computed from either end of the link, but some of the links will be missing compared to those available in a complete hypercube. As for routing, the basic algorithm still works after a small modification[22] as follows;

```
Compute relativeaddr = currentaddr ⊕ destinationaddr.
Starting with the most significant bit of relativeaddr:
        let $i$ be the bit number of first 1 in relativeaddr,
        where link $i$ exists from current node.
Forward the packet on link $i$.
```

For an incomplete hypercube with $\mathcal{N}$ nodes, this algorithm has a worst case path length of $\lceil \log_2 \mathcal{N} \rceil$. The main disadvantage of hypercube is that the nodal connectivity degree increases logarithmically with the network size $\mathcal{N}$ [Muk92].

5.6.2 Shuffle–like (Minimum Diameter) Networks

Because of its relationship to throughput, it is natural to consider the networks designs that minimize $\mathcal{D}$ for a given number of nodes and degree of nodal connectivity. It is known that the minimum diameter $\mathcal{D}$ and the maximum connectivity degree p of a directed graph are related to each other:

$$\mathcal{N} \leq 1 + p + p^2 + \cdots + p^{\mathcal{D}} = \frac{p^{(\mathcal{D}+1)} - 1}{p - 1} \tag{5.6}$$

If the equality holds, then it is said that the *Moore Bound* is achieved. Note that for directed Moore graphs the lower bound on diameter is:

$$\mathcal{D} = \lceil \log_p(\mathcal{N}(p-1)+1) \rceil - 1 \tag{5.7}$$

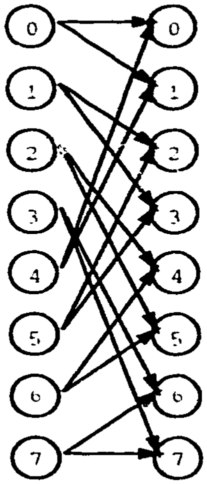
and $\overline{h}$ is [HIK88],

$$\overline{h} = \frac{p - p^{\mathcal{D}+1} + \mathcal{N}\mathcal{D}(p-1)^2 + \mathcal{D}(p-1)}{(\mathcal{N}-1)(p-1)^2} \tag{5.8}$$

It is also known that there are no directed Moore graphs for the nontrivial values of $\mathcal{D}$ and p [BrT80]. Shuffle like networks provide mean internodal distances approaching the Moore limit

[22] Which is printed in boldface in the code.

[Ro92b]. However, these networks are not defined for networks of arbitrary size[23]. Figures 5.1–5.3 depict three shuffle–like networks for $p = 2$.



From \ To	0	1	2	3	4	5	6	7	
0	▨	1	2	2	3	3	3	3	17/7 = 2.43
1	3	▨	1	1	2	2	2	2	13/7 = 1.86
2	2	2	▨	2	1	1	3	3	14/7 = 2.00
3	3	3	3	▨	2	2	1	1	15/7 = 2.14
4	1	1	2	2	▨	3	3	3	15/7 = 2.14
5	3	3	1	1	2	▨	2	2	14/7 = 2.00
6	2	2	2	2	1	1	▨	3	13/7 = 1.86
7	3	3	3	3	2	2	1	▨	17/7 = 2.43
									Avg. = 2.13

Figure 5.1: Topology of 8-node de Bruijn Network.

The shuffle–like networks based on the *de Bruijn graphs* are defined when $\mathcal{N}$ is a power of node connectivity degree p. Figure 5.1 shows a special case of $p = 2$ and $\mathcal{N} = 2^3$ in which $node(i)$ is connected to $node(2 * i \bmod \mathcal{N})$ and $node((2 * i + 1) \bmod \mathcal{N})$. In this network, even if the offered traffic is fully symmetric, the link loads can be unbalanced [Muk92]. This is due to the existence of self loops[24] that carry no traffic. For de Bruijn network, the diameter is $\mathcal{D} = \log_p \mathcal{N}$. The deflection penalty is also $(\log_p \mathcal{N})$ since a packet can travel back to the point where it was deflected in at most $(\log_p \mathcal{N})$ steps. Networks based on de Bruijn graphs are discussed in [SaP89, EsH85, SiR91].

Figure 5.2 shows the configuration that is discussed in [Max89] and referred to as SXN. The self–loops are eliminated by connecting nodes 0 and $\mathcal{N} - 1$ to each other. Although $\overline{h}$ is the lowest of the three variations, an effective routing algorithm that can use these new connections is not available and therefore they are used only in case of link failures or deflections.

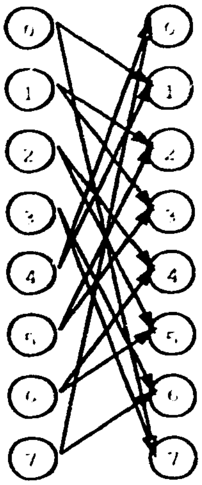
Another variation of shuffle–like networks was proposed in [AKH87] under the name of *ShuffleNet* and discussed further in [Aca87, HK88]. In ShuffleNet $\mathcal{N} = kp^k$ nodes are arranged in k columns of p^k nodes and a node is addressed with its row and column coordinate pair (r, c). Columns are ordered left to right from 0 to $(k - 1)$ and rows are numbered top to bottom from 0 to $(p^k - 1)$. The row coordinates are represented in p–ary notation, $(r = r_{k-1}r_{k-2} \cdots r_0)$. Accordingly, the p nodes that any given node (r, c) is connected to are as follows:

$$
\begin{aligned}
(r, c) \text{ is connected to : } & ((c + 1) \bmod k, r_{k-2}r_{k-3} \cdots r_0 0) \\
& ((c + 1) \bmod k, r_{k-2}r_{k-3} \cdots r_0 1) \\
& \vdots \\
& ((c + 1) \bmod k, r_{k-2}r_{k-3} \cdots r_0 (p - 1))
\end{aligned}
$$

Figure 5.3(a), redrawn in figure 5.3(b), shows a special case of $p = k = 2$. Th[illegible]tion pattern also has no self–loops. Furthermore, the variation in $\overline{h}_i$ across the nodes becomes zero since every

[23] One of the best known general family of graphs [HoP88] has been proposed by Imase and Itoh in [ImI83, ISO85] which contains de Bruijn graphs as a special class. The design procedure proposed in [ImI83] has the upper bound $\mathcal{D} \leq \lceil \log_p \mathcal{N} \rceil$ on network diameter for the arbitrary values of $\mathcal{N}$ and p. Another approach to construct networks with low mean internodal distances, based on simulated annealing, can be found in [Ro92b].

[24] The links 0 → 0 and 7 → 7 in the figure. In general, there are p of them in a p connected network.



From \ To	0	1	2	3	4	5	6	7	
0	▨	1	2	2	3	3	2	1	14/7 = 2.00
1	3	▨	1	1	2	2	2	2	13/7 = 1.86
2	2	2	▨	2	1	1	3	3	14/7 = 2.00
3	2	3	3	▨	2	2	1	1	14/7 = 2.00
4	1	1	2	2	▨	3	3	2	14/7 = 2.00
5	3	3	1	1	2	▨	2	2	14/7 = 2.00
6	2	2	2	2	1	1	▨	3	13/7 = 1.86
7	1	2	3	3	2	2	1	▨	14/7 = 2.00
									Avg = 1.96

Figure 5.2: Topology of 8-node SXN.

node has the same number of nodes that lies within a given shortest path distance (i.e., there are 2 nodes reachable in 1 hop, 3 nodes reachable in 2 hops and 2 nodes reachable in 3 hops from any node). Note that the edge effects are removed from the network by placing the nodes into 2 groups (shown as columns in figure 5.3(b)).

The diameter of the network is $D = 2k - 1$ with the deflection penalty of k. The average number of hops is given in [HIK88] as:

$$\bar{h} = \frac{kp^k(p-1)(3k-1) - 2k(p^k-1)}{2(p-1)(kp^k-1)} \tag{5.9}$$

For the special case of $p = 2$, the above equation takes the following form [Aca87]:

$$\bar{h} = \frac{1}{2^k}\left[3(k-1)2^{k-1} + 2\right] \tag{5.10}$$

giving the achievable network throughput under uniform traffic as

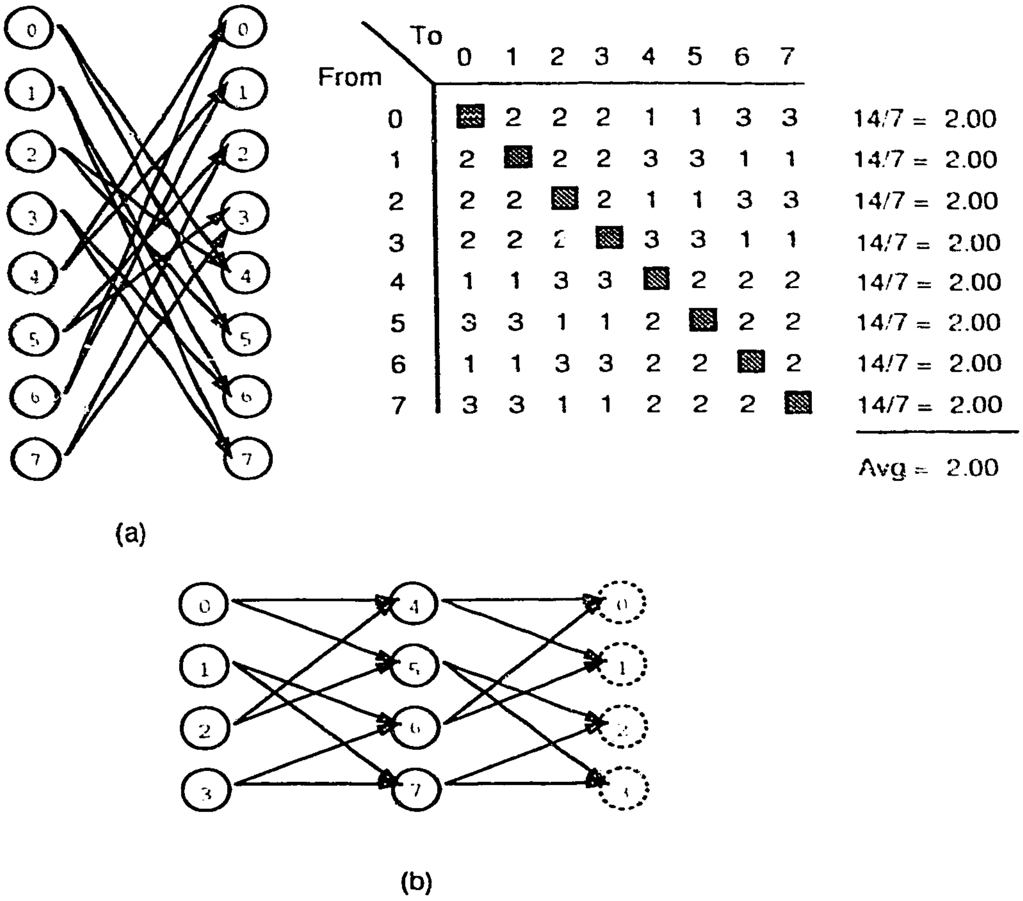
$$U = \frac{k2^{2k+1}}{3(k-1)2^{k-1} + 2} \tag{5.11}$$

Consequently, the throughput per user is

$$U_i = \frac{2^{k+1}}{3(k-1)2^{k-1} + 2} \approx \frac{4}{3}\frac{1}{k-1} \qquad \text{for } i = 1, \cdots, k2^k \tag{5.12}$$

Note that, the value of $\bar{h}$ increases with increasing k impeding the rate of growth in achievable throughput. The 2-column configuration offers the highest throughput rate [HIK88] and, for $k > 2$, no routing algorithm is known that yields a balanced use of links, even for a perfectly balanced load [EiM88]. On the other hand, the increased number of columns increases the number of multiple shortest paths; the destination nodes that are k to $(2k-1)$ hops away from a given source can be reached via more than one shortest path[25]. The following table [EiM88] gives the average hop counts, and throughput rates for different values of k.

[25] This property of ShuffleNet is investigated in [Aya89] with the help of signal flow graphs and the results are compared to that of MSN.



From \ To	0	1	2	3	4	5	6	7	
0	▨	2	2	2	1	1	3	3	14/7 = 2.00
1	2	▨	2	2	3	3	1	1	14/7 = 2.00
2	2	2	▨	2	1	1	3	3	14/7 = 2.00
3	2	2	2	▨	3	3	1	1	14/7 = 2.00
4	1	1	3	3	▨	2	2	2	14/7 = 2.00
5	3	3	1	1	2	▨	2	2	14/7 = 2.00
6	1	1	3	3	2	2	▨	2	14/7 = 2.00
7	3	3	1	1	2	2	2	▨	14/7 = 2.00
									Avg = 2.00

Figure 5.3: Topology of 8 node ShuffleNet.

k	N	h	U_i	U
2	8	2.0	1.0	8.0
3	24	3.25	0.617	14.8
4	64	4.625	0.433	27.7
5	160	6.06	0.33	52.8
6	384	7.53	0.265	101.9
7	896	9.09	0.222	198.8
8	2048	10.51	0.19	389.8
9	4608	12.001	0.167	767.7

For shuffle-like networks, many different routing algorithms are possible [EsH85, SiH88, Sie90, Hui90]. Two routing algorithms proposed for ShuffleNet are discussed below.

Under uniform traffic conditions, ShuffleNet is highly effective with a simple fixed routing algorithm. The fixed routing algorithm uses a single path for a given origin destination pair in which a given node $(\hat{r}, \hat{c})$ forwards a transient packet bound for (r^d, c^d) according to the following rule [KaS91]:

Route packet to $node((\hat{c}+1) \bmod k, \hat{r}_{k-2}, \cdots, \hat{r}_0 r^d_{X-1})$ where X denotes the number of columns between the current node and the destination and given as

$$X = \begin{cases} (k + c^d - \hat{c}) \bmod k & \text{if } c^d \neq \hat{c} \\ k & \text{if } c^d = \hat{c} \end{cases}$$

In [EiM88] it is shown that, the allowable throughput per node with fixed routing algorithm and realistic (nonuniform) traffic patterns is reduced by a factor between 0.3 and 0.5 with respect to that predicted for uniform load. In that case, a proper use of available multiple shortest paths can alleviate the aforementioned problem. To that end, an adaptive routing scheme is proposed in [KaS91] for ShuffleNet. Following the notation used in the description of the fixed routing algorithm, let D denote the number of columns between source (r^s, c^s) and destination (r^d, c^d).

$$D = \begin{cases} (k + c^d - c^s) \bmod k & \text{if } c^d \neq c^s \\ k & \text{if } c^d = c^s \end{cases}$$

If a packet cannot reach its destination in k hops, then the minimum–hop routing path length is $D + k$, regardless of what routing decisions are made in the first D hops. Therefore, if the source and destination are more than k hops apart, the packet can be routed arbitrarily for the first D hops until it reaches column c^d of its destination. Then a single path of length k leads to (r^d, c^d). The routing algorithm requires marking each packet at the source according to the distance to its destination either as M (stands for multiple minimum-hop paths) or as S (stands for single minimum–hop path). Clearly, type M packets require more than k hops to reach their destinations. At each intermediate node the remaining distance is calculated for each packet and the type field is updated if necessary. There are two ways for a packet's type to change. First, a type S packet is deflected to a longer path increasing the remaining distance by k hops and becomes type M. Second, a type M packet reaches the column of its destination and becomes type S. The test for type can be performed as follows.

TYPE S if $r^s_{k-1-D} = r^d_{k-1}, r^s_{k-2-D} = r^d_{k-2}, \cdots$, and $r^s_0 = r^d_D$
TYPE M otherwise.

The routing scheme also employes buffers and deflects a packet at most once. Type M packets are always placed in the shortest queue at a given node. Type S packets are normally routed according to the fixed algorithm. But if the desired buffer is full beyond a certain threshold, it is placed in the shortest queue with this information is recorded in the header. A deflected packet is always placed in the appropriate queue regardless of the threshold. If the queue is full, then the packet is dropped. The throughput results for networks with sizes comparable to those above, are not available.

The throughput and delay performance of de Bruijn networks are compared with those of ShuffleNets in [SiR91]. For a given diameter, de Bruijn network can support more nodes than its ShuffleNet counterpart as can be seen from the following table (for p=2).

k	N	$\mathcal{D}$ (ShuffleNet)	$\mathcal{D}$ (de Bruijn)
2	$2^3 = 8$	3	3
4	$2^6 = 64$	7	6
6	$6 \times 2^6 = 384$	11	Not applicable
8	$2^{11} = 2048$	15	11
16	$2^{20} = 1048576$	31	20

The increase in the number of stations comes at the expense of some nonuniformity in edge loading under uniformly distributed traffic. In this regard, to illustrate the differences between ShuffleNet and de Bruijn netwoks, we performed the following counting operation on these sample networks shown in the figures 5.1 and 5.3. First, we listed all shortest paths for every origin–destination pair in the network, indicating also the intermediate nodes, e.g., the shortest path from *node*(0) to *node*(7) is (0 — 1 → 3 → 7) in de Bruijn network, whereas ShuffleNet offers two possibilities, namely, (0 — 4 — 1 → 7) and (0 → 5 → 3 → 7). Then we deleted the end nodes from the paths i.e., nodes 0 and 7 in the example. The results revealed the following:

1. There are no multiple–shortest paths in the de Bruijn network whereas the ShuffleNet offers 2 multiple–shortest paths to every node for 2 destinations out of 7.
2. For de Bruijn network, 56 different shortest paths are established. Due to the availability of multiple–shortest paths, this number is 72 for ShuffleNet. For each node, the number of times that the node acted as an intermediate was determined. The results are listed in the following table.

Node	*de Bruijn*	*ShuffleNet*
0	0	11
1	11	11
2	9	11
3	11	11
4	11	11
5	9	11
6	11	11
7	0	11

Note that in de Bruijn network, *node*(0) and *node*(7) never act as an intermediate, therefore their output ports are never used to convey any other traffic and are always available to them. The loads of other nodes are also unbalanced. This is the source of the aforementioned unbalanced link loading, which is also discussed in [SiR91].

5.6.3 2-Dimensional Toroidal Networks

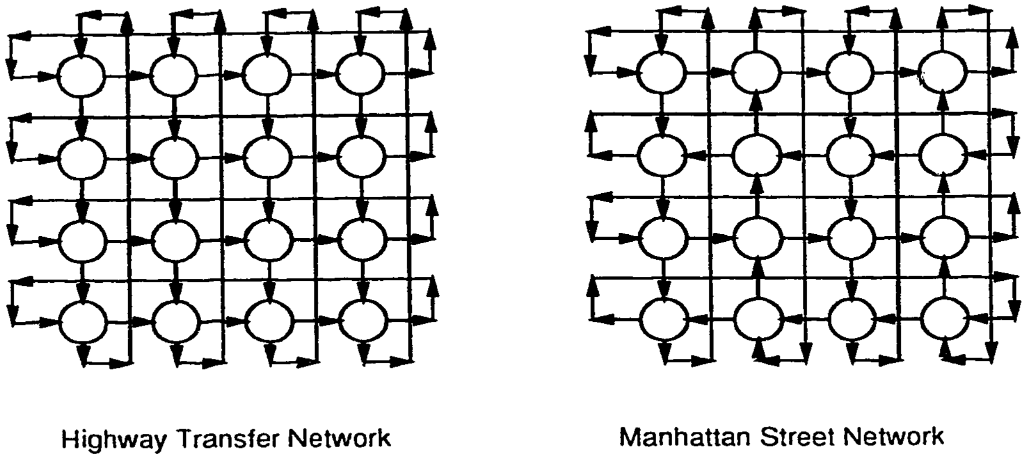
A 2–dimensional toroidal network is simply a rectangular mesh with orthogonal wraparaound connections. The reasons that made this structure a viable topology for MANs and WANs are stated in [Rob88] as follows:

1. Addressing and routing is straightforward.
2. The topology is *isotropic*, that is every node has a similar set of connections.
3. There are no edge effects and wraparaound connections decrease path lengths.
4. For MANs, the topology easily covers a rectangular grid of streets and avenues (i.e., the topology makes sense geograhically [Max85]).

Other important properties which are mentioned in [Max85] are the ability to keep the traffic of one community of interest from interfering with others and the ease of adding new nodes without resorting to major reconfigurations.

The torus removes the edge effects by the virtue of being a manifold, that is a finite space without a boundary. Not every network has this property; the edge effects in de Bruijn network are dicussed previously.

In figure 5.4, two different toroidal mesh networks are given with nodal connectivity degree of two. They differ by the orientation of the links. Bidirectional Manhattan Street Network (BMSN) is also considered. For these networks, connection rules are defined below; a square network is assumed



Highway Transfer Network

Manhattan Street Network

Figure 5.4: Topology of HTN and MSN.

to keep the definitions simple (i.e. $\mathcal{N} = n^2$ nodes) and every node is addressed by its $(row, column)$ coordinates. In Highway Transfer Network (HTN):

$$(r, c) \text{ is connected to } \begin{cases} ((r+1) \bmod n, c) \\ (r, (c+1) \bmod n) \end{cases} \tag{5.13}$$

and in unidirectional Manhattan Street Network (MSN):

$$(r, c) \text{ is connected to } \begin{cases} ((r+1) \bmod n, c) & \text{if } c \text{ is even} \\ ((r-1) \bmod n, c) & \text{if } c \text{ is odd} \\ (r, (c+1) \bmod n) & \text{if } r \text{ is even} \\ (r, (c-1) \bmod n) & \text{if } r \text{ is odd} \end{cases} \tag{5.14}$$

BMSN offers all these four connections at every node. For these networks, the aforementioned topological properties of interest are listed below.

1. $\mathcal{D}$: For HTN it is $2(n-1)$. For MSN, it is n when $n/2$ is odd and $n+1$ when $n/2$ is even [ChA90]. For BSMN it is $(n-1)$ when n is odd and n when n is even. It is interesting to note that the increased connectivity of BSMN does not improve its diameter over its unidirectional counterpart.

2. $\overline{h}$: For MSNs of different size different formulas are given in [ChA90]. The most succinct form is obtained when $n/2$ is even:

$$\overline{h} = \frac{n}{2}\frac{\mathcal{N}}{\mathcal{N}-1} + \frac{\mathcal{N}-4}{\mathcal{N}-1} \tag{5.15}$$

For BMSN we have ([BoC87, BoC90]);

$$\overline{h} = \begin{cases} \frac{n^3}{2(\mathcal{N}-1)} & \text{when } n \text{ is even} \\ \frac{n}{2} & \text{when } n \text{ is odd} \end{cases} \tag{5.16}$$

For HTN, the following formula can be derived:

$$\overline{h} = \frac{\sum_{i=1}^{n-1} i(i+1) + \sum_{i=1}^{n-1} i(2n-i-1)}{\mathcal{N}-1} \tag{5.17}$$

which simplifies to $\overline{h} = \mathcal{N}/(n+1) \approx n$.

3. *Penalty of deflection*: For HTN, it is n. For MSN, it is 4, which is constant and independent of the network size. For BMSN, it is 2.

4. *Degree of connectivity*: It is 2 for MSN and HTN. It is 4 for BMSN.

In routing, a sequence of interconnection functions that add up to the difference between the source and the destination addresses is used to transfer the data. The execution order is not important, as in the hypercube, therefore multiple shortest paths are available. If the lines are unidirectional, the routing scheme needs to be adjusted according to the topology as being done in the routing scheme of MSN.

Manhattan Street Network

MSN is a regular two connected network designed for packet communications in local or metropolitan area [Max85, Max87]. The topology requires the number of columns and rows to be even and therefore is not defined for arbitrary number of nodes. However, a fractional addressing scheme is also provided that allows an arbitrary number of pairs of rows to be added at any position in the network, as well as a procedure to add one node at a time [Max87]. The latter makes the network less regular and effects the ability of distributed rules to find the shortest path to a destination, thus decreasing the network throughput. Here we will address the the networks with even number of columns and rows.

The routing algorithm works on relative addresses which are derived from the physical addresses. The current node assumes that the destination is located at the center of the network and therefore having its row and column coordinates equal to zero. Then the relative address of the current node is calculated. In an $\mathcal{N} = mn$ integer addressed network, the relative address (r, c) of a node with absolute (physical) address (r_{fr}, c_{fr}) with respect to the destination node with absolute address (r_{to}, c_{to}) is,

$$r = \frac{m}{2} - \left\{ \left(\frac{m}{2} - D_c \left(r_{fr} - r_{to} \right) \right) \bmod m \right\}$$

$$c = \frac{n}{2} - \left\{ \left(\frac{n}{2} - D_r \left(c_{fr} - c_{to} \right) \right) \bmod n \right\}$$

where D_c and D_r are dependent upon the direction of the links at the destination and are either -1 or $+1$. Since the destination lies at the center, the current node has to be located in one of the quadrants.

$$(r, c) \text{ is in } \begin{cases} Q_1 & \text{if } r > 0 \text{ and } c > 0 \\ Q_2 & \text{if } r > 0 \text{ and } c \leq 0 \\ Q_3 & \text{if } r \leq 0 \text{ and } c \leq 0 \\ Q_4 & \text{if } r \leq 0 \text{ and } c > 0 \end{cases}$$

Based on this quadrant location, the preferred links can be determined. For the rules see [Max87].

MSN functions better when the number of rows and columns are equal [MyZ90]. When $\mathcal{N} = n^2$, the throughput is limited to $2n^2/(n/2) = 4n$ since, $\overline{h} = n/2$ and there are $2n^2$ links in the network. The performance of MSN is investigated in [GrG86, Max89] and compared to that of SXN. In [Max89], it is showed that the characteristics of MSN make it well suited for deflection routing, so that even without buffering, 55-70% of the maximum throughput possible (with an infinite number of buffers) can be obtained under uniform traffic load, while the addition of a single buffer increases this figure to 80-90%. A simple flow-control mechanism at the local source preventing the node from transmitting when both of the input links are busy is sufficient to guarantee that no packet loss will occur within the network.

Both networks have simple distributed routing rules. However, while it is easy to add a single node or bypass failed links in MSN, no simple means of doing these exist for SXN. The simulations have provided the following results:

1. If nodes insert packets into empty slots with a probability of p_a, as p_a increases towards unity[26], the throughput does not decrease in MSN. On the contrary, the throughput in the SXN decreases as p_a increases beyond a certain point. The effect becomes more pronounced as the number of nodes in the network increases, because there is a greater penalty for deflecting packets. Consequently, the SXN must be flow-controlled to achieve a throughput rate close to the maximum value. It is noted that the penalty for not flow-controlling in the SXN is between 10% and 50% of the throughput, while the penalty for not flow-controlling in the MSN is less than 3%. Therefore, with uniform load, flow-control is required in the SXN, but not in the MSN.

2. On the other hand, the throughput per node that can be achieved with the SXN is greater than that in the MSN. As the number of nodes increases, the throughput per node decreases more rapidly in the MSN than in the SXN. Although, MSN achieves a higher throughput per node when there are no buffers at the node, as buffers are added, the SXN achieves a higher throughput per node.

This comperative study shows that deflection mechanism is a way of avoiding flow-control and buffering in a network. As for the throughput loss, it can be remedied by increasing the connectivity of the network, so that decreasing the number of hops between two nodes and increasing the number of links in which packets can be deflected. Next we will discuss another 2–connected toroidal network, the Highway Transfer Network (HTN).

Highway Transfer Network

The routing scheme of HTN is suitable for either mesh connected or arbitrary topologies and presented in [KuY90]. A *highway* is defined as a collection of contiguous links with each link belonging to only one highway. The flow of packets on a highway is unidirectional. A highway may be of loop-shaped or consist of a single link. Each node maintains a routing table associated with each incoming link that has entries for every destination. The nodal routing mechanism favors highway connections over cross-highway connections (figures 5.5 and 5.6) and can process the packets going along the highway with less overhead.

The data packet consists of a routing header (RH) and user data (DT) part. The first bit of the RH indicates whether the slot is full of empty. The RH part carries the destination node address of the packet. Each node maintains a routing table associated with each incoming link. This table is indexed by the destination address and contains an outgoing link identifier and the highway indicator information. The highway indicator shows whether the outgoing link on which the packet must be forwarded is part of the highway that the packet arrived on or not. As can be seen in figure 5.6, each incoming link (IL) is associated with a shift register (SR[27]) and a routing table (RT). When the packet is to be forwarded upon the highway, the received slot is fed into the outgoing link (OL). On the other hand, if the packet is to be deflected from one highway to another, the time slot is marked empty and the packet is transferred to the FIFO buffer of the related outgoing link (figure 5.6). In this scheme, the packets going along the highway do not wait in the buffers whereas the FIFO buffer is served only when the time slot in the outgoing link is marked as empty. The basic assumption of the scheme is that the packets *en route* to the destinations are relayed in along-highway mode in most cases, and hence are expected to experience shorter delay compared to other existing methods. Therefore, it is obvious that the key to the success of this scheme is the proper placement of highways. The designers stated that the technique is still in its primitive stage and the optimal design rules were not clear at the time of writing. However, it is argued that with the proper placement of the highways on the network this particular scheme can produce better results than store-and-forward and cut-through switching methods as observed on a square torus network.

[26] Also increasing the number of deflections.

[27] Its length is equal to that of a packet header.

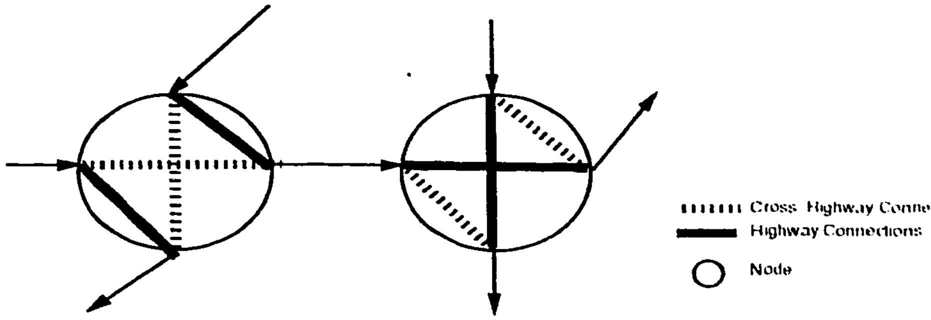


Figure 5.5: Examples of Highway and Cross Highway Connections in HTN.

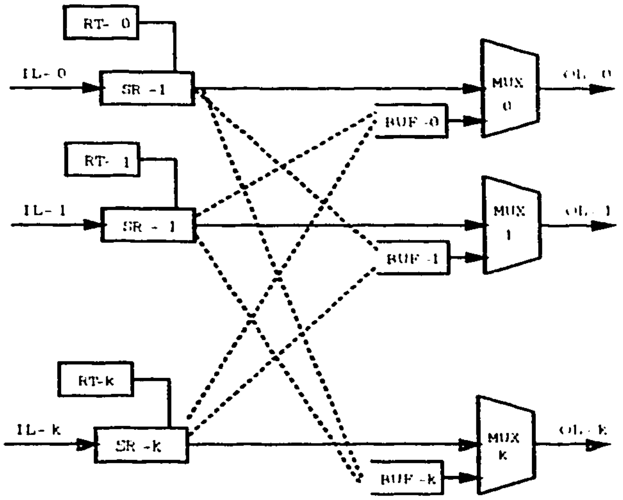


Figure 5.6: Node Configuration.

Triangularly Arranged Network

Another routing strategy that we will investigate is defined on the so-called *Triangularly Arranged Connection Network* (TAC) and described in [MyZ90]. TAC is a 3-connected toroidal network in which nodes are located on vertices of equilateral triangles and referenced by unique cartesian pairs *(x,y)*. The number of nodes needs to be a multiple of 4 in order for the links to be oriented properly, therefore the network is not defined for an arbitrary number of nodes. In this design, the next hop in the path between any two nodes can be found solely by the location of the destination relative to the current node (figure 5.7).

Before going into the details of the algorithm, one geometric property of the network should be clarified: in figure 5.7, the triangle that nodes $(3,1)$, $(5,1)$ and $(4,2)$ form is supposed to be equilateral. Therefore, the difference between the y-coordinates of the node $(3,1)$ and $(4,2)$ is not 1, but instead $\sqrt{3}$. Consequently, in the following algorithm, y-coordinates of the node are multiplied by $\sqrt{3}$ when the distance or the magnitude of some vectors is calculated. The distance between two nodes ($d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$) and the magnitude of vectors ($m = \sqrt{x^2 + y^2}$) can both be evaluated in the standard Euclidean sense. Due to the directionality of the triangular links, there appear to be eight different combinations of output link directions which are denoted by three digits. The binary numbers are chosen such that each bit represents a particular line, and a 1 in

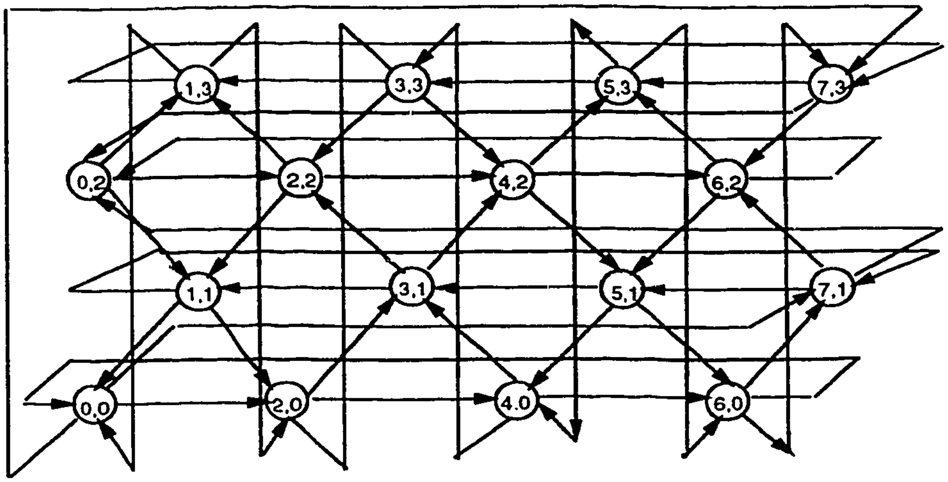


Figure 5.7: Basic 4×4 TAC Network.

any position implies a rightward pointing arrow. The first bit position denotes the main diagonal, the second the off diagonal and the third represents the horizontal line. The routing algorithm is executed in three steps:

Step 1: Suppose that a packet is to be transmitted by node (5, 1) to node (0, 2) in figure 5.8[28]. The relative coordinate of the destination is calculated as $((x_d - x_s, y_d - y_s) = (-5, 1))$. That means, the x-coordinate of the destination is 5 units smaller (thus, the negative sign) and y-coordinate is 1 unit greater than that of source node[29]. In other words, the minus sign of the x-coordinate of the destination shows that the destination lies 5 units left to the current node. By the same token, it is also located 1 unit up when the difference between y-coordinates of the source and the destination is considered.

Step 2: Due to the torus structure, it is clear that the destination can be reached via four different paths. These alternatives and the distances involved are :

1. Route 1 $((-5, 1)$ i.e., 5 units left and 1 unit up from the current coordinate) has the distance of $\sqrt{28}$.

2. Route 2 $((-5, -3)$ i.e., 5 units left and 3 units down from the current coordinate) has the distance of $\sqrt{52}$.

3. Route 3 $((3, 1)$ i.e., 3 units right and 1 unit up from the current coordinate) has the distance of $\sqrt{12}$.

4. Route 4 $((3, -3)$ i.e., 3 units right and 3 units down from the current coordinate) has the distance of $\sqrt{36}$.

Note that the distance calculations have nothing to do with the actual connections. The aim of this step is simply to determine the proper direction in which the packet will be forwarded. These routes and the distance involved in each case are depicted in figure 5.9. The minimum direct distance on the *xy plane* is offered by Route 3.

Step 3: No physical connection exists that can take the packet 3 units right and 1 unit up. Consequently, the algorithm tries to choose the link that can cover as much distance as possible in

[28] To illustrate the torus structure, the destination node is redrawn at the right of the figure.

[29] Remember that the units in the vertical direction are $\sqrt{3}$ times as long as the horizontal units.

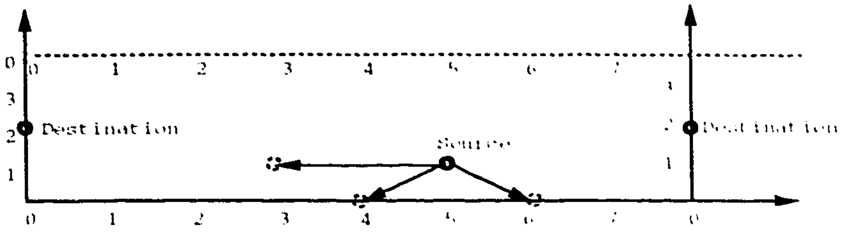


Figure 5.8: A TAC Network Routing Example step 1.

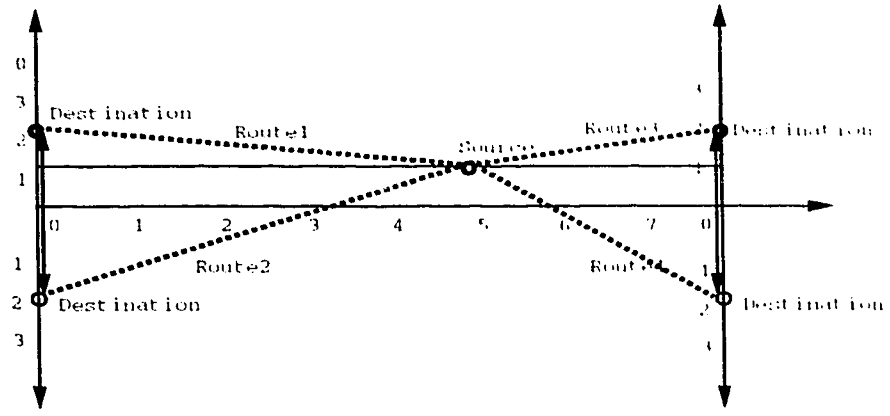


Figure 5.9: A TAC Network Routing Example step 2.

the indicated direction. The minimum distance vector calculated in the previous step enables the algorithm to choose between the three outgoing links available. The best choice is the direction on the main diagonal, leading the packet to node (6, 0) (figure 5.10).

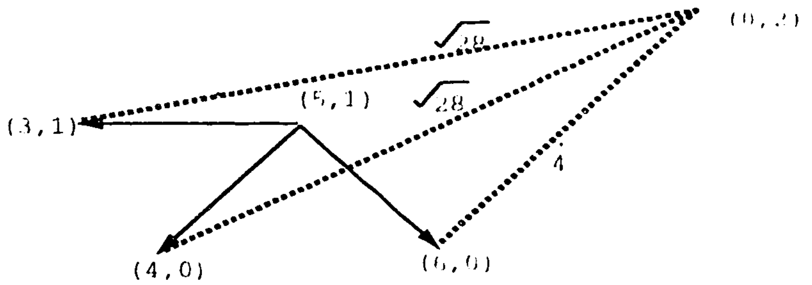


Figure 5.10: A TAC Network Routing Example step 3.

The same operation will be performed in every intermediate node until the packet arrives at its destination. In a simulation that destroys packets that have not reached their destination after 7 hops on a 4×4 TAC network, 90% of the packets were able to reach their designated destinations while queue lengths remained acceptable (fewer than 1/8 of the packets had to wait in queues).

The cost of implementing a TAC network is higher than that of a MSN: there are more links which must be placed and the individual nodes are more complex and thus more expensive. However, the designers argue that the actual incremental cost may be small enough to be justified by the potential increase in performance. The table below summarizes the path length data for the MSN and TAC networks.

Network Size	*Number of links*		*Average Hops*		*Maximum Hops*	
	MSN	TAC	MSN	TAC	MSN	TAC
4x4	32	48	2.93	2.18	5	4
8x8	128	192	5.02	3.76	9	6
12x12	288	432	7.02	5.32	13	10
16x16	512	768	9.02	6.89	17	12

On the other hand, the results are open to criticism: it seems to us that the better performance of TAC is related to the increased connectivity rather than to the triangular vs. rectangular network topology, as claimed in [MyZ90]. For example, we find the maximum hop count to be 6 on a 8×8 MSN when we increase the connectivity of a node from 2 to 3. Moreover, the TAC routing algorithm introduces considerably more processing overhead when compared to that of MSN.

Next we will discuss bidirectional toroidal networks.

Bidirectional Toroidal Networks

Bidirectional toroidal networks that we will discuss are similar to MSN in topology except that half-duplex links are replaced with full-duplex ones. In that case, the unidirectionality of optical fibre requires a two-fold increase in the number of links. Thus, there are no collisions due to two packets travelling on the same link in the opposite directions. The throughput of this topology is limited to $4n^2/(n/2) = 8n$ for the case of uniform traffic patterns and unlimited buffer space.

A number of routing protocols are suggested for this connection pattern. Differences between them lie in the way the contentions are resolved. Note that, although the links are increased by two-fold numberwise, the complexity of contention resolution is increased by more than that: in MSN 2 incoming packets can be assigned to 2 outgoing links in 2 ways whereas the corresponding number is 24 for the bidirectional case.

HR^4-NET [BoC87] has a routing mechanism that organizes the network into two different ring structures at two levels. Low level rings (L-rings, streets) are connected by high level rings (H-rings, avenues) at each node. Each L-ring is identified by its own address which is used as routing information. Each packet travels on the H-ring until it reaches a node which is located on the same L-ring as its destination. The routing decisions are precomputed and stored in a ROM which is addressed according to the preference of the incoming packets (3^4 possible cases; empty, H-ring, L-ring) and link availability (2^4 possibilities; failed, normal). The retrieved memory content is the switching configuration which maximizes the number of packets satisfied with their demands. In any case, the number of misdirected packets is not greater than two and its uniform average is 0.75 when 4 packets are always present. Note that in this mechanism there is no attempt to differentiate between two horizontal and vertical connections according to the packets's destination information. Consequently, $\overline{h} = n^2/(n+1)$ and $H = 4(n+1)$. In [BoC87], the loss of efficiency is justified with the simplcity of implementation. A shortest-path routing scheme for HR^4-NET can be found in [WoK90]. It is a combination of ideas used in MSN (i.e., use of relative addresses and quadrants) and HR^4-NET (i.e., 10^4 precomputed switching decisions).

Another routing scheme applicable to bidirectional toroidal networks is that of SIGnet's (Slotted Interconnected-Grid Network) [ToB90]. Routing in SIGnet makes use of a concept referred to as the preference vector (PV). It is simply an ordered list which indicates the prefered links for a packet, given the quadrant which contains the packet's destination. There is also a secondary counter which indicates the number of remaining hops in vertical or horizontal directions, whichever is the smaller. Six different routing mechanisms are considered in [ToB90]. The first classification is made according to the setting of PV, namely in conformance with orthogonal (O) or diagonal (D) routing. The first algorithm routes the packets in the row direction first, followed by the column direction.

Note that this is the approach employed in HR^4-NET and HTN. The second algorithm attempts to route packets along a stepwise diagonal from source to destination, therefore tries to decrement the larger of the column or row counters of a given packet until a "corner node" is reached. After the packet preferences are determined, contention is resolved according to packet priorities which can be assigned in three different ways: the (S) algorithm sorts the packets according to their secondary counter values and the priority is given to the smallest value. The (R) algorithm resolves contention randomly. The (D) algorithm sorts the packets according to the sums of row and column counters and the priority is given to the smallest value (i.e., to the packet closer to its destination than the other(s)). Consequently, six distinct routing protocols are possible, i.e., OS, OR, OD, DS, DR, DD. Simulation studies show that under uniform traffic and for a wide range of network sizes, the OS algorithm achieves the best overall mean delay and throughput performance.

In [BoC90], another routing algorithm is proposed for BMSN which gives higher priority to packets that are closest to their destinations using a weighted cumulative distance function. The deflections are forced towards the topological antipode with respect to the destination so that the link preferences of the deflected packet will be increased at the next node. With the help of simulations, it is assessed that under this policy the throughput increases as the offered load increases until saturation is reached. However, this result is not general. As reported in [Max89], networks with lesser degrees of connectivity (e.g., MSN and SXN) reach their highest throughput rates before the saturation point and it may happen that it decreases as the load increases towards the saturation [BoC90]. A variation of diagonal routing is also suggested in [BaP89] under the name of Z^2 *(Zig Zag Routing) policy.*

5.6.4 Flooding Networks

In this section we will discuss some implementations whose routing mechanisms are based on flooding. The first three of them, namely, Flooding Sink, Arbnet+ and Noahnet, are designed for small to medium size local area networks and for the reasons that will become clear, are not suitable for metropolitan or wide area networks. They should rather be considered as altenatives to Ethernet-like networks which flood the whole network during transmission. The last one is not limited in scope, but it leads to an unbalanced use of transmission links, especially under heterogeneous traffic patterns.

Flooding Sink

Flooding Sink [HPU86] has a node architecture that makes it possible for a switching element to remember the IDs of the last 255 packets it forwarded previously. Using this mechanism packets visiting a certain node more than once are destroyed by hardware intervention alone.

The Flooding Sink (FS) is a network interface unit with an equal number of input and output ports. It receives messages from other FSs in the network and from its host. Messages have a header containing a source address, a serial number and a destination address. Within a certain time frame, the source address and serial number uniquely identify a message and together they will be referred to as the *message identifier*. Any message arriving on one of the input ports is considered old if the FS remembers it – such a message will be discarded. Otherwise, it is simply sent to all of the other FSs to which it is linked. Each input port of FS is connected to an input buffer. After performing a CRC check on an incoming message, the *eliminator* is addressed using the message identifier. If a match occurs, that means the message is old and should be discarded. Otherwise, the pointers to the start and the end of the message in the input buffer are stored. The destination field is compared to the address of the FS and a bit representing the result is also stored with this pointers.

Arbnet+

The basic routing elements of Arbnet+ are *switches* connected by bidirectional point to point links to form the Arbnet+ Network [Pun89, Pun90]. User devices are connected to the network via *Network Interface Units* (NIUs) that can also be used as multiplexors to support multiple user devices. The packet routing is performed by switches and the *interswitch routing protocol* is topologically indifferent.

The MAC layer of Arbnet+ is quite similar to the IEEE 802.3 CSMA-CD medium access control. When an NIU has a frame to transmit, it listens to the port that is connected to the switch. If no incoming signal is detected at the port, the NIU transmits the frame after an inter frame delay. Should a collision occur during transmission, the NIU stops the transmission immediately and jams the line. Then it enters a random back-off mode. Otherwise the transmission will continue until the end of the frame has been reached. The NIU is then ready to transmit or receive another frame. The reception begins with the detection of an incoming signal. As soon as the address information is gathered, the NIU checks it against its own. If no match occurs, the NIU jams the incoming signal. Otherwise it performs an error check and passes it to the host.

The switch is actually a transceiver and it neither supports store-and-forward transmission nor provides error and flow control. The arbitration of switch contention is based on the principle of first-come-first-served with blocking. The selected frame is repeated on all free output ports *on the fly*. In case of a collision, the switch stops the transmission and sends a *Clear Link Signal* (CLS) on the link on which the collision has occurred. All the switch protocol are performed by hardware.

In Arbnet+, a collision can be of two types: *unintentional* or *intentional*. An unintentional collision occurs due to the frames traveling on the same link in the opposite directions. An intentional collision is created by the switch that cannot accept a data frame for routing. In that case the switch jams the incoming signal by transmitting a CLS. The effect of a collision in Arbnet+ is usually confined to the link where the collision occurs except during the *Clear Backward* process.

A switch can be in one of the three states: *idle*, *routing* and *transmission*. A switch enters the routing state upon detecting an incoming signal at one of its ports. Subsequently, it repeats the incoming frame on its all free ports with only one or two bits delay. During this process, no store-and-forward or consultation of routing table is involved. A frame that arrives at the switch when it is already in routing state or when no free output port is available will not be repeated and a CLS will be generated to the port of arrival.

Arbnet+'s interswitch routing is based on a shortest path tree-search technique, of which a routing tree rooted at the source NIU is formed for each routing attempt. A leaf collapses when a routing transmission along that leaf is blocked[30] and a branch collapses when all the leaves along that branch also collapse. To prevent looping of frames within the network, it is necessary that the root shall not exhaust its transmission before the leaf of the farthest branch begins to collapse. Consequently, this implies a minimum frame size constraint for Arbnet+. The minimum frame size should be longer than the longest possible loop delay, although, shorter frames are reported to be eventually absorbed by the network due to the collisions at the expense of some degradation in performance. Otherwise better delay and throughput rates than Ethernet are observed.

Noahnet

Noahnet is a LAN architecture, implemented at the University of Delaware [FaP86]. Noahnet uses a randomly-connected graph topology, a flooding protocol to route messages and high bandwidth communication media.

In Noahnet, there are three types of messages, namely, *data*, *status* and *command*. Data messages carry the actual information. Status messages are used for two purposes: to indicate if a downstream node has received a message with/without error and to indicate the flood status of a downstream

[30] In other words, jammed by a CLS signal and Clear Backward process began.

node. The flood status of a node can be *forwarding, blocked* or *got to destination* (GTD). At the time of writing, the only command was *stop flooding*. All status messages are transmitted by a downstream node to its immediate upstream node whereas the command messages are transmitted in the opposite direction.

In Noahnet, a switch, upon receiving a message, tries to send it to all unoccupied adjacent nodes. The adjacent nodes also repeat the process until the message reaches its destination or cannot be forwarded anymore. A forwarding node gets flood status messages from its all downstream nodes and sends one resultant status message to its immediate upstream node. Consequently, the path of the message forms a tree rooted at the source node. When a message is looking for its destination, it spans this tree For efficiency, the nodes lying outside of the successful path should be released as quickly as possible. To achieve this, releasing is done both from leaves upwards and from root to leaves. The blocked status message starts releasing leaf nodes and proceeds upwards. Meanwhile, the stop flooding command starts releasing nodes downward from the top of various branches.

The designers argue that the throughput of Noahnet is expected to be better than Ethernet-like or ring LANs since Noahnet allows multiple messages to be active in the network at the same time. Although many nodes get occupied by the same message due to flooding, most of them become free before the transmission of a message is over. On the contrary, every node has to remain occupied for the whole transmission time of a message in case of the Ethernet network.

Controlled Flooding

Controlled Flooding scheme is introduced in [LeR90] and investigated further in [ANR92]. The extent of the flooding is limited by assigning costs to all link traversals and allowing a packet to expand only a limited total cost for network traversals. The source asssigns a numerical value to every packet which is referred as the *wealth* of the packet. In order to traverse a link, the current wealth of the packet must equal or exceed the cost of the link. Upon traversing a link, the cost of it is deducted from the wealth of the packet. Therefore, at every intermediate node, the packet is repeated only on the links that it can afford. To perform the flooding, every node must only know the costs of its outgoing links; no other routing tables are necessary.

It is obvious that the costs assigned to links and the wealth assigned to packets control the scope of the flooding and results in different routing patterns. A heuristic link–cost assignment algorithm that is aimed to obtain a better performance by minimizing the number of nodes that receive every message is also given in [LeR90]. A later study [ANR92] claims that the proposed scheme is not likely to yield a balanced use of resources and compares it to two other routing algorithms which choose the routes along breadth–first search trees and shortest paths.

5.7 Conclusion

In a point–to–point network, the need for routing and flow control is self–evident. The effect of good routing is to increase the throughput for the same value of average delay per packet under high offered load conditions and to decrease average delay per packet under low offered load conditions. Therefore, the design of the routing algorithms is extremely crucial in any network since the two main performance measures are substantially affected by its efficiency — throughput (*quantity of service*) and average message/packet delay (*quality of service*) [BeG87]. The flow and congestion control is also an essential part of any packet switching network architecture to guarantee its performance level (packet loss, packet delay, total network throughput) under unpredictable and changing traffic conditions. The coupling between routing and flow control mechanisms is obvious: as the routing algorithm is more successful in keeping packet delay low, the flow control algorithm allows more traffic into the network [BeG87]. Considering the large variety of the applications that the future high–speed networks are supposed to support, it is obvious that more should be done within shorter time in a high–speed network compared to its conventional counterpart. In this regard, the importance of simple yet effective routing protocols is self evident. In this chapter, we tried to discuss the challenges posed by Gbps networks and examined some proposals designed to meet these challenges As a result, we submit to reader the following observations regarding the design of a routing protocol that can effectively perform in a high-speed networking environment;

1. High transmission speeds force us to return to simplicity There is no time for software intervention and an effective subnet design should solely depend upon very high speed specialized hardware. This point is related to execution (switching) speed and denies any software related approach as well as complexity.

2. Table lookups or long computations to accomplish optimal or near optimal routing introduce nonnegligible delays and therefore should be avoided; one possible implication being the use of packets carrying their own routing information.

3. In case of congestion, the information regarding the current status of the network may never be available to the nodes and switches involved soon enough to guarantee the proper response. Therefore, rather than relying upon backpressure or feedback mechanisms for congestion control, each switch must be able to function on the available local information without degrading the performance. In this regard, some designs enlist the help of routing mechanism to perform some trivial tasks on behalf of the flow and/or congestion control mechanism, such as not accepting a packet from the local source if all the output links are already assigned to transit packets and/or if the packet cannot be forwarded on the shortest path, as well as limiting the number of packets inserted by the local resource according to a certain criteria, i.e , some probability threshold, the intensity of the traffic, etc.

4. Extensive buffering should be avoided for two main reasons: Firstly, it tends to slow down the speed of the switch. Secondly, large buffers may have an adverse effect in congestion control as discussed earlier. On the other hand, small number of buffers are shown to yield a better throughput rate even for perfect'y balanced load, especially when the nodal degree is constant and independent of network size. Lack of the optical equivalent of electronic buffer memories results in the mixed electro-optical solutions of photonic fast packet switches. Although fully optical switches have been demonstrated, their practical applications do not seem to be viable in the near future [JaM93]. Consequently, inclusion of small number of buffers into the switch architecture seems to be feasibie.

5. The addition or deletion of a node should not cause any maintainability problems for the other nodes (assuming that the directly connected nodes can detect the loss of signal on the line and react accordingly). Failed nodes should be bypassed naturally without requiring any global coordination effort on the part of the routing algorithm.

Bibliography

[Aca87] A.S. Acampora, "A Multichannel Multihop Local Lightwave Network", GLOBECOM'87, 1459–1467.

[AcS92] A.S. Acampora, S.I.A. Shah, "Multihop Lightwave Networks: A Comparison of Store-and Forward and Hot Potato Routing", *IEEE Trans. on Communications*, 40, 6 (June 1992), 1082–1090.

[AdD74] P.R. Adby, M.A.H. Dempster, *Introduction to Optimization Methods*, Halsted Press, 1974

[AKH87] A.S. Acampora, M. Karol, M.G. Hluchyj, "Terabit Lightwave Networks: The Multihop Approach", *AT&T Technical Journal*, 66, 6 (November/December 1987), 21–34.

[ANR92] Y. Azar, J. Naor, R. Rom, "Routing Strategies for Fast Networks", INFOCOM'92, 170–179.

[Aya89] E. Ayanoğlu, "Signal-Flow Graphs for Path Enumeration and Deflection Routing Analysis in Multihop Networks", GLOBECOM'89, 1022–1029.

[Bae80] J L. Baer, *Computer Systems Architecture*, Computer Science Press, 1980.

[BaP89] H.G. Badr, S. Podar, "An Optimal Shortest-Path Routing Policy for Network Computers with Regular Mesh-Connected Topologies", *IEEE Trans. on Computers*, 38, 10 (October 1989), 1362–1371.

[BCS90] K. Bala, I. Cidon, K. Sohraby, "Congestion Control for High Speed Packet Switched Networks", INFOCOM'90, 520–526.

[BeG87] D. Bertsekas, R.Gallager, *Data Networks*, Prentice-Hall, 1987.

[BeG92] D. Bertsekas, R.Gallager, *Data Networks, second edition*, Prentice-Hall, 1992.

[BFM90] J.A. Bannister, L. Fratta, M. Gerla, "Topological Design of the Wavelength-Division Optical Network", INFOCOM'90, 1005–1013.

[BoC87] F. Borgonovo, E. Cadorin, "HR^{4}Net: A Hierarchical Random-Routing, Reliable and Reconfigurable Network for Metropolitan Area", INFOCOM'87, 320–326.

[BoC90] F. Borgonovo, E. Cadorin, "Locally-Optimal Routing in the Bidirectional Manhattan Network", INFOCOM'90, 458–464.

[BrT80] W.G. Bridges, S. Toueg, "On the Impossibility of Directed Moore Graphs", *Journal of Combinatorial Theory*, Series B, 29 (1980), 339–341.

[ChA90] T.Y. Chung, D.P. Agrawal, "On the Network Characterization of and Optimal Broadcasting in the Manhattan Street Network", INFOCOM'90, 465–472.

[CLR90] T.H. Cormen, C.E. Leiserson, R.L. Rivest, *Introduction to Algorithms*, McGraw Hill, 1990.

[EiM88] M. Eisenberg, N. Mehravari, "Performance of the Multichannel Multihop Lightwave Network Under Nonuniform Traffic", *IEEE Journal on Selected Areas in Communications*, 6, 7 (August 1988), 1063–1078.

[EsH85] A. Esfahanian, S.L. Hakimi, "Fault-Tolerant Routing in De Bruijn Communication Networks", *IEEE Trans on. Computers*, 34, 9 (September 1985), 777–788.

[FaP86] D. J. Farber, G.M. Parulkar, "A Closer Look at Noahnet", SIGCOMM'86, 205–213.

[Fen81] T. Feng, "A Survey of Interconnection Networks", *IEEE Computer*, December 1981, 12-27.

[GaJ79] M.R. Garey, D.S. Johnson, "Computers and Intractability: A Guide to NP Completeness", W.H. Freeman and Co., 1979.

[GMW92] M. Gumbold, P. Martini, R. Wittenberg, "Temporary Overload in High Speed Backbone Networks", INFOCOM'92, 2280–2289, 1992.

[GoM93] M.X. Goemans, Y. Myung, "A Catalog of Steiner Tree Formulations", *Networks*, 23, 1993, 19–28.

[GrG86] A.G. Greenberg, J. Goodman, "Sharp Approximate Models of Adaptive Routing in Mesh Networks", *Teletraffic Analysis and Computer Performance Evaluation*, Elsevier 1986, 255–270.

[GrH92] A.G. Greenberg, B. Hajek, "Deflection Routing in Hypercube Networks", *IEEE Trans. on Communications*, 40, 6 (June 1992), 1070–1081.

[Haj91] B. Hajek, "Bounds on Evacuation Time for Deflection Routing", *Distributed Computing*, 5 (1991), 1–6.

[Hir91] A. Hiramatsu, "Integration of ATM Call Admission Control and Link Capacity Control by Distributed Neural Networks", *IEEE Journal on Selected Areas in Communications*, September 1991, 1131–1138.

[HlK88] M.G. Hluchyj, M.J. Karol, "ShuffleNet: An Application of Generalized Perfect Shuffles to Multihop Lightwave Networks", INFOCOM'88, 379–390.

[HoP88] N. Homobono, C. Peyrat, "Connectivity of Imase and Itoh Digraphs", *IEEE Trans. on Computers*, 37, 11 (November 1988), 1459–1461.

[HPU86] N. Hutchinson, T. Patten, B. Unger, "The Flooding Sink: A New Approach to Local Area Networking", *Computer Networks and ISDN Systems*, 11 (1986), 1–14.

[Hui90] J.Y. Hui, *Switching and Traffic Theory for Integrated Broadband Networks*, Kluwer Academic Publishers, 1990.

[HwB84] K. Hwang, F.A. Briggs, *Computer Architecture and Parallel Processing*, McGraw Hill, 1984.

[ImI83] M. Imase, M. Itoh, "A Design for Directed Graphs with Minimum Diameter", *IEEE Trans. on Computers*, 32, 8 (August 1983), 782–784.

[ISO85] M. Imase, T. Soneoka, K. Okada, "Connectivity of Regular Directed Graphs with Small Diameters", *IEEE Trans. on Computers*, 34, 3 (March 1985), 267–273.

[Jai90] R. Jain, "Congestion Control in Computer Networks: Issues and Trends", *IEEE Networks Magazine*, 4, 3 (May 1990), 24-30.

[JaM93] A. Jajszczyk, H.T. Mouftah, "Photonic Packet Switching", *IEEE Communications Magazine*, 31, 2 (February 1993), 58-65.

[KaS91] M.J. Karol, S.Z. Shaikh, "A Simple Adaptive Routing Scheme for Congestion Control in ShuffleNet Multihop Lightwave Networks", *IEEE Journal on Selected Areas in Communications*, 9, 7 (September 1991), 1040-1050.

[KAS91] B. Khasnabish, M. Ahmadi, M. Shridhar, "Congestion Avoidance in Large Supra-High-Speed Packet Switching Networks Using Neural Arbiters", GLOBECOM'91, 140-144.

[Kat88] H.P. Katseff, "Incomplete Hypercubes", *IEEE Trans. on Computers*, 37, 5 (May 1988), 604-608.

[Kle92] L. Kleinrock, "The Latency/Bandwidth Tradeoff in Gigabit Networks; Gigabit Networks are Really Different!", *IEEE Communications Magazine*, 30, 4 (April 1992), 36-40.

[KuY90] T. Kubo, K. Yoguchi, "Highway Transfer: A New Forwarding Technique for Real-Time Applications", INFOCOM'90, 403-408.

[LaA91] J.P. Labourdette, A.S. Acampora, "Logically Rearrangeable Multihop Lightwave Networks", *IEEE Trans. on Communications*, 39, 8 (August), 1991, 1223-1230.

[Law76] E. Lawler, *Combinatorial Optimization: Networks and Matroids*, Holt, Rinehart and Winston, 1976.

[LeR90] O. Lesser, R. Rom, "Routing by Controlled Flooding in Communication Networks", INFOCOM'90, 910-917.

[Max85] N.F. Maxemchuk, "Regular Mesh Topologies in Local and Metropolitan Area Networks", *AT&T Technical Journal*, 64, 7 (September 1985), 1659-1685.

[Max87] N.F.Maxemchuk, "Routing in the Manhattan Street Network", *IEEE Trans. on Communications*, 35, 5 (May 1987), 503-512.

[Max89] N.F.Maxemchuk, "Comparison of Deflection and Store-and-Forward Techniques in the Manhattan Street and Shuffle-Exchange Networks", INFOCOM'89, 800-809.

[Max90] N.F.Maxemchuk, "Problems Arising from Deflection Routing: Live-Lock, Congestion and Message Reassembly", Proc. of NATO Workshop on Architecture and Performance Issues of High Capacity Local and Metropolitan Area Networks, 1990, 209-233.

[MoG90] J.A.S. Monteiro, M. Gerla, "Topological Reconfiguration of ATM Networks", INFOCOM'90, 207-214.

[Muk92] B. Mukherjee, "WDM-Based Local Lightwave Networks, Part II: Multihop Systems", *IEEE Network Magazine*, 6, 4 (July 1992), 20-32.

[MyZ90] G.E. Myers, M. E. Zarki, "Routing in TAC — a Triangulary Arranged Network", INFOCOM'90, 481-486.

[OSM90] Y. Oie, T. Suda, M. Murata, D. Kolson, H. Miyahara, "Survey of Switching Techniques in High-Speed Networks and Their Performance", INFOCOM'90, 1242-1251.

[PSU88] A.L. Peressini, F.E. Sullivan, J.J. Uhl, Jr., *The Mathematics of Nonlinear Programming*, Springer-Verlag, 1988.

[Pun89] H. K. Pung, et al., "Arbnet+: An Experimental Mesh like Local Area Network", SICON'89, Singapore, 301–306.

[Pun90] H.K. Pung, et al., "Performance of Arbnet from the Logical Link Control Point of View", Singapore ICCS'90, 1133–1137.

[ReG87] D.A. Reed, D.G. Grunwald, "The Performance of Multicomputer Interconnection Networks", *IEEE Computer*, June 1987, 63–73.

[Rob88] T.G. Robertazzi, "Toroidal Networks", *IEEE Communications Magazine*, 26, 4 (June 1988), 45–50.

[Ro92a] C. Rose, "Mean Internodal Distance in Regular and Random Multihop Networks", *IEEE Trans. on Communications*, 40, 8 (August 1992), 1310–1318.

[Ro92b] C. Rose, "Low Mean Internodal Distance Network Topologies and Simulated Annealing", *IEEE Trans. on Communications*, 40, 8 (August 1992), 1319–1326.

[SaP89] M.R. Samatham, D.J. Pradhan, "The De Bruijn Multiprocessor Network: A Versatile Parallel Processing and Sorting Network for VLSI", *IEEE Trans. on Computers*, 38, 4 (April 1989), 567–581.

[SaS88] Y. Saad, M.H. Schultz, "Topological Properties of Hypercubes", *IEEE Trans. on Computers*, 37, 7 (July 1988), 867–872.

[Sch80] M. Schwartz, "Routing and Flow Control in Data Networks", IBM Research Report 36329, 1980.

[Sch87] M. Schwartz, *Telecommunication Networks, Protocols, Modeling and Analysis*, Addison Wesley, 1987.

[Sed88] R. Sedgewick, *Algorithms, second edition*, Addison Wesley, 1988

[Sie90] H. J. Siegel, *Interconnection Networks for Large-Scale Parallel Processing*, McGraw Hill, 1990.

[SiH88] H.J. Siegel, W.T. Hsu, "Interconnection Networks", chapter 6 in *Computer Architectures, Concepts and Systems*, V.M. Milutinovic, ed., Elsevier Science Publishing, 1988.

[SiR91] K. Sivarajan, R. Ramaswami, "Multihop Lightwave Networks Based on De Bruijn Graphs", INFOCOM'91, 1001–1011.

[SLL81] J.M. Smith, D.T. Lee, J.D. Liebman, "An $O(n \log n)$ Heuristic for Steiner Tree Problems on the Euclidean Metric", *Networks*, 11, 1981, 23–29.

[Sto87] H.S. Stone, *High Performance Computer Architecture*, Addison Wesley, 1987.

[Szy90] T. Szymanski, "An Analysis of Hot-Potato Routing in a Fiber Optic Packet Switched Hypercube", INFOCOM'90, 918–925.

[Tah82] H.A. Taha, *Operation Research, An Introduction*, Collier Macmillan, 1982.

[TrD90] P. Tran-Gia, R. Dittmann, "Performance Analysis of the CRMA-Protocol in High-Speed Networks", Univ. of Würzburg, Institute of Computer Science Research Report Series, Report No. 23, December 1990.

[ToB90] T.D. Todd, A.M. Bignell, "Performance Modelling of SIGnet MAN Backbone", INFOCOM'90, 192–199.

[Tur86] J.S. Turner, "New Directions in Communications", *IEEE Communications Magazine*, 24, 10 (October 1986), 8–15.

[Tur92] J.S. Turner, "Managing Bandwidth in ATM Networks with Bursty Traffic", *IEEE Network Magazine*, 6, 5 (September 1992), 50–58.

[VWD91] R.J. Vetter, K.A. Williams, D.H.C. Du, "Topological Design of Optically Switched WDM Networks", IEEE 742, 114–127.

[Win87] P. Winter, "Steiner Problem in Networks: A Survey", *Networks*, 17, 1987, 126–167.

[WoS89] L. Wong, M. Schwartz, "Flow Control in Metropolitan Area Networks", INFOCOM'89, 826–833.

[YaA87] S. Yalamanchili, J.K. Aggarwal, "A Characterization and Analysis of Parallel Processor Interconnection Networks", *IEEE Trans. on Computers*, 36, 6 (June 1987), 680–691.

[Zha91] L. Zhang, "The Virtual Clock: A New Traffic Control Algorithm for Packet Switching Networks", *ACM Trans. on Computer Systems*, 9, 2 (1991), 101–124.

[ZhA90] Z. Zhang, A.S. Acampora, "Analysis of Multihop Lightwave Networks", GLOBECOM'90, 1873–1879.

Chapter 6

Compass Routing

6.1 Introduction

In the last decade, fiber optic technology has matured to the point where it can support a transmission rate far beyond 1 Gbps. Its impact on the design of network topologies and routing and congestion control schemes was discussed earlier to stress the importance of swifter switch architectures and protocols. The contemporary design proposals also seem to acknowledge this prospect; for example, ATM networks are stripped of most of the congestion and flow control procedures found in conventional networks in order to improve switch throughput [MoG90]. On the other hand, each asset of a packet switching network, such as optimal routing, load distribution, fairness, predictability, packet losslessness, flexibility to satisfy heterogeneous traffic demands, congestion control, etc., comes with its own demand for processing capacity and time. Improving the network throughput and the quality of the service provided by the network are not necessarily mutually compatible goals. For example, it is possible to enhance the throughput by sacrificing fairness. Conversely, a fairness mechanism may reduce utilization. Therefore, a successful routing protocol design should strike a balance between sometimes conflicting demands. In the previous chapter, we stated some observations regarding the design of a routing protocol for high-speed MANs and WANs. In this chapter, we present a routing protocol which is designed according to those observations.

In a packet switching network, the packet delay is the sum of propagation, queueing and switching delays. Each routing decision is aimed at minimizing them in order to increase the overall network throughput. In this regard, a routing scheme may recognize one of these factors as the major contributor to the average packet delay and direct its minimization effort accordingly[1]. To this end, the networks which are primarily designed for multiprocessor computers, with special routing algorithms to minimize the average hop count between source-destination pairs, are suggested as viable alternatives for local and metropolitan area networks. The obvious advantage of this approach is very simple routing procedures which do not require lengthy routing tables or calculations. On a multiprocessor computer backplane, only a few bits of a packet can exist on a link, therefore the exclusion of a link length parameter from routing criteria makes sense. On the other hand, as one moves from LANs to MANs the value of *parameter–a* increases as with the propagation delays. For WANs, as mentioned in the previous chapter, propagation delays dominate switching delays by a large margin. Consequently, minimization of propagation delays takes precedence over the minimization of hop counts. When links are assumed to be of equal length, minimum hop routing also results in relaying the packets along the shortest path, lengthwise. However, for wide area networks, a path of minimum hop count does not necessarily imply a path of minimum distance. The traditional solution to this problem has been the use of routing tables. For high speed networks, embedding

[1] In the absence of extensive buffering, we are concerned with switching and propagation delays only.

a virtual topology into a physical topology has been suggested so that simple routing rules of the virtual topology can be maintained. Examples include shufflenets embedded into multiple rings and hypercubes embedded into minimum spanning trees. For these methods to succeed, it is imperative that the nodes closer in the virtual topology should also be closer in the physical topology. This mapping problem is known to be NP-hard [BFM90]. Regarding this approach, following observations can be made:

1. The routing procedures exercised on a virtual topology cannot offer a shorter path than the physical topology in which it is embedded.

2. A given virtual/physical topology may not be suitable for different networking environments. Regarding the physical topologies of the future high-speed networks, the following classification seems reasonable:

   (a) A richly connected, minimum diameter, regular network topology within a city block (e.g., tree networks, shuffle-based networks).

   (b) A regular, richly connected network topology for an intra-city (metropolitan area) network (e.g., toroidal or grid networks).

   (c) A sparse, mostly planar network topology with nodes generally connected to their nearest neighbours for inter-city (wide area) networks.

   Whether this model is accurate or not is not the issue here. Rather we would like to indicate the implication of a hierachical routing structure just like in the slow networks. This structure introduces many bottleneck points such as bridges and protocol adapters. On the other hand, an integrated structure requires a routing protocol which can perform as effectively as the ad-hoc ones in all these networks so that a standard switch can be conceived. In terms of manufacturing costs and standardization, the appeal of such a scheme is obvious. The main difficulty involved is the design of a routing algorithm which can minimize average hop count when propagation delay is negligible or links are of equal length, and propagation delay otherwise.

3. A given topology may not be easy to maintain. For example, addition and/or deletion of a single node is not easy in shuffle-like networks. Ideally, the addition of a single node should not affect any node other than its immediate neighbours and should not require global reconfiguration.

6.2 Compass Routing

The availability of special knowledge about the problem domain can yield a simpler solution than the one required for a more general case thereby improving the computational efficiency. For example, Dijkstra's shortest path algorithm on graphs requires that all link lengths are positive. One can easily infer the validity of this assumption for data network applications. Another example can be found in [HNR68] which presents an algorithm for the heuristic determination of minimum cost paths on the Euclidean plane for applications such as robot navigation. Since sites are located on the Euclidean plane, the distance matrix satisfies the triangle inequality. In such a case, more effective methods of computing shortest paths and minimum spanning trees can be designed [SLL81, GoB78, SeV86, Fre87].

The multiprocessor computer interconnection networks also provide a similar support to the routing algorithm. For example, in hypercube, a simple *exclusive-OR* operation indicates the output links on the shortest path to a given destination. The existence of these paths and thus the correctness of the routing decision is guaranteed by the underlying topology. In the absence of a similar guarantee, routing tables are required. Trying to avoid the use of routing tables, our routing protocol also is founded upon an assumption regarding the interconnection structure of the net work. But it is more flexible and covers some irregular and regular structures alike making the application domain larger in terms of network topology. This particular assumption and therefore the constraints involved will be discussed after the details of the protocol have been introduced

6.2.1 Description

The proposed scheme supports packet switching and data is assumed to be transmitted in fixed size blocks. Each packet carries its own routing information. Each node only knows the location of the nodes to which it is directly connected. No other information is stored in the nodes. The numbers of incoming and outgoing links are equal. The slot structure consists of the following items:

1. A direction indicator of four bits which are referred as N,S,E,W, (North, South, East, West) individually and named as *compass field* collectively.
2. 2-bit priority information.
3. 2-bit type information (unused).
4. The number of remaining horizontal steps to the destination, x *steps*.
5. The number of remaining vertical steps to the destination, y *steps*. The step counts are related to the the grid structure and their sum may not be equal to the remaining hop count.
6. The segment payload of 48 bytes.

The slot length is in conformance with the ATM cell size and four priority levels is taken from the DQDB standard.

For addressing, we assume that the nodes are located on the cross points of a mesh so that they can be referenced with their (x, y) coordinate pairs which are expressed as integers. The distance between two given nodes can be calculated according to different metrics. With this addressing scheme, it is possible to address $(2^{16} \times 2^{16} = 4,294,967,256)$ possible nodes in North America[2] on a grid with the step size of roughly $133m$.

In a given node, outgoing links have distance and direction information associated with them indicating the location of the nodes to which they are directly connected. The format is the same as of the packet header. The following gives an example for a $node(i, j)$ with degree of connectivity of four.

Link Number	*NSEW*	*x-steps*	*y-steps*	*The Coordinate of the Neighbour*
1	0010	4	0	*node(i,j+4)*
2	0100	0	3	*node(i+3,j)*
3	0101	1	2	*node(i+1,j-2)*
4	1010	2	1	*node(i-2,j+1)*

Routing decisions are made by the cooperation of two types of control units. Each input line is controlled by a separate *line controller* and sychronization and arbitration functions are carried out by a single control unit which we call *arbitrator*. The line controller starts to work as soon as the header is received and builds up an output link preference list for an incoming transient packet. There are four types of links:

[2] North American continent reaches its maximum length between Point Barrow (Alaska) and Punta Mariato (Panama) a total length of roughly 8700 km; its width from the most westernly point of Alaska as far as Canso on the peninsula of Nova Scotia (Canada) measures 5950 km. Our addressing scheme has the resolution of 8700000/65536 = $132.75m$ when the actual distances between nodes are used. In a 1 Gbps network, this distance can be covered roughly in 665 ns.

1. A link can take a packet closer to its destination than the others.

2. A link can forward a packet towards its destination, though covers less distance than some other links.

3. A link may cause a packet to cover some distance in one direction while pushing it further away in the other.

4. A link may cause deflection.

A link that takes a packet towards its destination is called a *feasible link*, regardless of the distance covered. If two links take a packet to the same distance from the destination, the one that reduces the greater of the *x steps* and *y-steps* is considered first. The number of feasible links is reported to the arbitrator along with the priority of the packet. Up to this point, line controllers function in parallel. The arbitrator allocates the links beginning with the packet which has the highest priority and smallest number of feasible links. In the absence of feasible links, the preference list is constructed according to the criteria of minimalising the remaining distance.

6.2.2 Basic Assumptions and Applicability Constraints

Our routing protocol operates on the proximity information of the nodes on the Euclidean plane (and hence the name, Compass Routing). Given the coordinates of a node and the coordinates of its immediate neighbours, the distance between them can be calculated according to a family of metrics. We claim that, with a proper distance metric and with a proper assignment of node coordinates, the following routing criteria performs efficiently in different networking environments:

For every node-destination pair in the network, a packet at an intermediate node prefers the outgoing link which takes it closer to its destination on the Euclidean plane. The geometry of the network:

1. *guarantees that the destination can be reached according to this routing criteria, and*

2. *the path covered is the shortest path available lengthwise.*

An example geometry in which our protocol cannot operate is given in figure 6.1. In the absence of competition, a packet transmitted by $node_1$ to $node_5$ will be relayed to $node_3$ at the first step due to its closeness to the destination. Since $node_3$ is not connected to any node other than $node_1$ and the protocol is designed not to reflect a packet to the node it is received from, the packet has to be absorbed by the $node_3$. Therefore, $node_5$ is not reachable from $node_1$ unless the packet is deflected in the first step. Note that this problem will disappear if this network is presented in a geometrically different but topologically equivalent way. This mapping can be done statically when the network is built.

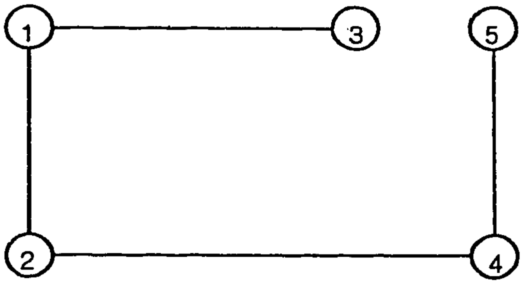


Figure 6.1: An unsuitable topology.

The exact classification of the topologies (thus graphs) that satisfy these two requirements is difficult. However, based on the observations of the infrastructure of the wide area networks in existence, we can establish the validity of the protocol for a well studied group of networks.

1. Wide area networks which tend to be sparse and planar, with nodes which are generally adjacent to their nearest neighbours [GoB78]. The coordinates of the sites to be connected is known in advance and used as they are. The critical decision involves the placement of the links. For such networks, we will introduce a link placement procedure based on Voronoi diagram[3].

2. A grid network for metropolitan area with or without wraparound connections and/or with or without orthogonal connections.

3. A minimum diameter network in which links are of equal length and a path with minimum hop count implies the shortest path lengthwise.

In the next section, we discuss the applicability of Compass Routing for these networks. For the first two cases, the routing protocol is suitable. For minimum diameter networks (de Bruijn network is examined), a proper assignment of node coordinates is not found.

6.3 Applications

6.3.1 Compass Routing in Wide Area Networks

For wide area netwoks, it is assumed that locations of nodes do not follow a regular pattern, similar to the locations of cities on the map. The internodal distances are very large compared to switching delays so that propagation delays dominate. As the network is expanded, new nodes are likely be connected to their nearest neighbours in the Euclidean sense giving rise to a planar network. The planarity is not strict since some nodes can be connected directly with links that violate this particular property, due to the load requirements. However, their percentage is small. Another important point is the consideration of the clusters formed by the users on the map. Since different areas have different populations and therefore different numbers of potential users, the connectivity and bandwidth requirements will vary from region to region. The intra cluster connections are relatively richer than those of inter–cluster connections. Furthermore, it is also logical to assume that the clusters will be connected to each other via their closest pairs in order to reduce the cable length. Note that these observations are in accordance with the telephone networks and imply the importance of proximity and population information in the construction, expansion and resource allocation of a wide area network.

Let us assume that we are given the task of constructing a wide area network along with the coordinates of the sites to be interconnected on the Euclidean plane and the task of designing a fast routing algorithm for this network. Our approach to these problems can be outlined as follows:

1. Construct the convex hull[4] of the given point set.

2. Divide the interior of the convex hull into non–overlapping polygonal regions such that there is exactly one point in each region.

3. Connect the points in the neighbouring regions with direct links. Now the routing problem can be examined with respect to closest point problems and related searching problems on the plane.

[3] Refer to section 3.1 for definition.

[4] The *convex hull* of a set of points S in d–dimensional Euclidean space (E^d) is the boundary of the smallest convex domain in E^d containing S [PrS88].

The key point is how to divide the convex hull into subregions. Given a set of points, there is a planar subdivision[5] algorithm such that the network obtained by connecting the points in the neighbouring regions by direct links has the following characteristics:

1. The number of links used grows linearly with the number of nodes.
2. The shortest path distance offered by the network is less than approximately 2.42 times of the direct distance between a given source-destination pair (i.e., the network 2.42-approximates the complete Euclidean graph of the network.).
3. Addition of a single node requires only setting the values of the registers associated with a single link at its immediate neighbours.
4. The network contains the Euclidean minimum spanning tree as its subgraph.
5. Provides the necessary infrastructure for Compass Routing to operate.

Such a planar subdivision can be achieved using *Voronoi diagrams* which contain all the proximity information defined by a given point set [PrS88]. The set of all points closer to a given point in a point set than all the other points in the set is a geometric structure called the *Voronoi polygon* for the point. The union of all the Voronoi polygons for a point set is called it *Voronoi diagram* [Sed88]. At this point, the following definitions are in order [Lee80]:

Definition 1. Given two points q_i and q_j in the Cartesian plane R^2 with coordinates (x_i, y_i) and (x_j, y_j) respectively, and a real number p, $1 \leq p < \infty$, the distance between them in the $\mathcal{L}_p$ metric is;

$$d_p(q_i, q_j) = \left(|x_i - x_j|^p + |y_i - y_j|^p\right)^{1/p}$$

and in the $\mathcal{L}_\infty$ metric is;

$$d_\infty(q_i, q_j) = \max\left(|x_i - x_j|, |y_i - y_j|\right)$$

Since we are interested only in $\mathcal{L}_2$ (Euclidean) metric in the context of Voronoi diagrams, the subscript p is removed from the following definitions.

Definition 2. The distance between a point q and a set A of points in R^2 is;

$$d(q, A) = \min_{a \in A} d(q, a)$$

and the distance between two sets A and B of points is

$$d(A, B) = \min_{a \in A, b \in B} d(a, b)$$

Definition 3. The bisector $B(q_i, q_j)$ of q_i and q_j is the locus of points equidistant from q_i and q_j, i.e.;

$$B(q_i, q_j) = \left\{r \mid r \in R^2, d(r, q_i) = d(r, q_j)\right\}$$

Definition 4 [LeW80]. Given a set $S = \{q_1, q_2, \cdots q_n\}$ of n points in R^2, the locus of points closer to q_i than to q_j, denoted by $h(q_i, q_j)$, is one of the half planes determined by the bisector $B(q_i, q_j)$, i.e.;

$$h(q_i, q_j) = \left\{r \mid r \in R^2, d(r, q_i) \leq d(r, q_j)\right\}$$

The locus of points closer to q_i than to *any* other point, denoted by $V(q_i)$ is thus given by, $V(q_i) = \cap_{q_j \in S} h(q_i, q_j)$. The region $V(q_i)$ is called the *Voronoi polygon* (not necessarily bounded) associated with q_i.

Some properties of the Voronoi diagrams can be stated as follows under the simplifying assumption that no four points of the given set are cocircular [LeP84]:

[5] A straight line planar embedding of a planar graph determines a partition of the plane called *planar subdivision* or *map*.

1. Every vertex, called Voronoi point of the Voronoi diagram, has degree of three.
2. Every nearest neighbour q_j of point q_i defines an edge of $V(q_i)$ which is a portion of the bisector $B(q_i, q_j)$.
3. $V(q_i)$ is an unbounded polygon iff the point q_i is on the convex hull of set S.
4. The straight–line dual of the Voronoi diagram is a triangulation[6] S known as *Delaunay triangulation* (figure 6.2).

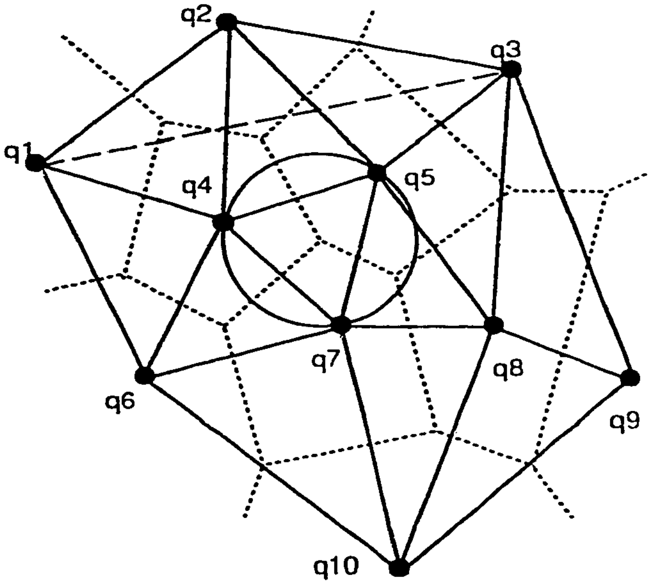


Figure 6.2: Voronoi diagram (dotted) and its dual (black) in $\mathcal{L}_2$ metric.

Assuming that the network is constructed according to Delaunay triangulation, the network:

1. is planar and contains the Euclidean Minimum Spanning Tree (EMST) as its subgraph [PrS85],
2. is constructed with less than $3N$ links for an N node network, since for $N \geq 3$ points, the number of triangles (I) and the number of edges (E) in Delaunay triangulation are:

   $$E = 3(N - 1) - C$$

   $$I = 2(N - 1) - C$$

   where C is the number of vertices on the convex hull [illegible]
3. has a convex boundary and thus is bridgeless[7] for any set not lying on a single line,
4. t–approximates the complete Euclidean graph for $t \approx 2.42$.

Note that the number of required links grows linearly and slowly with the number of nodes. In terms of networking applications in the wide area, the most important property is the ability to approximate the complete Euclidean graph for a small constant. More specifically, let S be any set

[6] A planar subdivision is a *triangulation* if all its bounded regions are triangles [PrS88].

[7] A *bridge* of graph G is an edge whose removal disconnects G [CLR90].

of N points in the plane and let *DT(S)* be the graph of the Delaunay triangulation of S. For all points a and b of S, let *d(a,b)* the shortest Euclidean distance between them and let *DT(a,b)* the length of the shortest path in *DT(S)*. There is a constant $t \leq 2\pi(3\cos(\pi/6)) \approx 2.42$ independent of S and N such that

$$\frac{DT(a,b)}{d(a,b)} \leq t.$$

In other words, *DT(S)* offers an interconnection structure with less than $3N$ links in which the distance between two given nodes is not worse than 2.42 times of the best possible[8] for an N node network. This property of the Delaunay triangulation is examined in [Che86, Che89, DFS87, DFS90, KeG92]. The lower bound for t is known to be $\pi/2 \approx 1.57$ [Che89].

In $DT(S)$, Compass Routing works as follows: *DT(S)* has a convex boundary. Due to convexity, any straight line (representing the shortest possible distance) that connects two nodes, completely lies within the convex hull. Given two points p and q, there is a path between them through the Voronoi polygons contained within the circle C (figure 6.3). On C, p and q are antipodes (i.e., diametrically opposed points) and the length of the diameter defines the maximum value of *DT(a,b)*. A packet needs to be directed along the links which are monotone[9] with respect to line segment $\overline{pq}$. These links can be determined by inspecting the Voronoi polygons that line segment $\overline{pq}$ intersects. For example, in figure 6.2, line segment $\overline{q_1q_3}$ intersects the polygons in which q_1, q_4, q_2, q_5, q_3 are located. Obviously, a path obtained in this manner can be unnecessarily long. To this end, in [KeG92], it is shown that going through the neighbours which are closer to line segment $\overline{pq}$ is sufficient to construct the proper path. In other words, the packet should be kept as close to line segment $\overline{pq}$ as possible. Assume that in figure 6.3, a choice need to be made at p so that the packet destined for q is relayed through either r or s. In our protocol, the link to r is preferred over the link to s if $(l_1 + l_2 < l_3 + l_4)$. Consequently, the packet is relayed on the triangle with the smallest perimeter that takes line segment $\overline{pq}$ as one of its sides. This is the method used in Compass Routing to keep the packet as close to the line segment $\overline{pq}$ as possible. The distance calculations are made in Euclidean metric. However, there is no need for the square root operation since we are interested only with the rank of the distances not their actual values.

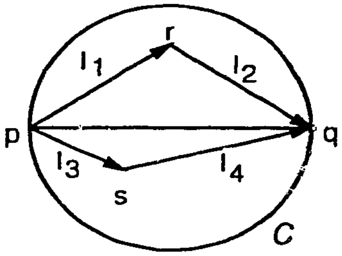


Figure 6.3: Compass Routing on *DT(S)*.

6.3.2 Compass Routing in Grid Networks

In this section, we will discuss the Compass Routing for the four different networks shown in figure 6.4. Note that networks in figure 6.4(a) and figure 6.4(b) can be implemented with or without toroidal links. Figure 6.4(a) without toroidal connections refers to a rectangular mesh. When horizontal and vertical wraparaound links are added to this configuration, the topology of the bidirectional

[8] Such a network requires N^2 links.

[9] A chain $C = (v_1, v_2, \cdots, v_p)$ is a planar straight graph with vertex set $\{v_1, v_2, \cdots v_p\}$ and edge set $\{(v_i, v_{i+1}) \mid i = 1 \cdots p-1\}$. A chain C is said to be *monotone* with respect to a straight line l if the orthogonal projections $\{l(v_1), l(v_2), \cdots, l(v_p)\}$ of the vertices of C on l are as ordered as $(l(v_1), l(v_2), \cdots, l(v_p))$ [LeP77].

Manhattan Street Network is obtained. The network shown in figure 6.4(b), without wraparaound connections, is presented under the name of *Octagonal Mesh* (O Mesh) in [TrM92]. Here we refer to its counterpart with toroidal connections as *c8-Net*.

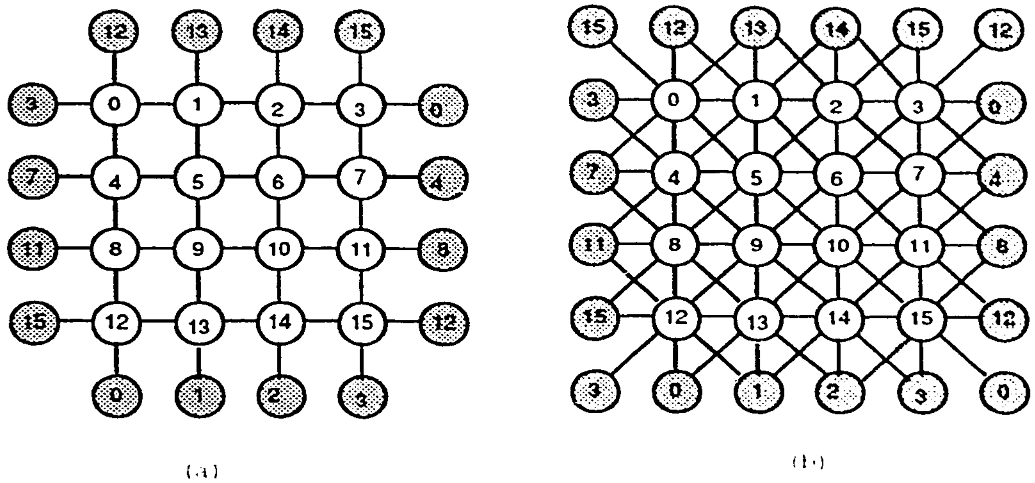


Figure 6.4: 4 × 4 Grid Networks.

The suitability of Compass Routing for grid networks is obvious: A path between any two nodes can be constructed by relaying the packet to a node which is closer to the destination than the current node at each intermediate step. This is exactly what the Compass Routing does. Furthermore, the packets are relayed such that the number of available shortest paths is maximum. This is aimed to reduce the possiblity of deflection. Consider the mesh network first: Assume that, without the loss of generality, a given source is located at coordinates $(0, 0)$ and a given destination is located at (n, m). The distance between them is $d = n + m$ and there are $C(n + m, n)$ equal distance paths through $n \times m$ nodes lying within the rectangular region of which the source and the destination are the opposite corners. In this topology, each routing step can change the value of *x steps* or *y steps*, but not both. In order to keep the number of available shortest paths at its maximum, the shape of the rectangle should be kept as close to a square as possible. Note that when the packet reaches the row/column of the destination, there is only a single shortest path. In Compass Routing, this is the reason for the rule to decrease first the largest of *(x-steps,y-steps)* pair of a given packet. The internodal distance, obviously, is calculated according to $\mathcal{L}_1$ metric.

The extension of the protocol to O-Mesh requires no modification. Note that internodal hop distance between a given source destination pair can be expressed in $\mathcal{L}_\infty$ metric more properly: A packet follows orthogonal links until it reaches the column/row of the destination and then is relayed on a horizontal or vertical direction. The total distance is also proportional to the hop count. This implies a change of distance metric from $\mathcal{L}_1$ to $\mathcal{L}_\infty$ metric: Note that $\mathcal{L}_1$ metric causes some loss of information in orthogonal networks. Consider the transmission between nodes 5 and 3 in figure 6.4(b). There are two equal length paths namely 5 → 6 → 3 and 5 → 2 → 3. If the distance calculations are made in $\mathcal{L}_1$ metric at node 5, the route via node 2 will be preferred over the route via node 6. On the other hand, $\mathcal{L}_\infty$ metric recognizes that both paths are equally preferable[10].

Now we consider the addition of toroidal links to both mesh network and O Mesh in Compass Routing. If the link lengths are long enough so that the propagation delays dominate over the switching delays and the length of the toroidal link is equal to the length of the side of the grid, the

[10] Actually the path via node is 6 better when there are no toroidal links: At node 2 there are two feasible links available, which are the links to nodes 3 and 7. At node 6, the number of feasible links is three (i.e., links to nodes 3, 2 and 7).

use of toroidal links is meaningful only in case of deflections and for transmission between the nodes that they connect directly. Otherwise, a method similar to the mechanism of TAC (see chapter 5) can be implemented: consider the transmission from station 9 to 1 in figure 6.4(a). When toroidal links are present, there are two equal length paths that could be taken. Packet compass field can be set to either (1000,2,0) or (0100,2,0). In other words, packet can be transmitted via station 5 or 13. If there are no wraparound links, the shortest path through station 13 does not exist. Consequently, stations need to be informed about the existence of the toroidal links in order to be able to set the compass fields of the packets correctly. Furthermore, toroidal links presents an opportunity to improve the performance of deflection handling in a way related to congestion control. In an $m \times n$ network with toroidal connections, assume that the values of $m/2$ and $n/2$ are known to the link controller. If in the header of a transient packet, *x-steps* is equal to $n/2$ and/or *y-steps* is equal to $m/2$, it is clear that the packet is in the center of the network with respect to its destination in either one or both directions. This situation also indicates that the original compass setting of the packet led it to a congested region. Note that we do not claim to have enough information to correctly estimate the longevity and the extent of the congestion. The packet may have lost the contention in a single hop away or the current position may be the result of a series of deflections. However, if it is received, for example, from the northernly neighbour with *y-steps* equal to $m/2$, the link controller assumes that the packet has been deflected and changes its compass field so that the southern port is preferred. This approach is aimed to force the packets to explore other feasible directions. For example, consider the transmission from node 9 to 1 again. Assume that station 9 decides to sent the packet through station 5 and sets its compass field to (1000,2,0). However the packet is deflected to station 13 due to contention and its compass field is updated to (1000,3,0). Station 13, knowing that $n/2 = m/2 = 2$ and $3 > 2$, alters the direction of the packet by changing its compass field to (0100,1,0) and delivers the packet to its destination via its southern port.

6.3.3 Compass Routing in Shuffle-like Networks

The networks that we have considered so far have topologies with diameters and average hops counts that grow linearly with the network size. Next, we will discuss logarithmic case of the de Bruijn networks. Compass Routing requires the mapping of this topology onto the Euclidean plane. However, a distance-preserving mapping of this topology does not exist. Consider figure 6.5.

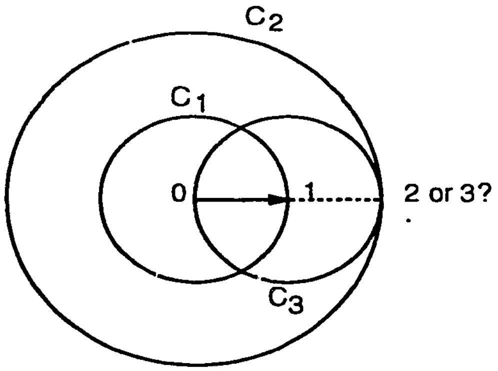


Figure 6.5: Nonexistence of a Distance-Preserving Mapping of de Bruijn Topology into Euclidean Plane.

According to de Bruijn topology, node 0 is one step away from node 1 which is, in turn, one step away from nodes 2 and 3. Furthermore, node 0 is two steps away from nodes 2 and 3. Accordingly, node 1 needs to be on circle C_1 and node 2 and 3 should be placed on C_2. Since nodes 2 and 3 are required to be on C_3 as well, a distance preserving mapping is not possible due to the fact that C_2 and C_3 intersect at a single point only.

Since routing decisions are based on the rank of the distances offered by the links rather than their actual values, another alternative is to look for a mapping which preserves the rank of internodal distances. Such an arrangement is possible for 8 node de Bruijn network (figure 6.6) which works perfectly; an exhaustive test yielded no deviations from the original algorithm. However, when network size is increased to 16, on a similar structure we observed problems (figure 6.7). As an example, consider the transmission from node 13 to 12. The proper route is 13 → 11 → 6 → 12. But, our protocol forwards the packet to node 10 in the first step causing an endless loop, (13 → 10 → 4 → 8 → 1 → 2 → 4 → 8 · · ·). It is obvious that the network should be set up in a different way. After many trials, the arrangement shown in figure 6.8 is found to provide the closest performance to the original. In this case, an exhaustive trial spotted the following deviations, out of all source destination pairs.

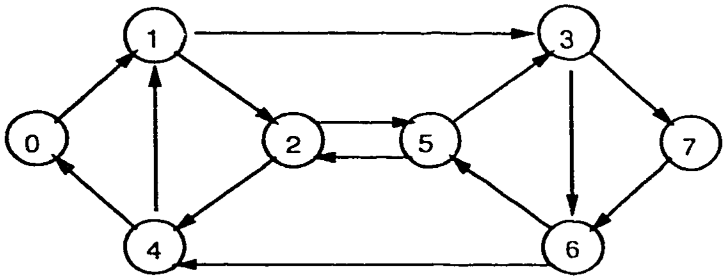


Figure 6.6: 8-node de Bruijn network.

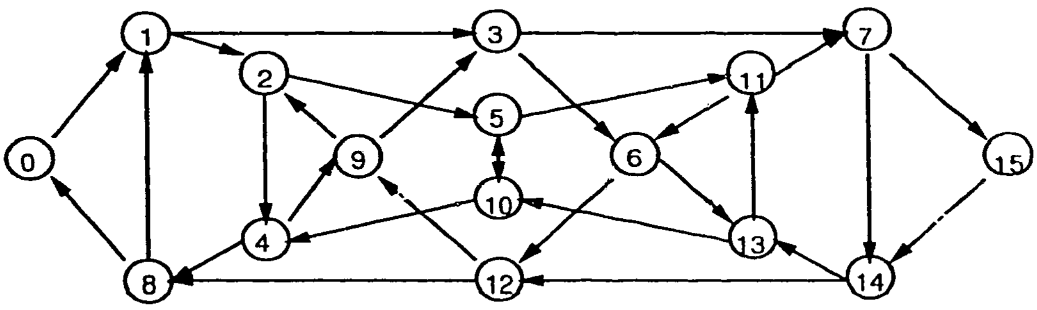


Figure 6.7: 16-node de Bruijn network.

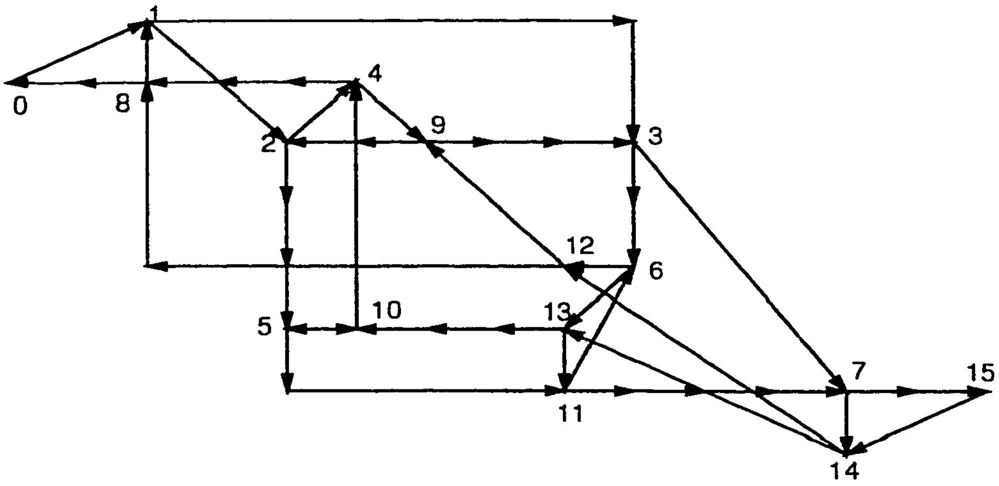


Figure 6.8: 16–node de Bruijn network, redrawn.

from → *to*	Path used by Compass Routing	Path in de Bruijn network
0 → 11	0 → 1 → 3 → 6 → 13 → 11	0 → 1 → 2 → 5 → 11
3 → 4	3 → 6 → 12 → 9 → 2 → 4	3 → 6 → 13 → 10 → 4
4 → 11	4 → 9 → 3 → 6 → 13 → 11	4 → 9 → 2 → 5 → 11
5 → 3	5 → 11 → 6 → 12 → 9 → 3	5 → 10 → 4 → 9 → 3
7 → 4	7 → 14 → 12 → 9 → 2 → 4	7 → 14 → 13 → 10 → 4
7 → 6	7 → 14 → 12 → 9 → 3 → 6	7 → 14 → 13 → 11 → 6
8 → 11	8 → 1 → 3 → 6 → 13 → 11	8 → 1 → 2 → 5 → 11
11 → 4	11 → 6 → 12 → 9 → 2 → 4	11 → 6 → 13 → 10 → 4
12 → 11	12 → 9 → 3 → 6 → 13 → 11	12 → 9 → 2 → 5 → 11
13 → 3	13 → 11 → 6 → 12 → 9 → 3	13 → 10 → 4 → 9 → 3
15 → 4	15 → 14 → 12 → 9 → 2 → 4	15 → 14 → 13 → 10 → 4
15 → 6	15 → 14 → 12 → 9 → 3 → 6	15 → 14 → 13 → 11 → 6

For a de Bruijn graph of 16 nodes, the network diameter is 4 hops. In our protocol, this limit is exceeded by 1, 12 times out of 240 and the total number of hops is greater by almost 4% compared to the corresponding de Bruijn graph in this particular case. This rate can be considered acceptable; however, no similar arrangement is known for configurations with 32 or more nodes[11].

6.4 Conclusion

In this chapter, we presented a routing protocol which is suitable for a number of networks. The protocol is designed to route the packets on the shortest path lengthwise to their destinations. No global connectivity information and/or routing tables are required. The node addition process is simple since only some register values at the directly connected nodes need to be updated and it is not necessary to disseminate this information to other nodes in the network. The protocol is independent of the degree of node connectivity and simple enough to yield an implementation directly in hardware.

[11] For the 16–node graph, the problems disappear if a node uses a 2–hop lookahead. But this is not a solution for larger graphs.

Bibliography

[BFM90] J.A. Bannister, L. Fratta, M. Gerla, "Topological Design of the Wavelength Division Optical Network", INFOCOM'90, 1005–1013.

[Che86] L.P. Chew, "There is a Planar Graph Almost as Good as the Complete Graph", Second Annual Symposium on Computational Geometry, 1986, 169–177.

[Che89] L.P. Chew, "There are Planar Graphs Almost as Good as the Complete Graph", *Journal of Computer and System Sciences*, 39, 1989, 205–219.

[CLR90] T.H. Cormen, C.E. Leiserson, R.L. Rivest, *Introduction to Algorithms*, McGraw Hill, 1990.

[DFS87] D.P. Dobkin, S.J. Friedman, K.J. Supowit, "Delaunay Graphs are Almost as Good as Complete Graphs", 28th Annual Symposium on Foundations of Computing, 1987, 20–26.

[DFS90] D.P. Dobkin, S.J. Friedman, K.J. Supowit, "Delaunay Graphs are Almost as Good as Complete Graphs", *Discrete and Computational Geometry*, 5, 1990, 399–407.

[Fre87] G.N. Frederickson, "Fast Algorithms for Shortest Paths in Planar Graphs with Applications", *SIAM J. Comput.*, 16, 6 (December 1987), 1004–1022.

[GoB78] B.L. Golden, M. Ball, "Shortest Paths with Euclidean Distances: An Explanatory Model", *Networks*, 8 (1978), 297–314.

[HNR68] P.E. Hart, N.J. Nilsson, B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths", *IEEE Trans. on Systems Science and Cyberbetics*, 4, 2 (July 1968), 100–107.

[KeG92] J.M. Keil, C.A. Gutwin, "Classes of Graphs Which Approximate the Complete Euclidean Graph", *Discrete and Computational Geometry*, 7, 1992, 13–28.

[Lee80] D.T. Lee, "Two–Dimensional Voronoi Diagrams in the $\mathcal{L}_p$- Metric", *JACM*, 27, 4 (October 1980), 604–618.

[LeP84] D.T. Lee, F.P. Preparata, "Computational Geometry–A Survey", *IEEE Trans. on Computers*, 33, 12 (December 1984), 1072–1101.

[LeW80] D.T. Lee, C.K. Wong, "Voronoi Diagrams in $\mathcal{L}_1(\mathcal{L}_\infty)$ Metrics with 2-Dimensional Storage Applications", *SIAM J. Comput.*, 9, 1 (February 1980), 200–211.

[MoG90] J.A.S. Monteiro, M. Gerla, "Topological Reconfiguration of ATM Networks", INFOCOM'90, 207–214.

[PrS88] F.P. Preparata, M.I. Shamos, *Computational Geometry, An Introduction*, Springer Verlag, 1985.

[Sed88] R. Sedgewick, *Algorithms, second edition*, Addison-Wesley, 1988.

[SeV86] R. Sedgewick, J.S. Vitter, "Shortest Path in Euclidean Graphs", *Algorithmica*, 1, 1, 1986, 31–48.

[SLL81] J.M. Smith, D.T. Lee, J.D. Liebman, "An $O(n \log n)$ Heuristic for Steiner Tree Problems on the Euclidean Metric", *Networks*, 11, 1981, 23–29.

[TrM92] C. Trefftz, P.K. McKinley, "Performance Evaluation of Wormhole Routing in Octagonal Mesh Direct Networks", International Conference on Parallel and Distributed Systems, 1992, 25–33.

Chapter 7

General Discussion and Conclusions

In this thesis, we proposed two new MAC–level protocols for dual/folded bus networks and one new routing protocol for point–to–point networks. In doing so, our primary aim was to address the requirements of high–speed networks.

The results are presented in two parts. The first part is dedicated to MAC level protocols which provide controlled access to a shared transmission medium. In this environment, no routing is required and the main issue is to provide fairness in bandwidth allocation while maintaining a high aggregate throughput rate. To this end, we proposed CBRMA++/SR and SP/R MAC level protocols and compared them to DQDB. In chapter 2, we outlined the requirements of high speed networks in regard to MAC–level protocol design and discussed the deficiencies of DQDB which mostly stem from the inconsistency of the distributed reservation queue at a given instant. Based on the simulation results, we claim that CBRMA++/SR and SP/R MAC protocols satisfy the requirements of a high speed networking environment better than DQDB.

The DQDB standard provides two access modes: *Pre–Arbitrated (PA) Access* for isochronous connection–oriented traffic, such as voice and video and *Queue Arbitrated (QA) Access* for non-isochronous traffic such as regular data. We considered only the QA access mode. A further study may involve the introduction of *Pre–Arbitrated access* for real time traffic (such as voice packet) into CBRMA++/SR and SP/R along with a priority mechanism and broadcasting/multicasting capabilities.

In the second part, we considered point–to–point networks. In chapter 5, we discussed the new challenges posed by the Gbps networks in the design of routing protocols, network topology and congestion control mechanisms. Following a survey of the contemporary design proposals, we introduced the *Compass Routing* protocol in chapter 6. Compass Routing is suitable for a number of network topologies and does not require any connectivity information and/or routing tables at the nodes. It can be implemented directly in hardware and provides support for deflection routing.

For wide area networks, we discussed a network topology based on Delaunay Triangulation. In Delaunay graphs the degree of connectivity is not uniform across the nodes. Further investigation of graphs produced by bounded–degree planar subdivision methods may yield useful results in terms of a standard switch design. However, the existence of a bounded degree graph which approximates the Complete Euclidean graph is currently an open problem [KeG92].

Another line of further study may involve the investigation of a particular claim made in [BFM90]:

> ***Given a specific network geography and traffic matrix, the choice of a physical topology does not exercise a very strong influence on the performance that can be subsequently attained through the optimization of the virtual topology.***

This claim is in sharp contrast with our approach which constructs the network topology according to the criterion of nodal proximity. Although the method of optimization by simulated annealing and genetic algorithms is limited in applicability due to its dependence on a s.atic traffic matrix, a comparative study should provide valuable insight regarding the performance of Compass Routing on Delaunay graphs.

Bibliography

[BFM90] J.A. Bannister, L. Fratta, M. Gerla, "Topological Design of the Wavelength Division Optical Network", INFOCOM'90, 1005–1013.

[KeG92] J.M. Keil, C.A. Gutwin, "Classes of Graphs Which Approximate the Complete Euclidean Graph", *Discrete and Computational Geometry*, 7, 1992, 13–28.

Arbnet+

The basic routing elements of Arbnet+ are *switches* connected by bidirectional point to point links to form the Arbnet+ Network [Pun89, Pun90] User devices are connected to the network via *Network Interface Units* (NIUs) that can also be used as multiplexors to support multiple user devices The packet routing is performed by switches and the *interswitch routing protocol* is topologically indifferent.

The MAC layer of Arbnet+ is quite similar to the IEEE 802.3 CSMA-CD medium access control. When an NIU has a frame to transmit, it listens to the port that is connected to the switch If no incoming signal is detected at the port, the NIU transmits the frame after an inter frame delay. Should a collision occur during transmission, the NIU stops the transmission immediately and jams the line. Then it enters a random back-off mode. Otherwise the transmission will continue until the end of the frame has been reached. The NIU is then ready to transmit or receive another frame. The reception begins with the detection of an incoming signal. As soon as the address information is gathered, the NIU checks it against its own. If no match occurs, the NIU jams the incoming signal. Otherwise it performs an error check and passes it to the host.

The switch is actually a transceiver and it neither supports store-and-forward transmission nor provides error and flow control. The arbitration of switch contention is based on the principle of first-come-first-served with blocking. The selected frame is repeated on all free output ports *on the fly*. In case of a collision, the switch stops the transmission and sends a *Clear Link Signal* (CLS) on the link on which the collision has occurred. All the switch protocol are performed by hardware

In Arbnet+, a collision can be of two types: *unintentional* or *intentional*. An unintentional collision occurs due to the frames traveling on the same link in the opposite directions. An intentional collision is created by the switch that cannot accept a data frame for routing. In that case the switch jams the incoming signal by transmitting a CLS. The effect of a collision in Arbnet+ is usually confined to the link where the collision occurs except during the *Clear Backward* process

A switch can be in one of the three states: *idle, routing* and *transmission*. A switch enters the routing state upon detecting an incoming signal at one of its ports. Subsequently, it repeats the incoming frame on its all free ports with only one or two bits delay. During this process, no store-and-forward or consultation of routing table is involved. A frame that arrives at the switch when it is already in routing state or when no free output port is available will not be repeated and a CLS will be generated to the port of arrival.

Arbnet+'s interswitch routing is based on a shortest path tree-search technique, of which a routing tree rooted at the source NIU is formed for each routing attempt. A leaf collapses when a routing transmission along that leaf is blocked[30] and a branch collapses when all the leaves along that branch also collapse. To prevent looping of frames within the network, it is necessary that the root shall not exhaust its transmission before the leaf of the farthest branch begins to collapse. Consequently, this implies a minimum frame size constraint for Arbnet+. The minimum frame size should be longer than the longest possible loop delay, although, shorter frames are reported to be eventually absorbed by the network due to the collisions at the expense of some degradation in performance. Otherwise better delay and throughput rates than Ethernet are observed.

Noahnet

Noahnet is a LAN architecture, implemented at the University of Delaware [FaP86]. Noahnet uses a randomly-connected graph topology, a flooding protocol to route messages and high bandwidth communication media.

In Noahnet, there are three types of messages, namely, *data*, *status* and *command*. Data messages carry the actual information. Status messages are used for two purposes: to indicate if a downstream node has received a message with/without error and to indicate the flood status of a downstream

[30] In other words, jammed by a CLS signal and Clear Backward process began.

node. The flood status of a node can be *forwarding, blocked* or *got to destination* (GTD). At the time of writing, the only command was *stop flooding.* All status messages are transmitted by a downstream node to its immediate upstream node whereas the command messages are transmitted in the opposite direction.

In Noahnet, a switch, upon receiving a message, tries to send it to all unoccupied adjacent nodes. The adjacent nodes also repeat the process until the message reaches its destination or cannot be forwarded anymore. A forwarding node gets flood status messages from its all downstream nodes and sends one resultant status message to its immediate upstream node. Consequently, the path of the message forms a tree rooted at the source node. When a message is looking for its destination, it spans this tree For efficiency, the nodes lying outside of the successful path should be released as quickly as possible. To achieve this, releasing is done both from leaves upwards and from root to leaves. The blocked status message starts releasing leaf nodes and proceeds upwards. Meanwhile, the stop flooding command starts releasing nodes downward from the top of various branches.

The designers argue that the throughput of Noahnet is expected to be better than Ethernet-like or ring LANS since Noahnet allows multiple messages to be active in the network at the same time. Although many nodes get occupied by the same message due to flooding, most of them become free before the transmission of a message is over. On the contrary, every node has to remain occupied for the whole transmission time of a message in case of the Ethernet network.

Controlled Flooding

Controlled Flooding scheme is introduced in [LeR90] and investigated further in [ANR92]. The extent of the flooding is limited by assigning costs to all link traversals and allowing a packet to expand only a limited total cost for network traversals. The source asssigns a numerical value to every packet which is referred as the *wealth* of the packet. In order to traverse a link, the current wealth of the packet must equal or exceed the cost of the link. Upon traversing a link, the cost of it is deducted from the wealth of the packet. Therefore, at every intermediate node, the packet is repeated only on the links that it can afford. To perform the flooding, every node must only know the costs of its outgoing links; no other routing tables are necessary.

It is obvious that the costs assigned to links and the wealth assigned to packets control the scope of the flooding and results in different routing patterns. A heuristic link–cost assignment algorithm that is aimed to obtain a better performance by minimizing the number of nodes that receive every message is also given in [LeR90]. A later study [ANR92] claims that the proposed scheme is not likely to yield a balanced use of resources and compares it to two other routing algorithms which choose the routes along breadth-first search trees and shortest paths.

5.7 Conclusion

In a point-to-point network, the need for routing and flow control is self–evident. The effect of good routing is to increase the throughput for the same value of average delay per packet under high offered load conditions and to decrease average delay per packet under low offered load conditions. Therefore, the design of the routing algorithms is extremely crucial in any network since the two main performance measures are substantially affected by its efficiency — throughput (*quantity of service*) and average message/packet delay (*quality of service*) [BeG87]. The flow and congestion control is also an essential part of any packet switching network architecture to guarantee its performance level (packet loss, packet delay, total network throughput) under unpredictable and changing traffic conditions. The coupling between routing and flow control mechanisms is obvious: as the routing algorithm is more successful in keeping packet delay low, the flow control algorithm allows more traffic into the network [BeG87]. Considering the large variety of the applications that the future high–speed networks are supposed to support, it is obvious that more should be done within shorter time in a high-speed network compared to its conventional counterpart. In this regard, the importance of simple yet effective routing protocols is self evident. In this chapter, we tried to discuss the challenges posed by Gbps networks and examined some proposals designed to meet these challenges As a result, we submit to reader the following observations regarding the design of a routing protocol that can effectively perform in a high-speed networking environment;

1. High transmission speeds force us to return to simplicity There is no time for software intervention and an effective subnet design should solely depend upon very high speed specialized hardware. This point is related to execution (switching) speed and denies any software related approach as well as complexity.

2. Table lookups or long computations to accomplish optimal or near optimal routing introduce nonnegligible delays and therefore should be avoided; one possible implication being the use of packets carrying their own routing information.

3. In case of congestion, the information regarding the current status of the network may never be available to the nodes and switches involved soon enough to guarantee the proper response. Therefore, rather than relying upon backpressure or feedback mechanisms for congestion control, each switch must be able to function on the available local information without degrading the performance. In this regard, some designs enlist the help of routing mechanism to perform some trivial tasks on behalf of the flow and/or congestion control mechanism, such as not accepting a packet from the local source if all the output links are already assigned to transit packets and/or if the packet cannot be forwarded on the shortest path, as well as limiting the number of packets inserted by the local resource according to a certain criteria, i.e , some probability threshold, the intensity of the traffic, etc.

4. Extensive buffering should be avoided for two main reasons: Firstly, it tends to slow down the speed of the switch. Secondly, large buffers may have an adverse effect in congestion control as discussed earlier. On the other hand, small number of buffers are shown to yield a better throughput rate even for perfect'y balanced load, especially when the nodal degree is constant and independent of network size. Lack of the optical equivalent of electronic buffer memories results in the mixed electro-optical solutions of photonic fast packet switches. Although fully optical switches have been demonstrated, their practical applications do not seem to be viable in the near future [JaM93]. Consequently, inclusion of small number of buffers into the switch architecture seems to be feasibie.

5. The addition or deletion of a node should not cause any maintainability problems for the other nodes (assuming that the directly connected nodes can detect the loss of signal on the line and react accordingly). Failed nodes should be bypassed naturally without requiring any global coordination effort on the part of the routing algorithm.

Bibliography

[Aca87] A.S. Acampora, "A Multichannel Multihop Local Lightwave Network", GLOBECOM'87, 1459–1467.

[AcS92] A.S. Acampora, S.I.A. Shah, "Multihop Lightwave Networks: A Comparison of Store-and Forward and Hot Potato Routing", *IEEE Trans. on Communications*, 40, 6 (June 1992), 1082–1090.

[AdD74] P.R. Adby, M.A.H. Dempster, *Introduction to Optimization Methods*, Halsted Press, 1974

[AKH87] A.S. Acampora, M. Karol, M.G. Hluchyj, "Terabit Lightwave Networks: The Multihop Approach", *AT&T Technical Journal*, 66, 6 (November/December 1987), 21–34.

[ANR92] Y. Azar, J. Naor, R. Rom, "Routing Strategies for Fast Networks", INFOCOM'92, 170–179.

[Aya89] E. Ayanoğlu, "Signal–Flow Graphs for Path Enumeration and Deflection Routing Analysis in Multihop Networks", GLOBECOM'89, 1022–1029.

[Bae80] J L. Baer, *Computer Systems Architecture*, Computer Science Press, 1980.

[BaP89] H.G. Badr, S. Podar, "An Optimal Shortest–Path Routing Policy for Network Computers with Regular Mesh–Connected Topologies", *IEEE Trans. on Computers*, 38, 10 (October 1989), 1362–1371.

[BCS90] K. Bala, I. Cidon, K. Sohraby, "Congestion Control for High Speed Packet Switched Networks", INFOCOM'90, 520–526.

[BeG87] D. Bertsekas, R.Gallager, *Data Networks*, Prentice-Hall, 1987.

[BeG92] D. Bertsekas, R.Gallager, *Data Networks, second edition*, Prentice-Hall, 1992.

[BFM90] J.A. Bannister, L. Fratta, M. Gerla, "Topological Design of the Wavelength–Division Optical Network", INFOCOM'90, 1005–1013.

[BoC87] F. Borgonovo, E. Cadorin, "HR^{4}Net: A Hierarchical Random-Routing, Reliable and Reconfigurable Network for Metropolitan Area", INFOCOM'87, 320–326.

[BoC90] F. Borgonovo, E. Cadorin, "Locally–Optimal Routing in the Bidirectional Manhattan Network", INFOCOM'90, 458–464.

[BrT80] W.G. Bridges, S. Toueg, "On the Impossibility of Directed Moore Graphs", *Journal of Combinatorial Theory*, Series B, 29 (1980), 339–341.

[ChA90] T.Y. Chung, D.P. Agrawal, "On the Network Characterization of and Optimal Broadcasting in the Manhattan Street Network", INFOCOM'90, 465–472.

[CLR90] T.H. Cormen, C.E. Leiserson, R.L. Rivest, *Introduction to Algorithms*, McGraw Hill, 1990.

[EiM88] M. Eisenberg, N. Mehravari, "Performance of the Multichannel Multihop Lightwave Network Under Nonuniform Traffic", *IEEE Journal on Selected Areas in Communications*, 6, 7 (August 1988), 1063–1078.

[EsH85] A. Esfahanian, S.L. Hakimi, "Fault-Tolerant Routing in De Bruijn Communication Networks", *IEEE Trans on. Computers*, 34, 9 (September 1985), 777–788.

[FaP86] D. J. Farber, G.M. Parulkar, "A Closer Look at Noahnet", SIGCOMM'86, 205–213.

[Fen81] T. Feng, "A Survey of Interconnection Networks", *IEEE Computer*, December 1981, 12-27.

[GaJ79] M.R. Garey, D.S. Johnson, "Computers and Intractability: A Guide to NP Completeness", W.H. Freeman and Co., 1979.

[GMW92] M. Gumbold, P. Martini, R. Wittenberg, "Temporary Overload in High Speed Backbone Networks", INFOCOM'92, 2280–2289, 1992.

[GoM93] M.X. Goemans, Y. Myung, "A Catalog of Steiner Tree Formulations", *Networks*, 23, 1993, 19–28.

[GrG86] A.G. Greenberg, J. Goodman, "Sharp Approximate Models of Adaptive Routing in Mesh Networks", *Teletraffic Analysis and Computer Performance Evaluation*, Elsevier 1986, 255–270.

[GrH92] A.G. Greenberg, B. Hajek, "Deflection Routing in Hypercube Networks", *IEEE Trans. on Communications*, 40, 6 (June 1992), 1070–1081.

[Haj91] B. Hajek, "Bounds on Evacuation Time for Deflection Routing", *Distributed Computing*, 5 (1991), 1–6.

[Hir91] A. Hiramatsu, "Integration of ATM Call Admission Control and Link Capacity Control by Distributed Neural Networks", *IEEE Journal on Selected Areas in Communications*, September 1991, 1131–1138.

[HlK88] M.G. Hluchyj, M.J. Karol, "ShuffleNet: An Application of Generalized Perfect Shuffles to Multihop Lightwave Networks", INFOCOM'88, 379–390.

[HoP88] N. Homobono, C. Peyrat, "Connectivity of Imase and Itoh Digraphs", *IEEE Trans. on Computers*, 37, 11 (November 1988), 1459–1461.

[HPU86] N. Hutchinson, T. Patten, B. Unger, "The Flooding Sink: A New Approach to Local Area Networking", *Computer Networks and ISDN Systems*, 11 (1986), 1–14.

[Hui90] J.Y. Hui, *Switching and Traffic Theory for Integrated Broadband Networks*, Kluwer Academic Publishers, 1990.

[HwB84] K. Hwang, F.A. Briggs, *Computer Architecture and Parallel Processing*, McGraw Hill, 1984.

[ImI83] M. Imase, M. Itoh, "A Design for Directed Graphs with Minimum Diameter", *IEEE Trans. on Computers*, 32, 8 (August 1983), 782–784.

[ISO85] M. Imase, T. Soneoka, K. Okada, "Connectivity of Regular Directed Graphs with Small Diameters", *IEEE Trans. on Computers*, 34, 3 (March 1985), 267–273.

[Jai90] R. Jain, "Congestion Control in Computer Networks: Issues and Trends", *IEEE Networks Magazine*, 4, 3 (May 1990), 24–30.

[JaM93] A. Jajszczyk, H.T. Mouftah, "Photonic Packet Switching", *IEEE Communications Magazine*, 31, 2 (February 1993), 58–65.

[KaS91] M.J. Karol, S.Z. Shaikh, "A Simple Adaptive Routing Scheme for Congestion Control in ShuffleNet Multihop Lightwave Networks", *IEEE Journal on Selected Areas in Communications*, 9, 7 (September 1991), 1040–1050.

[KAS91] B. Khasnabish, M. Ahmadi, M. Shridhar, "Congestion Avoidance in Large Supra-High-Speed Packet Switching Networks Using Neural Arbiters", GLOBECOM'91, 140–144.

[Kat88] H.P. Katseff, "Incomplete Hypercubes", *IEEE Trans. on Computers*, 37, 5 (May 1988), 604–608.

[Kle92] L. Kleinrock, "The Latency/Bandwidth Tradeoff in Gigabit Networks; Gigabit Networks are Really Different!", *IEEE Communications Magazine*, 30, 4 (April 1992), 36–40.

[KuY90] T. Kubo, K. Yoguchi, "Highway Transfer: A New Forwarding Technique for Real-Time Applications", INFOCOM'90, 403–408.

[LaA91] J.P. Labourdette, A.S. Acampora, "Logically Rearrangeable Multihop Lightwave Networks", *IEEE Trans. on Communications*, 39, 8 (August), 1991, 1223–1230.

[Law76] E. Lawler, *Combinatorial Optimization: Networks and Matroids*, Holt, Rinehart and Winston, 1976.

[LeR90] O. Lesser, R. Rom, "Routing by Controlled Flooding in Communication Networks", INFOCOM'90, 910-917.

[Max85] N.F. Maxemchuk, "Regular Mesh Topologies in Local and Metropolitan Area Networks", *AT&T Technical Journal*, 64, 7 (September 1985), 1659–1685.

[Max87] N.F.Maxemchuk, "Routing in the Manhattan Street Network", *IEEE Trans. on Communications*, 35, 5 (May 1987), 503–512.

[Max89] N.F.Maxemchuk, "Comparison of Deflection and Store-and-Forward Techniques in the Manhattan Street and Shuffle-Exchange Networks", INFOCOM'89, 800–809.

[Max90] N.F.Maxemchuk, "Problems Arising from Deflection Routing: Live-Lock, Congestion and Message Reassembly", Proc. of NATO Workshop on Architecture and Performance Issues of High Capacity Local and Metropolitan Area Networks, 1990, 209–233.

[MoG90] J.A.S. Monteiro, M. Gerla, "Topological Reconfiguration of ATM Networks", INFOCOM'90, 207–214.

[Muk92] B. Mukherjee, "WDM-Based Local Lightwave Networks, Part II: Multihop Systems", *IEEE Network Magazine*, 6, 4 (July 1992), 20–32.

[MyZ90] G.E. Myers, M. E. Zarki, "Routing in TAC — a Triangulary Arranged Network", INFOCOM'90, 481–486.

[OSM90] Y. Oie, T. Suda, M. Murata, D. Kolson, H. Miyahara, "Survey of Switching Techniques in High-Speed Networks and Their Performance", INFOCOM'90, 1242–1251.

[PSU88] A.L. Peressini, F.E. Sullivan, J.J. Uhl, Jr., *The Mathematics of Nonlinear Programming*, Springer-Verlag, 1988.

[Pun89] H. K. Pung, et al., "Arbnet+: An Experimental Mesh like Local Area Network", SICON'89, Singapore, 301–306.

[Pun90] H.K. Pung, et al., "Performance of Arbnet from the Logical Link Control Point of View", Singapore ICCS'90, 1133–1137.

[ReG87] D.A. Reed, D.G. Grunwald, "The Performance of Multicomputer Interconnection Networks", *IEEE Computer*, June 1987, 63–73.

[Rob88] T.G. Robertazzi, "Toroidal Networks", *IEEE Communications Magazine*, 26, 4 (June 1988), 45–50.

[Ro92a] C. Rose, "Mean Internodal Distance in Regular and Random Multihop Networks", *IEEE Trans. on Communications*, 40, 8 (August 1992), 1310–1318.

[Ro92b] C. Rose, "Low Mean Internodal Distance Network Topologies and Simulated Annealing", *IEEE Trans. on Communications*, 40, 8 (August 1992), 1319–1326.

[SaP89] M.R. Samatham, D.J. Pradhan, "The De Bruijn Multiprocessor Network: A Versatile Parallel Processing and Sorting Network for VLSI", *IEEE Trans. on Computers*, 38, 4 (April 1989), 567–581.

[SaS88] Y. Saad, M.H. Schultz, "Topological Properties of Hypercubes", *IEEE Trans. on Computers*, 37, 7 (July 1988), 867–872.

[Sch80] M. Schwartz, "Routing and Flow Control in Data Networks", IBM Research Report 36329, 1980.

[Sch87] M. Schwartz, *Telecommunication Networks, Protocols, Modeling and Analysis*, Addison Wesley, 1987.

[Sed88] R. Sedgewick, *Algorithms, second edition*, Addison Wesley, 1988

[Sie90] H. J. Siegel, *Interconnection Networks for Large-Scale Parallel Processing*, McGraw Hill, 1990.

[SiH88] H.J. Siegel, W.T. Hsu, "Interconnection Networks", chapter 6 in *Computer Architectures, Concepts and Systems*, V.M. Milutinovic, ed., Elsevier Science Publishing, 1988.

[SiR91] K. Sivarajan, R. Ramaswami, "Multihop Lightwave Networks Based on De Bruijn Graphs", INFOCOM'91, 1001–1011.

[SLL81] J.M. Smith, D.T. Lee, J.D. Liebman, "An $O(n \log n)$ Heuristic for Steiner Tree Problems on the Euclidean Metric", *Networks*, 11, 1981, 23–29.

[Sto87] H.S. Stone, *High Performance Computer Architecture*, Addison Wesley, 1987.

[Szy90] T. Szymanski, "An Analysis of Hot-Potato Routing in a Fiber Optic Packet Switched Hypercube", INFOCOM'90, 918–925.

[Tah82] H.A. Taha, *Operation Research, An Introduction*, Collier Macmillan, 1982.

[TrD90] P. Tran-Gia, R. Dittmann, "Performance Analysis of the CRMA-Protocol in High Speed Networks", Univ. of Würzburg, Institute of Computer Science Research Report Series, Report No. 23, December 1990.

[ToB90] T.D. Todd, A.M. Bignell, "Performance Modelling of SIGnet MAN Backbone", INFOCOM'90, 192–199.

[Tur86] J.S. Turner, "New Directions in Communications", *IEEE Communications Magazine*, 24, 10 (October 1986), 8–15.

[Tur92] J.S. Turner, "Managing Bandwidth in ATM Networks with Bursty Traffic", *IEEE Network Magazine*, 6, 5 (September 1992), 50–58.

[VWD91] R.J. Vetter, K.A. Williams, D.H.C. Du, "Topological Design of Optically Switched WDM Networks", IEEE 742, 114–127.

[Win87] P. Winter, "Steiner Problem in Networks: A Survey", *Networks*, 17, 1987, 126–167.

[WoS89] L. Wong, M. Schwartz, "Flow Control in Metropolitan Area Networks", INFOCOM'89, 826–833.

[YaA87] S. Yalamanchili, J.K. Aggarwal, "A Characterization and Analysis of Parallel Processor Interconnection Networks", *IEEE Trans. on Computers*, 36, 6 (June 1987), 680–691.

[Zha91] L. Zhang, "The Virtual Clock: A New Traffic Control Algorithm for Packet Switching Networks", *ACM Trans. on Computer Systems*, 9, 2 (1991), 101–124.

[ZhA90] Z. Zhang, A.S. Acampora, "Analysis of Multihop Lightwave Networks", GLOBECOM'90, 1873–1879.

Chapter 6

Compass Routing

6.1 Introduction

In the last decade, fiber optic technology has matured to the point where it can support a transmission rate far beyond 1 Gbps. Its impact on the design of network topologies and routing and congestion control schemes was discussed earlier to stress the importance of swifter switch architectures and protocols. The contemporary design proposals also seem to acknowledge this prospect; for example, ATM networks are stripped of most of the congestion and flow control procedures found in conventional networks in order to improve switch throughput [MoG90]. On the other hand, each asset of a packet switching network, such as optimal routing, load distribution, fairness, predictability, packet losslessness, flexibility to satisfy heterogeneous traffic demands, congestion control, etc., comes with its own demand for processing capacity and time. Improving the network throughput and the quality of the service provided by the network are not necessarily mutually compatible goals. For example, it is possible to enhance the throughput by sacrificing fairness. Conversely, a fairness mechanism may reduce utilization. Therefore, a successful routing protocol design should strike a balance between sometimes conflicting demands. In the previous chapter, we stated some observations regarding the design of a routing protocol for high-speed MANs and WANs. In this chapter, we present a routing protocol which is designed according to those observations.

In a packet switching network, the packet delay is the sum of propagation, queueing and switching delays. Each routing decision is aimed at minimizing them in order to increase the overall network throughput. In this regard, a routing scheme may recognize one of these factors as the major contributor to the average packet delay and direct its minimization effort accordingly[1]. To this end, the networks which are primarily designed for multiprocessor computers, with special routing algorithms to minimize the average hop count between source-destination pairs, are suggested as viable alternatives for local and metropolitan area networks. The obvious advantage of this approach is very simple routing procedures which do not require lengthy routing tables or calculations. On a multiprocessor computer backplane, only a few bits of a packet can exist on a link, therefore the exclusion of a link length parameter from routing criteria makes sense. On the other hand, as one moves from LANs to MANs the value of *parameter–a* increases as with the propagation delays. For WANs, as mentioned in the previous chapter, propagation delays dominate switching delays by a large margin. Consequently, minimization of propagation delays takes precedence over the minimization of hop counts. When links are assumed to be of equal length, minimum hop routing also results in relaying the packets along the shortest path, lengthwise. However, for wide area networks, a path of minimum hop count does not necessarily imply a path of minimum distance. The traditional solution to this problem has been the use of routing tables. For high speed networks, embedding

[1] In the absence of extensive buffering, we are concerned with switching and propagation delays only.

a virtual topology into a physical topology has been suggested so that simple routing rules of the virtual topology can be maintained. Examples include shufflenets embedded into multiple rings and hypercubes embedded into minimum spanning trees. For these methods to succeed, it is imperative that the nodes closer in the virtual topology should also be closer in the physical topology. This mapping problem is known to be NP-hard [BFM90]. Regarding this approach, following observations can be made:

1. The routing procedures exercised on a virtual topology cannot offer a shorter path than the physical topology in which it is embedded.

2. A given virtual/physical topology may not be suitable for different networking environments. Regarding the physical topologies of the future high-speed networks, the following classification seems reasonable:

   (a) A richly connected, minimum diameter, regular network topology within a city block (e.g., tree networks, shuffle-based networks).

   (b) A regular, richly connected network topology for an intra-city (metropolitan area) network (e.g., toroidal or grid networks).

   (c) A sparse, mostly planar network topology with nodes generally connected to their nearest neighbours for inter-city (wide area) networks.

   Whether this model is accurate or not is not the issue here. Rather we would like to indicate the implication of a hierachical routing structure just like in the slow networks. This structure introduces many bottleneck points such as bridges and protocol adapters. On the other hand, an integrated structure requires a routing protocol which can perform as effectively as the ad-hoc ones in all these networks so that a standard switch can be conceived. In terms of manufacturing costs and standardization, the appeal of such a scheme is obvious. The main difficulty involved is the design of a routing algorithm which can minimize average hop count when propagation delay is negligible or links are of equal length, and propagation delay otherwise.

3. A given topology may not be easy to maintain. For example, addition and/or deletion of a single node is not easy in shuffle-like networks. Ideally, the addition of a single node should not affect any node other than its immediate neighbours and should not require global reconfiguration.

6.2 Compass Routing

The availability of special knowledge about the problem domain can yield a simpler solution than the one required for a more general case thereby improving the computational efficiency. For example, Dijkstra's shortest path algorithm on graphs requires that all link lengths are positive. One can easily infer the validity of this assumption for data network applications. Another example can be found in [HNR68] which presents an algorithm for the heuristic determination of minimum cost paths on the Euclidean plane for applications such as robot navigation. Since sites are located on the Euclidean plane, the distance matrix satisfies the triangle inequality. In such a case, more effective methods of computing shortest paths and minimum spanning trees can be designed [SLL81, GoB78, SeV86, Fre87].

The multiprocessor computer interconnection networks also provide a similar support to the routing algorithm. For example, in hypercube, a simple *exclusive-OR* operation indicates the output links on the shortest path to a given destination. The existence of these paths and thus the correctness of the routing decision is guaranteed by the underlying topology. In the absence of a similar guarantee, routing tables are required. Trying to avoid the use of routing tables, our routing protocol also is founded upon an assumption regarding the interconnection structure of the network. But it is more flexible and covers some irregular and regular structures alike making the application domain larger in terms of network topology. This particular assumption and therefore the constraints involved will be discussed after the details of the protocol have been introduced

6.2.1 Description

The proposed scheme supports packet switching and data is assumed to be transmitted in fixed size blocks. Each packet carries its own routing information. Each node only knows the location of the nodes to which it is directly connected. No other information is stored in the nodes. The numbers of incoming and outgoing links are equal. The slot structure consists of the following items:

1. A direction indicator of four bits which are referred as N, S, E, W, (North, South, East, West) individually and named as *compass field* collectively.
2. 2-bit priority information.
3. 2-bit type information (unused).
4. The number of remaining horizontal steps to the destination, x *steps*.
5. The number of remaining vertical steps to the destination, y *steps*. The step counts are related to the the grid structure and their sum may not be equal to the remaining hop count.
6. The segment payload of 48 bytes.

The slot length is in conformance with the ATM cell size and four priority levels is taken from the DQDB standard.

For addressing, we assume that the nodes are located on the cross points of a mesh so that they can be referenced with their (x, y) coordinate pairs which are expressed as integers. The distance between two given nodes can be calculated according to different metrics. With this addressing scheme, it is possible to address ($2^{16} \times 2^{16} = 4,294,967,256$) possible nodes in North America[2] on a grid with the step size of roughly $133m$.

In a given node, outgoing links have distance and direction information associated with them indicating the location of the nodes to which they are directly connected. The format is the same as of the packet header. The following gives an example for a $node(i, j)$ with degree of connectivity of four.

Link Number	*NSEW*	*x-steps*	*y-steps*	*The Coordinate of the Neighbour*
1	0010	4	0	*node(i,j+4)*
2	0100	0	3	*node(i+3,j)*
3	0101	1	2	*node(i+1,j-2)*
4	1010	2	1	*node(i-2,j+1)*

Routing decisions are made by the cooperation of two types of control units. Each input line is controlled by a separate ***line controller*** and sychronization and arbitration functions are carried out by a single control unit which we call ***arbitrator***. The line controller starts to work as soon as the header is received and builds up an output link preference list for an incoming transient packet. There are four types of links:

[2] North American continent reaches its maximum length between Point Barrow (Alaska) and Punta Mariato (Panama) a total length of roughly 8700 km; its width from the most westernly point of Alaska as far as Canso on the peninsula of Nova Scotia (Canada) measures 5950 km. Our addressing scheme has the resolution of $8700000/65536 = 132.75m$ when the actual distances between nodes are used. In a 1 Gbps network, this distance can be covered roughly in 665 ns.

1. A link can take a packet closer to its destination than the others.

2. A link can forward a packet towards its destination, though covers less distance than some other links.

3. A link may cause a packet to cover some distance in one direction while pushing it further away in the other.

4. A link may cause deflection.

A link that takes a packet towards its destination is called a *feasible link*, regardless of the distance covered. If two links take a packet to the same distance from the destination, the one that reduces the greater of the *x steps* and *y-steps* is considered first. The number of feasible links is reported to the arbitrator along with the priority of the packet. Up to this point, line controllers function in parallel. The arbitrator allocates the links beginning with the packet which has the highest priority and smallest number of feasible links. In the absence of feasible links, the preference list is constructed according to the criteria of minimalising the remaining distance.

6.2.2 Basic Assumptions and Applicability Constraints

Our routing protocol operates on the proximity information of the nodes on the Euclidean plane (and hence the name, Compass Routing). Given the coordinates of a node and the coordinates of its immediate neighbours, the distance between them can be calculated according to a family of metrics. We claim that, with a proper distance metric and with a proper assignment of node coordinates, the following routing criteria performs efficiently in different networking environments:

For every node-destination pair in the network, a packet at an intermediate node prefers the outgoing link which takes it closer to its destination on the Euclidean plane. The geometry of the network:

1. guarantees that the destination can be reached according to this routing criteria, and

2. the path covered is the shortest path available lengthwise.

An example geometry in which our protocol cannot operate is given in figure 6.1. In the absence of competition, a packet transmitted by $node_1$ to $node_5$ will be relayed to $node_3$ at the first step due to its closeness to the destination. Since $node_3$ is not connected to any node other than $node_1$ and the protocol is designed not to reflect a packet to the node it is received from, the packet has to be absorbed by the $node_3$. Therefore, $node_5$ is not reachable from $node_1$ unless the packet is deflected in the first step. Note that this problem will disappear if this network is presented in a geometrically different but topologically equivalent way. This mapping can be done statically when the network is built.

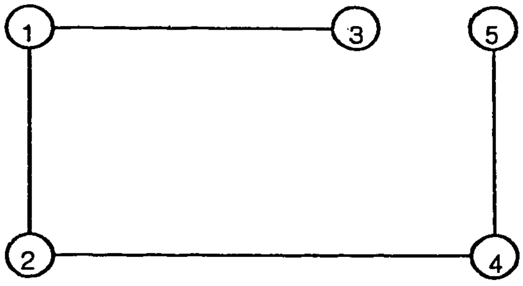


Figure 6.1: An unsuitable topology.

The exact classification of the topologies (thus graphs) that satisfy these two requirements is difficult. However, based on the observations of the infrastructure of the wide area networks in existence, we can establish the validity of the protocol for a well studied group of networks.

1. Wide area networks which tend to be sparse and planar, with nodes which are generally adjacent to their nearest neighbours [GoB78]. The coordinates of the sites to be connected is known in advance and used as they are. The critical decision involves the placement of the links. For such networks, we will introduce a link placement procedure based on Voronoi diagram[3].

2. A grid network for metropolitan area with or without wraparound connections and/or with or without orthogonal connections.

3. A minimum diameter network in which links are of equal length and a path with minimum hop count implies the shortest path lengthwise.

In the next section, we discuss the applicability of Compass Routing for these networks. For the first two cases, the routing protocol is suitable. For minimum diameter networks (de Bruijn network is examined), a proper assignment of node coordinates is not found.

6.3 Applications

6.3.1 Compass Routing in Wide Area Networks

For wide area netwoks, it is assumed that locations of nodes do not follow a regular pattern, similar to the locations of cities on the map. The internodal distances are very large compared to switching delays so that propagation delays dominate. As the network is expanded, new nodes are likely be connected to their nearest neighbours in the Euclidean sense giving rise to a planar network. The planarity is not strict since some nodes can be connected directly with links that violate this particular property, due to the load requirements. However, their percentage is small. Another important point is the consideration of the clusters formed by the users on the map. Since different areas have different populations and therefore different numbers of potential users, the connectivity and bandwidth requirements will vary from region to region. The intra cluster connections are relatively richer than those of inter-cluster connections. Furthermore, it is also logical to assume that the clusters will be connected to each other via their closest pairs in order to reduce the cable length. Note that these observations are in accordance with the telephone networks and imply the importance of proximity and population information in the construction, expansion and resource allocation of a wide area network.

Let us assume that we are given the task of constructing a wide area network along with the coordinates of the sites to be interconnected on the Euclidean plane and the task of designing a fast routing algorithm for this network. Our approach to these problems can be outlined as follows:

1. Construct the convex hull[4] of the given point set.

2. Divide the interior of the convex hull into non-overlapping polygonal regions such that there is exactly one point in each region.

3. Connect the points in the neighbouring regions with direct links. Now the routing problem can be examined with respect to closest point problems and related searching problems on the plane.

[3] Refer to section 3.1 for definition.

[4] The *convex hull* of a set of points S in d-dimensional Euclidean space (E^d) is the boundary of the smallest convex domain in E^d containing S [PrS88].

The key point is how to divide the convex hull into subregions. Given a set of points, there is a planar subdivision[5] algorithm such that the network obtained by connecting the points in the neighbouring regions by direct links has the following characteristics:

1. The number of links used grows linearly with the number of nodes.
2. The shortest path distance offered by the network is less than approximately 2.42 times of the direct distance between a given source-destination pair (i.e., the network 2.42-approximates the complete Euclidean graph of the network.).
3. Addition of a single node requires only setting the values of the registers associated with a single link at its immediate neighbours.
4. The network contains the Euclidean minimum spanning tree as its subgraph.
5. Provides the necessary infrastructure for Compass Routing to operate.

Such a planar subdivision can be achieved using *Voronoi diagrams* which contain all the proximity information defined by a given point set [PrS88]. The set of all points closer to a given point in a point set than all the other points in the set is a geometric structure called the *Voronoi polygon* for the point. The union of all the Voronoi polygons for a point set is called it *Voronoi diagram* [Sed88]. At this point, the following definitions are in order [Lee80]:

Definition 1 Given two points q_i and q_j in the Cartesian plane R^2 with coordinates (x_i, y_i) and (x_j, y_j) respectively, and a real number p, $1 \leq p < \infty$, the distance between them in the $\mathcal{L}_p$ metric is;

$$d_p(q_i, q_j) = \left(|x_i - x_j|^p + |y_i - y_j|^p\right)^{1/p}$$

and in the $\mathcal{L}_\infty$ metric is;

$$d_\infty(q_i, q_j) = \max(|x_i - x_j|, |y_i - y_j|)$$

Since we are interested only in $\mathcal{L}_2$ (Euclidean) metric in the context of Voronoi diagrams, the subscript p is removed from the following definitions.

Definition 2 The distance between a point q and a set A of points in R^2 is;

$$d(q, A) = \min_{a \in A} d(q, a)$$

and the distance between two sets A and B of points is

$$d(A, B) = \min_{a \in A, b \in B} d(a, b)$$

Definition 3 The bisector $B(q_i, q_j)$ of q_i and q_j is the locus of points equidistant from q_i and q_j, i.e.;

$$B(q_i, q_j) = \left\{ r \mid r \in R^2, d(r, q_i) = d(r, q_j) \right\}$$

Definition 4 [LeW80] Given a set $S = \{q_1, q_2, \cdots q_n\}$ of n points in R^2, the locus of points closer to q_i than to q_j, denoted by $h(q_i, q_j)$, is one of the half planes determined by the bisector $B(q_i, q_j)$, i.e.;

$$h(q_i, q_j) = \left\{ r \mid r \in R^2, d(r, q_i) \leq d(r, q_j) \right\}$$

The locus of points closer to q_i than to *any* other point, denoted by $V(q_i)$ is thus given by, $V(q_i) = \cap_{q_j \in S} h(q_i, q_j)$. The region $V(q_i)$ is called the *Voronoi polygon* (not necessarily bounded) associated with q_i.

Some properties of the Voronoi diagrams can be stated as follows under the simplifying assumption that no four points of the given set are cocircular [LeP84]:

[5] A straight line planar embedding of a planar graph determines a partition of the plane called *planar subdivision* or *map*.

1. Every vertex, called Voronoi point of the Voronoi diagram, has degree of three.
2. Every nearest neighbour q_j of point q_i defines an edge of $V(q_i)$ which is a portion of the bisector $B(q_i, q_j)$.
3. $V(q_i)$ is an unbounded polygon iff the point q_i is on the convex hull of set S.
4. The straight-line dual of the Voronoi diagram is a triangulation[6] S known as *Delaunay triangulation* (figure 6.2).

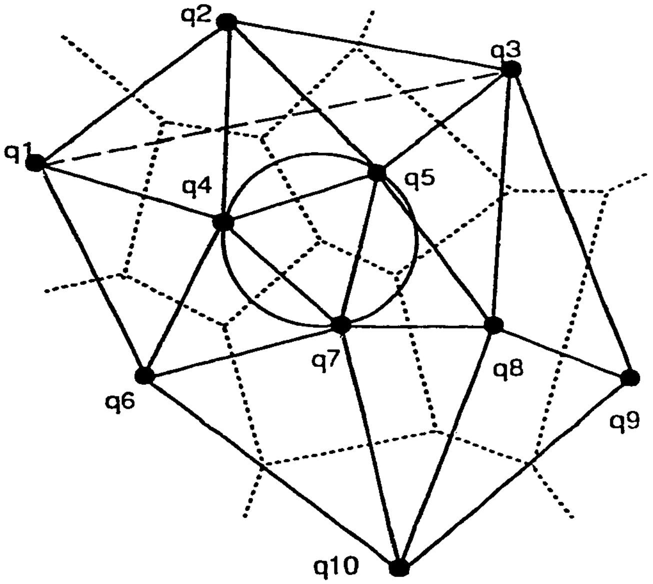


Figure 6.2: Voronoi diagram (dotted) and its dual (black) in $\mathcal{L}_2$ metric.

Assuming that the network is constructed according to Delaunay triangulation, the network:

1. is planar and contains the Euclidean Minimum Spanning Tree (EMST) as its subgraph [PrS85],
2. is constructed with less than $3N$ links for an N node network, since for $N \geq 3$ points, the number of triangles (I) and the number of edges (E) in Delaunay triangulation are:

$$E = 3(N - 1) - C$$

$$I = 2(N - 1) - C$$

   where C is the number of vertices on the convex hull [illegible]

3. has a convex boundary and thus is bridgeless[7] for any set not lying on a single line,
4. t-approximates the complete Euclidean graph for $t \approx 2.42$.

Note that the number of required links grows linearly and slowly with the number of nodes. In terms of networking applications in the wide area, the most important property is the ability to approximate the complete Euclidean graph for a small constant. More specifically, let S be any set

[6] A planar subdivision is a *triangulation* if all its bounded regions are triangles [PrS88].

[7] A *bridge* of graph G is an edge whose removal disconnects G [CLR90].

of N points in the plane and let *DT(S)* be the graph of the Delaunay triangulation of S. For all points a and b of S, let *d(a,b)* the shortest Euclidean distance between them and let *DT(a,b)* the length of the shortest path in *DT(S)*. There is a constant $t \leq 2\pi(3\cos(\pi/6)) \approx 2.42$ independent of S and N such that

$$\frac{DT(a,b)}{d(a,b)} \leq t.$$

In other words, *DT(S)* offers an interconnection structure with less than $3N$ links in which the distance between two given nodes is not worse than 2.42 times of the best possible[8] for an N node network. This property of the Delaunay triangulation is examined in [Che86, Che89, DFS87, DFS90, KeG92]. The lower bound for t is known to be $\pi/2 \approx 1.57$ [Che89].

In $DT(S)$, Compass Routing works as follows: *DT(S)* has a convex boundary. Due to convexity, any straight line (representing the shortest possible distance) that connects two nodes, completely lies within the convex hull. Given two points p and q, there is a path between them through the Voronoi polygons contained within the circle C (figure 6.3). On C, p and q are antipodes (i.e., diametrically opposed points) and the length of the diameter defines the maximum value of *DT(a,b)*. A packet needs to be directed along the links which are monotone[9] with respect to line segment $\overline{pq}$. These links can be determined by inspecting the Voronoi polygons that line segment $\overline{pq}$ intersects. For example, in figure 6.2, line segment $\overline{q_1q_3}$ intersects the polygons in which q_1, q_4, q_2, q_5, q_3 are located. Obviously, a path obtained in this manner can be unnecessarily long. To this end, in [KeG92], it is shown that going through the neighbours which are closer to line segment $\overline{pq}$ is sufficient to construct the proper path. In other words, the packet should be kept as close to line segment $\overline{pq}$ as possible. Assume that in figure 6.3, a choice need to be made at p so that the packet destined for q is relayed through either r or s. In our protocol, the link to r is preferred over the link to s if $(l_1 + l_2 < l_3 + l_4)$. Consequently, the packet is relayed on the triangle with the smallest perimeter that takes line segment $\overline{pq}$ as one of its sides. This is the method used in Compass Routing to keep the packet as close to the line segment $\overline{pq}$ as possible. The distance calculations are made in Euclidean metric. However, there is no need for the square root operation since we are interested only with the rank of the distances not their actual values.

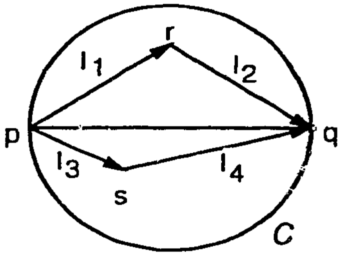


Figure 6.3: Compass Routing on *DT(S)*.

6.3.2 Compass Routing in Grid Networks

In this section, we will discuss the Compass Routing for the four different networks shown in figure 6.4. Note that networks in figure 6.4(a) and figure 6.4(b) can be implemented with or without toroidal links. Figure 6.4(a) without toroidal connections refers to a rectangular mesh. When horizontal and vertical wraparaound links are added to this configuration, the topology of the bidirectional

[8] Such a network requires N^2 links.

[9] A chain $C = (v_1, v_2, \cdots, v_p)$ is a planar straight graph with vertex set $\{v_1, v_2, \cdots v_p\}$ and edge set $\{(v_i, v_{i+1}) \mid i = 1 \cdots p-1\}$. A chain C is said to be *monotone* with respect to a straight line l if the orthogonal projections $\{l(v_1), l(v_2), \cdots, l(v_p)\}$ of the vertices of C on l are as ordered as $(l(v_1), l(v_2), \cdots, l(v_p))$ [LeP77].

Manhattan Street Network is obtained. The network shown in figure 6.4(b), without wraparound connections, is presented under the name of *Octagonal Mesh* (O-Mesh) in [TrM92]. Here we refer to its counterpart with toroidal connections as *cS-Net*.

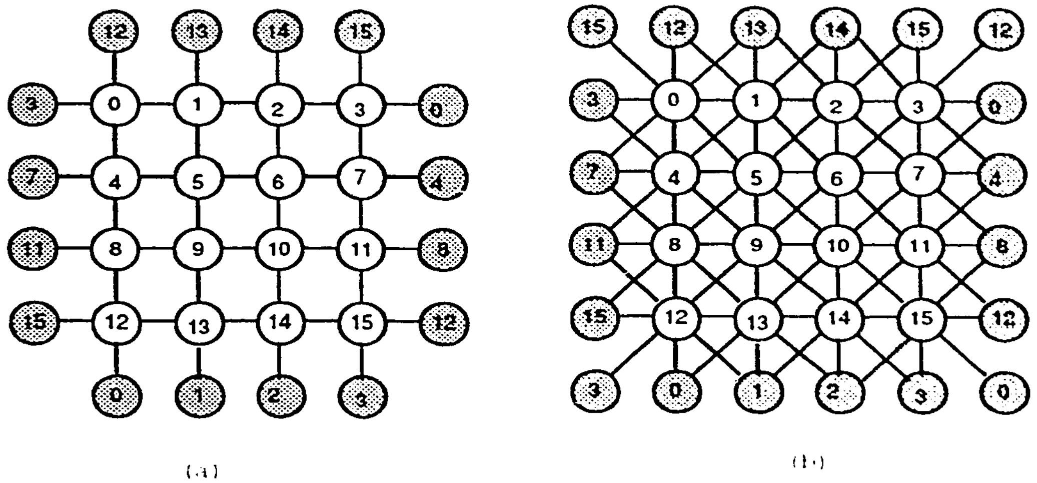


Figure 6.4: 4 × 4 Grid Networks.

The suitability of Compass Routing for grid networks is obvious: A path between any two nodes can be constructed by relaying the packet to a node which is closer to the destination than the current node at each intermediate step. This is exactly what the Compass Routing does. Furthermore, the packets are relayed such that the number of available shortest paths is maximum. This is aimed to reduce the possiblity of deflection. Consider the mesh network first: Assume that, without the loss of generality, a given source is located at coordinates $(0,0)$ and a given destination is located at (n,m). The distance between them is $d = n + m$ and there are $C(n+m,n)$ equal distance paths through $n \times m$ nodes lying within the rectangular region of which the source and the destination are the opposite corners. In this topology, each routing step can change the value of *x steps* or *y steps*, but not both. In order to keep the number of available shortest paths at its maximum, the shape of the rectangle should be kept as close to a square as possible. Note that when the packet reaches the row/column of the destination, there is only a single shortest path. In Compass Routing, this is the reason for the rule to decrease first the largest of *(x-steps,y-steps)* pair of a given packet. The internodal distance, obviously, is calculated according to $\mathcal{L}_1$ metric.

The extension of the protocol to O-Mesh requires no modification. Note that internodal hop distance between a given source destination pair can be expressed in $\mathcal{L}_\infty$ metric more properly: A packet follows orthogonal links until it reaches the column/row of the destination and then is relayed on a horizontal or vertical direction. The total distance is also proportional to the hop count. This implies a change of distance metric from $\mathcal{L}_1$ to $\mathcal{L}_\infty$ metric: Note that $\mathcal{L}_1$ metric causes some loss of information in orthogonal networks. Consider the transmission between nodes 5 and 3 in figure 6.4(b). There are two equal length paths namely 5 → 6 → 3 and 5 → 2 → 3. If the distance calculations are made in $\mathcal{L}_1$ metric at node 5, the route via node 2 will be preferred over the route via node 6. On the other hand, $\mathcal{L}_\infty$ metric recognizes that both paths are equally preferable[10].

Now we consider the addition of toroidal links to both mesh network and O Mesh in Compass Routing. If the link lengths are long enough so that the propagation delays dominate over the switching delays and the length of the toroidal link is equal to the length of the side of the grid, the

[10] Actually the path via node is 6 better when there are no toroidal links: At node 2 there are two feasible links available, which are the links to nodes 3 and 7. At node 6, the number of feasible links is three (i.e., links to nodes 3, 2 and 7).

use of toroidal links is meaningful only in case of deflections and for transmission between the nodes that they connect directly. Otherwise, a method similar to the mechanism of TAC (see chapter 5) can be implemented: consider the transmission from station 9 to 1 in figure 6.4(a). When toroidal links are present, there are two equal length paths that could be taken. Packet compass field can be set to either (1000,2,0) or (0100,2,0). In other words, packet can be transmitted via station 5 or 13. If there are no wraparound links, the shortest path through station 13 does not exist. Consequently, stations need to be informed about the existence of the toroidal links in order to be able to set the compass fields of the packets correctly. Furthermore, toroidal links presents an opportunity to improve the performance of deflection handling in a way related to congestion control. In an $m \times n$ network with toroidal connections, assume that the values of $m/2$ and $n/2$ are known to the link controller. If in the header of a transient packet, *x-steps* is equal to $n/2$ and/or *y-steps* is equal to $m/2$, it is clear that the packet is in the center of the network with respect to its destination in either one or both directions. This situation also indicates that the original compass setting of the packet led it to a congested region. Note that we do not claim to have enough information to correctly estimate the longevity and the extent of the congestion. The packet may have lost the contention in a single hop away or the current position may be the result of a series of deflections. However, if it is received, for example, from the northernly neighbour with *y-steps* equal to $m/2$, the link controller assumes that the packet has been deflected and changes its compass field so that the southern port is preferred. This approach is aimed to force the packets to explore other feasible directions. For example, consider the transmission from node 9 to 1 again. Assume that station 9 decides to sent the packet through station 5 and sets its compass field to (1000,2,0). However the packet is deflected to station 13 due to contention and its compass field is updated to (1000,3,0). Station 13, knowing that $n/2 = m/2 = 2$ and $3 > 2$, alters the direction of the packet by changing its compass field to (0100,1,0) and delivers the packet to its destination via its southern port.

6.3.3 Compass Routing in Shuffle–like Networks

The networks that we have considered so far have topologies with diameters and average hops counts that grow linearly with the network size. Next, we will discuss logarithmic case of the de Bruijn networks. Compass Routing requires the mapping of this topology onto the Euclidean plane. However, a distance–preserving mapping of this topology does not exist. Consider figure 6.5.

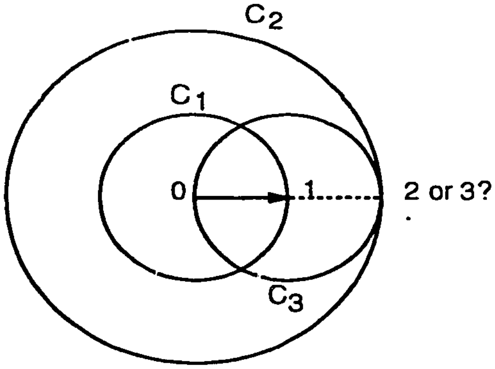


Figure 6.5: Nonexistence of a Distance–Preserving Mapping of de Bruijn Topology into Euclidean Plane.

According to de Bruijn topology, node 0 is one step away from node 1 which is, in turn, one step away from nodes 2 and 3. Furthermore, node 0 is two steps away from nodes 2 and 3. Accordingly, node 1 needs to be on circle C_1 and node 2 and 3 should be placed on C_2. Since nodes 2 and 3 are required to be on C_3 as well, a distance preserving mapping is not possible due to the fact that C_2 and C_3 intersect at a single point only.

Since routing decisions are based on the rank of the distances offered by the links rather than their actual values, another alternative is to look for a mapping which preserves the rank of internodal distances. Such an arrangement is possible for 8-node de Bruijn network (figure 6.6) which works perfectly; an exhaustive test yielded no deviations from the original algorithm. However, when network size is increased to 16, on a similar structure we observed problems (figure 6.7). As an example, consider the transmission from node 13 to 12. The proper route is 13 → 11 → 6 → 12. But, our protocol forwards the packet to node 10 in the first step causing an endless loop, (13 → 10 → 4 → 8 → 1 → 2 → 4 → 8 · · ·). It is obvious that the network should be set up in a different way. After many trials, the arrangement shown in figure 6.8 is found to provide the closest performance to the original. In this case, an exhaustive trial spotted the following deviations, out of all source destination pairs.

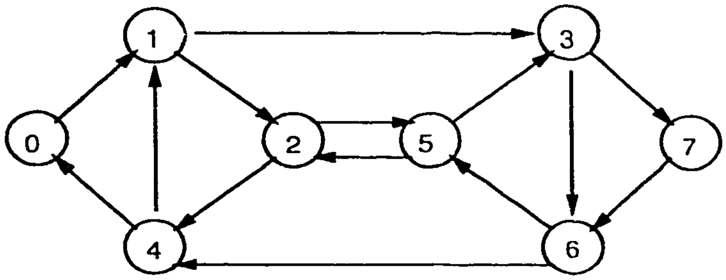


Figure 6.6: 8-node de Bruijn network.

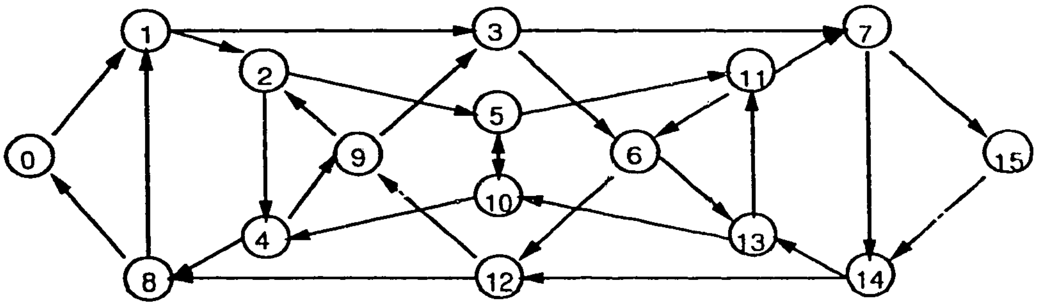


Figure 6.7: 16-node de Bruijn network.

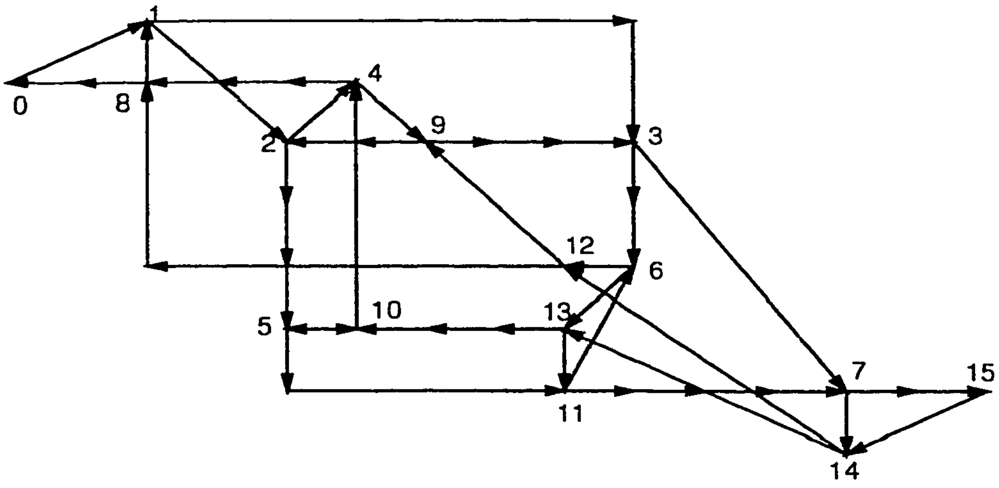


Figure 6.8: 16–node de Bruijn network, redrawn.

from → *to*	Path used by Compass Routing	Path in de Bruijn network
0 → 11	0 → 1 → 3 → 6 → 13 → 11	0 → 1 → 2 → 5 → 11
3 → 4	3 → 6 → 12 → 9 → 2 → 4	3 → 6 → 13 → 10 → 4
4 → 11	4 → 9 → 3 → 6 → 13 → 11	4 → 9 → 2 → 5 → 11
5 → 3	5 → 11 → 6 → 12 → 9 → 3	5 → 10 → 4 → 9 → 3
7 → 4	7 → 14 → 12 → 9 → 2 → 4	7 → 14 → 13 → 10 → 4
7 → 6	7 → 14 → 12 → 9 → 3 → 6	7 → 14 → 13 → 11 → 6
8 → 11	8 → 1 → 3 → 6 → 13 → 11	8 → 1 → 2 → 5 → 11
11 → 4	11 → 6 → 12 → 9 → 2 → 4	11 → 6 → 13 → 10 → 4
12 → 11	12 → 9 → 3 → 6 → 13 → 11	12 → 9 → 2 → 5 → 11
13 → 3	13 → 11 → 6 → 12 → 9 → 3	13 → 10 → 4 → 9 → 3
15 → 4	15 → 14 → 12 → 9 → 2 → 4	15 → 14 → 13 → 10 → 4
15 → 6	15 → 14 → 12 → 9 → 3 → 6	15 → 14 → 13 → 11 → 6

For a de Bruijn graph of 16 nodes, the network diameter is 4 hops. In our protocol, this limit is exceeded by 1, 12 times out of 240 and the total number of hops is greater by almost 4% compared to the corresponding de Bruijn graph in this particular case. This rate can be considered acceptable; however, no similar arrangement is known for configurations with 32 or more nodes[11].

6.4 Conclusion

In this chapter, we presented a routing protocol which is suitable for a number of networks. The protocol is designed to route the packets on the shortest path lengthwise to their destinations. No global connectivity information and/or routing tables are required. The node addition process is simple since only some register values at the directly connected nodes need to be updated and it is not necessary to disseminate this information to other nodes in the network. The protocol is independent of the degree of node connectivity and simple enough to yield an implementation directly in hardware.

[11] For the 16-node graph, the problems disappear if a node uses a 2–hop lookahead. But this is not a solution for larger graphs.

Bibliography

[BFM90] J.A. Bannister, L. Fratta, M. Gerla, "Topological Design of the Wavelength Division Optical Network", INFOCOM'90, 1005–1013.

[Che86] L.P. Chew, "There is a Planar Graph Almost as Good as the Complete Graph", Second Annual Symposium on Computational Geometry, 1986, 169–177.

[Che89] L.P. Chew, "There are Planar Graphs Almost as Good as the Complete Graph", *Journal of Computer and System Sciences*, 39, 1989, 205–219.

[CLR90] T.H. Cormen, C.E. Leiserson, R.L. Rivest, *Introduction to Algorithms*, McGraw Hill, 1990.

[DFS87] D.P. Dobkin, S.J. Friedman, K.J. Supowit, "Delaunay Graphs are Almost as Good as Complete Graphs", 28th Annual Symposium on Foundations of Computing, 1987, 20–26.

[DFS90] D.P. Dobkin, S.J. Friedman, K.J. Supowit, "Delaunay Graphs are Almost as Good as Complete Graphs", *Discrete and Computational Geometry*, 5, 1990, 399–407.

[Fre87] G.N. Frederickson, "Fast Algorithms for Shortest Paths in Planar Graphs with Applications", *SIAM J. Comput.*, 16, 6 (December 1987), 1004–1022.

[GoB78] B.L. Golden, M. Ball, "Shortest Paths with Euclidean Distances: An Explanatory Model", *Networks*, 8 (1978), 297–314.

[HNR68] P.E. Hart, N.J. Nilsson, B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths", *IEEE Trans. on Systems Science and Cyberbetics*, 4, 2 (July 1968), 100–107.

[KeG92] J.M. Keil, C.A. Gutwin, "Classes of Graphs Which Approximate the Complete Euclidean Graph", *Discrete and Computational Geometry*, 7, 1992, 13–28.

[Lee80] D.T. Lee, "Two–Dimensional Voronoi Diagrams in the $\mathcal{L}_p$- Metric", *JACM*, 27, 4 (October 1980), 604–618.

[LeP84] D.T. Lee, F.P. Preparata, "Computational Geometry–A Survey", *IEEE Trans. on Computers*, 33, 12 (December 1984), 1072–1101.

[LeW80] D.T. Lee, C.K. Wong, "Voronoi Diagrams in $\mathcal{L}_1(\mathcal{L}_\infty)$ Metrics with 2-Dimensional Storage Applications", *SIAM J. Comput.*, 9, 1 (February 1980), 200–211.

[MoG90] J.A.S. Monteiro, M. Gerla, "Topological Reconfiguration of ATM Networks", INFOCOM'90, 207–214.

[PrS88] F.P. Preparata, M.I. Shamos, *Computational Geometry, An Introduction*, Springer Verlag, 1985.

[Sed88] R. Sedgewick, *Algorithms, second edition*, Addison-Wesley, 1988.

[SeV86] R. Sedgewick, J.S. Vitter, "Shortest Path in Euclidean Graphs", *Algorithmica*, 1, 1, 1986, 31–48.

[SLL81] J.M. Smith, D.T. Lee, J.D. Liebman, "An $O(n \log n)$ Heuristic for Steiner Tree Problems on the Euclidean Metric", *Networks*, 11, 1981, 23–29.

[TrM92] C. Trefftz, P.K. McKinley, "Performance Evaluation of Wormhole Routing in Octagonal Mesh Direct Networks", International Conference on Parallel and Distributed Systems, 1992, 25–33.

Chapter 7

General Discussion and Conclusions

In this thesis, we proposed two new MAC–level protocols for dual/folded bus networks and one new routing protocol for point–to–point networks. In doing so, our primary aim was to address the requirements of high–speed networks.

The results are presented in two parts. The first part is dedicated to MAC level protocols which provide controlled access to a shared transmission medium. In this environment, no routing is required and the main issue is to provide fairness in bandwidth allocation while maintaining a high aggregate throughput rate. To this end, we proposed CBRMA++/SR and SP/R MAC level protocols and compared them to DQDB. In chapter 2, we outlined the requirements of high speed networks in regard to MAC–level protocol design and discussed the deficiencies of DQDB which mostly stem from the inconsistency of the distributed reservation queue at a given instant. Based on the simulation results, we claim that CBRMA++/SR and SP/R MAC protocols satisfy the requirements of a high speed networking environment better than DQDB.

The DQDB standard provides two access modes: *Pre–Arbitrated (PA) Access* for isochronous connection-oriented traffic, such as voice and video and *Queue Arbitrated (QA) Access* for non-isochronous traffic such as regular data. We considered only the QA access mode. A further study may involve the introduction of *Pre–Arbitrated access* for real time traffic (such as voice packet) into CBRMA++/SR and SP/R along with a priority mechanism and broadcasting/multicasting capabilities.

In the second part, we considered point–to–point networks. In chapter 5, we discussed the new challenges posed by the Gbps networks in the design of routing protocols, network topology and congestion control mechanisms. Following a survey of the contemporary design proposals, we introduced the *Compass Routing* protocol in chapter 6. Compass Routing is suitable for a number of network topologies and does not require any connectivity information and/or routing tables at the nodes. It can be implemented directly in hardware and provides support for deflection routing.

For wide area networks, we discussed a network topology based on Delaunay Triangulation. In Delaunay graphs the degree of connectivity is not uniform across the nodes. Further investigation of graphs produced by bounded–degree planar subdivision methods may yield useful results in terms of a standard switch design. However, the existence of a bounded degree graph which approximates the Complete Euclidean graph is currently an open problem [KeG92].

Another line of further study may involve the investigation of a particular claim made in [BFM90]:

> ***Given a specific network geography and traffic matrix, the choice of a physical topology does not exercise a very strong influence on the performance that can be subsequently attained through the optimization of the virtual topology.***

This claim is in sharp contrast with our approach which constructs the network topology according to the criterion of nodal proximity. Although the method of optimization by simulated annealing and genetic algorithms is limited in applicability due to its dependence on a s.atic traffic matrix, a comparative study should provide valuable insight regarding the performance of Compass Routing on Delaunay graphs.

Bibliography

[BFM90] J.A. Bannister, L. Fratta, M. Gerla, "Topological Design of the Wavelength Division Optical Network", INFOCOM'90, 1005–1013.

[KeG92] J.M. Keil, C.A. Gutwin, "Classes of Graphs Which Approximate the Complete Euclidean Graph", *Discrete and Computational Geometry*, 7, 1992, 13–28.