Volume 23 Number 3 September 1999 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

Special Issue:
Gigabit Networking
Guest Editors:
Mohsen Guizani, Kenneth Henriksen

Informatica **23** (1999) Number 3, pp. 303–446

The Slovene Society Informatika, Ljubljana, Slovenia

Informatica

An International Journal of Computing and Informatics

Basic info about Informatica and back issues may be FTP'ed from `ftp.arnes.si` in magazines/informatica ID: `anonymous` PASSWORD: *<your mail address>*
FTP archive may be also accessed with WWW (worldwide web) clients with
URL: `http://www2.ijs.si/~mezi/informatica.html`

Subscription Information Informatica (ISSN 0350-5596) is published four times a year in Spring, Summer, Autumn, and Winter (4 issues per year) by the Slovene Society Informatika, Vožarski pot 12, 1000 Ljubljana, Slovenia.
The subscription rate for 1999 (Volume 23) is
– DEM 100 (US$ 70) for institutions,
– DEM 50 (US$ 34) for individuals, and
– DEM 20 (US$ 14) for students
plus the mail charge DEM 10 (US$ 7).
Claims for missing issues will be honored free of charge within six months after the publication date of the issue.

LaTeX Tech. Support: Borut Žnidar, Kranj, Slovenia.
Lectorship: Fergus F. Smith, AMIDAS d.o.o., Cankarjevo nabrežje 11, Ljubljana, Slovenia.
Printed by Biro M, d.o.o., Žibertova 1, 1000 Ljubljana, Slovenia.

Orders for subscription may be placed by telephone or fax using any major credit card. Please call Mr. R. Murn, Jožef Stefan Institute: Tel (+386) 61 1773 900, Fax (+386) 61 219 385, or send checks or VISA card number or use the bank account number 900–27620–5159/4 Nova Ljubljanska Banka d.d. Slovenia (LB 50101-678-51841 for domestic subscribers only).

According to the opinion of the Ministry for Informing (number 23/216-92 of March 27, 1992), the scientific journal Informatica is a product of informative matter (point 13 of the tariff number 3), for which the tax of traffic amounts to 5%.

Informatica is published in cooperation with the following societies (and contact persons):
Robotics Society of Slovenia (Jadran Lenarčič)
Slovene Society for Pattern Recognition (Franjo Pernuš)
Slovenian Artificial Intelligence Society; Cognitive Science Society (Matjaž Gams)
Slovenian Society of Mathematicians, Physicists and Astronomers (Bojan Mohar)
Automatic Control Society of Slovenia (Borut Zupančič)
Slovenian Association of Technical and Natural Sciences (Janez Peklenik)

Informatica is surveyed by: AI and Robotic Abstracts, AI References, ACM Computing Surveys, Applied Science & Techn. Index, COMPENDEX*PLUS, Computer ASAP, Computer Literature Index, Cur. Cont. & Comp. & Math. Sear., Current Mathematical Publications, Engineering Index, INSPEC, Mathematical Reviews, MathSci, Sociological Abstracts, Uncover, Zentralblatt für Mathematik, Linguistics and Language Behaviour Abstracts, Cybernetica Newsletter

The issuing of the Informatica journal is financially supported by the Ministry for Science and Technology, Slovenska 50, 1000 Ljubljana, Slovenia.

Post tax payed at post 1102 Ljubljana. Slovenia taxe Percue.

Introduction: Design Issues of Gigabit Networking

The ongoing revolution in the wide area network infrastructure and services is creating tremendous opportunities for network equipment vendors, operators, and service providers. On the other hand, it is also creating new challenges for them to be tackled. These challenges include a number of issues, such as design infrastructure of gigabit networking, SONET/SDH architectures, protocol design for multimedia, fault-tolerant of backbone networks, routing and congestion protocols, switching issues, QoS of ATM networks, etc. In particular, a multitude of choices, the unpredictability of future technologies and traffic, the need to operate heterogeneous networks, and the speed of change necessitate an understanding of design issues in early stages.

This continuing explosive growth of multimedia traffic coupled with a high degree of uncertainty in future traffic patterns call for new approaches to planning network technologies, topologies, and capacities. This feature issue provides some leads to recent efforts by researchers in this field. There are eight articles for this theme that touches on major issues of concern in this field.

The first article by Marosits et al suggests a novel traffic control framework to support all services traffic in ATM. The control is based on a complete buffer partitioning architecture and on the associated buffer scheduling rule with adaptive weighting functions. After presenting the formulation of the traffic control problem, a comprehensive performance evaluation of the method has been discussed and simulation results and QoS dependence on different traffic services are detailed.

Cseleny and Szabo, in the second article, discuss an efficient resource allocation for Internet and ATM multimedia applications. They go on to present an intelligent allocation approach that utilizes the knowledge of multimedia application and user behavior during connection establishment. They analyze and model their approach using a video-phone service as an example. The performance of the proposed schemes is evaluated and the benefits are highlighted in terms of connection setup time, blocking probability, and signaling intensity.

Third, recognizing that fault-tolerance is of a great importance, especially in high capacity networks, Rayhan et al present a rigorous study of a fault-tolerant ATM switch using logical neighborhood network. The switch is targeted for terabit switching through localized traffic control and management, use of input/output queuing, distributed simple cell routing algorithm, and soft fault tolerance. In addition, they also propose a dynamic routing algorithm that detects blocked and/or faulty links. The switch throughput, cell loss probability, and performance under faults are studied as well.

Issues on the network level are as important as at the switch level. The next three papers deal with issues at the network level. Liotopoulos examines the scalability potential of 3-stage Clos networks for Gig-abit ATM switching. The proposed switching fabric, a nonblocking with capacity scalable to 160 Gbps, consists of a number of modular switching elements, interconnected in a 3-stage Clos network, with a total number of up to 1024 x 1024 input/output ports (OC-3). Results show that nonblocking operation is feasible with small internal buffering that leads to very low cell latencies. Given the appropriate Call Admission Control algorithm and an adequate number of middle-stage switches, the Clos network can also provide nonblocking operation at call setup.

Then, Venkatesan et al investigate the performance analysis of pipelined multistage interconnection networks to combat internal and output blocking. They present the effects of the buffer size on the performance of pipelined multistage interconnection networks based on banyan as well as Balanced Gamma networks. Performance analyses with finite input and finite output buffers are addressed under traffic patterns created using mixes of uniform random traffic as well as bursty traffic. These traffic patterns are closer to ATM traffic than permutation traffic or pure uniform random traffic. They conclude that output queuing is an architecture independent problem.

Still at the network level, Battou and Khan propose an approach to ATM network topology design that is driven by the performance of its routing protocol, PNNI. Towards this end, they define performance indicators based on the time and traffic required for the protocol to first enter and subsequently return to the meta-stable state of global synchrony, in which switch views are in concordance with physical reality. They argue that the benefits of high call admission rate and low setup latency are guaranteed by our indicators. Then, use the PNNI Routing and Simulation Toolkit (PRouST) to conduct simulations of PNNI networks, with the aim of discovering how topological characteristics such as the diameter, representation size, and geodesic structure of a network affect its performance.

Then, Nong et al present a scheduling algorithm for multiple input-queued ATM switches. They assume that the interconnection network of the ATM switch is internally non blocking and each input port try to reduce the head-of-line (HOL) blocking of conventional input queuing switches. Each input is allowed to send only one cell per time slot, and each output port is allowed to accept only one cell per time slot. The cells to be transmitted in a time slot are selected using a fast, fair and efficient scheduling algorithm called iSLIP. The switch throughput, mean cell delay; and cell loss probability are computed from the analytical model. With the switch being under heavy traffic loads, the accuracy of the analytical model is verified using simulation.

Finally, Mickle looks at the possibility of determining an absolute network performance for a collection of networks connected to form an Internet. However, the Internet in this case is captive and created to serve a single purpose, as is the case with numerous control problems such as the beam control of a linear accelerator. A captive Internet has a theoretical capability at the physical level depending on the type of networks that have been interconnected. This capability, in terms of bits per second, i.e., data rates, represents a convex domain of feasibility in which the network can be characterized as a single point at any instant of time. The characterization or modeling of a captive Internet in this fashion also provides a basis for analysis of the well-known Internet under certain assumptions of protocols and priorities.

This is a modest list of contributions to the vast topic of broadband network architectures. It is essential to note that many other important points that fall under this topic could not be accommodated here. We believe that the articles presented here will shed some light that will lead to better research, about the infrastructure of gigabit networks, in the future.

Guest Editor's Bio

Mohsen Guizani is currently the Chair of the Computer Science Department at the University of West Florida, Pensacola, FL. From 1996 to 1999, he was an Associate Professor of Electrical and Computer Engineering at the University of Missouri, Columbia. Prior to joining the University of Missouri, he was a Research Fellow at the University of Colorado, Boulder. From 1989 to 1996, he held academic positions at the Computer Engineering Department at the University of Petroleum and Minerals, Dhahran, Saudi Arabia. He was also a Visiting Professor in the Electrical and Computer Engineering Department at Syracuse University, Syracuse, New York during academic year 1988-1989. His research interests include Computer Networks, Design and Analysis of Computer Systems, Parallel and Distributed Computing, Fault-Tolerant Systems, Modeling and Performance Evaluation of Communication Systems, Wireless Communications and Computing, and Optical Interconnection Networks. He served as a guest editor of the IEEE Communication Magazine, Informatica Journal, International Journal of Computer Systems and Networks, International Journal of Communication Systems, International Journal of Computing Research. Dr. Guizani is the founder and Editor-In-Chief of a new Journal entitled "Wireless Systems and Mobile Computing", by John Wiley. He is the author of two books: "Designing ATM Switching Networks," McGraw-Hill, 1999 and "Optical Networking and Computing for Multimedia Systems," Marcel Dekker, to appear December 1999. He presented Tutorials in conferences all over the world. He has chaired and developed many technical Sessions/Symposia in national and international conferences. He has more than 80 publications in refereed journals and conferences. Dr. Guizani received his B.S. and M.S. degrees in Electrical Engineering; M.S. and Ph.D. degrees in Computer Engineering in 1984, 1986, 1987, and 1990, respectively, from Syracuse University, Syracuse, New York. Dr. Guizani is a senior member of IEEE, member of IEEE Communication Society, IEEE Computer Society, ASEE, ACM, ISCA, OSA, SCS, and Tau Beta Pi.

Mohsen Guizani

Supporting All Service Classes In ATM: A Novel Traffic Control Framework

Tamás Marosits and Sándor Molnár
Department of Telecommunications and Telematics
Technical University of Budapest
Pázmány Péter sétány 1/D, Budapest, Hungary H-1117
{molnar, marosits}@ttt-atm.ttt.bme.hu
AND
Gábor Fodor
Mobile Networks and Systems Research Department
Ericsson Radio Systems
Kistagangen 20-26, Kista, Stockholm, Sweden SE-16480
Gabor.Fodor@era-t.ericsson.se

Keywords: ATM Quality of Service, cell loss, cell delay, cell delay variation, service classes dynamic cell scheduling algorithm

Edited by: Marcin Paprzycki

Received: January 31, 1999 **Revised:** July 31, 1999 **Accepted:** August 31, 1999

This paper presents a general traffic control framework for Asynchronous Transfer Mode (ATM) networks with its performance evaluation. The proposed traffic control scheme can incorporate all the recently considered ATM service classes including Constant Bit Rate (CBR), real time Variable Bit Rate (rtVBR), non-real time Variable Bit Rate (nrVBR), Available Bit Rate (ABR) and Unspecified Bit Rate (UBR) services. The control is based on a complete buffer partitioning architecture and on the associated buffer scheduling rule with adaptive weighting functions. We present the formulation of the traffic control as an optimization problem in a 3-dimensional Quality of Service (QoS) state space. A solution approach based on dynamic programming is also suggested. A comprehensive performance evaluation of the method has been performed based on simulations and results are presented with several examples. The QoS dependence on CBR load, VBR load, VBR burstiness, UBR load are investigated and results are demonstrated with explanations.

1 Introduction

Since ATM networks are to support CBR, rtVBR, nrVBR, ABR and UBR service classes, the simplest 2-level priority based control policies become inadequate [11, 19]. Furthermore, within these service classes different VCs may require different cell loss, cell delay and cell delay variation parameters. It is therefore essential that the traffic control strategies be capable for the provision of the negotiated QoS parameters and for high network utilization by statistical multiplexing traffic classes with strict (CBR and VBR) or limited (ABR) QoS guarantees with a pure best effort type service class (UBR). Indeed, one of the key issues in the success of ATM is the traffic integration, and specifically, the design and analysis of control strategies which make the integration possible [9, 15, 18, 20, 21] . The need for a fine granularity of traffic control in ATM has been recognized e.g. in [19] and [11] where a 4-level priority based control mechanism is proposed. This model classifies traffic classes as "sensitive" or "less sensitive" to loss and delay. Gelenbe et al. concentrates on minimizing the impact of cell loss where cells belonging to different classes are assigned a cost function representing the importance of cells belonging to different "sessions" i.e. VCC's [12]. However, the priority based control algorithms are simple and easy to evaluate, their behaviour are static, they cannot be adapted to variable traffic. In addition, there are too many loser services because of static rules. The mixed approaches, such as Partial Buffer Sharing [14] are better, but they are not suitable for both loss and delay sensitive traffic, for example rtVBR. In a static priority system the cells with higher priority level can completely push out the lower priority traffic. This is an important problem in the case the network can provide ABR traffic. The ABR cannot have the lowest priority because it also has QoS guarantees. However, if it has a medium priority level, it can push out all the lower priority cell streams that are neglected in the control. ABR increases its rate to utilize the full bandwidth left by higher priority classes.

It is envisaged that second generation ATM switches will employ the Generalized Processor Sharing (GPS) and its packetized version PGPS, as the basic principle for buffer management [16]. However GPS is a static rule which means that it is reconfigured only when a new connection is established. There is no adaption of the fluctuation of traffic between two reconfigurations and the instantaneous QoS parameters have not been taken into account. The ABR traffic has some problems also with this scheduling discipline, because it can change its rate during the connection setup. In the more theoretical vein, recently there has been a growing interest in devising stochastic control methods, which can serve as a theoretical basis in the engineering of control enabling traffic integration in ATM, see e.g. [13, 17].

The purpose of this paper is to present a traffic control framework which allows for arbitrary degree of granularity in terms of guaranteed QoS parameters in an ATM network where service classes with and without QoS guarantee, with and without congestion control, and with and without real time guarantees are present. This means that the control cannot be based on a single bit, like the Cell Loss Priority (CLP) bit found in the ATM header. This is not only because a single bit cannot contain enough information on a complex QoS measure, but also because the continuous ("real time") QoS monitoring of VCC's is both required and feasible by current and next generation ATM switches. We propose a scheme where the control is based on the (1) negotiated QoS parameters, (2) instantaneous (current) QoS of the VCC under control, and (3) network resources allocated for the VCC under control. This is achieved by (1) defining a 3-dimensional QoS state space where the QoS parameters of each VCC can exactly (or with arbitrary precision) be described and (2) by a complete buffer partitioning with complete link capacity sharing [14] architecture of ATM multiplexers, which allows for "an individual handling" of VCC's requesting sharply different QoS measures from the network. It follows that a buffer partition arbitration algorithm is needed, which decides (possibly at each time slot) which partition's cell(s) gets served next. The basic requirements to this algorithm are that (1) it should guarantee the negotiated QoS parameters to each VCC and (2) it should optimize the "overall" network performance in the sense that provided that (1) is kept, each VCC gets the highest possible quality of service while network utilization is also kept high. Since the algorithm is to be executed real time, it should be simple and feasible by current technologies. The authors published the basic idea of this traffic control algorithm in [10] with. This paper presents the performance evaluation of this scheme.

The paper is organized as follows. Section 2 presents the reference architecture of the ATM buffers and the reference model of a VC connection, which serve as the basic model for the traffic control scheme under study. The QoS state space, is described in Section 3. In this space the contracted traffic parameters define an acceptance region, within which the points representing the QoS of the current VCC's must fall. The buffer partition arbitration algorithm is formulated as a dynamic programming optimization problem. Next, in Section 4, simulation results are presented, where the impact of the so called weighting function settings on network behavior is studied. The role of the weighting functions assigned to the partitions is to define the partition to be scheduled next. The section discusses numerical results and relates some results to related work. Section 5 draws conclusions and outlines future extensions of the model.

2 The Scheduling Technique

2.1 Buffer architecture

The traffic control method based on the complete buffer partitioning architecture (see Figure 2.1) where the total switch memory is divided to five FIFO buffers according to the presently considered service classes: CBR, rtVBR, nrVBR, UBR and ABR [4]. For the ABR class the end-to-end rate based EFCI mechanism [8] is used to reduce the cell loss. The motivation behind choosing the complete

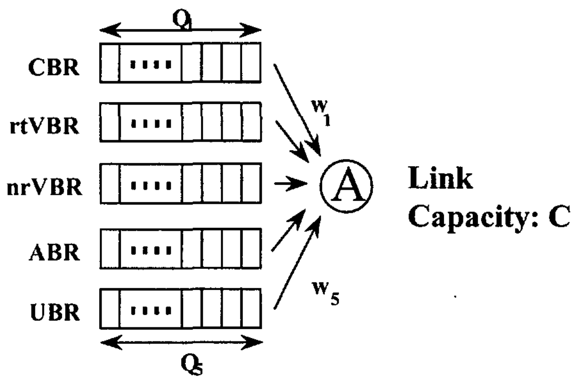


Figure 2.1 The buffer architecture

buffer partitioning is that previous works clearly demonstrated that the presence of all these inherently different services having also rather diverse QoS requirements the complete buffer partitioning based schemes are superior over shared buffering techniques [11, 14, 19]. The Connection Traffic Descriptors should be selected in corporation with the proper dimensioning of Usage Parameter Control (UPC) [1, 4, 7], see below. An efficient Call Admission Control (CAC) can be performed with buffer partitioning and bandwidth allocation based on the listed parameters. In our model a weighting function (W_i) is dedicated to each buffer. The cell scheduling rule is the following: all weighting functions are evaluated at each time slot and the Head of Line (HOL) cell of the buffer with the greatest weighting function value is forwarded to the output link. The appropriate choice of the weighting function is a crucial point of the control. The weighting function uses the Connection Traffic Descriptors, QoS Requirements, Network Resource settings and also the current (instantaneous) QoS information of the VCC under control. This idea of weighting functions allows us to set flexible and adaptive control method. An application example of setting the weighting functions can be found below. We use the results of the CAC as starting point so we get the buffer sizes (Q_i) as input parameters to our traffic control method.

2.2 Traffic Control Parameters

The main goal of traffic control is to protect the network and the user in order to achieve network performance objectives with optimum allocation of network resources [2]. To fulfill these objectives QoS requirements, traffic descriptors and network information needed for the generic traffic control functions. We have chosen the following parameters for our traffic control framework which is in agreement with the standardization work of ATM Forum [3, 4] and ITU-T [1, 2]:

- **Connection Traffic Descriptors**: Peak Cell Rate (PCR), Cell Delay Variation Tolerance (CDVT), Sustainable Cell Rate (SCR), Maximum Burst Size (MBS), Minimum Cell Rate (MCR) and the conformance definition: the Generic Cell Rate Algorithm (GCRA) [2]
- **Quality of Service Parameters**: Cell Loss Ratio (CLR), average Cell Transfer Delay (CTD), peak-to-peak Cell Delay Variation (CDV)
- **Network Resources**: link capacities (C), memory size for buffering (Q)

3 The QoS Control

3.1 QoS specification

Recently there are five service classes with different traffic descriptors and QoS requirements defined in ATM. Correlation can be discovered between descriptor parameters and QoS requirements, which are specified in Table 3.1. Our assignment is mainly based on the ATM Forum specification [4].

The weighting function related to a service class should reflect the parameters specified in the appropriate column of Table 3.1. The end-to-end performance objectives of traffic contract should be allocated among the connection portions. We use the allocation principles specified in the standardization works, i.e. the CLR and CTD objectives are allocated by additive rules and CDV objectives are determined by the square root rule [1, 5]. In this way, our control method can be performed locally in the switches, because each switch has the performance objectives after the above decomposition for itself. Note that using the local performance objectives, local resource settings and the instantaneous local QoS information with the given Connection Traffic Descriptors the general end-to-end traffic control problem can be handled as a local traffic control problem in each switch. We avoid the overload caused by the transmission of lot of information necessary for a global control too. Also note, that this control can coexist with the end-to-end ABR control mechanism.

Attribute	CBR	rtVBR	nrVBR	UBR	ABR
Traffic Parameters					
PCR, CDVT	X	X	X	X	X
SCR, MBS, CDVT		X	X		
MCR					X
QoS Parameters					
CDV	X	X			
CTD	X	X			
CLR	X	X	X		X

Table 1: Table 3.1 Parameters of traffic contract and QoS requirements

3.2 The 3-dimensional abstract QoS space

To connect the quality of service requirements negotiated by the traffic contract to weighting function parameters we define a 3-dimensional state space with co-ordinates of measures of cell loss, delay and delay variation characteristics [8]. We choose for these measures the instantaneous CLR, CTD, and CDV parameters of a connection. That means each connection represented as a point of this space in each time slot (see Figure 3.1).

In this state space the QoS evolution of VCs can be observed where acceptance region can also be identified based on the negotiated QoS requirements. We define a cost function as an abstract distance of the actual QoS from the origin in the state space. The task of the traffic control method is thereafter formulated as to find the appropriate weighting functions such that:

- the actual QoS values for each VC should be within the negotiated region
- the total cost of all VC connections should be minimal

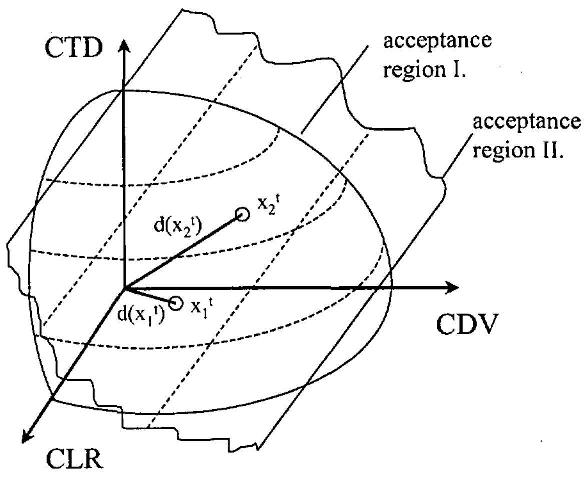


Figure 3.1 The abstract QoS state space

To fulfill these objectives we face with an optimization problem. One of the possible solutions is to define weighting functions, which parameters are evaluated with a dynamic programming algorithm [10].

3.2.1 System equation

Consider the buffer partition in Figure 3.2. The lower index i refers to the service class ($i = 1..5$, i=1: CBR, i=2: rtVBR, i=3: nrVBR, i=4: ABR, i=5: UBR), the upper index (k) refers to the kth time slot. The state of the system at each time slot is specified by the following variables assigned to the ith queue at time (k): $n_i^{(k)}$: the number of processed cells; $l_i^{(k)}$: the number of discarded cells; $q_i^{(k)}$: the length of the queue (i.e. the number of cells in the queue); $\tau_i^{(k)}$: time stamp of the current HOL cell. This stamp is assigned to this HOL cell when entering the buffer. The aggregation of these quantities gives the system state column vector $(X_i^{(k)})^T = (n_i^{(k)}, l_i^{(k)}, q_i^{(k)}, \tau_i^{(k)})$. Q_i is the buffersize of the ith partition, $a_i^{(k)}$ is the number of arriving cells and $u_i^{(k)}$ is the number of served cells during the kth time slot. In this paper, for simplicity, we do not allow batch arrivals or batch departures, i.e. the latter two quantities are either 0 or 1. Note the a will correspond to the random disturbance while u to the control in the DP algorithm. Also note that all these variables are quantities which can be stored locally at the switch, which ease the formulation of a local control law.

lower index i	index of the service class
upper index (k)	number of the current time slot
$n_i^{(k)}$	the number of processed cells
$l_i^{(k)}$	the number of discarded cells
$q_i^{(k)}$	the number of cells in the queue
$\tau_i^{(k)}$	the time stamp of the current HOL cell
Q_i	the buffersize of the ith partition
$a_i^{(k)}$	the number of arriving cells
$u_i^{(k)}$	number of served cells

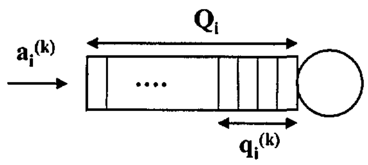


Figure 3.2 Notations of cell buffering

With the above notation and assuming $l_i^{(0)} = 0$ the system dynamics is described by the following discrete time system equations:

$$\begin{aligned} q_i^{(k+1)} &= min(q_i^{(k)} - u_i^{(k)} + a_i^{(k)}, Q_i) \\ l_i^{(k+1)} &= max(q_i^{(k)} - u_i^{(k)} + a_i^{(k)} - Q_i, 0) + l_i^{(k)} \\ n_i^{(k+1)} &= n_i^{(k)} + a_i^{(k)} \end{aligned}$$

Note that an alternative to (1) could be:

$$n_i^{(k+1)} = n_i^{(k)} + u_i^{(k)},$$

but we will assume that $n_i^{(k+1)}$ is large enough to make this difference irrelevant. Obviously, we have the following control constraint: $\sum_i u_i^{(k)} \leq 1$ for all k, indicating that at most one queue can get served at any one time.

3.2.2 Instantaneous QoS Characteristics

We proceed with relating cell loss probability $c_{i1}^{(k)}$, average (instantaneous) cell delay $c_{i2}^{(k)}$ and average (instantaneous) cell delay variation $c_{i3}^{(k)}$ to the system variables. These definitions are important, because they map the system description to the abstract model of the QoS state space ($i = 1..5$).

$$\begin{aligned} c_{i1}^{(k+1)} &= \frac{l_i^{(k)}}{n_i^{(k)}} \\ c_{i2}^{(k+1)} &= \begin{cases} \frac{n_i^{(k)} * c_{i2}^{(k)} + (k - \tau_i^{(k)})}{n_i^{(k)}} & \text{if } u_i^{(k)} = 0 \\ c_{i2}^{(k)} & \text{if } u_i^{(k)} = 1 \\ & \text{OR } q_i^{(k)} = 0 \end{cases} \\ c_{i3}^{(k+1)} &= \begin{cases} \frac{n_i^{(k)} * c_{i3}^{(k)} + (k - \tau_i^{(k)} - c_{i2}^{(k)})}{n_i^{(k)}} & \text{if } u_i^{(k)} = 1 \\ c_{i3}^{(k)} & \text{if } u_i^{(k)} = 0 \end{cases} \end{aligned}$$

3.2.3 Cost Functional

In this subsection we define the cost functional applicable for optimization with dynamic programming, since it facilitates a straightforward formulation of the Bellman equation [6]. Throughout we restrict our attention to the case when a single traffic source generates traffic to each queue, i.e. each traffic class is represented by a single source (one VCC per queue). This assumption is not especially restricting, since with proper aggregated traffic models the multiple source per service class case can readily be represented. To obtain an overall scaler valued cost functional suitable for optimization, we first need the cost function of a single session, i.e. the cost assigned to a VCC:

$$\begin{aligned} J_i^{(k)}(c_i^{(k)}) &= J_i^{(k)}(c_{i1}^{(k)}, c_{i2}^{(k)}, c_{i3}^{(k)}) = ||J_i^{(k)}|| = \\ &= \sqrt{f_{i1}^{(k)2}(c_{i1}^{(k)}) + f_{i2}^{(k)2}(c_{i2}^{(k)}) + f_{i3}^{(k)2}(c_{i3}^{(k)})} \end{aligned}$$

where the $f_{i1}^{(k)}(.)$, $f_{i2}^{(k)}(.)$, $f_{i3}^{(k)}(.)$ functions are the loss, delay and delay-variation weighting functions, respectively, of VCC_i, and $c_i^{(k)}$ is a column vector, corresponding to the QoS of VCC_i at time k, $(c_i^{(k)}) = (c_{i1}^{(k)}, c_{i2}^{(k)}, c_{i3}^{(k)})$. Assuming stationarity and equal VCC (session) weighting functions for all i, we can neglect the dependency of the weight functions from the i_1, i_2 and i_3 parameters, as well as from the time index (k). In other words, we simply have:

$$J_i(c_i^{(k)}) = \sqrt{f_1{}^2(c_{i1}^{(k)}) + f_2{}^2(c_{i2}^{(k)}) + f_{i3}{}^2(c_{i3}^{(k)})}$$

Let Γ denote the overall cost functional vector, and C denote the cost matrix:

$$\Gamma(C^{(k)}) = (J_i(c_i^{(k)})); C^{(k)} = (c_i^{(k)}); c_i^{(k)} = (c_1^{(k)}, c_2^{(k)}, c_3^{(k)})$$

With this notation the overall scaler cost functional to be minimized takes the form:

$$\gamma(C^{(k)}) = \sum_{i=1}^{5} J_i(c_i^{(k)}) \text{ subject to } C^{(k)} \leq (QoS)_{i,j}$$

where $i = 1..5, j = 1..3$, and the QoS matrix contains the negotiated QoS parameters (CLR, CTD, CDV), see Subsection 2.2. With the above formulation of the optimization problem we are facing the challenge of a DP task, where we wish to find the optimal (and feasible) control law μ which only depends on the state of the physically observable state vector X, such that the cost functional is optimized over the 3 dimensional QoS state space represented by the C matrix. The other way is to solve the optimization problem is using direct cost functions in the algorithm. This means that instead of weighting functions cost functions are evaluated before departure and the HOL cell of the most "expensive" queue must be sent. The cost function should be discounted in order to slowly forget the past. We are recently working on this topic.

3.3 An example for the set of weighting functions

These objectives are mathematically formulated as follow [10]:

$$\begin{aligned}
W_1 &= a_1 * \frac{LC_1}{SUM_1 * CLR_1} + b_1 * \frac{T_1}{CTD_1} + \\
&\quad + c_1 * max(T_1 - CTD_1 - \frac{2}{3} * CDV_1, 0) \\
W_2 &= a_2 * \frac{LC_2}{SUM_2 * CLR_2} + b_2 * \frac{T_2}{CTD_2} + \\
&\quad + c_2 * max(T_2 - CTD_2 - \frac{2}{3} * CDV_2, 0) \\
W_3 &= a_3 * \frac{LC_3}{SUM_3 * CLR_3} \\
W_4 &= a_4 * \frac{LC_4}{SUM_4 * CLR_4} \\
W_5 &= \begin{cases} w_5 & \text{if } K_1, K_2, K_3, K_4 \text{ are all} > 1 \\ 0 & \text{otherwise} \end{cases}
\end{aligned}$$

where:

$$K_1 = \frac{d_1 * (a_1 + b_1 + c_1)}{W_{CBR}}; K_2 = \frac{d_2 * (a_2 + b_2 + c_2)}{W_{rtVBR}};$$

and

$$K_3 = \frac{d_3 * a_3}{W_{nrVBR}}; K_4 = \frac{d_4 * a_4}{W_{ABR}}$$

The value of a weighting function is equal to minus infinity if the queue is empty. Let be LC_i is the number of lost cells of class i, SUM_i is the total number of cells of class i, and T_i is the waiting time of HOL cell in the queue of class i. Note that $[x]^+$ is equal to x if $x > 0$ else 0. These weighting functions obtained by heuristics based on Table 3.1, that means they reflect the different service classes sensitivity to cell loss, delay and delay variations and also take into account the required QoS parameters. Specifically, the weighting parameters a_i, b_i, c_i and d_i are to determine the relative "importance" of a given QoS parameter in the weight of a given service class, while the constants CLR_i, CTD_i, and CDV_i are the negotiated (contracted) cell loss ratio, cell transfer delay and cell delay variation of the respective VCC's. These latter three parameters are referred to as QoS in this paper.

4 Performance Evaluation

4.1 The input traffic

In the next following simulation scenarios we consider a link of capacity 45 Mbps, and a multiplexer with 5 input ports corresponding to the 5 service classes. The basic state of the traffic sources is the following: The CBR source is of 1.5 Mbps representing DS-1 circuit emulation. The rtVBR, nrVBR and UBR sources are all bursty and modeled as Interrupted Bernoulli Processes (IBPs) and are characterized by their peak and sustainable cell rates. The ABR source is assumed to be of rate based and is also modeled by an IBP. It is characterized by its peak and minimum cell rate (see Table 4.1). We have given the burstiness parameters of all services measured by the squared coefficient of variation of the interarrival time (i.e. the c^2 parameter).

	PCR	SCR	MCR	c^2
CBR	1.5	-	-	0
rtVBR	15.0	3.0	-	9.44
nrVBR	22.5	1.0	-	20.75
UBR	45.0	5.0	-	26.06
ABR	22.5	-	4.5	-

Table 4.1 Basic input traffic characteristics
(the rates are given in Mbps)

Note that with the above link capacity a time slot in our discrete time model correspond to 9.422 μs, which will be used as the time unit in the CTD and CDV values below. Tables 4.2-4.4 display the QoS requirements of different services, the buffer sizes available for different service classes and an appropriate parameter set for weighting functions, respectively. These weighting function parameters obtained by heuristics. Note that no delay or delay variation parameters are negotiated for the nrVBR or the ABR service classes and no QoS requirements are given for the UBR service.

	CBR	rtVBR	nrVBR	UBR	ABR
CLR_i	10^{-5}	10^{-6}	10^{-7}	-	10^{-7}
CTD_i	3.0	5.0	-	-	-
CDV_i	1.0	2.0	-	-	-

Table 4.2 The QoS requirements
(CTD and CDV requirement are given in time unit)

Service class	CBR	rtVBR	nrVBR	UBR	ABR
Buffersize	5	8	12	250	80

Table 4.3 Buffer sizes in cells

	a_i	b_i	c_i	d_i
CBR	0.1	0.6	0.9	0.5
rtVBR	0.2	0.3	0.4	0.5
nrVBR	0.6	-	-	0.7
UBR	-	-	-	-
ABR	0.4	-	-	0.6

and $w_5 = 6.0$

Table 4.4 The parameter set of weighting functions

4.2 The QoS dependence on CBR load

Figures 4.1-4.3 display simulation result on CLR, CTD and CDV respectively, when we increase the CBR load from 1.5 Mbps up to 7.5 Mbps and the other sources are in basic state. In this example we consider a single multiplexer with the weighting function parameter set described above. Due to the lower utilization of the connections (between 0.73 and 0.87) there is a considerable decrease in the QoS parameters of the traffic classes, which have a strict traffic contract with the network. We can see that all the negotiated QoS parameters met their requirements. The CLP and CDV of the CBR service class is slightly increasing according to the increasing load, but this increase effects the increase of the value of the weighting function of CBR class, i.e. the CBR service class gets more bandwidth and the QoS parameters finally rest within the negotiated region.

UBR service class has no any QoS requirements, so the load change causes changes only in the QoS parameters of this service, as it can be seen in the Figures 4.1-4.3.

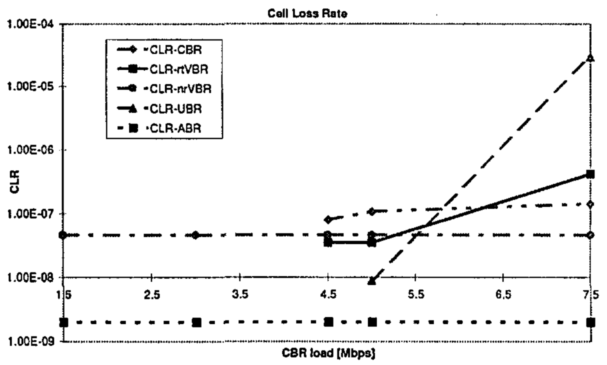


Figure 4.1 Cell Loss Ratio vs. CBR load

In the following scenario, we increase the sustainable cell rate of UBR traffic to 12 Mbps end we set the parameter d_3 to 0.9. The remaining three sources are in basic state and the other parameters are the same as in the previous scenario.

Figures 4.4-4.6 display the QoS parameters of a highly utilized link. The utilization goes from 88% up to 95%. The CLP parameters are similar to the previous case. The

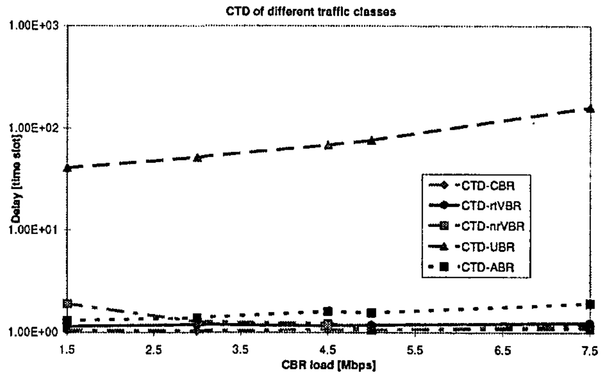


Figure 4.2 Cell Transfer Delay vs. CBR load

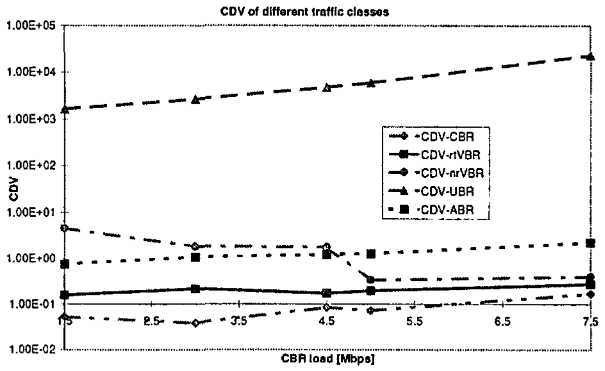


Figure 4.3 Cell Delay Variation vs. CBR load

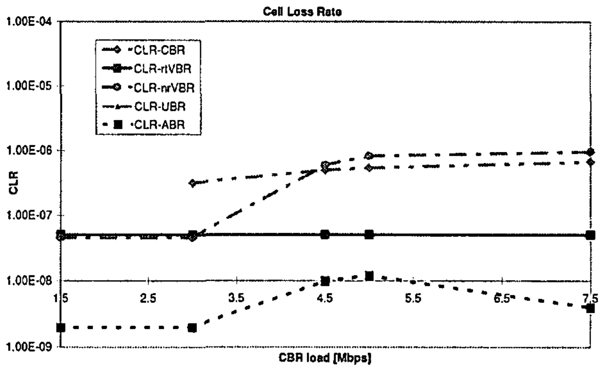


Figure 4.4 Cell Loss Rate vs. CBR load under heavy UBR traffic

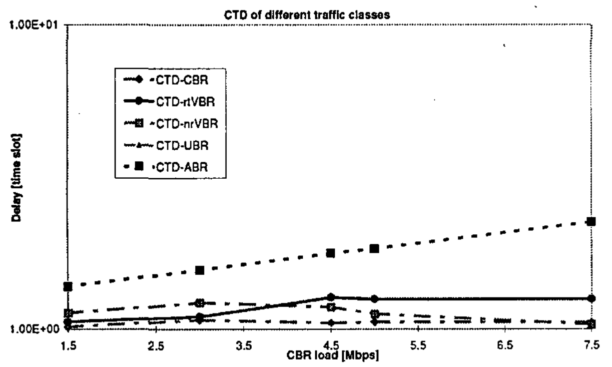


Figure 4.5 Cell Transfer Delay vs. CBR load under heavy UBR traffic

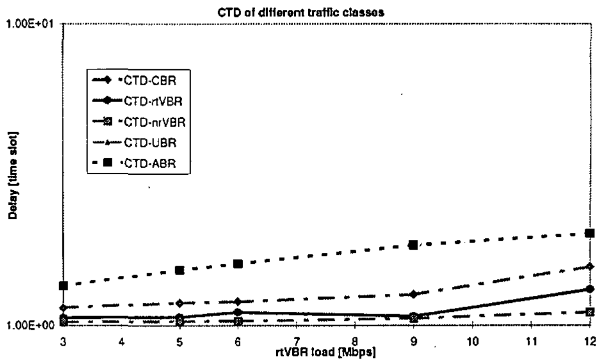


Figure 4.8 Cell Transfer Delay vs. rtVBR load under heavy UBR traffic

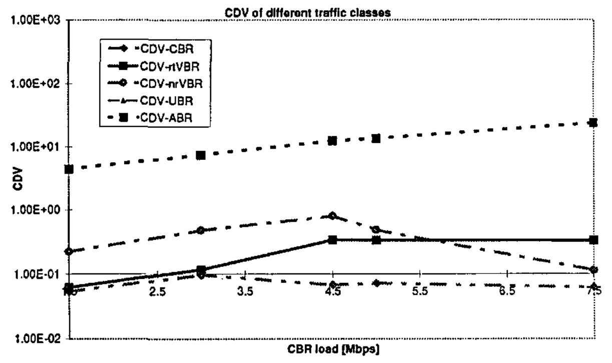


Figure 4.6 Cell Delay Variation vs. CBR load under heavy UBR traffic

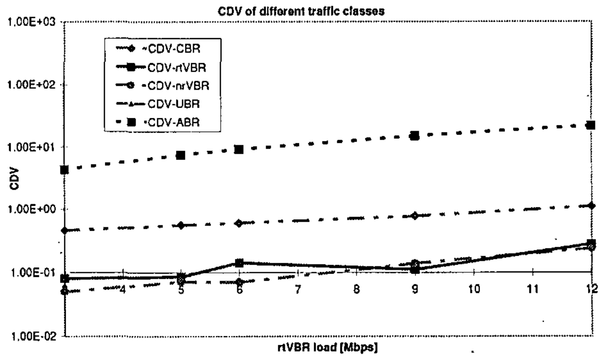


Figure 4.9 Cell Delay Variation vs. rtVBR load under heavy UBR traffic

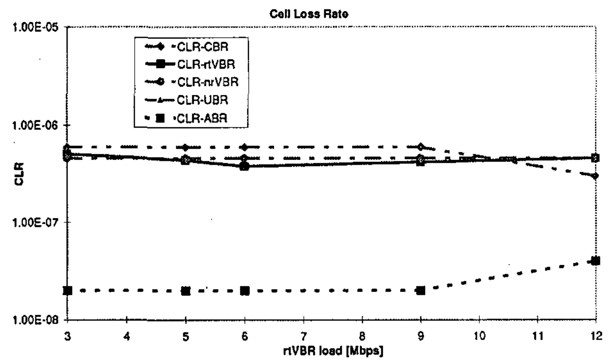


Figure 4.7 Cell Loss Rate vs. rtVBR load under heavy UBR traffic

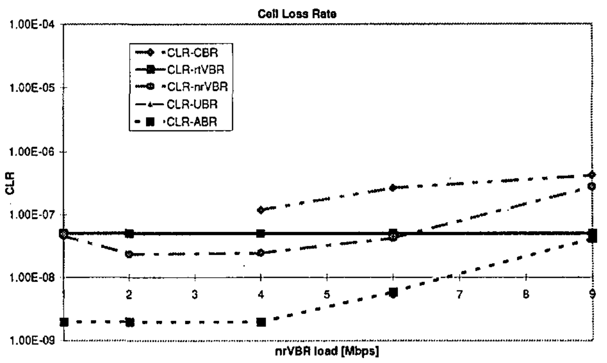


Figure 4.10 Cell Loss Rate vs. nrVBR load under heavy UBR traffic

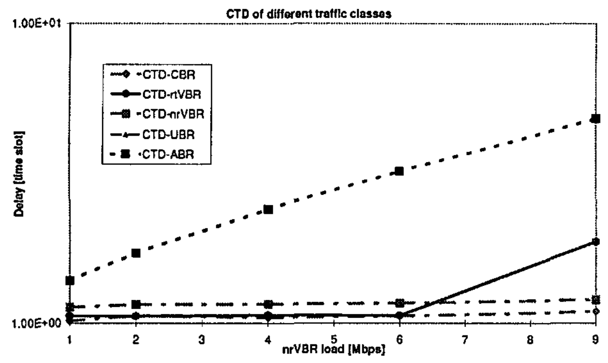


Figure 4.11 Cell Transfer Delay vs. nrVBR load under heavy UBR traffic

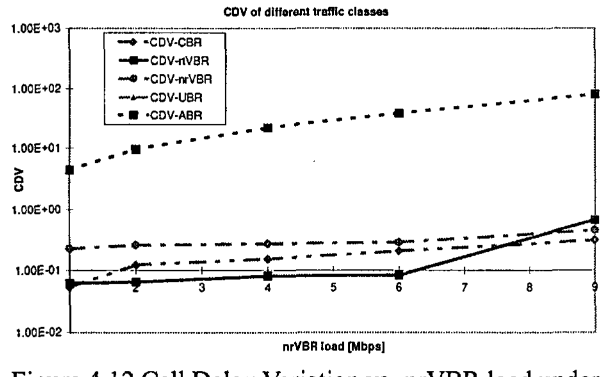


Figure 4.12 Cell Delay Variation vs. nrVBR load under heavy UBR traffic

guaranteed services have constant cell loss except CBR, which has an increase by a decade. This resulted in the slow decreasing of the CDV parameter. The CDV of the other regarded class (rtVBR) is normal. The nrVBR traffic class has no CDV assurance; the non-monotony of the curve comes from the abrupt step of its CLP at the same point.

Observe that the load increase affects the CLR of UBR only, as desired, since all other classes have strictly prescribed CLR values. The same behaviour can be observed for the CTD and CDV parameters of CBR and rtVBR classes. The ABR class is congestion controlled and sensitive to CLR only so its CTD and CDV behaviour is determined by the other classes.

4.3 The QoS dependence on VBR load

In the next following simulation studies we examine the dependence of QoS parameters on the increasing load of VBR traffic. In Figures 4.7-4.9 the load of rtVBR goes from 3 Mbps up to 12 Mbps. The sustainable cell rate of UBR source is set to 15 Mbps and the d_3 is set to 0.9; the other sources and parameters are in basic state.

The utilization is about 0.97 in the Figures 4.7-4.9. In this cases the CLP requirements of nrVBR and ABR classes are increased to 10^{-6} and 10^{-7}, respectively. Because rtVBR is a bursty traffic, there are more significant changes in the QoS parameters of the guaranteed classes. The CDV of CBR class gets in the near of QoS requirement (1.0). This, in consideration of the increasing average delay of CBR, effects the decreasing of CLP at the last measuring point. The other curves meet their QoS requirements. In Figures 4.10-4.12 the load of nrVBR goes from 1 Mbps up to 9 Mbps. The sustainable cell rate of UBR source is set to 18 Mbps and the d_3 is set to 0.7; the other sources and parameters are in basic state.

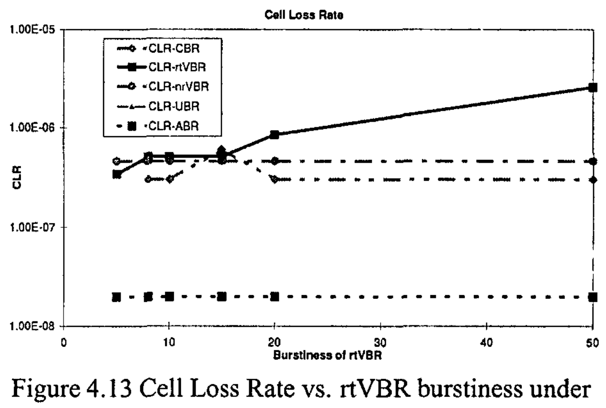


Figure 4.13 Cell Loss Rate vs. rtVBR burstiness under heavy UBR traffic

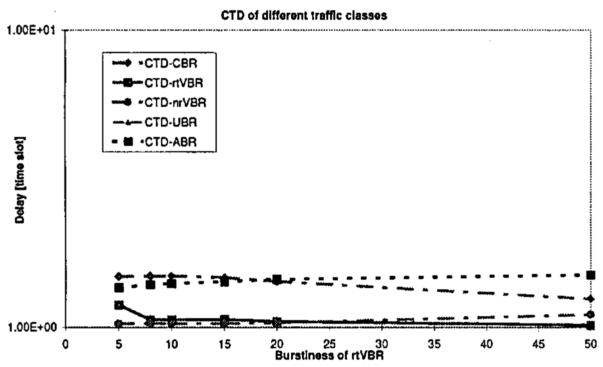


Figure 4.14 Cell Transfer Delay vs. rtVBR burstiness under heavy UBR traffic

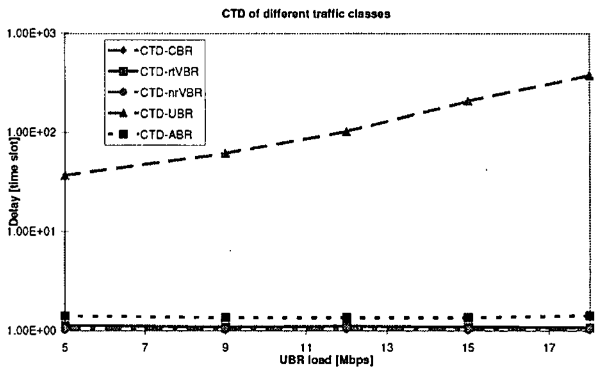


Figure 4.17 Cell Transfer Delay vs. UBR load

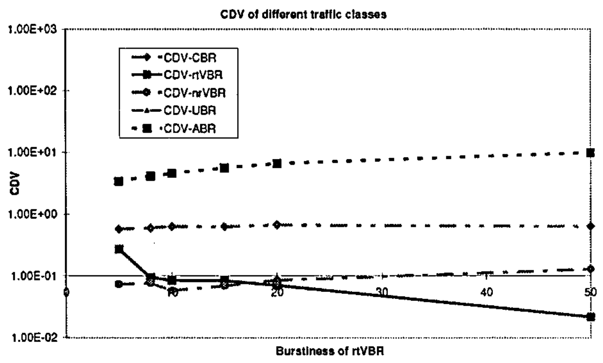


Figure 4.15 Cell Delay Variation vs. rtVBR burstiness under heavy UBR traffic

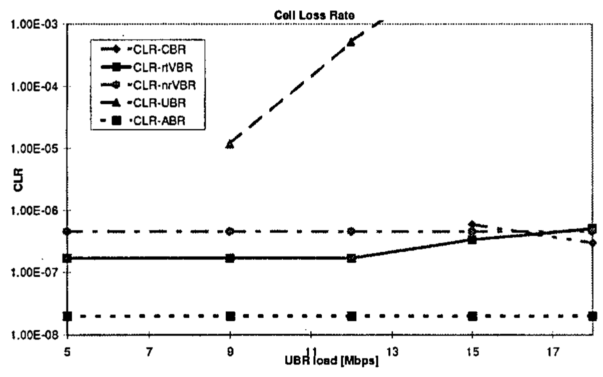


Figure 4.16 Cell Loss Rate vs. UBR load

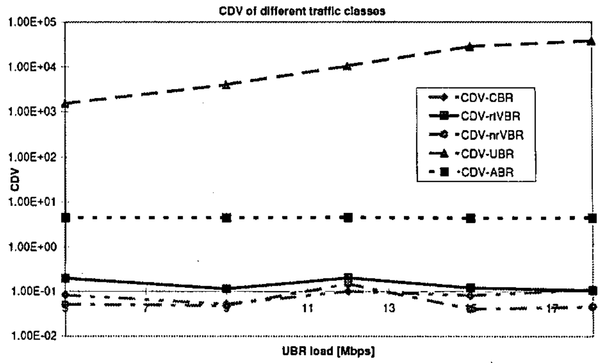


Figure 4.18 Cell Delay Variation vs. UBR load

4.4 The QoS dependence on the burstiness on VBR

In the next following simulation studies we examine the dependence of QoS parameters on the increasing burstiness of VBR traffics. In Figures 4.13-4.15 the burstiness of rtVBR (measured by the squared coefficient of variation of the rtVBR interarrival time) goes from 5 up to 50. The sustainable cell rate of UBR source is set to 18 Mbps and the load of rtVBR source is 3 Mbps; the other sources and parameters are in basic state.

In the Figures 4.13-4.15 can be seen excellently, that the weighting functions handle the different services independent from each other. Real-time VBR traffic with increasing burstiness is arriving to the short buffer described in Table 4.3. The CLP of the rtVBR has linear increase with the burstiness. This causes a decreasing in the CTD and CDV of the rtVBR, but for other classes it seems to be neutral.

4.5 The QoS dependence on UBR load

At the last we show how is the dependence of the QoS parameters of service classes on the increasing load of UBR traffic. In Figures 4.16-4.18 the load of UBR goes from 5 Mbps up to 18 Mbps. Note that the burstiness of UBR traffic is constantly 26.06 in all cases. The other sources and parameters are in basic state.

The increase of the UBR load does not have any impacts on the QoS parameters of the other classes. It can be seen in Figure 4.17. The average cell transfer delay of UBR traffic significantly increases, while other classes have the same CTD. Note that in our model the UBR service is not totally transparent for the other services. However, there are cells of other classes in the buffer, it maybe delivered an UBR cell, because of the adaptability of our model. We give a chance to the UBR if all other classes meet their QoS with a given reserve. Although, the UBR has poor prestige in the network, if the other services needs the bandwidth.

4.6 Comparison to static scheduling schemes

We made simulations to compare the performance of our algorithm with static scheduling rules - FCFS and Round Robin. Our results show that to achieve the same CLP the static traffic control schemes need 15-20% more buffer space for the guaranteed services, moreover, the utilization of the network decreases. With FCFS the delay requirements can not be fulfilled. In the Round Robin-case the delay is limited by the number of served queues. We have an ongoing research on the detailed comparison of traffic control methods.

5 Conclusions

We have considered the issue of optimal cell scheduling in an integrated services ATM network and proposed a general traffic control framework which is based on a complete buffer partitioning architecture and on an adaptive weighting function based buffering schedule. The method can incorporate all the presently considered service classes with their diverse QoS requirements and it is capable of providing an optimal scheduling considering also the temporary traffic load at the switches with a simple information processing which requires only summation and multiplication.

We suggest a dynamic programming solution for the optimization problem. Moreover, in the paper a performance evaluation study of the control framework is demonstrated with several examples investigating the QoS dependence on CBR load, VBR load, VBR burstiness and UBR load. >From the results we can conclude that the QoS characteristics of each service are within the negotiated QoS region and that the remaining resources are efficiently used by best effort type service classes. We can see that the utilization is achieved by keeping the actual QoS characteristics close to the negotiated parameters rather than overfulfilling them. The examples also show the advantage of statistical multiplexing of the five different service classes sharing only 45 Mbps capacity instead of 106.5 Mbps, which would be the case of peak rate allocation.

We can conclude that the proposed traffic control scheme is capable to keep traffic contracts for the classes with strict QoS requirements by an optimal resource sharing and also distributes resources to the best effort type service classes as such resources become available. In our future research we concentrate to find optimal weighting functions for different scenarios, and apply and analyze the method for different traffic environments with using real traffic sources.

References

[1] ITU-T Recommendations I.356 *B-ISDN ATM Layer Cell Transfer Performance*, 1996.

[2] ITU-T Recommendations I.371 *Traffic Control and Congestion Control in B-ISDN*, 1995.

[3] ATM Forum, *ATM User Network Interface Specification Version 3.1*, September 1994.

[4] ATM Forum, *Traffic Management Specification Version 4.0*, April 1996.

[5] ATM Forum, *B-ISDN Inter Carrier Interface (B-ICI) Specification Version 1.0*, August 1993.

[6] Dimitri P. Bertsekas, "Dynamic Programming, Deterministic and Stochastic Models", *Prentice-Hall Inc., Englewood Cliffs, N.J. 07632*, 1987.

[7] S. Blaabjerg, S. Molnár, "Methods for UPC Dimensioning of a CDV Perturbated Cell Stream", RACE BRAVE Workshop, Milano, Italy, June 14-15, 1995.

[8] F. Bonomi, K. W. Fendick, "The Rate-Based Flow Control Framework for the Available Bit Rate ATM Service", *IEEE Network*, p 25-39, March/April 1995.

[9] I. Cidon, L. Georgiadis, R. Guérin, A. Khamisy, "Optimal Buffer Sharing", *IEEE J-SAC*, vol. 13, No. 7, September 1995.

[10] G. Fodor, T. Marosits, S. Molnár, "A General Traffic Control Framework in ATM Networks", in *Proceedings of the IEEE ICCS/ISPACS '96*, p 160-164, Singapore, November 25-29, 1996.

[11] P. A. Ganos, M. N. Koukias, G. K. Kokkinakis, S. A. Kotsopulos, "A Novel Dynamic Priority Scheduling Method for Multiple Classes of ATM Traffic in an ATM Statistical Multiplexer", in *Performance Modelling and Evaluation of ATM Networks*, Vol. 1, Ed. Demetres Kouvatsos, IFIP, Chapman and Hall, 1995.

[12] Erol Gelenbe, Vilay Srinivasan and Sridhar Seshadri, "Single Node and End-to-End Buffer Control in Real Time", in *Performance Modelling and Evaluation of ATM Networks*, vol. 1, Ed. Demetres Kouvatsos, IFIP, Chapman and Hall, 1995.

[13] Harold J. Kusnher, "Analysis of Controlled Multiplexing Systems via Numerical Stochastic Control Methods", *IEEE J-SAC*, vol. 13, No 7, September 1995.

[14] A. Y.-M. Lin, J. A. Sylvester, "Priority Queueing Strategies and Buffer Allocation Protocols for Traffic Control at an ATM Integrated Broadband Switching System", *IEEE J-SAC*, vol. 9, No. 9, p 1524-1536, December 1991.

[15] J. W. Mark, Jing-Fei Ren, "A Traffic Management Framework for ATM Networks", *IFIP Performance Modelling and Evaluation of ATM Networks*, Chapman and Hall, 1996

[16] A. K. Parekh, R. G. Gallager, "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single Node Case", *IEEE/ACM Transactions on Networking*, 1, p 344-357, 1993.

[17] S. Rajagopal, V. G. Kulkarni and S. Stidham, Jr., "Optimal Flow Control of a Stochastic Fluid-Flow System", *IEEE J-SAC*, vol. 13, No 7, September 1995.

[18] J.-F. Ren, J. W. Mark, J. W. Wong, "A Dynamic Priority Queueing Approach to Traffic Regulation and Scheduling in B-ISDN",*Proceeding of the IEEE Globecom*, p 612-618, 1994.

[19] Chen ShanZi and Yan Liemin, "A New Priority Control of ATM Output Buffer", *Telecommunication System Journal*, 4, p 61-69, 1995.

[20] S. Valaee, M. A. Kaplan, "Congestion Control in a Constraint-Worst-case Framework",*IFIP Performance Modelling and Evaluation of ATM Networks*, Chapman and Hall, 1996

[21] Hui Zhang, "Service Disciplines for Guaranted Performance Service in Packet-Switching Networks", *Proceedings of the IEEE*, 83(10), October 1995.

Service Specific Information Based Resource Allocation For Multimedia Applications

István Cselényi
Broadband Internet Group, Network Services Department, Telia Research AB
Vitsandsgatan 9, S-123 86 Farsta, Sweden
E-mail: istvan.i.cselenyi@telia.se
AND
Róbert Szabó
High Speed Networks Laboratory, Dept. of Telecommunications and Telematics
Technical University of Budapest Budapest, H-1111 Sztoczek u. 2, Hungary
E-mail: szabor@ttt-atm.ttt.bme.hu

Keywords: resource reservation, performance analysis, multimedia services, signaling

Edited by: Mohsen Guizani

Received: January 13, 1999 **Revised:** August 30, 1999 **Accepted:** September 15, 1999

Efficient resource allocation with proper quality of service support is a frequently addressed topic both in the Internet and ATM community; especially in case of multimedia applications. This paper presents an intelligent allocation approach that utilizes the knowledge of multimedia application and user behavior during connection establishment. The general problem is analyzed and modeled for a simple scenario including video-phone service. The performance of proposed schemes is evaluated and benefits are highlighted in terms of connection setup time, blocking probability, over-provisioning and signaling intensity based on simulation study.

1 Introduction

There is an unambiguous mapping between the telephony service and the corresponding resources in the telephony network. However, resource management is much more complicated in case of novel computer and telecommunication networks that support multi-rate, multimedia and multi-party services with different quality requirements. Network resources are usually managed by bearer protocols, which are not able to take into account the behavior of the specific telecommunication service. The aim of this paper is to prove that the usage of service specific information can enhance the performance of resource allocation. More concretely, if services enabling discrete downgrading in quality could support information about discrete media scaling levels of their application and the corresponding resource demand, then network nodes can perform an *intelligent downgrading function* when processing incoming connection requests. The efficiency could be further improved by considering the *user's preference index* of each quality setting for the selection of next downgrade step. In this way, both the amount of over provisioned network resources and the number of repeated connection setup trials can be reduced. A new allocation scheme has to be designed which takes this special teleservice information into account for setting up connections.

The rest of this article is organized as follows. First, we describe the service specific information that is used by the investigated resource allocation schemes in Section 2. Two new and a traditional allocation schemes are introduced in Section 3. Several performance measures and our videophone service are defined in Section 4. Finally, numerical results are presented in Section 5 and conclusions are drawn in the last section.

2 Service Specific Information

The separation of *tele-service* and *bearer service* layers (or in other words application and bearer network layers) is widely proposed for the control of distributed multimedia services and applications [2, 11, 9]. One of the benefits of this architecture is that the resource allocation scheme running in the bearer network layer can retrieve not only the usual connection and QoS descriptors from the higher layer but can also gain information about the multimedia application and its users. The rest of this section details this service specific information.

2.1 Resource Demand Vector of the Application

Multimedia applications usually have several parameters, which influence the amount of network resources (e.g. bandwidth or multiplexing buffer size) they require for a certain QoS. For instance, by altering video coding parameters of a scalable MPEG application, its bandwidth requirement scales from 2.7, 3.085, 3.6, 4.32, 5.4, 7.2 up to 10.8 Mbps [12]. For describing a general case, let us denote the ith independent *quality parameter* of the application by q_i; the number of independent parameters by N and the number of quality levels each parameter can take by N_{Q_i} i.e.

$$q_i \in \mathcal{Q}_i = \{q_{i,1}, \dots, q_{i,N_{Q_i}}\}, \ i = 1, \dots, N. \quad (1)$$

Further, the number of independent *network resource types* (e.g. bandwidth, token bucket size, route, priority, ...) that applications are competing for is denoted by K. Let's also define a mapping function between the quality parameters and the network resource types ($\mathbb{R}$) in the following way:

$$r_j \in \mathcal{R}_j := \mathbb{R}_j\{\mathcal{Q}_1 \times \dots \times \mathcal{Q}_N\} \quad (2)$$
$$j = 1, \dots, K$$

where r_j is the value of the jth network resource type and $\mathcal{R}_j$ denotes the *resource set* which contains the possible resource values that the application may take from the jth resource type in case of different quality settings. The number of elements in each $\mathcal{R}_j$ is

$$L = \prod_{i=1}^{N} N_{Q_i}. \quad (3)$$

The elements of each resource set can be sorted in descending order yielding K *resource vectors*:

$$\mathbf{r}_k = [r_{k,1}, r_{k,2}, \dots, r_{k,L}]$$
$$r_{k,i} \leq r_{k,j} \iff i > j$$
$$i, j = 1, 2, \dots, L; \quad k = 1, 2, \dots, K \quad (4)$$

It can happen that some combinations of the N parameters result the same resource requirement, thus neighboring elements of a resource vector can be equal. The $\mathbf{r}$ resource vectors represent the special service specific information one can determine based on the capabilities of the multimedia application.

2.2 User's Preference Function

The second task is to describe how much the user prefers the application's quality resulted by the different parameter settings. This can be expressed by a *preference index*, i.e. a positive number which is assigned to each combinations of the quality parameters. These preference values can be expressed by an N dimensional vector-scalar function i.e.

$$p = P(q_1, \dots, q_N) \in \{0, 1, \dots, N_P\} \quad (5)$$

where N_P denotes the maximum preference index (e.g. 255).

The goal of user's *preference function* is two-fold in our case; in one hand our user model initiates calls with QoS requests selected from the available settings through the distribution given by the user-preference function and on the other hand, if downgrading happens inside the network, the user's decision is also emulated through the user preference function.

2.3 Downgrade Vector for the Network

Application specific information i.e. the resource vector can be used for downgrading reservation requests. Elements of the $\mathbf{r}_k$ vector can be ordered into a downgrade vector, denoted by $\mathbf{d}$, that specifies the discrete resource downgrading steps the application may require from the network, in a descending order:

$$\mathbf{d}_k = [d_{k,1}, d_{k,2}, \dots, d_{k,J_k}] \qquad \begin{array}{l} d_{k,i} \in \mathcal{R}_k \\ i = 1, 2, \dots, J_k \\ k = 1, 2, \dots, K \end{array} \quad (6)$$

where J_k denotes the number of *different* elements in the downgrade vector belonging to the kth resource type. This can be equal or less than L because every non unique resource value should be omitted from the downgrade vector. The $\mathbf{d}_k$ downgrade vector expresses information that network nodes shall consider during the reservation for the kth resource. The allocation of either more or less resource than the quantities described in $d_{k,i}$'s is just a waste of resource.

3 Resource Allocation Schemes

This section presents two resource allocation schemes that uses the resource demand and preference matrices described previously. These schemes are compared to an ordinary resource allocation scheme, which is somewhat similar to recent IP resource reservation concepts [15, 1, 4]. For the sake of comparability, we consider call admission control (CAC) taking place in nodes and the sender initializes all reservation. Moreover, only one network resource is considered in the following, namely the bandwidth. Others like switching capacity and buffer size are excluded from the recent investigation.

3.1 Admission Control

The basic idea of our CAC algorithm is to reject each incoming connection request if the available bandwidth in the node is less than the smallest element of the $\mathbf{d}$ vector, but anyway accept the connection request and allocate bandwidth according to the largest element of the $\mathbf{d}$ vector that fits into the available capacity. If the available bandwidth at the jth node is denoted by f_j, then the call admission control mechanism applied at the jth node can be expressed as follows:

$$\begin{array}{lll} f_j < \min\limits_i(d_i) & \Longrightarrow & \text{reject request} \\ \exists d_i \leq f_i & \Longrightarrow & \text{allocated bandwidth} = \\ & & = \max\limits_i\{d_i | d_i \leq f_j, \ i = 1, \dots, J\} \end{array} \quad (7)$$

If downgrading happens, a reservation tear message is sent backward to the up-stream nodes in order to release excess reservations resulting from recent downgrading. The tear message is also propagated along the backward path.

3.2 User Revision

However, the mere fact that the network downgrades to those quality levels that bear with rational quality settings for the application, does not necessarily mean that the user will at all accept the established connection. In fact, if downgrading happens inside the network, the user will decide whether the quality level of the established service is still acceptable or not. The preference function can characterize this decision, further referred as *User Revision* in this paper.

3.3 Types of Allocation Scheme

Apart from the "baseline" allocation scheme (type 0), there are two basic types of the allocation scheme we propose in this paper. The first (type 1) uses only the downgrade vector, while the second (type 2) utilizes the preference function too, as service specific information. In spite of this principal difference, the reservation request is processed for both according to the following rules: (i) a reservation request is sent to the network with a downgrading vector describing the discrete meaningful bandwidth steps of the specific application; (ii) in the nodes along the path CAC is performed as described previously; (iii) if downgrading happens, a tear message is sent backward along the path. The three basic allocation types are further modified by disabling the user revision function (type 0* and type 1*). The different allocation types are summarized in Table 1.

Table 1: Resource Allocation Schemes

Allocation Type	Service Specific Information	Downgrading in the Network	User Revision
Type 0	No	No	Based on **P**
Type 0*	No	No	No
Type 1	**d**	Based on **d**	Based on **P**
Type 1*	**d**	Based on **d**	No
Type 2	**d and P**	**d and P**	No

Allocation Type 0

This is the reference reservation model without any service specific information for processing reservations. The reservation request is launched into the network, and progresses forward until it finds a bottleneck link or reaches the destination. If in a certain node there is not enough available bandwidth, it stops the reservation message, creates a tear message for the total requested bandwidth to the previous nodes. This tear message also contains the bottleneck bandwidth. The caller, receiving this tear message either tries a downgraded request considering both the preference function and the bottleneck capacity or considers the call to be blocked.

Allocation Type 0*

This is a variant of Type 0, where the user always accepts an established reservation when it meets any of the elements of the resource set.

Allocation Type 1

In this case, after a reservation request has been launched to the network the source either gets a release message, that means the failure of the reservation request, or an acknowledgment message with the reserved bandwidth. As downgrading could happen inside the network, the acknowledged bandwidth expresses the effective reservation. Now the user - i.e. the preference function in our case - decides whether the acknowledged allocation is acceptable or not. If no downgrading happened then the user accepts the established reservation with probability 1. However, in case of downgrading, the user may reject the reservation or choose a more preferred resource setting according to the preference function. In either case, the excess bandwidth reservation is released and the necessary bandwidth is allocated for the application by issuing allocate and tear excess[1] messages.

Allocation Type 1*

This type is similar to Type 1 except that the User Revision is disabled, i.e. any non-zero allocation is accepted by the initiator user.

Allocation Type 2

In this case, not only the downgrade vector but also the preference function is included in the reservation request message. If a request can not be admitted, the node considers both the downgrade vector and the preference function for downgrading. Hence, the user behavior is modeled within the network nodes, so user revision is not required.

4 Performance Analysis Framework

This section introduces the performance measures, the simulation tool and the investigated service and its parameters.

4.1 Performance Measures

Several measures can be used for comparing the performance of resource allocation schemes. Firstly, the *connection setup time*, i.e. the time elapsed between the request and establishment of a connection. The number of rejected connection requests divided by the total number of requests i.e. the *blocking probability* of connection requests can be the second measure. If resource reservation is considered in the network, it implies that the control architecture allocates resources, which may become unneeded and will be released later. This performance characteristic can be quantified by the *over-provisioning factor* that characterizes the amount of excess resources and the duration of their reservation:

$$O_m = \sum_{i=0}^{I_m} e_{m,i} \Delta t_{m,i}, \quad m = 0, 1, \ldots, M \tag{8}$$

where $e_{m,i}$ denotes the amount of excess resource released by the ith tear message received by the mth node;

[1]note that the allocated bandwidth may be zero

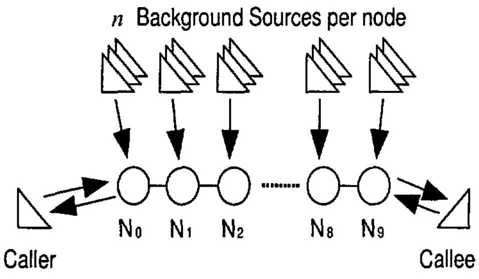


Figure 1: The investigated Video-phone service

$\Delta t_{m,i}$ represents the duration of the ith reservation at the mth node; M denotes the number of nodes and I_m the number of tear messages received by the mth node.

The network nodes process different number of atomic signaling messages for a successful connection setup in case of the different allocation schemes. The intensity of message arrivals is an important factor, because each signaling message requires special treatment from the router affecting its forwarding performance. Therefore the *signaling intensity* measure is defined:

$$S_m = \frac{\#\text{ of atomic signaling messages}}{\#\text{ of successful reservations}}\bigg|_m \tag{9}$$

where $m = 1, 2, \ldots, M$.

The proposed reservation types were compared using these measures.

4.2 Simulation Environment

We developed an object oriented *discrete event simulator* in C++ that emulates the necessary signaling and resource contention in a multi-rate network environment. The designed simulation model and object hierarchies exploit the possible flexibility of object oriented languages.

4.3 Video-Phone Scenario

The simple simulation scenario presented in Figure 1 was used for our analysis. A fix route of ten network nodes (N0-N9) is assumed between the video-phone terminals of the caller and the callee parties. The requests for audio and video connections are handled together, since these media should be synchronized. Infinite switching capacity and 155.2 Mbps output cell rate are assumed for the nodes. Each node receives the connection requests of the investigated foreground source forwarded by its neighboring node and the requests of n independent background sources connected to it. The background requests have a different route than N0-N9, i.e. there is only one serving node per background source. Furthermore, background requests were not downgraded but considered as blocked in case of congestion. The nodes have a processing time of 10 ms per signaling message, the terminals' response time is 100 ms, while the propagation delay is neglected. Peak rate allocation is assumed in the nodes with no priority scheduling. Both the foreground and background sources generate requests according to a Poisson process with $\lambda = 0.0011/s$ arrival rate and $1/\mu = 100s$ mean holding time (see related works: [10, 6, 14, 7, 13]).

Table 2: Quality Parameters and Levels of Video

Video				
Quality Parameter	Picture Size [Pixels]		Frame Rate [fps]	
Level	"Small"	"Large"	"Slow"	"Quick"
Value	192x144	384x288	10	25

Table 3: Quality Parameters and Levels of Audio

Audio				
Quality Parameter	Audio Channel		Sampling Rate [kHz]	
Level	"Mono"	"Stereo"	"Low"	"High"
Value	1	2	24	48

4.4 Service Specific Parameters

The investigated video-phone service has two media (video and audio) both having two independent quality parameters (see Table 2,3).

The combinations of the two parameters and values yield four discrete quality levels for both media, which levels require different amount of network resources from the bearer network. These requirements are summed and given in the resource vector. The peak cell rate demand for each quality settings can be determined from Table 2-3, and are shown in Table 4.

Although these calculations are very simple, the resultant bandwidth requirements are in the range of a real audio and a coded video channel [5, 8]. Instead of a single peak, mean or minimum cell or bit rate (which are commonly used in standards), a set of bandwidth values are given which the application can exactly use for data transport. But are these values equally preferred by the service user?

Table 4: Mapping between quality settings and resource set of video-phone service

User Quality Levels for Video	User Quality Levels for Audio			
	low sampling rate		high sampling rate	
	mono	stereo	mono	stereo
small size simple frame rate	25	25	28	32
small size double frame rate	61	62	64	68
large size simple frame rate	97	98	100	104
large size double frame rate	241	242	244	248

Table 5: User's Preference Values

Video	Audio			
	worse	bad	good	best
worse	4	2	2	1
bad	3	4	2	2
good	3	3	6	4
best	2	3	5	8

In general, users prefer the similar quality settings for audio and video. It happens rather rarely that a user requests a high quality video channel with mono, low sampling rate audio or vice-versa. Thus the 16 quality settings given in

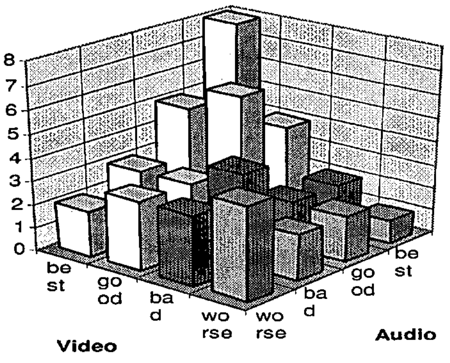


Figure 2: User Preference Function

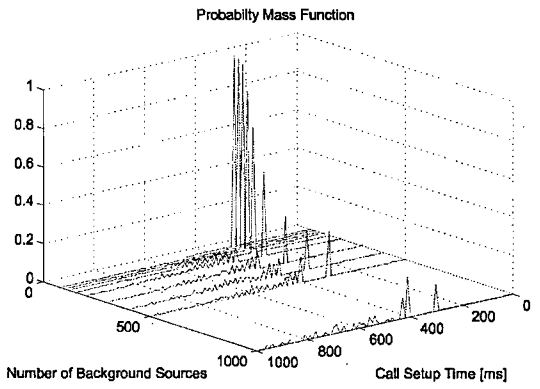


Figure 3: PMF of call setup time for Type 0

in the nodes, setting up a connection takes always a fix time (if it successes) for type 1, type 1* and type 2.

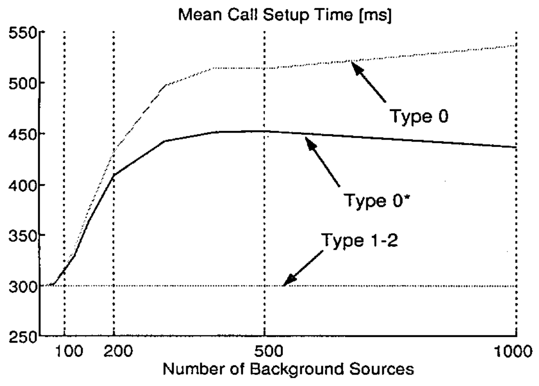


Figure 4: Mean Call Setup Time for the Five Allocation Types

5 Numerical Results

Numerical results were generated in the aforementioned simulation environment. The confidence level of each simulation runs was set according to model 100,000 to 1,000,000 foreground events, depending on the intensity of the background traffic.

5.1 Call Setup Time

Figure 3 shows the call setup time distribution for the type 0 scheme. It always starts from 300 ms, that is the minimum call setup time, and flattens with the increasing number of background sources. It is highly dependent on the actual network load contrary to the type 2 scheme which gives a constant setup delay of 300 ms whatever the network load is.

The mean call setup time of the five allocation schemes is presented in Figure 4. Comparing type 0 to type 0*, a moderate improvement can be noticed, which means that the free bandwidth determined by the random load situation in the network rarely matches the resource demand of quality settings preferred by the user. Due to downgrading

5.2 Setup Retries

The difference between the performance of type 0 and type 0* can be further stressed by visualizing the number of call setup retries (Figure 5,6). It can be seen that more retries are needed to make a successful reservation if there are higher network load. Moreover, type 0* requires much less retries for connection setup than type 0. The reason of this is that the user does not check the preference function but accept everything in type 0*.

5.3 Call Blocking Probability

In Figure 7 we show the foreground source's blocking probability as a function of the number of background sources. It can be seen that the blocking probability is an increasing function of the number of background sources for each type, and the curvatures are almost identical. It is also shown, that the advantage of instant downgrade capability of type 1 and type 2 does not really influence the blocking probability, however it slightly remains below the

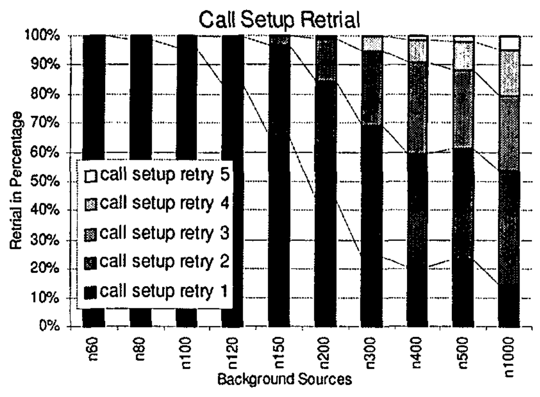


Figure 5: Call Setup Retries With (type 0)

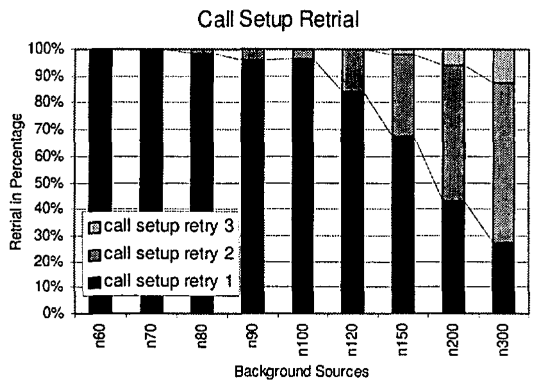


Figure 6: Call Setup Retries Without User Revision (type 0*)

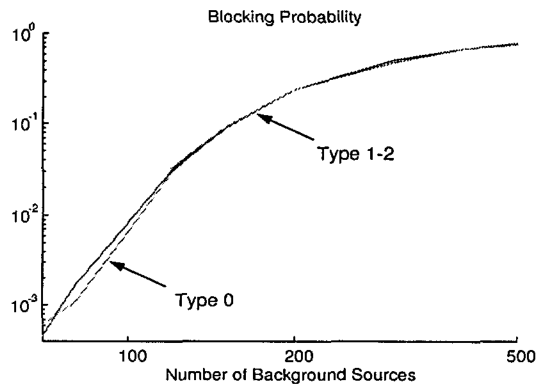


Figure 7: Blocking Probability Without User Downgrade

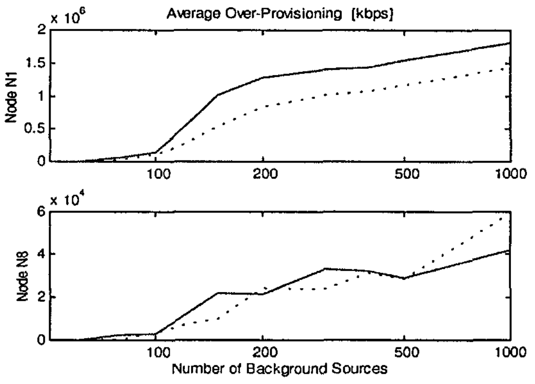


Figure 8: Over-Provisioning for Type 0 and Type 1

blocking probability of type 0 in case of large network load, while it is a little higher in case of small network load.

5.4 Over-Provisioning

Figure 8 shows the over-provisioning on node 1 and node 8 for allocation schemes type 0* and type 1*. It can be seen, that with the increasing number of call setup retries (see Figure 7) the over-provisioning of the type 0* allocation increases. Moreover, the curve of type 1* is always below of the type 0*. It means less waste of bandwidth which could result in higher efficiency in sharp situations. It is obvious that the worst case node is the first node next to the caller, where the unnecessary reservations are kept longest. For node 8 - far end size -, the over-provisioning factor straightens and is smaller with two magnitude for both allocation schemes. This unbalanced way of loading the network nodes is often called location bias [3].

Over-provisioning is also plotted for the allocation types in which the caller revises the acknowledged bandwidth. Reservation types 1 and 2 resulted in smaller over-provisioning than type 0, as it is pronounced in Figure 9. It is interesting that type 2 can utilize the additional service specific information (i.e. the user preference function) for sparing with excess bandwidth in case of small background load ($n < 200$), while type 1 performs better than type 2 in case of more background calls. The over-provisioning on node 1 has the same order of magnitude as in the previous case.

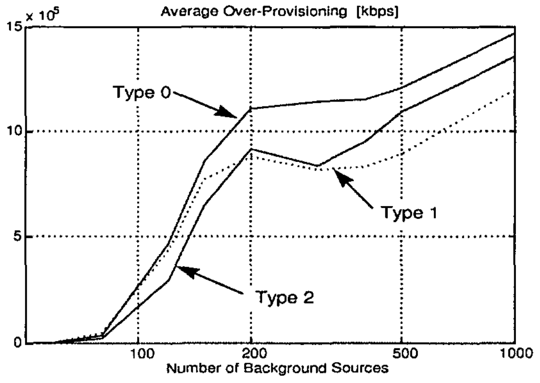


Figure 9: Over-Provisioning on Node 1

5.5 Signaling Intensity

Figure 11 presents the signaling intensity on the first (N0) and last node (N9) in case of type 0* and type 1*. Node 0 should handle more atomic signaling messages in case of allocation type 0, while this difference is very small on node 9. Allocation type 1 causes less location biased load regarding signal handling.

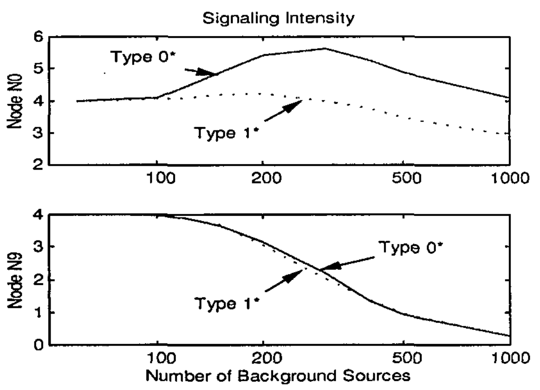


Figure 10: Signaling Intensity on Node 0 and Node 9

The same performance measure is plotted for the types implementing user revision (Figure 11). Only a small improvement can be observed in case of this variant of the baseline allocation type (i.e. type 0). Type 2 has the smallest location bias and less intensive signaling.

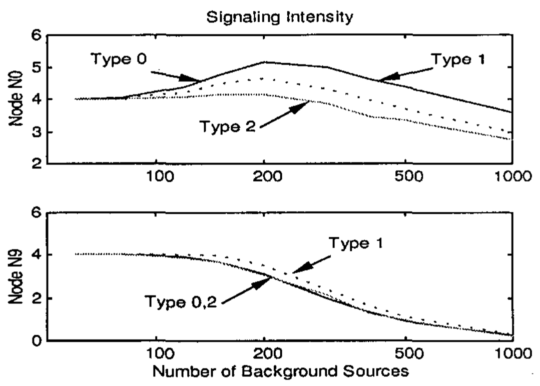


Figure 11: Signaling Intensity on Node 0 and Node 9 in Case of User Revision

It is noticeable, that the presented allocation schemes require signaling messages of different size, because types 1 and 2 include the Downgrade Vector, type 2 the Preference Function too, but type 0 does not transfer service specific information. Thus the overhead due to signaling is maximal for type 2 and minimal for type 0.

6 Conclusion

Two intelligent resource allocation schemes were introduced and analyzed in this paper, which utilize service specific information (i.e. downgrade vector and user preference function) in order to minimize over-provisioning and call setup time. Two different reservation types were defined and compared with an ordinary reservation scheme in terms of performance evaluation based on simulation. The numerical results highlight that the service specific information results in a fix and shorter call setup time, less reservation trials, less over-provisioning and less intensive signal handling in the nodes by maintaining the same blocking probability. However, the usage of further service specific information is questionable, as one always have to find a trade-off between complexity and efficiency.

Our future plan is to test our reservation approach on more complex network topologies since the presented scenario sidesteps the potential conflicts between automatic downgrading and routing. Moreover, we also want to to apply this concept for other teleservices and to make an analytical verification of the proposed allocation scheme.

References

[1] R. Barden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. Resource ReSerVation Protocol (RSVP)-Version 1 Functional Specification. Technical report, Internet Draft. RFC 2205, 1997.

[2] I. Cselényi, I. Szabó,.P. Haraszti, N. Bjorkman, and C. Gisgard. A versatile multimedia platfor. In *15th IASTED Internation Conference on Applied Informatics*, pages 276–279, 1997. Innsbruck, Austria.

[3] I. Cselényi, R. Szabó, I. Szabó, A.L. Henner, C. Gisgard, and N. Bjorkman. Experimental platform for telecommunication resource management. *Computer Communication Journal: Special Issue on the Stochastic Analysis and Optimization of Cummunication Sstems*, 1998.

[4] G. Fehér, K. Németh, M. Maliosz, I. Cselényi, J. Bergkwist, D. Ahlard, and T. Engborg. Boomerang - A simple protocol for resource reservation in IP networks. In *IEEE Workshop on QoS Support for Real-Time Internet Applications*, June 1999. Vancouver, Canada.

[5] H. Haapasalo, I. Norros, and T. Raij. Some experiments from ATM traffic measurements. In *The Twelfth Nordic Teletraffic Seminar*, Aug 1995. Espoo, Finland.

[6] Y.B. Lin and I. Chlamtac. Effective call holding times for PCS network. submitted for publication, available at http://liny.csie.nctu.edu.tw.

[7] D. Medhi and S. Guptan. Network dimensioning and performance of multiservice, multirate loss networks with dynamic routing. *IEEE/ACM Transaction on Networking*, 5(6):944–957, December 1997.

[8] S. Molnár, I. Cselényi, N. Bjorkman, C.G. Perntz, and M. Boda. ATM traffic measurements and analysis on a real testbed. In *10th Internation Teletraffic Congress Specialist Seminar*, pages 237–250, September 1997. Lund, Sweden.

[9] N. Natarajan. *Network Resource Information Model Specification*, 1997. Version 3.0.

[10] D. Niehaus. Performance benchmarking of signaling in ATM networks. *IEEE Communication Magazine*, pages 134–143, August 1997.

[11] RACE Project R2044. *MAGIC-Multiservice Applications Governing Integrated Control, Service Descritption Framework and B-ISDN Service Descripton*, 3rd deliverable edition, 1992.

[12] Litton Network Access Systems. *CAMVision-2 7615, User's Guide*, October 1998.

[13] W.C. Wong. Packet reservation multiple access in a metropolitan microcellular radio environment. *IEEE Journal on Selected Areas of Communication*, 1993.

[14] M.C. Yuang and Y.R. Haung. Bandwidth assignment paradigms for broadband integrated voice/data networks. *Computer Communications*, 21(3):243–253, 1998.

[15] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. RSVP: A new resource reservation protocol. *IEEE Networks Magazine*, September 1993.

Fault-Tolerant ATM Switch Using Logical Neighborhood Network

A. Rayhan and F. Elguibaly
Dept. of Electrical and Computer Eng.
University of Victoria
Victoria, B.C. V8W 3P6 Canada
AND
A. Almulhem
Dept. of Computer Eng.
King Fahd University of Petroleum and Minerals
Dhahran 31261, Saudi Arabia

Keywords: Multi-stage interconnection networks, Fault tolerance, complexity analysis, routing algorithm.

Edited by: Mohsen Guizani

Received: February 3, 1999 **Revised:** June 29, 1999 **Accepted:** July 31, 1999

A fault-tolerant ATM switch based on the logical neighborhood network is presented. The switch is targeted for terabit switching through localized traffic control and management, use of input/output queuing, distributed simple cell routing algorithm, and soft fault tolerance. The redundant hardware for fault tolerance is actively used by the switch to increase throughput, hardware utilization, and reduce cell loss and delay. For an $N \times N$ switch, the inherent structure of the LN-ATM network allows the simultaneous delivery of $\log_2 N + 1$ ATM cells to any output port. This is achieved using bit-serial operation and without speeding up the switching fabric. An attractive advantage of the proposed switch is support of data broadcast and hot-spot operation without the use of replicating hardware modules. We also propose a dynamic routing algorithm which detects blocked and/or faulty links. The switch throughput, cell loss probability, and performance under faults are studied using numerical simulation.

1 Introduction

ATM is designed to support switching for B-ISDN operating at high speed links such as OC-3 (155 Mbps) and OC-12 (622 Mbps). These high operating speeds place hardware design constrains on ATM switches. Some of the constraints include high throughput, and low cell loss probability (CLP), low delay, support of broadcast, multicast and service classes (QoS) [1]. Several ATM switches have been proposed in the literature addressing some of these issues [1]-[7].

Among the strategies used to design ATM switches, output buffering and shared-memory are the most efficient in terms of throughput and low delay [4]. However, these strategies have a limiting drawbacks. Shared-memory switches are limited by memory performance. The shared memory requires complex and fast management and scheduling policies and memory speed sets the limit on switch size and line rate [2]. Output buffering switches are limited by switching fabric performance. The switching fabric must be nonblocking and must run N times faster than the line rate in order to prevent output contention for a switch of size $N \times N$ [5].

The need to operate at OC-12 and even OC-192 (10 Gbps) in the not too distant future place unheard-of difficulties for the silicon designer. These line rates will only be satisfied by speed up techniques that result in *slowing down* of the operating speed of the switch. Thus we consider any schemes that enhance switch performance by speeding up as unacceptable.

One proposed approach for a fault-tolerant ATM switch is based on B-tree [8]. In this design, the switching fabric is based on a tree structure using 4×4 switching elements (SEs) to implement logical 2×2 SEs. The proposed ATM switch in that reference can be adapted to support cross-point buffering or partial shared memory. However, the SE complexity is doubled without being fully utilized. Furthermore, it provides limited redundant paths capabilities.

In another approach, a banyan network is connected using two types of SEs which have circular and shuffle type connections [10]. The first type of SEs provides the banyan structure while the other one provides the path redundancy to alleviate faults. This network gives a large number of redundant paths but has limited disjoint path capability. Moreover, path redundancy is not utilized to enhance the performance of the switch and the delay between SEs is not constant.

In [9], the approach of cross-linking two physical banyan networks is used. In this switch, the redundancy introduced by cross-linking SEs makes the switch fault tolerant. The main drawbacks of this design are doubling the amount of hardware used, limited disjoint path capability and excessive number of internal links. Another common drawback of the previously discussed approaches is that the need for switching fabric speed up is not eliminated in order to resolve output contention.

In this paper we propose an ATM switch that combines input and output queuing with the excellent fault-tolerance properties of the ILN network. This allows us to obtain excellent fault tolerance capabilities and low cell loss ratios and delays. The combination of input and output queuing also removes the memory access speed from presenting a bottleneck to the switch performance. In addition, we present a simple dynamic routing algorithm that is distributed throughout the ILN switching elements to provide fast path determination and allow back off mechanisms to choose alternate paths in when faults or internal blocking are encountered. The ILN design presented in [12] is used to provide a high degree of fault tolerance and achieve high performance. In this paper we propose an ATM switch that combines input/output. In our design, we enhance the utility of the the ILN [12] by implementing a dynamic and distributed routing algorithm at the switching elements level. This provides a high degree of fault tolerance and achieves high throughput and low cell loss. The main features of the switch are:

- Reduced complexity for output queuing by running at the same speed as the line rate.
- Providing $(n + 1)$-disjoint paths, where $n = \log_2 N$, for any source to any destination.
- Support for broadcast, multicast and hot-spot traffic.
- Dynamic routing of cells using an algorithm that detects blocked paths and faulty links.
- Provide n fault-tolerance.
- The additional stages added for fault tolerance are actively used to reduce cell loss.
- The redundant paths are used without internal speedup.
- The number of redundant paths increases dramatically as the network size grows.

Furthermore, the proposed ATM switch is modular and has low CLR and delay, and high throughput.

This paper is organized as follows. In Section 2, the detail of the proposed switch is presented. Section 3 outlines the routing algorithm for the switch. Section 4 discusses the fault-tolerance capability of the proposed switch. The simulations and numerical results are presented in Section 5. Conclusions are found in Section 6.

2 Proposed LN-ATM Switch

A block diagram of the proposed $N \times N$ Logical Neighborhood-ATM (LN-ATM) switch is shown in Figure 1. The switch consists of FIFO queues at all input and output ports and the Improved Logical Neighborhood Network (ILN) [12] to switch cells. The size of the input and output queues is B_i and B_o, respectively, where B_i is typically much smaller than B_o.

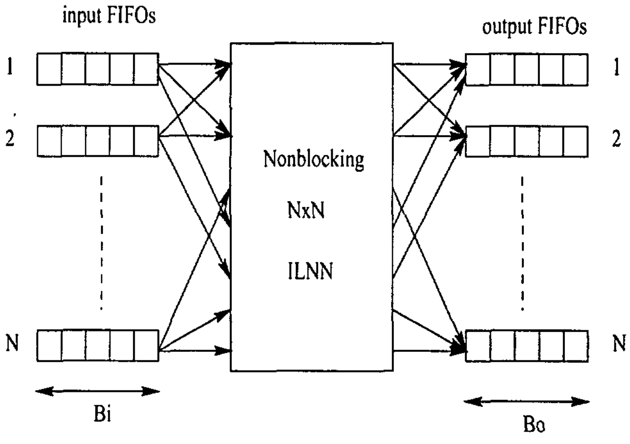


Figure 1: Proposed $N \times N$ LN-ATM switch.

When a cell arrives at an input port, it is stored in the corresponding FIFO queue. Once a cell reaches HOL, it is routed to its destination through the ILN using a dynamic routing algorithm. When the cell reaches the output port, it is stored in the output FIFO queue. The switch supports hot-spot traffic by allowing each output port to receive upto $n + 1$ cells in one routing cycle.

The rich connections in the ILN results in a nonblocking network that is n-fault-tolerant and $(n + 3)!/6$ redundant paths exist between source-destination pairs. In addition, it has $n + 1$ disjoint paths from any source to any destination. This allows the switching fabric to deliver $n + 1$ cells to any output port. Thus the switch supports broadcast and multicast traffic. The switch operates in queue loss mode where cells are routed to their corresponding outputs regardless of the output queue size. Hence, a cell is lost if the input or output queues are full. The ILN operates synchronously and cells received at the input ports are independently self-routed to the output ports.

The use of input and output queues allows operation at line rate or lower. This is perhaps the most desirable feature of our proposed switch since it allows support of OC-12 rates and higher.

An $N \times N$ ILN consists of $n + 1$ stages excluding the input and output, where $n = \log_2 N$, and each stage has N elements as illustrated in Figure 2. Input switching elements at stage 0 are $1 \times (n + 1)$. Switching elements at stages 1 to $(n+1)$ are $(n+1) \times (n+1)$ each. The elements at the output stage, $n+2$, are of $(n+1) \times 1$. Each output element can receive up to $(n + 1)$ cells in one routing cycles. The received cells are buffered at their output queues.

Switching elements in stage i are connected to the ones in stage $i + 1$ such that their binary addresses differ by at most one bit. For example, for $N = 8$, switch 001 in stage 1, will be connected to switches 000, 001, 011 and 101 in stage 2. The input and output elements are linked to the next and previous stages following the same way as internal stages.

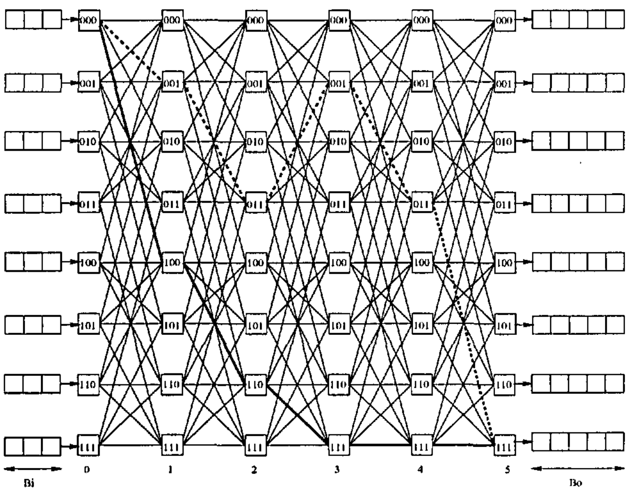


Figure 2: Block diagram of 8 × 8 LN-ATM switch with B_i input buffers and B_o output buffers.

2.0.1 Redundant Paths

The channel graph of ILN for all the redundant paths according to the routing algorithm is shown in Figure 3 for $n = 2$. Let R_n represent the number of redundant paths between input i and output j where $i \neq j$, for ILN, where $n = \log_2 N$. The nodes which are part of the paths between input and output terminals are labeled from A to J. Nodes H, I, and J each has one link to reach the output. Node D has 3×1 alternatives to reach the output, $a_1 = 3$. Nodes E, F and G each can reach output terminal in 2×1 different paths. Node A is connected to nodes D, E and G. Hence, the number of paths in which node A can reach the output is $2 \times 1 \times 2 + 3$,

$$a_2 = 2! \times 2 + a_1. \tag{1}$$

Node B has the same connectivity as A. Node C is connected to E, F, and G. Therefore, the number of redundant paths between C and output is $3 \times 2 \times 1$. Input terminal is linked to A, B, and C. Hence, the total number of redundant paths, R_2 is

$$R_2 = 3! + 2 \times a_2. \tag{2}$$

Generalizing the above argument for $n \geq 2$, we can formulated the recurrence relations as

$$\begin{aligned} R_n &= (n+1)! + na_n & (3)\\ a_m &= (n+2-m)m! + (m-1)a_{m-1} & (4)\\ a_1 &= n+1, & (5) \end{aligned}$$

where $2 \leq m \leq n$. The second terms of Eqs. 3 and 4 and Eq. 5 reduce to $(n+1)!$. The second term of R_n and first term of a_m, and second term of a_m and first term of a_{m-1}, etc., reduce to

$$\sum_{i=2}^{n} i!(n-i+2)\prod_{j=i}^{n} j. \tag{6}$$

Hence, we can write Eq. 3 as

$$R_n = 2(n+1)! + \sum_{i=2}^{n} i!(n-i+2)\prod_{j=i}^{n} j \tag{7}$$

Simplifying Eq. 7, we obtain

$$R_n = \frac{(n+3)!}{6}. \tag{8}$$

When $i = j$, the number of redundant paths increases by 1,

$$R'_n = R_n + 1. \tag{9}$$

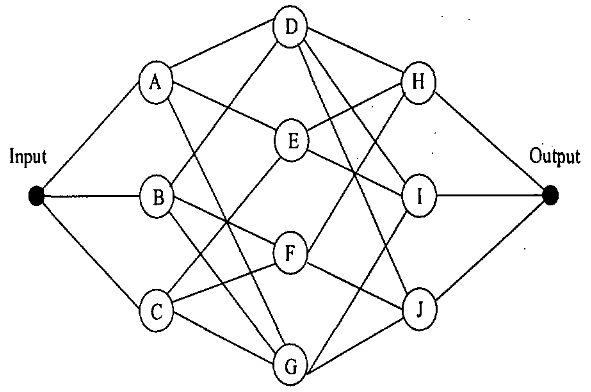


Figure 3: Channel graph for ILN showing all the possible paths between two ports for $n = 2$.

We performed a comparative study for the ILN network used in this work and various other MIN fault-tolerant networks that were reported in the literature. The results are summarized in two tables for comparison purposes. Table 1 shows the number of redundant paths for each network considered as the network size grows. Each network was varied in size from 4 to 256 input/outputs. For small network sizes between four and 16 input or output ports, our ILN network has the maximum number of redundant paths. However, for larger networks the network proposed by Lo [10] will be the clear winner. However, the link interconnection complexity does not make it a good candidate for VLSI implementation or packaging.

Table 2 shows compares the complexity of the various networks. The complexity variables considered were the number of nodes and the number of redundant paths. These variables were parameterized in terms of the network size. The ILN network compares well with the other networks since all of them have approximately the same number of switching nodes and redundant paths. Furthermore, the ILN is well suited to VLSI implementation because of its modularity and its regular wiring pattern in each stage.

Table 1: Comparison between redundant paths for various MIN networks.

n	2	3	4	5	6	7	8
P. Banyan [13]	2	2	2	2	2	2	2
Benes [11]	2	4	8	16	32	64	128
Tagle [9]	4	8	16	32	64	128	256
Itoh [15]	2	5	14	42	132	429	1430
Lin [11]	4	24	120	600	3000	15000	75000
ILN	20	120	820	6720	60480	604800	6652800
Tzeng [16]	2	8	64	1024	32768	2097152	268435456
Lo [10]	4	32	512	16384	1048576	134217728	34359738368

Table 2: Complexity figures for various MIN networks.

Network	Nodes	Redundant Paths
PBanyan [13]	Nn	2
Benes [11]	$N(n - \frac{1}{2})$	2^{n-1}
Tagle [9]	$N(n + 2)$	2^n
Itoh [15]	$N(n - 1) + 1$	$\frac{(2n)!}{n!(n+1)!}$
Lin [11]	$N(3n + 2)/4$	$24 \times 5^{n-3}$
Tzeng [16]	$Nn/2$	$2^{(n-1)n/2}$
Lo [10]	$N(3n - 1)/4$	$2^{(n-1)(n+2)/2}$
ILN	$N(n + 3)$	$(n + 3)!/6$

3 Routing Algorithm

In this section, we propose the Dynamic Routing Algorithm (DRA) to route cells between the input and output ports. The proposed algorithm is simple, localized, and is used to establish a path in a distributed manner. Furthermore, it has a constant delay and is capable of backtracking whenever a path is blocked due to a used or faulty link. The routing algorithm utilizes the Hamming distance property of the ILN and the concept of early routing to establish a path. In the ILN, two SEs in two adjacent stages are connected if the Hamming distance between their addresses is 0 or 1. Early routing means that routing is based on minimizing the vertical distance between the source and the destination at the early stages. This is to say that if a path branches into two or more, the branch which reduces the vertical separation to the destination is selected.

Before proceeding further we need to define the terminology we use in describing the proposed algorithm which is presented in Table 3. Switching occurs at switching stages numbered from 0 to $n + 1$ staring from the input stage as depicted in Figure 2. The current SE is defined by its stage index and address as $S(i, j)$. The SEs connected to $S(i, j)$ in the next and previous stages are $S(i + 1, k)$ and $S(i - 1, l)$, respectively. The set of SE which are connected to $S(i, j)$ and not used or faulty is P. The Hamming distance between SE $S(i, j)$ and destination y is defined as $\mathcal{H}(S(i, j), y)$.

Cell routing is carried out in two phases: path-establishment and cell-transfer [13]. In the cell-transfer phase, actual ATM cell transfer takes place. The path-establishment phase, on the other hand, is based on the request (*Req*) and acknowledgment (*Ack*) protocol. Each input port having an ATM cell to be routed generates a message that specifies the destination port y, and sends a *Req* on a selected link to the next SE. From the set of available links P, a link is selected that satisfies the two conditions explained below.

Link Selection Criterion

1. Potential next SE is selected from P such that $\mathcal{H}(k, y)$ is minimal.
2. When more than one SE in P have the same minimum Hamming distance away from the destination, the one that has minimum vertical distance to the destination is selected, i.e. $\min\{|k - y|\}$.

For simplicity, we will discuss the case of a single request at each SE. The argument can be easily modified for multi-requests using arbitration protocols discussed in [14]. As $S(i, j)$ receives a *Req*, it uses y to select the next SE $S(i + 1, k)$ in the successive stage from list P. The request process propagates through the stages till the destination (output) port y. The output port sends a positive *Ack* along the reverse path to the input port that initiated the initial *Req*. When a *Req* signal is blocked due to a used or faulty link, a negative *Ack* is sent back. The SE that receives the negative *Ack*, tries to initiate a new *Req* using a different link from P according to the selection criterion stated earlier. If it is not successful, it will send a negative *Ack* signal along the traversed route so far. When an input port receives a negative *Ack* on the selected link, it will issue a new *Req* on a different link using the selection criterion stated earlier.

Table 3: Terminology used in describing the routing algorithm.

Term	Symbol	Meaning
switching stage	i	index of the current stage and represented by $i =< i_{n-1} \cdots i_1 i_0 >$
current SE	j	address of the SE in stage i under consideration and represented by $j =< j_{n-1} \cdots j_1 j_0 >$
next SE	k	address of the SE in stage $i + 1$ which is connected to SE j and represented by $k =< k_{n-1} \cdots k_1 k_0 >$
previous SE	l	address of the SE in stage $i - 1$ which is connected to SE j and represented by $l =< l_{n01} \cdots l_1 l_0 >$
available list	P	list of next available SE connect to the current SE and represented by $P = \{k_1, k_2, \cdots, k_{n+1}\}$
destination	y	index of an output port and represented by $y =< y_{n-1} \cdots y_1 y_0 >$

To illustrate the operation of the DRA, we will present the example shown in Table 4. Consider the 8 x 8 LN-ATM shown in Figure 2. Suppose a cell is to be routed from input port 000, $S(0, 000)$, to output port $y = 111$, $S(5, 111)$. The list P of SEs in the next stage which are connected to $S(0, 000)$ is ordered according to the Hamming distance $\mathcal{H}$ first, then by the vertical distance, $\min\{|k - y|\}$. This means that if two next SEs have same Hamming distance away from y then condition 2 should be satisfied. Table 4 shows that $S(1, 100)$ is selected applying the selection criterion; and repeatedly the path shown in bold line in Figure 2 and Table 4 is selected. If a path is not available, an alternative path is sought using the same procedure as above. For example, an alternative path is shown in dotted line in Figure 2.

The pseudocode of the DRA is presented in Figure 4. It starts by switching element $S(i, j)$ receiving a *Req* from $S(i - 1, l)$ to establish a link to output port y in Step 1. Then, it checks if enough number of stages are left to reach the destination y in Step 2. This is necessary to know if the traversed route so far leads to the destination y. If it does not, then a negative *Ack* is sent to the previous SE $S(i-1, l)$ that initiated the *Req*. In Steps 3 and 4, $S(i, j)$ will select $S(i + 1, k)$ from the list of available SEs, P, that has minimum $H(S(i + 1, k), y)$ and $\min\{|k - y|\}$. A *Req* is sent to $S(i+1, k)$ and $S(i, j)$ waits on *Ack*. When a negative *Ack* is received by $S(i, j)$, the next SE from P is selected if there is any left; otherwise, a negative *Ack* is back propagated. However, a positive *Ack* is always propagated backwards till it reaches the input port at $S(0, l)$ indicating a path establishment.

4 Reliability of the ILN

To study the reliability of the ILN, the following fault-tolerance model is assumed [12]:

– Any component (switch or link) can fail.

– Faulty components are unusable.

– Faults occur independently.

– Faults are static.

– Faulty links are excluded from the list of available links for routing.

There are $n + 1$ disjoint paths for each source-destination pair; hence, the ILN is n fault-tolerant. The ILN tolerates n faults in any of the $n + 1$ stages. To study the terminal reliability (R_t) of the ILN, the failure probability of a switch or link is assumed to be q. Among the $n + 1$ paths, one path is sufficient to establish a route between a source and destination. Since there are $n + 1$ distinct paths and $n + 1$ stages, the terminal reliability is defined as:

$$R_t = (1 - q^{n+1})^{n+1}$$

which is based on the formula given in Ref [12].

Figure 5 shows terminal reliability for ILN for $N = 8$, 128, and 1024. The figure shows that the reliability decreases slowly as the probability of failure increases. Also, for a given failure probability, the reliability increases as the network size increases. This is advantageous, since larger switches have higher manufacturing cost and should be more immune to total failure. Furthermore, the ILN has reliability of almost 1 for failure probability of 50% when $N = 1024$.

5 Performance Analysis

To study the performance of the proposed switch, numerical simulations were conducted to find the cell loss probability, throughput, and cell loss when faults are present in the switching fabric. The following assumptions were made in our simulations.

Table 4: Example of establishing a route between input port 000 and output port $y = 111$. $k(a/b)^*$ indicates the selected link, where k is the address of the next SE, $a = \mathcal{H}(S(i+1,k),y)$ and $b = |k-y|$.

Stage i	SE j	P			
0	000	100(2/3)*	010(2/5)	001(2/6)	000(3/7)
1	100	110(1/1)*	101(1/2)	100(2/3)	000(3/7)
2	110	111(0/0)*	110(1/1)	100(2/3)	010(2/5)
3	111	111(0/0)*	110(1/1)	101(1/2)	011(1/4)
4	111	111(0/0)*	110(1/1)	101(1/2)	011(1/4)
5	111	111(0/0)*	110(1/1)	101(1/2)	011(1/4)

```
Start    DRA ($S(i,j)$):
Step 1 : Receive Req $y$ from $S(i-1,l)$.
Step 2 : If $H(S(i,j),y) > n+2-i$ then
             Send negative Ack to $S(i-1,l)$.
             Goto END .
         Endif.
Step 3:  Update & order list $P$.
Step 4:  Select $S(i+1,k)$ where $k$ = first SE in $P$
Step 5:  Send Req to $S(i+1,k)$.
Step 6:  If negative Ack is received then
             $k$ = next SE in $P$
             If no SE is left in $P$ then
                 Send negative Ack to $S(i-1,l)$.
                 Goto END.
             Endif.
             Goto Step 5.
         Else
             Send positive Ack to $S(i-1,l)$
         Endif.
END:     DRA.
```

Figure 4: Psuedocode of the Dynamic Routing Algorithm.

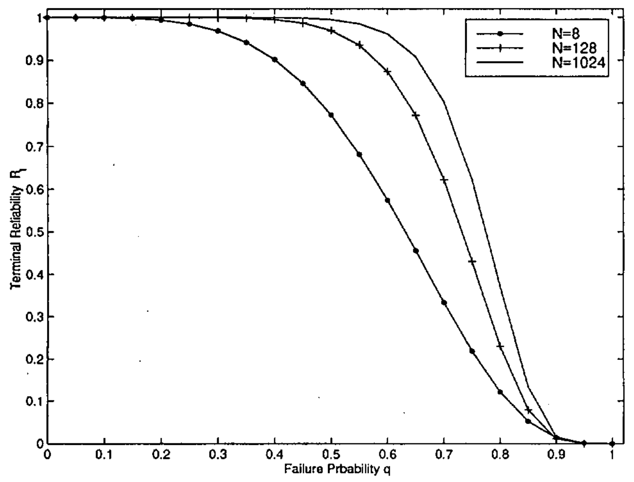


Figure 5: Terminal reliability for ILN when $N = 8, 128, 1024$.

1. ATM cells arrive at the input ports with probability p and are modeled using interrupted Bernoulli processes (IBP).

2. Arriving cells have uniform probability, $1/N$, of being destined to any output port.

3. At a given time slot, at most one cell leaves each input port and $n + 1$ cells can be delivered to any output port.

4. When more than $n+1$ cells are destined to a particular output port, $n + 1$ cells are selected for routing at random. The unselected ones remain at the input buffer HOL.

5. Cell loss takes place at two occasions; when the input or output queues are full.

6. Cell loss due to time out and faulty components are not considered.

7. The simulation was run for 5×10^8 time slots on Sun Ultra 10 for the case when no faults were present. When faults are present, the simulations were run for 2×10^8 times.

To study the effect of queuing, the input and output queues are modeled as FIFOs measured in number of ATM cells stored. The size of input FIFOs are fixed, $B_i = 10$, and output FIFOs are varied, $5 \le B_o \le 80$. The traffic intensity p was set to 0.9. LN-ATM of sizes 8, 16, 32, and 64 are simulated.

Figure 6 shows the effect of output queue size on CLP. The simulation results show that CLP of 10^{-6} can be reached at $B_o = 45$ for all networks. It was observed that the size of the input FIFOs utilized varied for different network sizes. For example, for switch $N = 8$, the input FIFO size was $B_i = 4$ when $B_o = 45$. As the network increases, B_i gets smaller, i.e. $B_i = 2$ when $B_o = 45$ and $N = 64$.

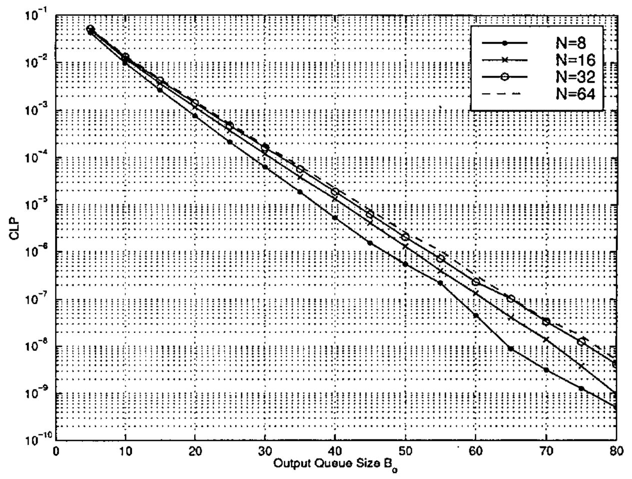


Figure 6: LN-ATM cell loss probability for various switch sizes when $p = 0.9$ and assuming no faults ($q = 0$).

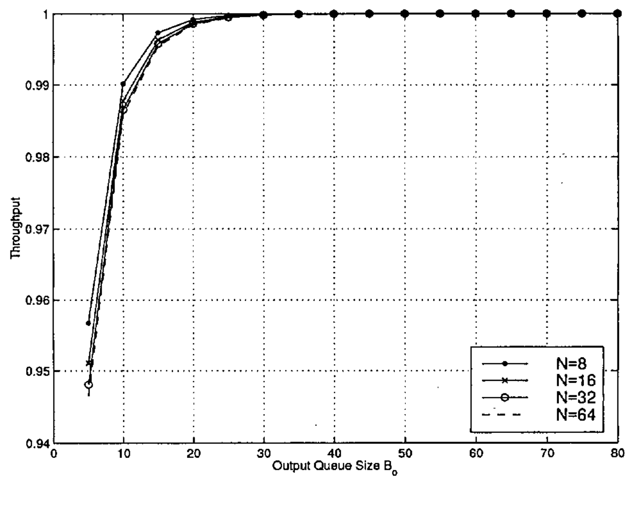


Figure 7: LN-ATM throughput input and output queues for $p = 0.9$ and assuming no faults ($q = 0$).

Figure 7 shows the effect output queue size on the throughput for network sizes 8, 16, 32, and 64. The simulation reveals that throughput of 100% is reached when $B_o = 35$ for all networks at high traffic intensity $p = 0.9$.

Figure 8 shows the mean packet waiting time (measured in time slots) versus the offered load when $B_i = 10$, $B_o = 45$, and no faults were assumed.

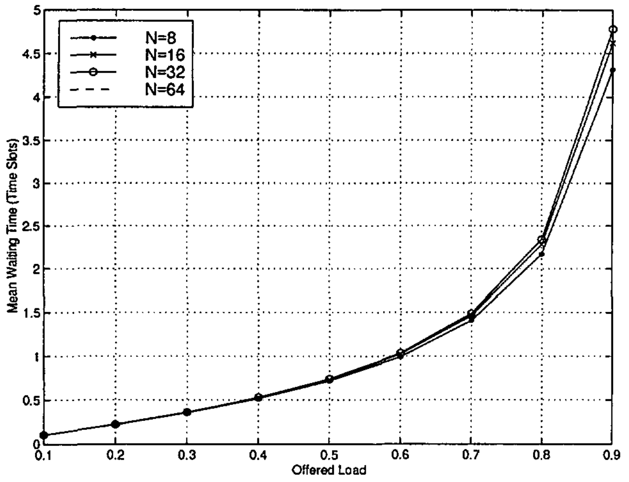


Figure 8: Mean waiting time (in time slots) vs. offered load in an LN-ATM with $B_i = 10$, $B_o = 45$, and no faults were assumed.

The simulations reveal that throughput of 100% is reached when $B_o = 30$ for all networks for high traffic intensity $p = 0.9$. Furthermore, low cell loss probability (10^{-6}) at output queues at reasonable size $B_o = 45$. In addition, cell loss at input queues is eliminated due to receiving capability of the output ports.

5.1 Effect of faults on performance

Faults within the switching fabric have different impact on the switch performance depending on the location of the fault. The most pronounced effect occurs when the fault happens at the last stage connecting the last stage of the fabric to the output port since this reduces the number of links. This has a direct impact on the amount of traffic arriving at an output port. On the other hand, faults occurring at the input or intermediate stages are masked by the inherent redundancy of alternative paths. To simplify the modeling and to reduce simulation time, we considered only faulty output links. We should point out here that other authors assumed in their studies that faults could not happen at the input or outputs stages.

To simulate the effect of faults on the switch performance, the following assumptions were employed.

1. Faults occur only on links connecting the switching nodes to the output ports.
2. Faults are static and do not change throughout the simulation.
3. Faulty links are unusable.
4. Faults occur independently and uniformly among the output links.
5. Faulty links are excluded from the list of available links for routing.

Figure 9 shows the effect of faults in the network on the cell loss probability. In this simulation, all switches had the following parameter values. $B_i = 20, B_o = 60, p = 0.9$. We notice in the figure that the network performance is not affected by the presence of faults up to a certain threshold for the fault probability. After the threshold is reached, the CLR increases at a fast rate. In effect the network becomes less fault-tolerant. Notice that the threshold is almost equal for all network sizes.

Our results report network performance for a range of fault probability extending from 0 to 0.9. This is contrasted with the results of Plate and Tan in which the fault probability was only 1% [8]. This value is way below the threshold we report here. In other words, their results could not have discovered the presence of a threshold above which the network loses its fault tolerance ability.

We should mention that the horizontal axis in our figures represents *fault probability*. Other references reported the effect of faults in a slightly different manner [9, 10, 11]. In these references, the switch performance was reported versus the *number* of faults present. The two techniques give different views of the same phenomenon but the reader must be aware of this point. In fact, the figures in the cited references imply that the authors used a very low effective fault probability. For example, the highest value for fault probability in reference [11] is only .25. This is gain below the fault threshold reported in this paper.

Other authors employed a different technique for investigating fault tolerance.

Figure 10 shows the effect of faults in the network on the throughput. In this simulation, all switches had the following parameter values. $B_i = 20, B_o = 60, p = 0.9$.

6 Conclusions

In this paper, a fault-tolerant LN-ATM switch was proposed. The proposed switch combines the advantages of input and output queuing, and soft fault-tolerance. The extra hardware for fault-tolerance is actively used to increase the throughput and reduce cell loss probability and delay. An $N \times N$ LN-ATM switch is capable of delivering $\log_2 N + 1$ ATM cells to any output port in bit-serial fashion and without speeding up the network. The LN-ATM also supports data broadcast and hot-spot operations. A dynamic routing algorithm was proposed to route cells from input to output ports. The proposed switch achieved low cell loss probability (10^{-6}) at the output ports for a reasonable queue size ($B_o = 45$) and eliminated cell loss at the input ports. Furthermore, the LN-ATM witch was shown to have unity terminal reliability for high failure probability ($q = 50\%$).

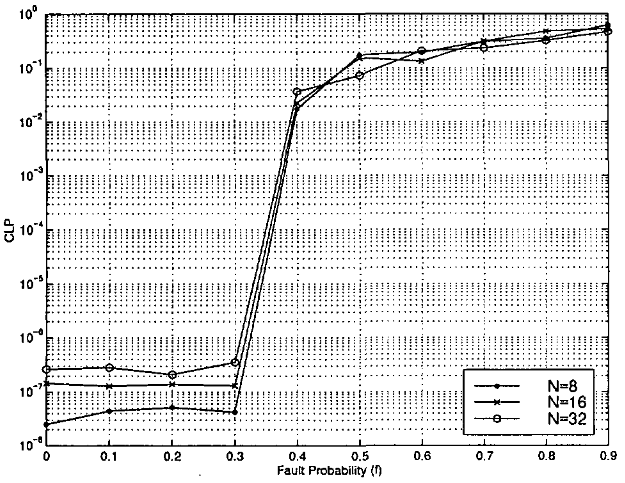


Figure 9: Effect of switching fabric faults on the CLP of switches with $N = 8, 16, 32$.

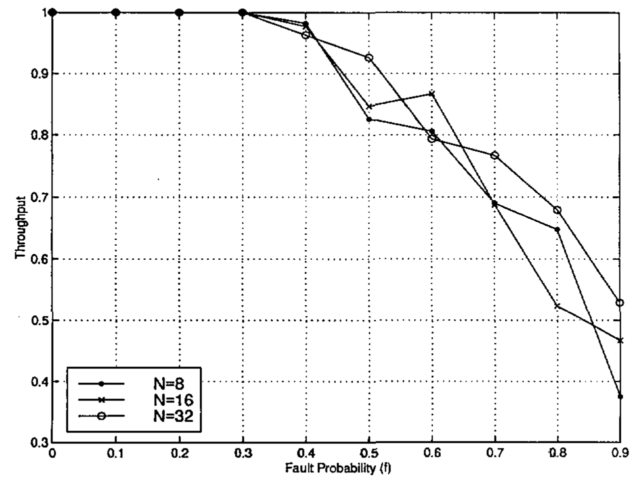


Figure 10: Effect of switching fabric faults on the throughput of switches with $N = 8, 16, 32$.

Acknowledgments

The authors would like to acknowledge King Fahd University of Petroleum and Minerals (Saudi Arabia) and Natural Sciences and Engineering Research Council (NSERC) of Canada for supporting this research. This work was also partially supported by contriubtions from PMC-Sierra Inc. of Vancouver and Micronet.

References

[1] J. Agrawal abd F. Yap, "Bodhi: A highly modular terabit ATM switch fabric architecture," *IEICE Trans. Commun.*, vol. E81-B, no. 2, pp. 182–193, Feb. 1998.

[2] M. Alimuddin, *Cost-Performance Trade-Offs in Large Scale Banyan-Based ATM Switches*, Ph.D. thesis, University of Brithish Columbia, 1997.

[3] S. Shiokawa and I. Sasase, "Input and output queueing two stage ATM switch with hot-spot route," *IEICE Trans. Commun.*, vol. E81-B, no. 2, pp. 194–200, Feb. 1998.

[4] R. Awdeh and H. Mouftah, "Survey of ATM switch architectures," *Computer Networks and ISDN Systems*, vol. 27, pp. 1567–1613, 1995.

[5] F. El-Guibaly and S. Agarwal, "Design and performance analysis of shift register-based ATM switch," in *1997 Proc. IEEE PacRim Conf. on Commun., Comp. and Sig. Process.*, 1997, pp. 70–73.

[6] S. Agarwal and F. El-Guibaly, "Design and performance analysis of shift register-based ATM switch," in *1997 Proc. IEEE PacRim Conf. on Commun., Comp. and Sig. Process.*, 1997, pp. 70–73.

[7] A. Sabaa, F. El-Guibaly, and D. Shpak, "Design and modeling of a nonblocking input-buffered ATM switch," *Can. J. Elec. and Computer Engg.*, vol. 22, no. 3, pp. 87–93, 1997.

[8] C. Plate and J. Tan, "Performance analysis of a fault-tolerance B-tree atm switch," in *21st IEEE Conf. on Local Computer Networks*, 1996, pp. 295–303.

[9] P. Tagle and N. Sharma, "Performance of fault tolerant ATM switches," *IEE Proc. commun.*, vol. 143, no. 5, pp. 317–324, Oct. 1996.

[10] C. Lo and C. Chiu, "A fault-tolerance architecture for atm networks," in *20th IEEE Conf. Local Computer Networks*, 1995, pp. 29–35.

[11] J.-F. Lin and S.-D. Wang, "A high performance fault-tolerant switching network for ATM," *IEICE Trans. Commun.*, vol. E78-B, no. 11, pp. 1518–1528, 1995.

[12] M. Abd-El-Barr, K. At-Tawil, and O. Abed, "Fault-tolerance and reliability analysis of multi-stage data manipulator networks," in *1995 Intl Conf. Dist. Computing*, 1995, pp. 275–280.

[13] S. Segkhoonthod and M. Sinclair, "Design of fault-tolerant ATM switch based on parallel architecture," *Electronic Letters*, vol. 33, no. 15, pp. 1289–1290, June 1997.

[14] F. Elguibaly, "Design and analysis of arbitration protocols," *IEEE Trans. Comput.*, vol. 38, no. 2, pp. 161–171, Feb. 1989.

[15] A. Itoh, "A fault-tolerant switching network for B-ISDN," *IEEE J. Selected Areas in Communications*, vol. 9, no. 8, pp. 1218–1226, 1991.

[16] N. Tzeng, P. Yew, and C. Zhu, "A fault-tolerant scheme for multistage interconnection networks," *12th International Symp. on Computer Architecture*, 1985, pp.368–375.

Issues On Gigabit Switching Using 3-Stage Clos Networks

Fotios K. Liotopoulos
Computer Technology Institute,
Akteou 11 & Poulopoulou, 11851 Thesseo, Athens, Greece.
Phone: +30-1-34.16.220, Fax: +30-1-34.16.700
E-mail: liotop@cti.gr

Keywords: Gigabit Switching, Broadband Switching, Clos Networks, ATM Switch Architecture.

Edited by: Mohsen Guizani

Received: March 4, 1999 **Revised:** August 9, 1999 **Accepted:** August 29, 1999

In this paper, we examine the scalability potential of 3-stage Clos networks for Gigabit ATM switching. We propose and evaluate an ATM switch architecture with a nonblocking switching capacity scalable to 160 Gbps. The switching fabric consists of a number of modular switching elements, interconnected in a 3-stage Clos network, with a total number of up to 1024 input ports and 1024 output ports (OC-3). Each switching element of the network consists of up to eight 4×4 switching modules, interconnected via a common bus. Internally, each module performs both input and output queuing. Cells are switched through a shared local bus. Fast arbitration logic is used to control the various busses of the network. Although there is shared resource contention, the proposed architecture can be nonblocking, if the internal bus operation is adequately fast. Simulation results show that nonblocking operation is feasible, with total internal buffering of as small as 18432 cells, achieving cell latencies of as low as 6 to 10 μsec. Given the appropriate Call Admission Control algorithm and an adequate number of middle-stage switches, the Clos network can also provide nonblocking operation at call setup.

1 Introduction

The proliferation of emerging multimedia technologies, supporting advance multimedia applications and services has set higher and more technologically challenging requirements for better multimedia networking support. Applications involving real-time audio and video transmission, multiple-party video-conferencing, video-on-demand, etc. require high switching capacities, high throughput, low latency, low or no blocking and a number of QoS guarantees.

Several of today's high performance switch implementations, used in multimedia application environments, including Cisco's LightStream 1010 and BPX-8600 and Fore Systems' Fore-Runner ASX-4000 ATM switches, do no exceed 60 Gbps of switching capacity. In a video-on-demand environment, it will soon not be uncommon to require service for tens of thousands of end-customers. For example, 50,000 MPEG-4 compressed video streams at 2 Mbps each, require a switching capacity of at least 100 Gbps. Such demands cannot be accommodated very easily, by today's state-of-the-art switch implementations.

In this paper[1], we present a scalable and modular architecture of a high-performance switching network, capable of achieving 160 Gbps throughput, with only a few microseconds end-to-end cell latency. This architecture is powerful enough to satisfy the switching capacity requirements of today's most demanding multimedia environments.

[1]Part of this research work has been presented in [16].

One design approach, adopted by several state-of-the-art switch architectures today, such as the Cisco's LightStream 1010 and 2020, is the *Shared Memory* approach [1, 12, 22, 23]. This approach, however, does not scale very well for high capacities and current implementations based on this approach rarely support more than 60 Gbps of switching capacity.

Another popular approach, implemented in IBM's PARIS and plaNET, NEC's ATOM switch and Fore Systems' ForeRunner ASX-4000 among others, involves routing cells from input to output through a *Shared Medium*, like a bus or a ring [1, 22, 23]. In these switches, the bus speed must be high enough in order to avoid blocking.

Orthogonally, all switch design approaches require a buffering strategy. Input buffering usually suffers from *head-of-the-line blocking*, while output buffering requires a mechanism able to deliver multiple cells to the same output during one cell-cycle [5, 23]. Internal buffers are often placed within the switching elements of a space division fabric. Such a strategy must be carefully evaluated as it may result to poor throughput and undesirable cell-latency variations. On the other hand, internal buffering can be used to isolate consecutive stages and improve on the parallel and independent operation of switching elements in a multistage switching network.

Our approach consists of interconnecting several high capacity switching elements (up to 5 Gbps each) into a multistage network, taking advantage of their distributed and concurrent operation. We propose a modular design of such a switching element, which allows it to scale efficiently to the required switching capacity of 5 Gbps. Up to 32 such switches can be interconnected per stage, thus achieving a total switching capacity of 160 Gbps. Our design uses internal buffering (both input and output) between stages and follows the shared bus paradigm in a hierarchical fashion to achieve the desired scalability.

We evaluate the proposed design with simulation results, which indicate the sizing conditions under which the internal operation of the switching network is nonblocking and the average cell-latency is optimized.

The rest of this paper is organized as follows. Section 2 provides an introduction to 3-stage Clos networks and their four nonblocking modes of operation. Section 3 describes the various parts of the proposed architecture, including the typical switching element, the switch module and the arbitration logic. Section 4 discusses issues related to call admission control (CAC) and ATM routing. Section 5 presents simulation results for the relative throughput and cell latency for various network configurations and sizes. Finally, Section 6 summarizes the contributions of this work and identifies new directions and future work.

2 Three-Stage Clos Networks

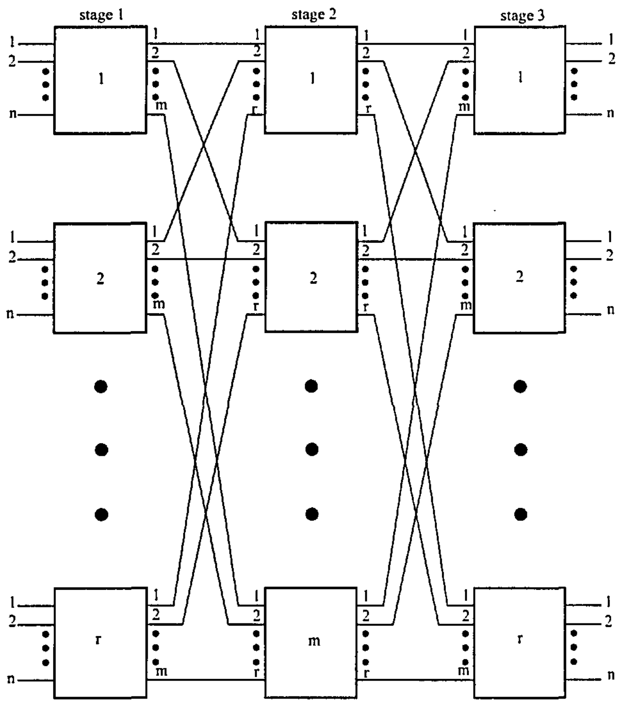


Figure 1: A symmetric three-stage Clos network.

Three-stage Clos networks [6] have been used for multiprocessor interconnection networks, as well as for data switching fabrics. For example, the Memphis switch of the IBM GF-11 parallel computer uses a symmetric three-stage Clos network to interconnect 576 processors [2]. A variant of the 3-stage Clos network has been designed for use in the High-Performance Switching System project at the Lawrence Livermore National Laboratory [25].

Clos networks can be used to efficiently implement low latency, high-bandwidth, connection-oriented ATM switching [7, 14]. Clos-type switching fabrics have also been found to be economically and technically attractive. Combined with their inherent fault-tolerant and multi-path routing properties, they pose as a very appealing choice for reliable broadband switching. Examples include the Fujitsu's FETEX-150, NEC's ATOM and Hitachi's hyper-distributed switching system.

A three-stage Clos network consists of three successive stages of switching elements which are interconnected by point-to-point links. In a symmetric three-stage network, all switching elements in a stage are uniform (see Figure 1). In the symmetric Clos network of Figure 1, there are r switches of size $n \times m$ in the first stage, m switches of size $r \times r$ in the second, and r switches of size $m \times n$ in the third. This network thus interconnects $n \cdot r$ input ports with $n \cdot r$ output ports.

Masson and Jordan introduced asymmetrical three-stage Clos networks as a generalization of the symmetric Clos network of Figure 1 [20]. These networks have r_1 switches of size $n_1 \times m$ in the first stage and r_2 switches of size $m \times n_2$ in the third; there are m switches in the middle stage each of size $r_1 \times r_2$. Hwang and Jajszczyk developed many routing strategies for these asymmetrical three-stage networks, and derived conditions under which these routing strategies enable nonblocking operation of the networks [10]. Koppelman considered more general asymmetrical Clos networks than those considered by Masson and Jordan, and Hwang and Jajszczyk. The networks considered by Koppelman may have nonuniform switches in each stage and arbitrary connection patterns between stages; he studied the rearrangeability of these networks [13]. Varma and Chalasani also designed connection-assignment algorithms for asymmetrical Clos networks under the rearrangeable mode of operation [24].

2.1 Generalized Clos Switching Networks

Figure 2 illustrates the general class of asymmetrical three-stage Clos networks. The set of switching elements in stage i of this network will be denoted by S_i, for $i = 1, 2, 3$. The number of switching elements in the three stages are f, m, and g, respectively. The interconnection pattern among the three stages is described below.

Stage 1 Switching element i has P_i input ports. It is connected to switching element j in the middle stage via a set of $R_{i,j}$ virtual channels. Each input port can accommodate u_i virtual channels.

Stage 3 Switching element k has Q_k output ports. It is connected to switching element j in the middle stage via a set of $T_{j,k}$ virtual channels. Each output port can accommodate v_k virtual channels.

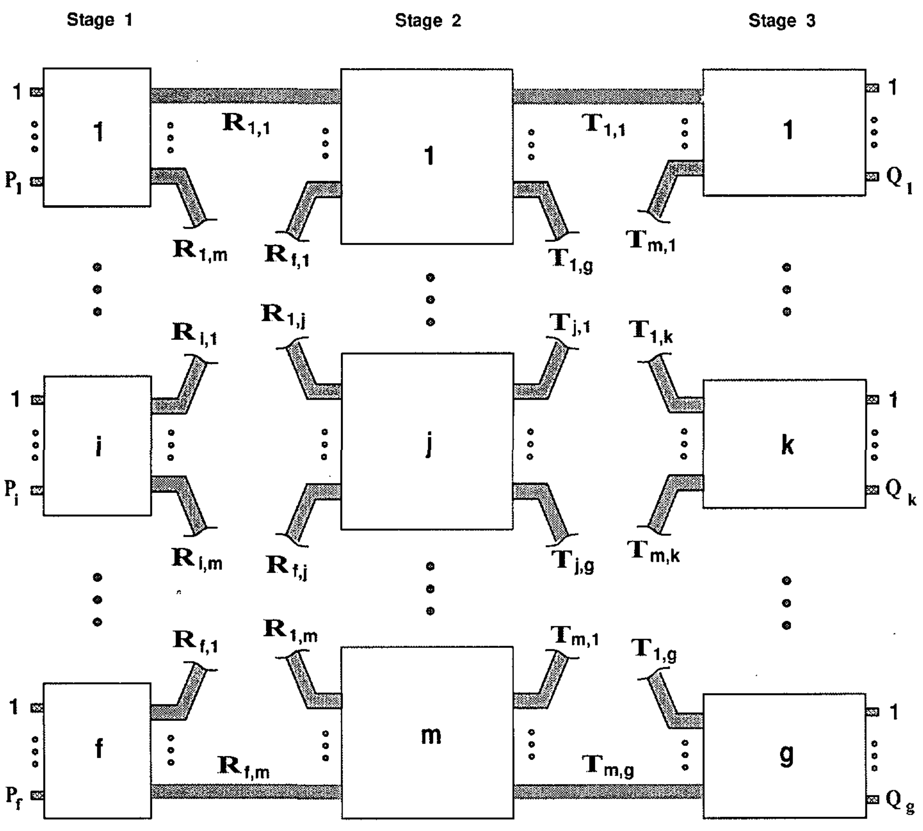


Figure 2: A general asymmetrical Clos network.

The total number of input virtual channels in the network (i.e. inputs to the first stage) is given by:

$$\mathcal{I} \stackrel{\text{def}}{=} \sum_{i=1}^{i=f} P_i \cdot u_i.$$

Similarly, the total number of output virtual channels in the network (i.e. outputs of the third stage) is:

$$\mathcal{O} \stackrel{\text{def}}{=} \sum_{k=1}^{k=g} Q_k \cdot v_k.$$

Nonblocking conditions have been derived for generalized asymmetrical 3-stage Clos networks, by Mellen & Turner, Chung & Ross, and Liotopoulos & Chalasani, both for the discrete as well as for the continuous bandwidth multirate environment.

Figure 3 shows an example of a $2 \times 4 \times 3$ asymmetrical 3-stage Clos network, consisting of 6 input ports and 12 output ports. Such a configuration can occur in the case of switch faults (e.g., when one switch from the first-stage is faulty and taken out). It can also be useful in cases where the input ports must have different capacities than the output ports. In the example of Figure 3, the input ports can have twice the capacity of the output ports.

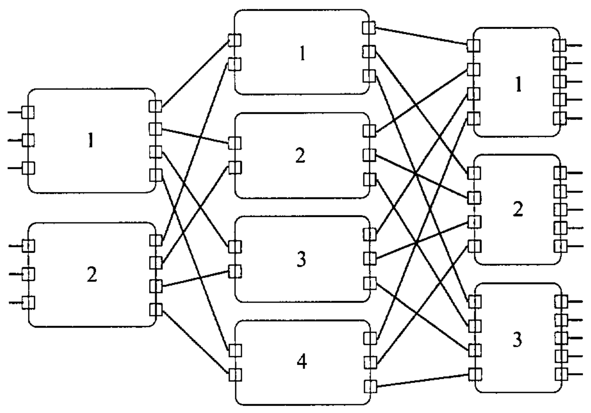


Figure 3: Example of an Asymmetrical 3-Stage Clos Network.

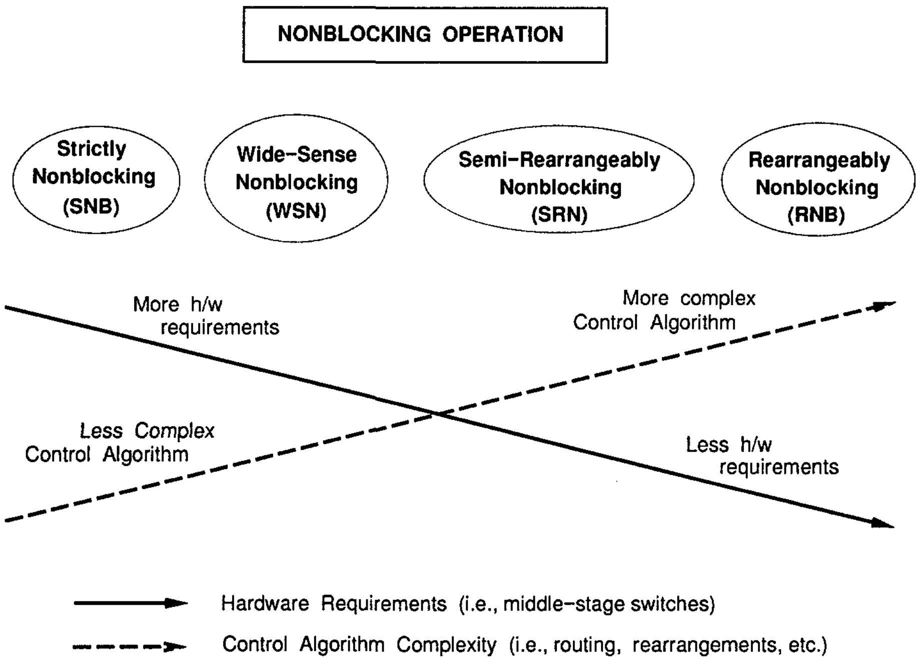

NONBLOCKING OPERATION
Strictly Nonblocking (SNB)
Wide-Sense Nonblocking (WSN)
Semi-Rearrangeably Nonblocking (SRN)
Rearrangeably Nonblocking (RNB)
More h/w requirements
More complex Control Algorithm
Less Complex Control Algorithm
Less h/w requirements
Hardware Requirements (i.e., middle-stage switches)
Control Algorithm Complexity (i.e., routing, rearrangements, etc.)

Figure 4: Tradeoffs for the various modes of nonblocking operation.

2.2 Nonblocking Switching Operation

Nonblocking operation of 3-stage Clos networks can be achieved in more than one ways, depending on how the incoming connections are distributed over the network's resources (Call Admission Control (CAC) policy). Each connection can be established by allocating network resources in a variety of ways, ranging from naive, arbitrary allocation to highly sophisticated and complex algorithms. Methods of higher complexity usually achieve better nonblocking performance at a cost slightly higher than that of the simple schemes [8].

In a *rearrangeably nonblocking* Clos network, a connection between any idle input port and any idle output port can always be realized by rearranging some of the existing connections if necessary. In a *strictly nonblocking* Clos network, an idle pair of ports can be connected *without* rearranging existing connections. Due to their more stringent requirements, nonblocking networks, in general, require more hardware resources than rearrangeable networks. For example, it has been shown that the symmetric three-stage Clos network of Figure 1 is rearrangeable if $m \geq n$, while the same network, for nonblocking operation requires at least $3n/2$ middle-stage switches [3].

Overall, in the context of CAC, we distinguish four modes of nonblocking operation for a 3-stage Clos network:

- Strictly Nonblocking (SNB).
- Wide-Sense Nonblocking (WSN).
- Semi-Rearrangeably Nonblocking (SRN).
- Rearrangeably Nonblocking (RNB).

Figure 4 shows the tradeoffs of the various modes, in terms of hardware requirements and control complexity. The SNB mode does not use any special CAC algorithm and therefore is very demanding in terms of the number of middle-stage switches needed for nonblocking switching, [19]. The WSN mode uses a special connection-placement (CAC) algorithm, which reduces its hardware requirements compared to the SNB mode. This mode is attractive mainly because of the simplicity of its control algorithms and the fact that it does not require rearrangements [11].

More sophisticated bandwidth and resource allocation schemes, involving connection placements and rearrangements, are employed in the SRN and RNB modes [4, 9, 21]. These schemes are even less demanding, in terms of hardware resources, for nonblocking operation. The RNB mode, though it requires the least amount of hardware resources, may be prohibitive as far as implementation is concerned. The main reason is that it may require a large number of rearrangements, which when performed may affect the quality of service of the established connections. On the other hand, in the SRN mode we attempt to restrict the number of rearrangements by rearranging only a limited number of connections per request, [17]. In this work, we specifically consider only those algorithms, which perform a small number of rearrangements and only after a disconnection request. Therefore, the SRN mode of operation appears as a viable compromise between resource requirements and implementation complexity.

3 A Clos Switch Architecture

Figure 5 shows a modular implementation of a typical switching element (i.e., a stage-switch) of the Clos network. It consists of two modules in cascade, which are interconnected via a shared bus.

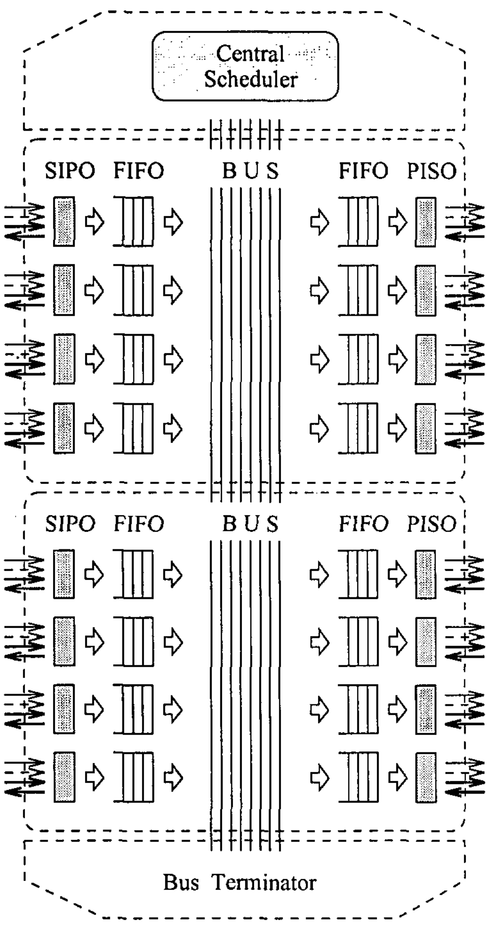


Figure 5: Cascaded Modules forming a Stage-Switch.

Each module has four serial input ports and four serial output ports (e.g., ATM ports operating at 155 Mbps (OC-3), also referred to as Synchronous Transport Signal level 3 (STS-3c)).

The serial data stream at each input port is transformed into parallel 32-bit words by a Serial-In-Parallel-Out (SIPO) shift register. These words are subsequently stored in a (local) input FIFO queue (`LiQ`). Cells can be switched from any input FIFO to any output FIFO within the same module or across different modules of the same switch.

A central scheduler performs control and bus arbitration, over the entire switch, and transfers one cell (14 32-bit words) at a time from an input FIFO (`LiQ`) to an output FIFO (`LoQ`), assuming that the former has at least one cell to send and the latter has adequate free buffer space to accommodate it.

Cells from the output FIFOs are then transformed into a serial stream by a Parallel-In-Serial-Out (PISO) shift register, in order to be switched to the next stage via an internal link or an output port. It is often desirable for internal links to have higher capacity than the input or output ports. This is usually implemented with wider data paths, or higher transfer rates.

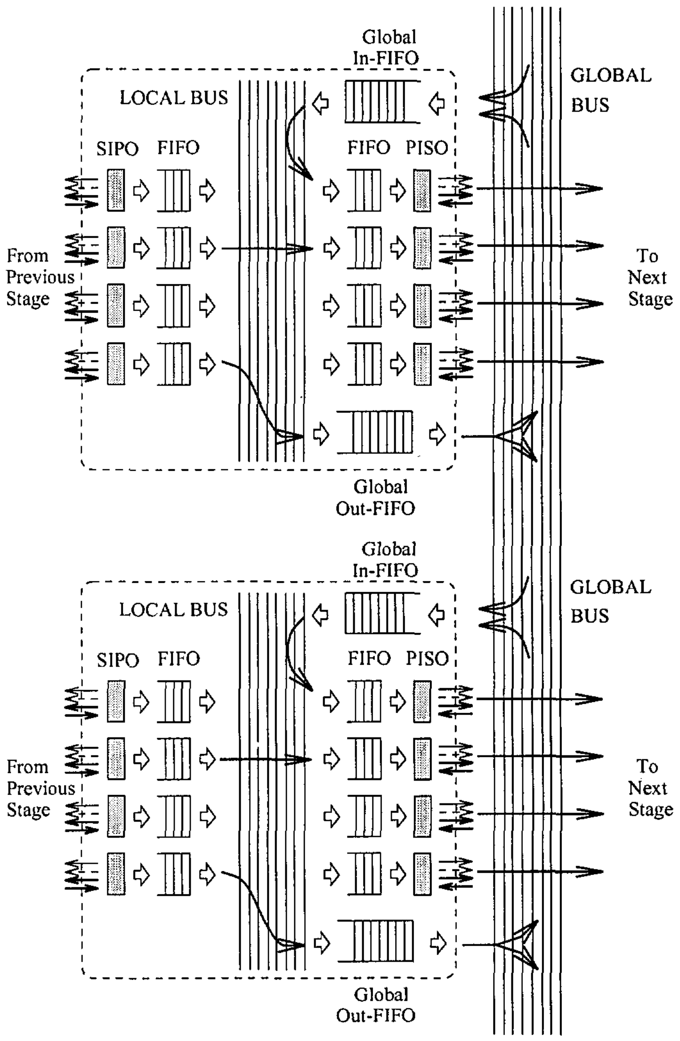


Figure 7: The proposed Module Architecture.

Figure 6 shows how two modules belonging to switches in consecutive stages are interconnected via an internal link. In this Figure, a cell originating from input port X1 and destined to output port Y2 of the next stage switch is first routed to output port Y1 of the first module. Then, it is transferred through the internal link, which connects the two modules, to input port X2 of the second module. Finally, the cell is switched to its destination, Y2, at the second module.

However, the above design approach is not very attractive for designing high capacity switches, since it cannot scale efficiently to more than 2-3 modules per switch. The main problem lies on the shared bus, which has: i) electrical load limitations, and ii) speed limitations.

To overcome this scaling problem, we propose an enhanced module architecture, described next.

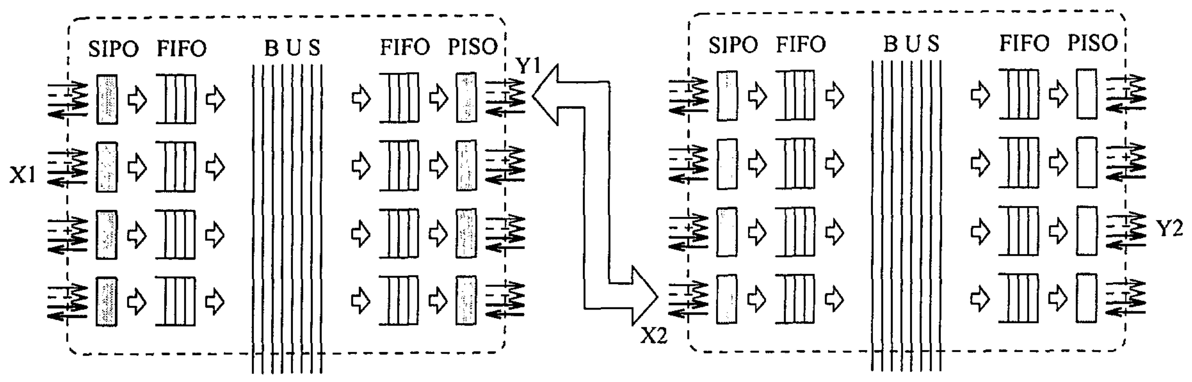


Figure 6: Inter-stage Module Communication.

3.1 An Enhanced Module Architecture

A more scalable approach to the design of the basic switch module is given in Figure 7.

In this design, we break down the single shared bus into a two-level hierarchy of shared busses, one local to the module (`LBus`) and one global for the entire switching element (`GBus`). Modules of the same switch can communicate with each other, via the `GBus`, by means of a pair of (global) input and output FIFOs (`GiQ`, `GoQ`).

A cell destined to an output port of the same module is transferred directly to the corresponding local FIFO `LoQ`, via the local bus, `LBus`. If the destination of the cell is an output port of a different module, within the same switch, then the cell is first transferred to the `GoQ` FIFO, and through the `GBus`, to the `GiQ` of the target module. At the remote module, the cell is transferred from the `GiQ` to the appropriate `LoQ`, before it exits the current switching element and moves to the next stage.

With this design approach, each module contributes only one input load and one output load to the total load of the `GBus`. Therefore, this design can scale to 8 or even 16 modules per switching element. Given that each module has a switching capacity of $4 \cdot 155 = 622$ Mbps, the switching element can scale up to 10 Gbps. A 3-stage Clos network consisting of such switching elements can therefore achieve up to 160 Gbps of strictly nonblocking switching capacity (in a $32 \times 64 \times 32$ configuration), or up to 320 Gbps of rearrangeably nonblocking switching capacity (in a $64 \times 64 \times 64$ configuration).

With respect to the `LBus`, arbitration is performed among the four `LiQ` and the `GiQ`, giving output to the four `LoQ` and the `GoQ`. Therefore, the `LBus` is electrically loaded only by 5 inputs and 5 outputs, which is well below its limitation, but kept at this level for overall performance reasons.

3.2 Bus Arbitration and Control

All busses of a typical switching element, either local or global, require fast control and arbitration logic, in order to carry out cell transfers more efficiently. The proposed Bus-Arbiter is depicted in Figure 8.

The proposed design for the bus arbitration logic ensures efficient data transfers, and high cell throughput. Idle bus cycles are minimized, since the arbiter examines and grants bus access to only those FIFOs that can truly send data (i.e., they have at least one cell to send and its destination FIFO has adequate free space to accept it).

A high level algorithm describing the operation and functionality of the bus arbiter, is given below:

1. The Arbiter queries all modules for the status of their `GiQ`.
2. Each module responds by setting its corresponding status line in the control bus.
3. The Arbiter queries all modules for bus requests.
4. Each module first determines the destination module of the top cell in its `GoQ` and the availability of this destination module's `GiQ`. if the destination module is available then the current module sets a valid bus request.
5. The Arbiter uses a Rotating Priority scheme to select a module to grant the bus to and outputs its ID number to the control bus.
6. The selected module takes control of the data bus and initiates the data transfer by first placing the destination module's address on the address bus.
7. The target module realizes that it is the destination module and initiates a read cycle.
8. The source module initiates the data transfer of an entire cell (14 32-bit words).
9. During the transfer of the last data word, the source module notifies the Arbiter of the completion of the transfer and releases control of the bus to the Arbiter.
10. Repeat the cycle from Step 1.

The above bus arbitration logic requires no more than 28 control bus signals for an 8-module switch, and no more than 18 control bus signals for a 4-module switch.

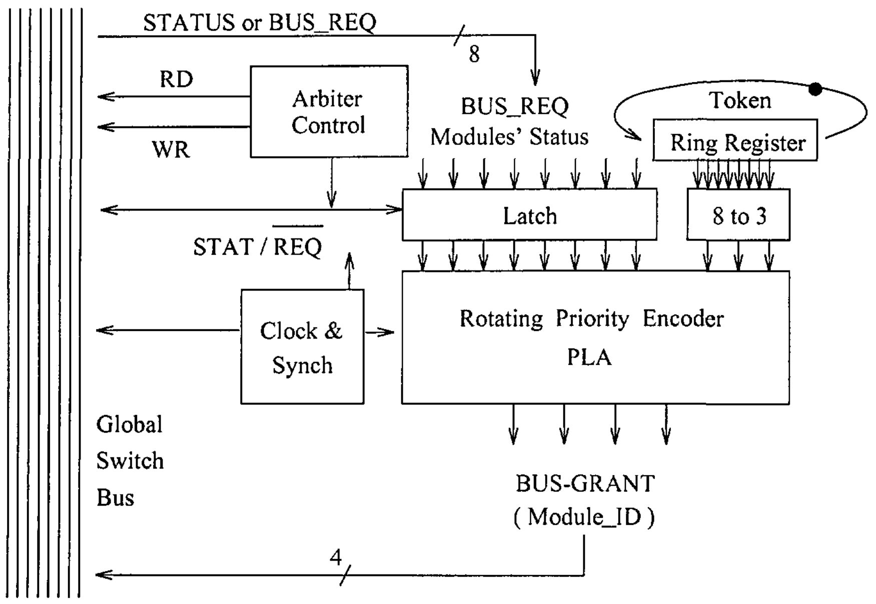


Figure 8: The proposed Bus-Arbiter architecture.

Currently, priority arbitration is done in a Round-Robin fashion, but other popular schemes, such as Fair-Queuing, are under investigation.

4 Call Admission Control (CAC) and Routing

Data stream transport through the switching network is achieved by means of Connection Oriented Network Services (CONS). CONS require the establishment of a connection between the origin and destination before transmitting the data. The connection is established by allocating bandwidth on a path of physical links through intermediate nodes of the network. Once established, all data travels over this same path in the network.

Control signaling is used to establish and take down a connection dynamically (Switched Virtual Circuits (SVC)).

For the proposed switch architecture, we can apply a variety of Call Admission Control (CAC) algorithms for SNB, WSN, SRN, or RNB operation, proposed by numerous researchers for generalized 3-stage Clos switching networks in the multirate environment. Next, we show an example of a semi-rearrangeably nonblocking (SRN) switching algorithm, proposed for Call Admission Control by Liotopoulos & Chalasani.

4.1 A CAC algorithm

This control algorithm performs Call Admission Control (CAC) by means of balancing the available paths of the switching matrix (Available Capacity Routing (ACR)). This is achieved by appropriately distributing incoming calls over all middle-stage switches, and performing at most one rearrangement per disconnection. The connection and disconnection rules for this routing algorithm are presented below.

Rules for establishing a new connection. Assume that $C(i,j,\star)$ denotes the number of connections routed through the middle-stage switch j from the first-stage switch i to all switches in the third stage. Similarly, $C(\star,j,k)$ represents the number of connections routed through the middle-stage switch j from all switches in the first stage to the third-stage switch k. In other words, $C(\star,j,k) = \sum_{i=1}^{i=f} C(i,j,k)$ and $C(i,j,\star) = \sum_{k=1}^{k=g} C(i,j,k)$.

Let the function $V(i,j,k)$ be defined as follows:

$$V(i,j,k) \stackrel{\text{def}}{=} \min\{(R_{ij} - C(i,j,\star)), (T_{jk} - C(\star,j,k))\},$$

$$(i,j,k) \in S_1 \times S_2 \times S_3.$$

For a given i and k, $V(i,j,k)$ represents the maximum number of *available* virtual paths in the network for the connection from i to k through middle-stage switch j. A middle stage switch to realize a connection from I to K is selected using the following rule.

Connection Rule
To realize a connection from I to K, select a middle-stage switch J, such that

$$V(I,J,K) = \max_p V(I,p,K)$$

The above connection rule distributes all the connections in such a way that the available virtual paths between a pair of first and third stage switches are uniformly distributed, to a large extent, over all the middle-stage switches.

Rules for disconnecting an existing connection. Suppose that a connection $[I, J, K]$ (that is, a connection from I to K through the middle-stage switch J) is to be disconnected. This is accomplished using the following disconnection rule.

```
Disconnection Rule
1 To disconnect $[I,J,K]$, first find a middle stage switch $J_m$ such that $V(I,J_m,K) = \min_p V(I,p,K)$.
2 if $V(I,J,K) == V(I,J_m,K)$ disconnect $[I,J,K]$.
3 else disconnect $[I,J,K]$ and reroute a connection from $I$ to $K$ through $J_m$ to use the middle-stage switch $J$. In other words, a connection of the form $[I, J_m, K]$ is rearranged so that it becomes a connection of the form $[I, J, K]$.
```

Observe that disconnection of $[I, J, K]$ will rearrange an existing connection only if $V(I, J, K)$ is greater than $V(I, J_m, K)$ by at least one; such a rearrangement ensures that the available virtual paths between the first-stage switch I and the third-stage switch K are divided approximately equally among all middle-stage switches.

4.2 ATM Cell Structure

Asynchronous Transfer Mode (ATM) switching is used to transport data from source to destination. Each ATM cell consists of the standard 5-octet header and 48-octet payload. Figure 9 depicts the format and parallel (32-bit wide) transmission of an ATM cell.

The cell header contains information, such as the cell's Virtual Path Identifier (VPI, 16 bits) and Virtual Channel Identifier (VCI, 8 bits), Generic Flow Control information (GFC, 4 bits), Payload Type (PT, 3 bits), Call Loss Priority (CLP, 1 bit), Header Error Check (HEC, 8 bits) and Payload Error Check (PEC, 24 bits). The PEC field is not part of the standard ATM cell format, but we have included it for enhanced reliability and in order for each cell to begin at a 32-bit word boundary.

From the above, we conclude that each cell is transmitted as 14 32-bit words and contains 48 octets of payload. Therefore, the protocol efficiency is $(48 \times 8)/(14 \times 32) = 85.714\%$.

Although the proposed ATM cell format is a slight variation of the standard, it is proposed because it provides a high efficiency routing protocol for OSI layer-3 routing and does not necessitate additional protocol encapsulation for this purpose.

In the proposed cell format, the VPI field contains all the routing information the cell requires to reach its destination output port at the third-stage of the switching network. The first 6 bits of this field ($\text{VPI}_{0:5}$) denote the selected middle-stage switch the cell is chosen to go through. This essentially identifies the output port number, which the specific cell should be switched to, inside the first-stage switch. The next 5 bits of the VPI field ($\text{VPI}_{6:10}$) denote the destination third-stage switch, or equivalently, the output port number of the middle-stage switch. Finally, the last 5 bits of the VPI field ($\text{VPI}_{11:15}$) denote the output port number of the third-stage switch.

The VCI field contains a virtual circuit ID number, which relatively identifies a virtual circuit (or service) within its corresponding ATM virtual path. Thus, the Absolute Identification of a Service (ASI) is derived from the concatenation of VPI and $\text{VCI}_{6:15}$, that is an 18-bit identifier.

5 Simulation Results

In order to evaluate the proposed architecture, we have developed a special purpose simulator, which is presented in [15]. In the following discussion, an $N \times M$ switch refers to a switch with N input ports and M output ports. Also, the variables LF and GF refer to the sizes of the local (`LiQ`, `LoQ`) and global (`GiQ`, `GoQ`) FIFOs, respectively.

The input traffic we applied consisted of uniformly distributed call requests over all output ports. Call lengths were exponentially distributed. Call interarrival times were assumed to be zero, in order to stress the switching fabric.

In this section, we present simulator results for the relative throughput[2] of the switch and for the average cell latency.

Figure 10 shows the overall relative throughput (a) and average cell latency (b), as a function of the bus speedup factor. A speedup factor of 16 indicates that the bus transfers cells 16 times faster than the input ports. For OC-3 input ports, this implies bus transfers at 310 MBytes per second, or 78 MWords per second, (assuming 32-bit wide busses).

All busses, both local and global, are assumed to have the same speedup factor. The behavior of the switch with different speedup factors for local and global busses are also under evaluation but results are not presented in this paper.

For various values of LF and GF we observed no effect in the overall relative throughput. No internal blocking was observed for bus speedups ≥ 16. For the same combination of FIFO sizes, we observed longer cell delays for larger internal buffering, but only for cases with internal blocking.

Figure 11 shows similar graphs, as in Figure 10, comparing two switch configurations with different number of output ports. The 256×256 configuration exhibits higher throughput and less cell delay, due to less path interdependencies, resulting in less resource contention.

Figure 12 compares two strictly nonblocking, $(16 \times 32 \times 16)$ switch configurations with capacities of 40 Gbps (256 OC-3 input ports) and 80 Gbps (512 OC-3 input ports), respectively. Graph (a) shows that in both cases, 100% rel-

[2]*Relative Throughput* is defined as the percent of the offered load, which is actually delivered by the switch. The relative throughput can be less than 100% due to internal blocking.

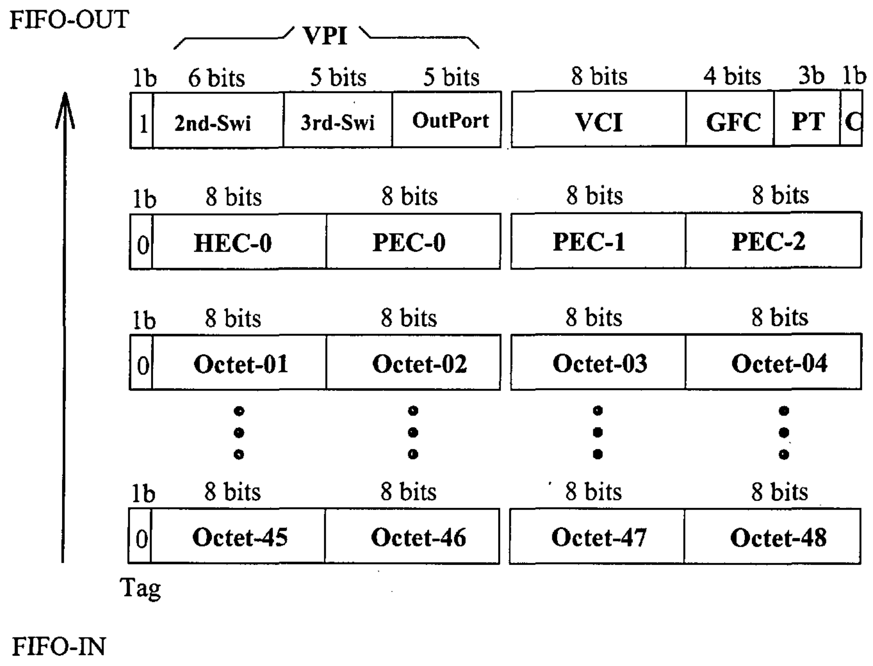


Figure 9: ATM cell transmission and format.

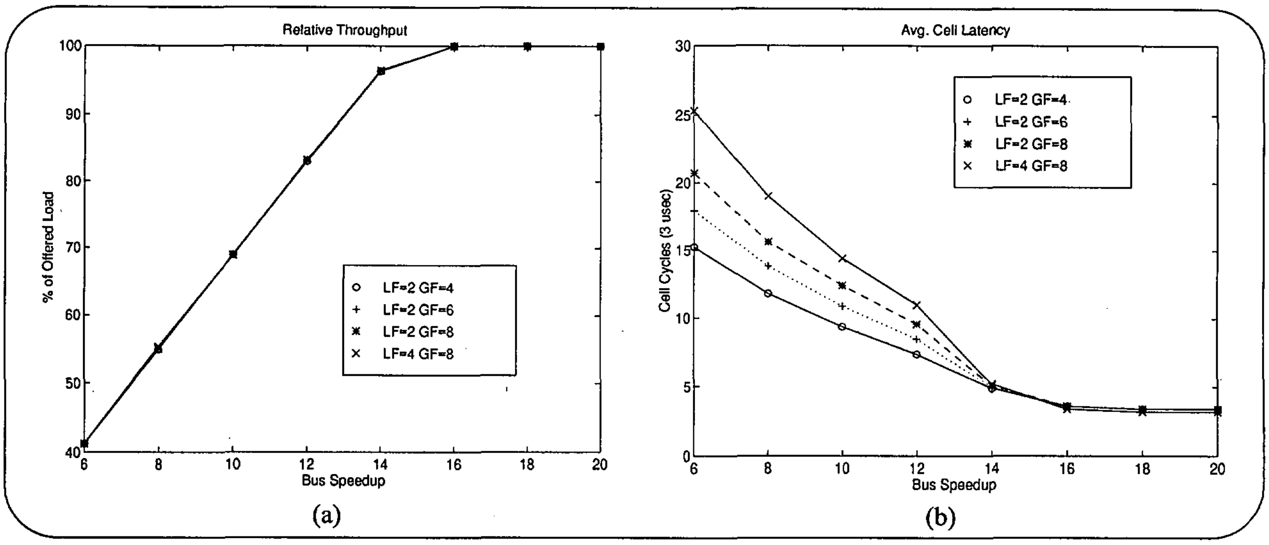


Figure 10: Throughput and Latency for various FIFO size configurations.

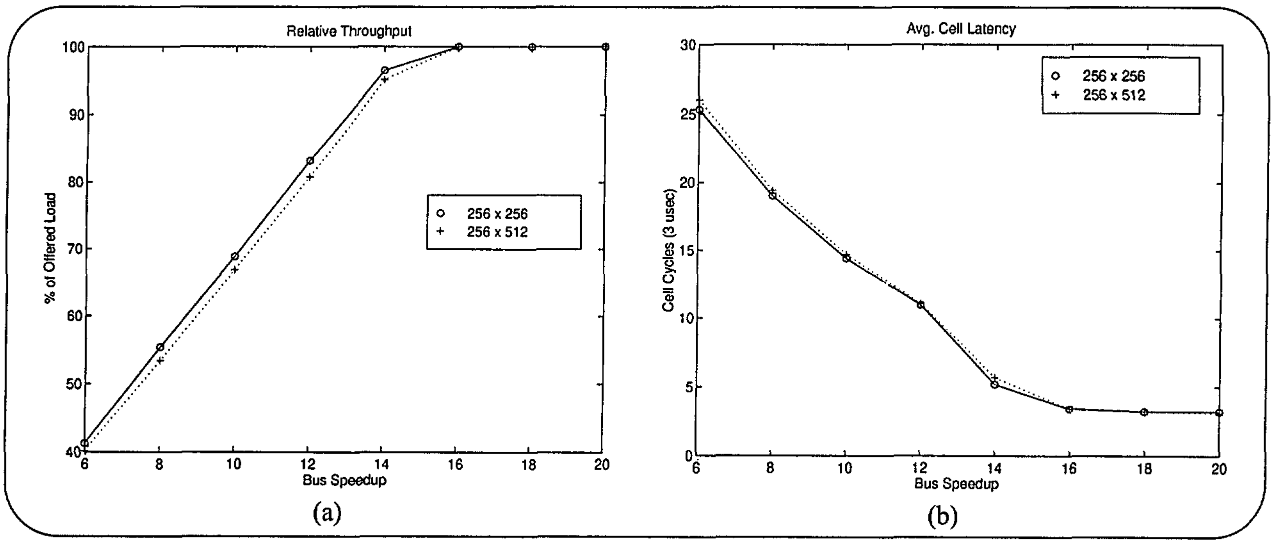


Figure 11: Throughput and Latency for a 256 × 256 and a 256 × 512 switch.

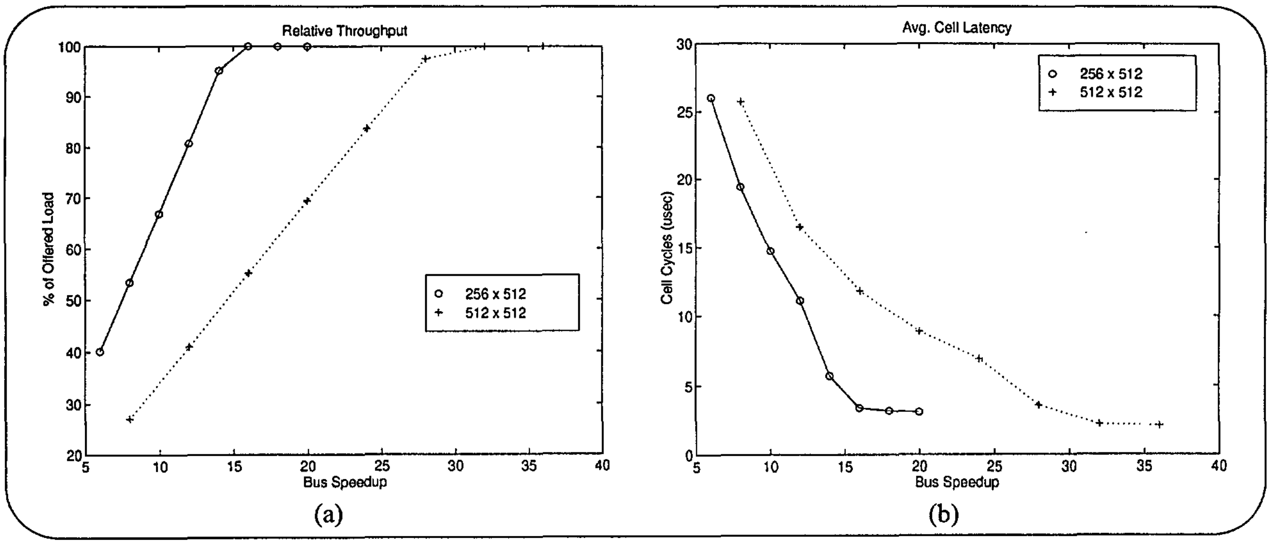


Figure 12: Throughput and Latency for a 256 × 512 and a 512 × 512 switch.

ative throughput is achievable. Graph (b) shows that the average cell latency can be under 10 μsec. (3 cell cycles), if no internal blocking occurs.

Figure 13 shows cell-latency distribution curves for two different capacities of a ($16 \times 32 \times 16$) switch ; a 40 Gbps and an 80 Gbps switch. In each case, the effect of the speedup factor is observed. As expected, the higher the speedup factor is, the more "steep" the distribution curves become. Steeper latency distribution curves indicate smaller latency variance.

6 Conclusions and Future Work

This paper advocates the use of 3-stage Clos networks, combined with ATM switching, for the implementation of large broadband switching networks for multimedia networking support.

Clos switches, controlled by intelligent CAC algorithms, offer high capacity switching and high switch utilization, while keeping the blocking probability very low, or near zero.

Building a high-capacity switch from a large number of switching modules takes advantage of the parallelism and resource decoupling exploited by the individual modules. Scalability is achieved by breaking down the shared bus bottleneck to a hierarchy of busses. This approach has the disadvantage that it increases the variance of cell latency. However, since all cells belonging to the same call follow the same physical path from source to destination, the latency variance among them is expected to be less than the overall latency variance and generally small.

Simulation results indicate that larger buffers do not improve the overall throughput significantly, while on the other hand they increase the average cell latency, due to the head-of-line blocking, experienced with input queuing. Bus speed and internal link speed are critical for the nonblocking operation of the entire switching network. We have shown through simulation that, for the proposed switching capacities, busses and internal links can be constructed and realisticly run at rates high enough to achieve 100% relative throughput and under 10 μsec. cell latencies with total internal buffering as small as 18432 cells.

More work needs to be done in incorporating traffic management and QoS features to the proposed switch architecture. We plan to develop an analytical model to evaluate the proposed architecture, and also implement a prototype module in VLSI. We also plan to evaluate the behavior and performance of this architecture under various types of ATM traffic (CBR, VBR, rt-VBR, bursty, etc.). Finally, we need to explore the multicasting capabilities of the proposed switch architecture, as well as potential extensions to IP-over-ATM switching.

References

[1] H. Ahmadi and W.E. Denzel, “A Survey of Modern High-Performance Switching Techniques,” *IEEE J. Selected Areas Commun.*, vol.7, no.7, pp.1091-1103, Sep.1989.

[2] J. Beetem, M. Denneau and D. Weingarten, “The GF11 supercomputer,” *Proc. of the 12th Annual Int. Symposium on Computer Architecture*, pp.108-115, June 1985.

[3] V.E. Beneš, *Mathematical Theory of Connecting Networks and Telephone Traffic*, Academic Press, New York, 1965.

[4] S. Chalasani and A. Varma, “Efficient time-slot assignment algorithms for SS/TDMA Systems with variable bandwidth beams,” *IEEE Trans Commun.*, vol. 42, no. 2/3/4, part II of 3 parts, pp. 1359-1370, Feb/Mar/Apr 1994.

[5] T.M. Chen and S.S. Liu, *ATM Switching Systems*, Artech House Inc., ch. 5-10, pp. 81-233, 1995.

[6] C. Clos, “A study of non-blocking switching networks,” *Bell Syst. Tech.J.*, pp.406-424, March 1953.

[7] P. Coppo, M. D’Ambrosio and R. Melen, “Optimal Cost/Performance Design of ATM Switches,” *IEEE Transactions on Networking*, vol. 1, no. 5, Oct. 1993, pp. 566–575.

[8] P. Feldman, J. Friedman and N. Pippenger, “Non-Blocking Networks,” *Proc. STOC 1986*, pp. 247-254, May 1986.

[9] F.K. Hwang, “Control Algorithms for Rearrangeable Clos Networks,” *IEEE Trans. Commun., COM-31(8)*, pp. 952-954, 1983.

[10] F.K. Hwang and A. Jajszczyk, “On nonblocking multiconnection networks,” *IEEE Trans. Commun. COM-34*, pp.1038-1041, 1986.

[11] A. Jajszczyk and G. Jekel, “A New Concept - Repackable Networks,” *IEEE Trans. Commun.*, vol. 41, no. 8, pp. 1232-1237, August 1993.

[12] S.K. Konakondla, “Design, Simulation and Performance Analysis of a Shared Memory ATM Switch for B-ISDN networks,” tech. rep., Cleveland State University, Cleveland, OH 1994.

[13] D.M. Koppelman, “A self-routing permutation network, a classification of all three-stage networks, some other permutation network designs and performance bounds,” *Ph.D. Thesis*, Rensselaer Polytechnic Institute, 1988.

[14] J.-Y. Le Boudec, “The Asynchronous Transfer Mode: a tutorial,” *Computer Networks and ISDN Systems*, 24 (1992), pp.279-309.

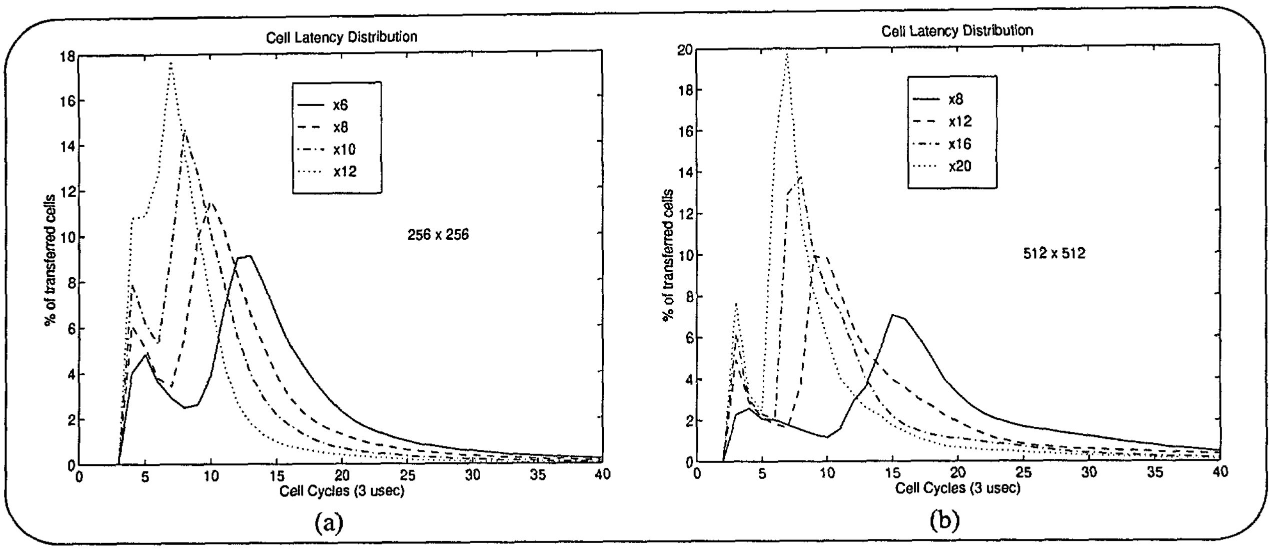


Figure 13: Latency distribution for a 256 × 512 and a 512 × 512 switch.

[15] F.K. Liotopoulos, "Simulator-Assisted Performance Evaluation of a High-Capacity 3-Stage Clos Network, used for Broadband ATM Switching," *Proc. 3rd IEEE Symposium on the Planning and Design of Broadband Networks*, Quebec, Canada, October 1998.

[16] F.K. Liotopoulos, "A Modular, 160 Gbps ATM Switch Architecture for Multimedia Networking Support, based on a 3-Stage Clos Network," *Proc. 16th Int'l Teletraffic Congress*, Edinburgh, UK, June 1999.

[17] F.K. Liotopoulos and S. Chalasani, "Semi-Rearrangeably Nonblocking Operation of Clos Networks in the Multirate Environment," *IEEE Trans on Networking*, vol.4, no.2, pp.281-291, April 1996.

[18] F.K. Liotopoulos and S. Chalasani, "Rearrangeably Nonblocking Operation of 3-Stage Clos Networks in the Multirate Environment," submitted to *IEEE Trans. on Communications*.

[19] F.K. Liotopoulos and S. Chalasani, "Strictly Nonblocking Operation of 3-Stage Clos Networks," *Performance Modelling and Evaluation of ATM Networks*, vol.2, edited by D.D.Kouvatsos, published by Chapman & Hall, London 1996.

[20] G.M. Masson and B.W. Jordan, "Realization of multiple connection assignment with asymmetrical multistage connection networks," *Networks 2(3)*, 1972.

[21] S. Ohta, "A simple control algorithm for rearrangeable switching networks with time division multiplexed links," *IEEE JSAC*, vol. SAC-5, no. 8, pp. 1302-1308, Oct. 1987.

[22] R.O. Onvural, *ATM networks: Performance Issues*, Artech House, ch. 7, pp. 207-252, Boston 1994.

[23] T.G. Robertazzi, *Performance evaluation of high speed switching fabrics and networks: ATM, B-ISDN and MAN technology*, IEEE Press, New York 1993.

[24] A. Varma and S. Chalasani, "Asymmetrical Multiconnection Three-Stage Clos Networks," *Networks*, vol. 23, pp. 427-439, 1993.

[25] A. Varma, V. Sahai and R. Bryant, "Performance Evaluation of a High-Speed Switching System Based on the Fibre Channel Standard," *Proc. of the 2nd Int. Symposium on High Performance Distributed Computing*, 1993, pp. 144-151.

Performance Analysis Of Pipelined Multistage Interconnection Networks

R. Venkatesan, Yaser El-Sayed, Rajagopalan Thuppal and H. Sivakumar
Faculty of Engineering and Applied Science
Memorial University of Newfoundland, St. John's, NF
Canada, A1B 3X5
R. Thuppal is currently working with Fore Systems, Ottawa, Canada.
H. Sivakumar is currently with University of Illinois, Chicago, also is working for IBM, Canada.
E-mail: {venky,yaser}@engr.mun.ca, rthuppal@fore.com, sharinat@eecs.uic.edu

Keywords: Broadband communications, Discrete simulation, Monte-Carlo simulation, Performance analysis.

Edited by: Mohsen Guizani

Received: January 17, 1999 **Revised:** June 16, 1999 **Accepted:** September 2, 1999

Pipelined multistage interconnection networks have been shown to exhibit excellent performance with low buffer requirements to combat internal and output blocking. Their characteristics are suitable for switch fabrics in broadband communication networks. Furthermore, their implementation is feasible using current VLSI technologies. In this paper we present the performance analyses of multistage interconnection networks and pipelined multistage interconnection networks. We also present the effects of the buffer size on the performance of pipelined multistage interconnection networks based on banyan as well as balanced gamma networks by decoupling the issues of output queueing from both internal and output blocking. Performance analyses with finite input and finite output buffers are investigated under traffic patterns created using mixes of uniform random traffic and bursty traffic; 20% of the total cells generated are assigned high priority in each simulation experiment. These traffic patterns are closer to the realistic traffic in broadband communication networks than permutation traffic or pure uniform random traffic. The results obtained demonstrate that output queuing is an architecture independent problem.

1 Introduction

Switching systems in broadband communication systems are network elements that support functions that include data transport, connection admission control, and network management. Apart from operating at very high speeds, switches deployed in such fast transmission environments should be able to handle the flood of data arriving at their input ports. The results in this paper are applicable to switches in broadband communication systems regardless of the protocol used for the following two reasons. Firstly, in the Internet protocol (IP) or fast Ethernet where packets potentially have variable sizes, the packets can be broken down into equal sized packets similar to the cells in the ATM protocol. Secondly, the switching architecture we are proposing does not adopt any internal buffering strategy, therefore; any traffic can be switched through this architecture as long as the packets used have the same packet size.

Over the past few years, several architectures (Tobagi 1990) have been proposed for ATM switching and many of these are based on multistage interconnection networks (MINs). MINs have many desirable features such as self-routing, low complexity, modularity and suitability for VLSI implementation. MINs are internally blocking with cells contending over the same internal links, even if they are destined to different output ports. Furthermore, when multiple cells request the same output port at the same time, output blocking results. Blocking severely degrades the throughput performance of MINs. To improve the throughput performance, many solutions have been proposed. The effects of internal blocking can be minimized by providing internal buffers or by providing input buffers and incorporating backpressure mechanism. The former solution is expensive, plus it leads to out of sequence receipt of cells in multipath networks, and so the latter solution is preferred in broadband communication networks. Output blocking can be contained by choosing a MIN architecture capable of accepting multiple cells at each output line at each cycle. Such MINs along with input buffers and backpressure mechanism would be capable of overcoming the drop in performance due to internal blocking as well as output blocking. Input buffered switch architectures suffer from Head Of Line (HOL) blocking, in which the temporarily untransmissible cell at the head of the buffers impedes the transmission of cells behind it, and reduce the effective switch throughput. However, if the unbuffered MIN possesses very high throughput (for example, better than 99%), then HOL blocking would not be significant most of the times. The MIN architectures reported in this paper come under this category.

When the incoming traffic is more bursty, the output buffer needs to be larger to prevent buffer overflow. This problem, which we refer to as output queuing problem, is a classical queuing theory issue. When we consider switch fabrics that are feasible to implement, we note that switches which display throughput performances closer to the ideal switch suffer more severely from the output queuing problem, as more cells arrive at the output buffers in each cycle. However, if we consider only those switch fabrics which possess very high throughputs, that is only those which route nearly every incoming cell to the output buffer without loss or delay, we can conclude that the output queuing problem is independent of the architecture employed in constructing these switch fabrics.

As we will discuss in this paper, it is possible to design switch fabrics which overcome the internal blocking and output blocking to a large extent. However, the loss of performance in switch fabrics due to output queuing problem is considerably more pronounced than other performance degradation factors such as internal or output blocking, especially if the incoming traffic is bursty. We could incorporate backpressure mechanism to prevent cell loss due to output buffer overflow. However, this solution causes the problem to be transferred to the input buffers making it worse, as the HOL effect causes further performance degradation. If no backpressure mechanism is provided, the switch operates in queue loss mode, where cells are dropped when overflow occurs. Thus, the combination of input and output buffering is essential in a switch. We demonstrate in this paper that the input buffer size can be chosen so that internal blocking and output blocking problems are overcome, and the output buffer size can be selected to limit the effects of the output queuing.

Many existing switch fabrics have only a few (8 or 16 or 32) input/output lines, and they employ huge amounts of internal and input buffering, apart from having large output buffers. Buffers are expensive not only in terms of the hardware required, but one has to consider the amount of delay and delay variance they cause. Furthermore, the switch architecture would be unscalable if large buffers are employed. That is, we will not be able to build switches with more input/output lines using the same architectures and larger buffers, as the HOL blocking would become prohibitively significant. We show in this paper that, while it is not possible to reduce the size of the output buffer, using proper selection of the basic MIN we can eliminate internal buffers and minimize the amount of input buffers required. We believe that the basic structure of the switch fabric should be a multipath MIN, and the output buffers at each output line should be capable of accepting all cells which reach the output stage of the MIN. This ensures that there is very low internal blocking and virtually no output blocking. A major reason to avoid internal buffers is that the actual data transfer can be effected in a circuit switched fashion. This can be accomplished by adopting a two-phase routing algorithm as in any switch fabric employing backpressure mechanism. In the first phase, termed reservation phase, only a few of the header bits (we term these pilots) of all cells at the head of the input buffer at each input line are transmitted through the network. The SEs are set in appropriate configurations based on the acknowledgment received. The cells corresponding to only those pilots which are positively acknowledged are transmitted in a pipelined and circuit switched fashion in the second phase. Such rapid transfer of cells would not be possible if the switch has internal buffers.

In Section 2 of this paper we briefly introduce the balanced gamma (BG) network and compare its performance with two other MINs, namely the 2-replicated 2-dilated banyan (R2D2) and Batcher- banyan networks. We employ neither backpressure nor buffering while performing this comparison. The reason is that we want to study the internal and output blocking behavior of the BG network compared to the other two networks without using any performance enhancing mechanisms. The comparison is conducted using uniform and non-uniform traffic loads. In Section 3, we investigate the buffer requirements of the BG network and the banyan network when implemented in a pipelined fashion. We extend the study provided in (Wong & Yeung 1992) to estimate the output buffer requirements for the pipelined networks. In addition, we estimate the buffer size not only under uniform traffic load, but also with realistic traffic mixes composed of both uniform random traffic and bursty traffic. Additionally, 20% of the cells generated in these mixes are assigned higher priority over the other generated cells to demonstrate the capability of the networks to handle priority traffic. We present concluding discussions in Section 4.

This work was supported partly by a Research Grant from NSERC. Parts of this work were presented in INFOCOM'92, CCECE'96 and SPECTS'98.

2 BG, BB AND R2D2 Networks

In this section, we compare the performance of three candidate MINs, namely BG, BB and R2D2 networks. The BG network is a multipath MIN. An $N \times N$ BG network, with N inputs and N outputs, consists of n stages ($n = log_2 N$). The first stage has 1×4 switching elements (SEs). Each of the following $n - 1$ stages has 4×4 crossbar SEs. The last stage is connected to the output buffers. Each stage has N SEs numbered from 0 to $N - 1$. An 8×8 BG is shown in Figure 1. The i^{th} SE in the j^{th} stage is connected to SEs i, $(i + 2^j)\text{mod}N$, $(i - 2^j)\text{mod}N$, and $(i + 2^{j+1})\text{mod}N$ in the $(j+1)^{th}$ stage. Links connecting the last stage SEs and output buffers follow the same pattern. Output link 0 and output link 2 are called regular links, whereas, output link 1 and output link 3 are called alternate links. The reason is that in any SE output, the cells arriving at the SE input links first target regular links. The BG network uses the reverse destination tag routing scheme. The routing tag bits for a cell, which needs to be routed from source S to destination D, is D in binary. The SEs interpret the routing tag in the reverse order, i.e., switching in the first stage (stage 0) is based on the least significant bit of D. In any SE, if the tag bit is 0 (1), then the cell is directed to one of the top (bottom) two output links. Accordingly, cells are dropped from

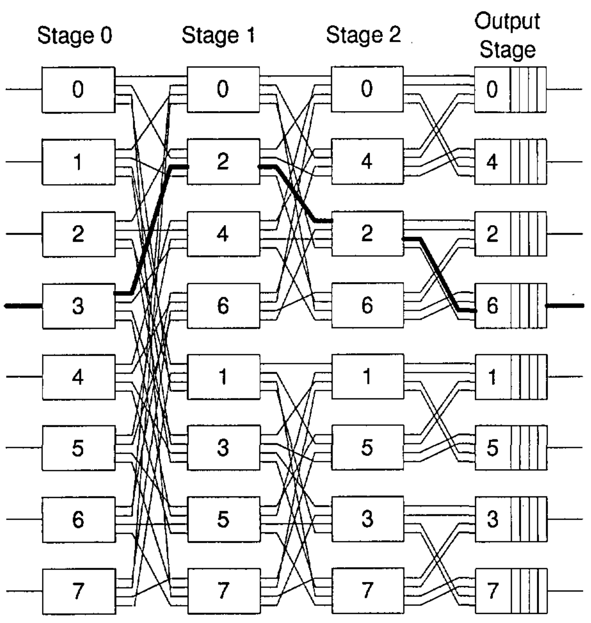


Figure 1: An 8 × 8 BG network.

any of the SEs only when more than two cells having the same routing bit arrive at the input of an SE. In Figure 1, the routing path from source 3 to destination 6 is shown in thick line. Further discussions on topology, fault tolerance and reliability of the BG network can be found in (El Sayed et al. 1999). We point out that the high throughput of this network is derived from the fact that alternative paths are available at every stage. We believe such judicious deployment of hardware resources is superior to adding numerous buffers to an inherently inefficient network to improve its performance. The performance would obviously improve further if we choose a MIN built using larger basic blocks, for example with 6 × 6 or 8 × 8 SEs. However, we find that the improvement in performance is not commensurate with the additional hardware complexity.

The second candidate is the BB network. In the BB network, a Batcher network first sorts the incoming cells in an ascending order. The sorted cells are handed to a banyan network after perfectly shuffling them. The hardware complexity of the Batcher network is quite high: there are $\frac{N}{2}(\frac{log_2 N+1}{2} log_2 N)$ sorting elements and $\frac{N}{2} log_2 N$ SEs. The implication is that they are more popular in the research community than for commercial applications (Lindberg 1994). Furthermore, however, this network has captured the imagination of several people, as the Batcher-banyan combination is known to be *non-blocking*. We should remember that this combination is non-blocking only if there are nonrepeated destination addresses, that is, if the incoming traffic is of permutation type. As it is well recognized that the anticipated traffic in an broadband communication network is bursty, a Batcher-banyan network is only of academic interest. Nevertheless, we have chosen to include the BB network in our study as many researchers employ this as the reference network for performance comparison. Also, the BB is identical to crossbar from the standpoint of performance.

The last candidate is the R2D2 network, which has a hardware complexity comparable to that of a BG network; hence the choice of the R2D2 network in our study. The topology of the R2D2 network is the same as the parallel banyan network (Itoh 1991). However, each input link (consequently each output link) of an SE is dilated by a factor of 2, i.e. each 2 × 2 SE, which is the building block of the banyan network, is replaced by a 4 × 4 SE. As in the case of the BG network, in R2D2 network too, a maximum of 4 cells can be accepted at each output port in any switching cycle.

To obtain a good insight into of the effects of internal and output blocking, no buffering schemes nor backpressure mechanism is employed within these candidates when we study their performance in this section. An infinite output buffering is assumed to eliminate the effects of output queuing.

2.1 Uniform Traffic Patterns

When voice, video, data and other payloads are carried in an integrated network, we can anticipate a complex traffic pattern. Although it may not be possible to exactly forecast and simulate realistic traffic patterns, study of MINs under certain easily synthesizable traffic patterns may help in understanding their performance under realistic traffic patterns.

We present below results obtained using a simulation software (in C++) developed at Memorial University of Newfoundland. The choice of a random number generator (RNG) is crucial for any Monte Carlo simulation. For the simulation of performance evaluation of MINs, an RNG is used to generate active cells, select destination for the cells, and assign priority based on a predefined probability. For this application an RNG, which has uniform distribution of numbers and long period is required. A newly reported algorithm called Mersenne Twister (Matsumoto & Nishimura 1998) for random number generation, which provides a period of $2^{19937} - 1$ and 623-dimensional equidistribution up to 32-bits accuracy, consuming a working area of 624 words, is used. The choice of the initial seed has no impact on any of these properties, which is an added advantage of using this generator.

Permutation Traffic

The *permutation* traffic pattern refers to the case where the output ports requested by cells arriving to the switch in the same time slot are distinct from one another. It is referred to as permutation traffic because, at full load (that is, when we have one cell arriving at each input line of the switch at each cycle), the list of requested destinations ordered by cells at the input lines forms a permutation of the set $(0, 1, \dots, N - 1)$. In the case of permutation traffic, at full load, only one cell is destined to each destination. Accordingly, cells will be lost only

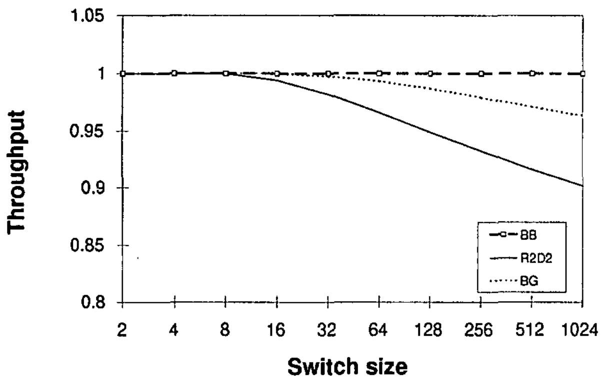


Figure 2: Throughput performance of MINs under permutation traffic.

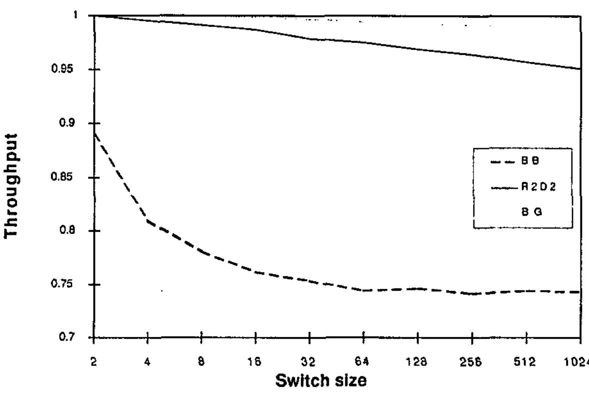


Figure 3: Throughput performance of MINs under uniform random traffic - 50% load.

due to internal blocking in the network and not due to output contention at a particular destination. Furthermore, if the departure rate of cells from the switch fabric onto the channel is 1, then there will be no output queuing problem either. To summarize, the permutation traffic can be considered, to a large extent, as a vehicle to demonstrate the internal blocking within MINs.

Both the banyan and the R2D2 networks do not have 100% throughput under permutation traffic patterns because they do not have non-conflicting paths for certain combinations of permutation traffic. Hence both of them are blocking networks under permutation traffic. In the BB network, it has been showed that non-conflicting paths exist between each source and destination and hence the throughput of the BB network is always 1 under permutation traffic pattern (Hui 1990). Therefore, BB network is a totally nonblocking network under permutation traffic.

In the BG network, due to the connection pattern, stage 0 and stage 1 of the BG network are nonblocking for any kind of traffic. Under permutation traffic, stage $(n-1)$ and stage $(n-2)$ are also nonblocking. Hence an $N \times N$ BG network is nonblocking under permutation traffic, iff $N \leq 16$ (Sivakumar 1996). Figure 2 shows the simulation results of the performance of the BB, the BG, and the R2D2 networks under permutation traffic at full load. The results in Figure 2 verify the facts discussed above.

Uniform Random Traffic

The uniform random traffic pattern refers to the case where each output port has an equal probability of being requested by cells arriving at the input ports. As destination addresses can be repeated, it is referred to as uniform random traffic. Even though the uniform random traffic does not properly describe the real traffic, it is a more realistic traffic pattern when compared to the permutation traffic pattern. Performance of MINs used in multiprocessor interconnections and in switch fabrics of broadband packet switching architectures is usually studied under uniform random traffic. It is a much simpler traffic pattern to be

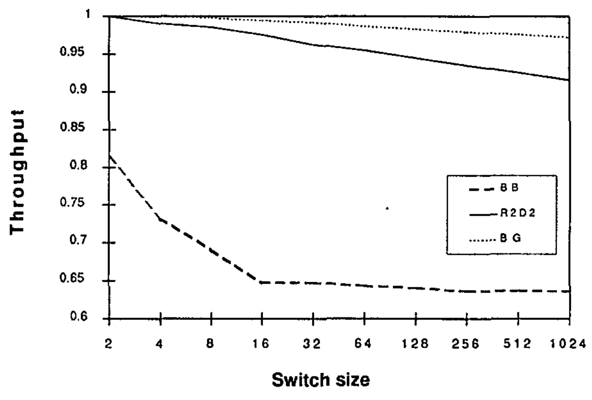


Figure 4: Throughput performance of MINs under uniform random traffic - 70% load.

analytically modeled as compared to the non-uniform traffic types. Therefore, analytical performance results obtained using uniform random traffic pattern are used to validate the software simulating packages. Figure 3, Figure 4, and Figure 5 show the performance of the three MINs under uniform random traffic of 50% load, 70% load, and full load, respectively. It can be clearly seen from the above figures that the throughput of the BG network is considerably better than that of the R2D2 and the BB networks under various percentage loads of uniform random traffic. Furthermore, we note that the BG network has a considerably better performance compared to the BB network under heavy traffic loads. Next, we present an approximate analysis of an $N \times N$ BG network. Let the probability of an output regular link at stage i carrying a cell is denoted by $x_{r,i}(1)$, and $x_{r,i}(0)$ denotes the probability that the output regular link does not carry a cell. Similarly, $x_{a,i}(1)$ ($x_{a,i}(0)$) refers to the probability that an alternate link at stage i carries (does not carry) a cell. We know that $x_{r,0}(0) = x_{r,0}(1) = 0.5$; $x_{a,0}(0) = x_{a,0}(1) = 0$; for uniform arrival at full load.

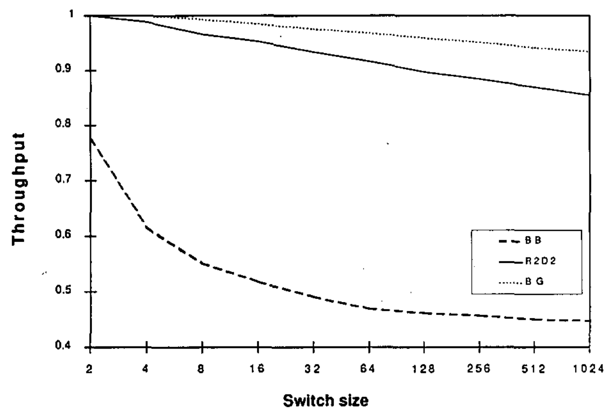


Figure 5: Throughput performance of MINs under uniform random traffic- full load.

The following recursive equations can be derived:

$$
\begin{aligned}
x_{r,i}(0) &= x^2_{r,i-1}(0)x^2_{a,i-1}(0) + x^2_{r,i-1}(0)x_{a,i-1}(0)\\
&\quad x_{a,i-1}(1) + x_{r,i-1}(0)x_{r,i-1}(1)x^2_{a,i-1}(0)\\
&\quad +\frac{1}{4}x^2_{r,i-1}(1)x^2_{a,i-1}(0) + \frac{1}{4}x^2_{r,i-1}(0)\\
&\quad x^2_{a,i-1}(1) + x_{r,i-1}(0)x_{r,i-1}(1)x_{a,i-1}(0)\\
&\quad x_{a,i-1}(1) + \frac{1}{4}x^2_{r,i-1}(1)x_{a,i-1}(0)\\
&\quad x_{a,i-1}(1) + \frac{1}{4}x_{r,i-1}(0)x_{r,i-1}(1)\\
&\quad x^2_{a,i-1}(1) + \frac{1}{16}x^2_{r,i-1}(1)x^2_{a,i-1}(1), \qquad (1)
\end{aligned}
$$

$$x_{r,i}(1) = 1 - x_{r,i}(0), \qquad (2)$$

$$
\begin{aligned}
x_{a,i}(1) &= \frac{1}{4}x^2_{r,i-1}(1)x^2_{a,i-1}(0) + \frac{1}{4}x^2_{r,i-1}(0)\\
&\quad x^2_{a,i-1}(1) + x_{r,i-1}(0)x_{r,i-1}(1)x_{a,i-1}(0)\\
&\quad x_{a,i-1}(1) + x^2_{r,i-1}(1)x_{a,i-1}(0)x_{a,i-1}(1)\\
&\quad +x_{r,i-1}(0)x_{r,i-1}(1)x^2_{a,i-1}(1)\\
&\quad +\frac{11}{16}x^2_{r,i-1}(1)x^2_{a,i-1}(1), \qquad (3)
\end{aligned}
$$

$$x_{a,i}(0) = 1 - x_{a,i}(1). \qquad (4)$$

When we assume that there exists a mechanism to accept up to 4 cells in a cycle then an approximate expression for the network throughput (TP) can be given by:

$$TP = 2x_{r,i}(1) + 2x_{a,i}(1). \qquad (5)$$

The results of this analysis are shown in Figure 6. The results suggest that the analytical model is more optimistic than simulation. However, even for large networks the discrepancy is less than 2%. We have further validated our simulator by reproducing results reported by other researchers.

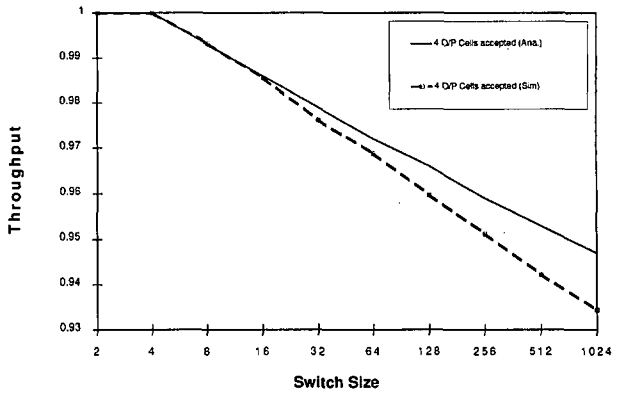


Figure 6: Performance analysis of the BG network-analytical and simulation results under full load.

2.2 Non - Uniform Traffic Patterns

The type of traffic expected in B-ISDN is likely not of the uniform traffic types discussed in Section 2.1. Considerable research is underway to determine a traffic model that would accurately characterize the anticipated ATM traffic. Since it is quite difficult to simulate and analyze the exact traffic expected in B-ISDN, researchers have come up with certain tractable non-uniform traffic types. Studying the performance of MINs under the non-uniform traffic patterns could give a reasonably good appreciation of the performance of the MINs under real traffic. The following sections explain these traffic types and provide simulation results for the performance of the three MINs under these traffic types.

Hotspot Traffic

Hotspot traffic (Scott & Sohi 1990; Badran & Mouftah 1990) is defined as the traffic type in which one or more nodes of the switch fabric, or of the destinations receive a given percentage of the incoming cells. The rest of the incoming traffic follows the uniform random pattern. These nodes or destinations are referred to as hot spots. It is difficult to model hotspots at nodes within the switch fabrics and hence performance study of MINs under hotspot destinations was carried out. Such hotspot traffic is also referred to as output-concentration traffic (Tobagi et al. 1991). Figure 6 gives the performance of MINs under 10% hotspot traffic with a probability of cell arrival at inputs equal to one. This indicates that 10% of the incoming traffic is destined to a particular output. Performance of the BG network is better than the BB network and the R2D2 network in the case of the hotspot traffic. Figure 8 gives the performance of the BG network under varying degrees of hotspots. Performance of the BG network under uniform random traffic is also provided for comparison. It is clearly seen that the throughput of the BG network drops down considerably with the increase in the percentage of hotspot load. This is to be expected in any realizable network.

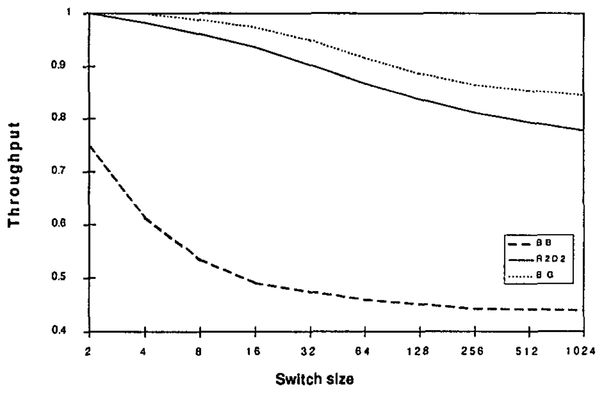


Figure 7: Throughput performance of MINs under 10% hotspot traffic.

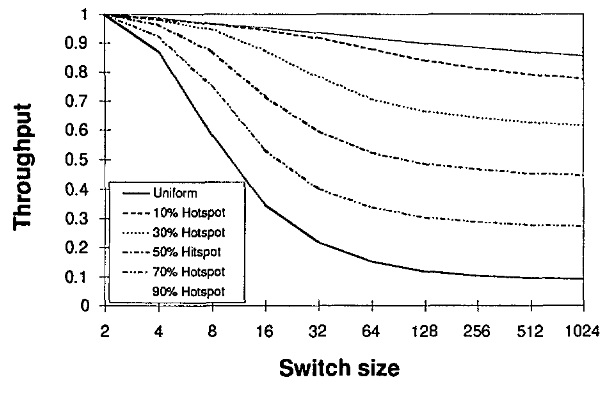


Figure 8: Throughput performance of the BG network for varying hotspot traffic loads.

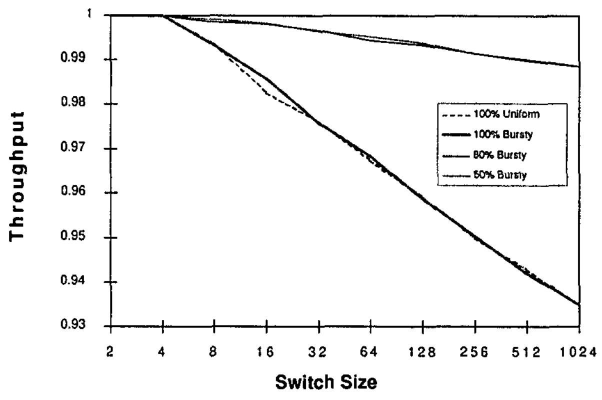


Figure 9: Throughput performance of BG network under varying loads of bursty traffic.

Bursty traffic

Bursty traffic is a traffic type in which the inputs of the switch fabric receive sudden bursts of packets (Ahmadi 1990; Badran & Mouftah 1990). The On-Off model is the least complex and the most widely used to model bursty sources in simulation. The On-Off model can describe most of the existing sources with a reasonable accuracy (Stamoulis et al. 1994). The model assumes that the source generates cells in a bursty manner: one active period (On period) followed by an idle one (Off period). The lengths of On and Off periods are independently evaluated from two geometric distributions given by (Banks et al. 1996):

$$L = 1 + \lceil \frac{\ln(1-R)}{\ln(1-p)} - 1 \rceil, \qquad (6)$$

where L is the period length in cells, $0 \leq R \leq 1$ is the random number generated, and $0 < p \leq 1$ is inverse of the average period length in cells. The cells arriving at each input line in a burst are destined to the same output line. This output is selected randomly.

Simulation results of the performance of the BG networks for different bursty traffic loads p, and a mean burst length equal to 5 cells/burst, are shown in Figure 9. The performance of the BG network under full load of uniform random traffic is also provided in Figure 9. It can be seen that the throughput of the BG network increases with a decrease in the probability of the bursty traffic but does not vary much between probabilities of 0.5 and 0.8. The throughput under 100% bursty traffic is slightly less than that under the uniform random traffic. The reason is some output ports, under bursty load, would be busy for longer periods as they are serving multiple bursts generated by some inputs ports. Later we will show how the performance degrades as the burst length increases. Note that the uniform random traffic is a special case of bursty traffic where the burst length is equal to one. The performance of the BG networks under bursty traffic at full load, with the same burst length used above, is compared with

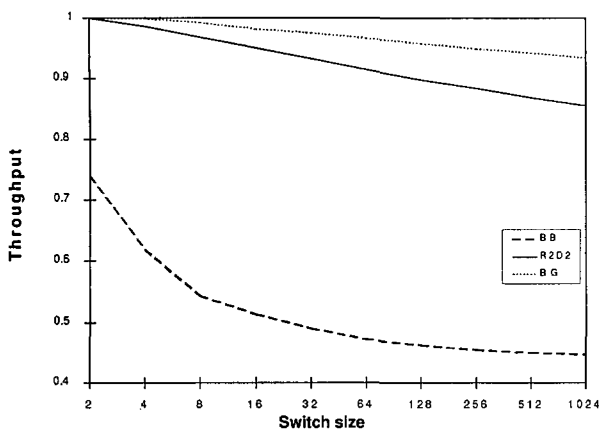


Figure 10: Performance analysis of the three MINs under Full Load of bursty traffic.

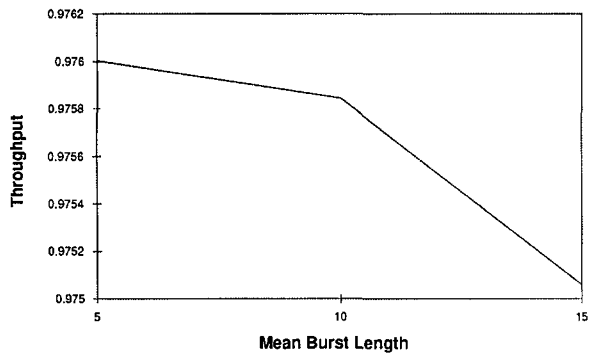


Figure 11: Throughput performance of a 32 × 32 BG network under varying burst lengths of bursty traffic.

the performances of the BB and R2D2 networks in Figure 10. It can be seen that the BG network features a better performance as compared to the BB and the R2D2 networks.

The throughput performance of a 32 × 32 BG network under full load for varying mean burst lengths is shown in Figure 11. The performance of the BG network decreases with increase in the burst length.

The performance results reported above are quite high, but they still do not meet the requirements in a broadband communication network. A straightforward solution to improve performance of MINs is to employ buffering. Buffers can be used at the main inputs and the main outputs or within the SEs themselves. Besides buffering, other solutions such as replication and dilation (as in R2D2 network), pipelining, speedup, incorporation of backpressure mechanism and cascaded (or tandem) networks, can be used.

3 Pipelined MINs

Several switch architectures have been proposed to build fast and efficient fast packet switching networks. Some ex-

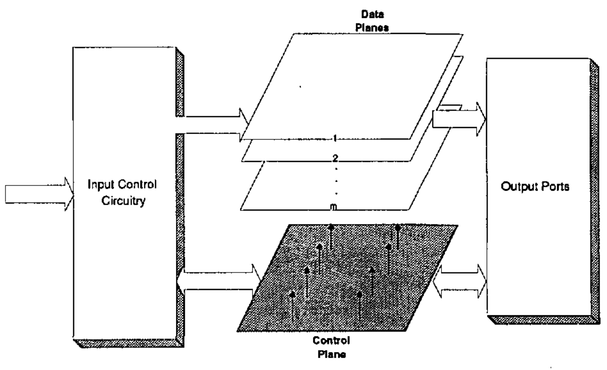


Figure 12: Block diagram of a pipelined architecture.

amples are: enhanced crossbar (Tobagi 1990; Mckeown et al. 1997), Tandem banyan (Tobagi 1991), speeded up banyan (Awdeh & Mouftah 1994), nonblocking Batcher-banyan (Giacopelli et al. 1991), Plane Interconnected Parallel Network (PIPN) (Oktug & Caglayan 1997), Parallel Tree banyan network (Al-Mohamed personal communication), and pipelined MINs. All these structures employ output buffers, and many employ input buffers as well. The number of banyan networks required for Tandem banyan and PIPN to achieve a cell loss probability of 10^{-9} is quite high. Furthermore, the cells in these two architectures are prone to variable cell delays and loss of sequence, which is undesirable for ATM traffic. The parallel tree banyan arrangement is very hardware intensive and it is difficult to implement this network using two-dimensional VLSI technology. The speeded up banyan network employs time redundancy to achieve an acceptable high performance, whereas pipelining employs hardware redundancy to attain the same. The implementation of pipelined MINs is therefore simpler in terms of operating speed. BB networks have, as we mentioned earlier, a high hardware complexity and are nonblocking only under permutation traffic. The performance of BB networks degrades more sharply than pipelined MINs under realistic traffic loads. If we exclude PIPN and pipeline structures, none of these architectures has inherent fault tolerance or a high reliability. The number of input buffers required to achieve a low cell loss probability is high for these networks. Pipelined MINs offer a good trade-off between performance and hardware complexity. The pipelined MINs have acceptably low hardware complexity, sequential cell delivery and a low cell delay, and can produce a very high throughput using few buffers to achieve acceptable cell loss probability.

The concept of pipelined MINs was first introduced in (Wong & Yeung 1992) based on a Banyan network. Figure 12 depicts a block diagram of a pipelined architecture. The control plane and each of the data planes are based on the same MIN. The input control circuitry consists of a buffer at each input line, and a set of input port controllers (IPCs), which are responsible for distributing the cells stored in the input buffers among the data planes. The control plane is used to switch only a few bits that represent relevant header

Load	Bursty	URT
0.20	0.10	0.10
0.30	0.10	0.20
0.50	0.20	0.30
0.70	0.20	0.50
0.80	0.20	0.60
0.90	0.20	0.70

Table 1: Traffic mixes under consideration

information contained in the cell header; we refer to these bits as pilots in the introduction. They are formed by the IPCs. The control plane transfers the routing configuration to the data planes, which transmit the whole cell in a pipelined and circuit-switched fashion. The output ports have buffer queues for each output line. An output port accepts all cells that arrive in the same switching cycle if there is room in its output buffer queue. Each cycle is divided into reservation slots and one data plane is selected on a round robin basis in each reservation slot. During a reservation slot, the IPC writes a copy of the cell into the transmit register of the selected data plane. The control plane sends 1 bit acknowledgement signal to each input, notifying the IPCs whether a cell is successfully routed or not. On receiving a positive acknowledgment signal, the IPC transmits the cell through the data plane to the output and keeps the negatively acknowledged cells for the next reservation slot, i.e. a backpressure mechanism is adopted. Further discussions about the design and operation of pipelined structures can be found in (Wong & Yeung 1995; Venkatesan et al. 1997).

As we mentioned above, the first pipeline structure proposed was based on banyan topology. In (Wong & Yeung 1995), analytical modeling, verified by simulation, was used to study the performance of the pipelined banyan network (PBN) under uniform random traffic load. The number of data planes and the size of each input buffer queue were determined under the assumption of infinite output buffering. In this section we extend the study to estimate the output buffering requirements. Also we further investigate the buffering requirements under some realistic traffic mixes of both uniform random and bursty traffic loads. These traffic mixes are shown in Table 1. The bursty traffic used in these mixes has the same properties of the bursty traffic we previously used. We will limit our simulations to network size of 256 $\times$ 256 and always obtaining a cell loss probability of 10^{-9}. In addition, we give 20% of the total generated cells during any of the simulation trials higher priority than other generated cells because in real ATM traffic it is expected to have two level priority cells. However, we discovered that the inclusion of these high priority cells did not result in a significance change in our simulations for both PBGN and PBN, therefore, we limit our discussion to simulation experiments that include high priority cells. Our procedure starts by estimating the required number of data planes for each traffic load under

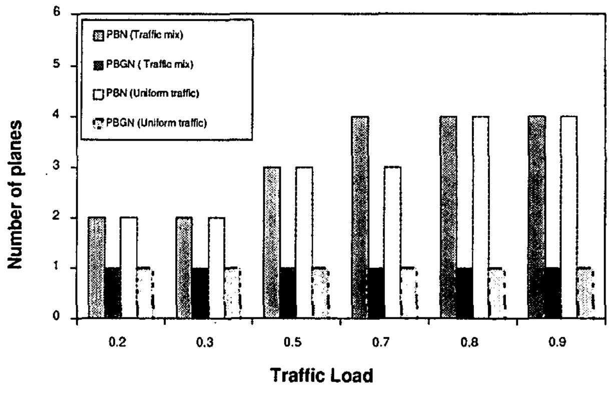


Figure 13: Number of planes required by PBN and PBGN under different loads.

the assumption of infinite input and infinite output buffers. Then we try to evaluate the input buffer requirements under the assumption of infinite output buffering using the same number of data planes we previously evaluated. Finally, we evaluate the output buffer requirements, using the number of data planes and the input buffer size estimated in the last two steps, by introducing the output queuing effect by setting the consumption rate of any of the output links to 1. That way no more than one cell can leave any of the output queues in a switching cycle.

Infinite Output Buffers

We first evaluate the number of the data planes assuming infinite input and infinite output buffers. From Figure 13 we can deduce that one data plane of the PBGN is sufficient to get the desired cell loss probability for most of the uniform traffic loads and traffic mixes given in Table 1. The PBGN requires one data plane while PBN requires 4 data planes for a traffic load above 80% to obtain the desired 10^{-9} cell loss probability. Two data planes would be required for the PBGN if the traffic load is above 90%. Figure 14 depicts the input buffer requirements for both PBGN and PBN using the number of data planes estimated in Figure 13. Obviously, the buffer requirements for traffic mixes are higher than those for plain uniform random traffic. This signifies the impact of the bursty traffic on the performance of switch fabrics. The dips in the input buffer requirements of the PBN in Figure 14 is due to using different numbers of data planes as we increase the load. For example, the input buffer requirements are almost the same for PBN when loaded with 70% uniform random traffic or 70% traffic mix because 3 data planes are used in the former case whereas 4 data planes are used in the latter case. For all subsequent simulation trials, only one data plane is used for the BG network.

Finite Output Buffers

Earlier researchers (Patavina & Bruzzi 1993; Awedh & Mouftah 1994) did not decouple the issue of output queuing from the internal and output blocking effects.

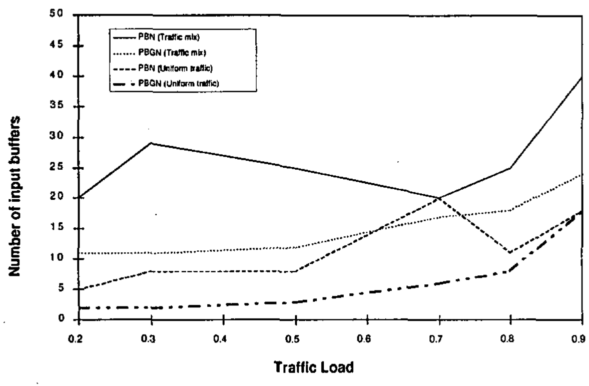


Figure 14: Input buffer size under traffic loads reported in Figure 13.

In this section we study the effect of output queuing by limiting the output buffer size and the consumption rate to 1, i.e. at most only one cell leaves any of the output buffer queues in any switching cycle. In fact, practically, only one cell can leave an output queue in each switching cycle.

Figure 15 introduces the output buffering requirements for PBN and PBGN under uniform random traffic and traffic loads reported in Table 1. Inspection of Figures 15.a, 15.b, and 14 shows that the input buffer size depends on the traffic and the switch architecture used, whereas the output buffer size depends on the traffic type used. Obviously, we see from Figure 15 that the output buffer requirements for both PBN and PBGN are almost the same when loaded with the same traffic. Moreover, this phenomenon are more apparent as the applied traffic load increases. The total number of buffers required is comparable to those reported in (Patavina & Bruzzi 1993). For example, it was reported that a nonblocking network would require 32 input buffers and 73 output buffers under uniform traffic load of 90%. In our simulations we have shown that a PBN requires about 16 input buffers and 75 output buffers while a PBGN requires about 18 input buffers and 75 output buffers. For this case the output buffer requirement is almost the same although fewer input buffers are required for the pipelined networks compared to the nonblocking network. For other cases the total buffers required is consistent with those reported in (Patavina & Bruzzi 1993).

Figure 16 depicts the effect of the burst length on the buffer requirements of the network. The simulation trials were carried out under a load of 70%, with same ratios of bursty and URT loads given in Table 1. The output buffer requirements jump significantly as the burst length increases. The reason is that the bursty loads intensify the output queuing problem. Accordingly, the output buffer size has to be increased to compensate the demands of these bursts. We have verified through simulation trials that the buffer requirements of the BG network are close to those of an ideal network.

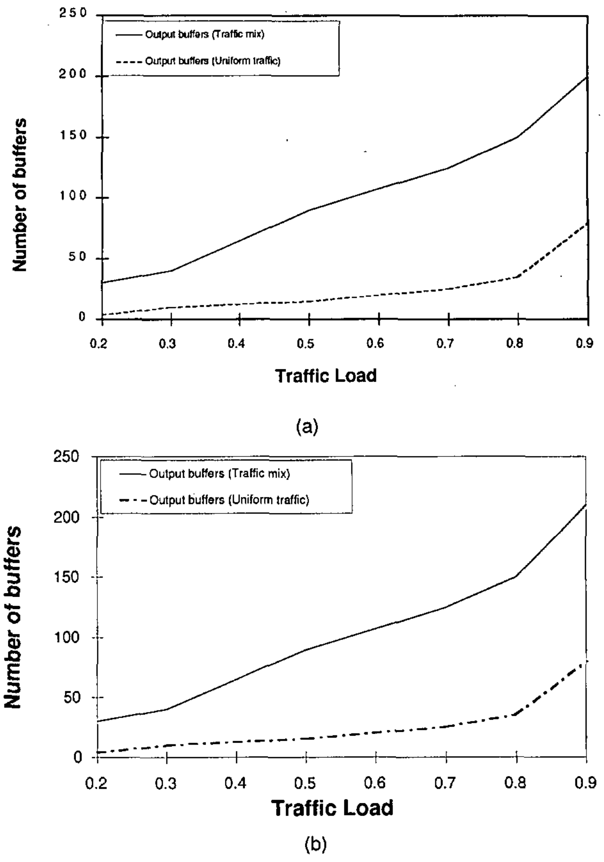


Figure 15: Output buffering required under different traffic loads (a) PGN (b) PBGN.

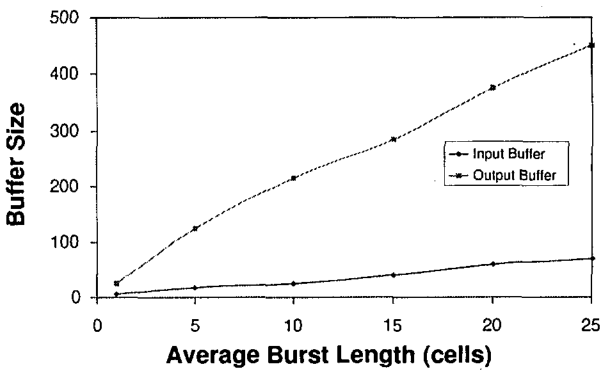


Figure 16: Buffer requirements of a single plane BG network under 70% mixed load.

4 Conclusion

We proposed reasons for the selection of multipath blocking MINs that possess high performance without any buffers in order to build large switch fabrics for integrated broadband switching network. We showed that it is possible to build switch fabrics that feature very low cell loss if we avoid using internal buffers. The BG network, built using 4×4 switching elements (SEs), meets these specifications.

We introduced the BG network and investigated its performance compared to other MINs. Firstly, we studied the network performance under uniform and non-uniform traffic loads without adopting either buffering or backpressure mechanisms. Except for the case of permutation traffic, the network was proved to be superior to the BB and R2D2 networks in terms of performance. This superiority is more significant as we increase the network size.

We discussed the basic ideas in pipelined MINs. The BG network was compared with banyan network when implemented in pipelined fashion. The issues of output queuing and blocking (internal and output) were decoupled and the sizes of the buffers were estimated for both PBN and PBGN to achieve a cell loss probability of 10^{-9}). We concluded that the number of output buffers required is the same for different architectures as this depends only on the traffic type. The amount of input buffering required depends on the characteristics of the switch architecture employed as well as on the traffic. Consequently, the output queuing can be considered as an architecture independent problem. We have also demonstrated that a BG network with a control plane and a data plane can handle many types of traffic loads with fewer input buffers as long as the traffic load is less than 90%. We are currently working on the VLSI realization of the BG network.

References

[1] Ahmadi H. (1990) Tree Switch: A modular Fast Packet Switching Fabric for Synchronous and Asynchronous Traffic. *Telecommunication Research Institute of Ontario.*

[2] Awdeh R.Y. & Mouftah H.T. (1994) Design and performance analysis of input-output buffering delta-based ATM switch with backpressure mechanism. *IEE Proceedings in Communications*, 141, 4, p. 225-244.

[3] Badran H.F. & Mouftah H.T. (1990) Performance of Broadband Integrated Switch Architectures with Input-Output-Buffering under Backpressure Mechanisms. *Research Report, no. 90-7, Queen's University, Kingston, Ontario, Canada*, 14 p.

[4] Banks J., Carson J.S., & Nelson B.L. (1996) Discrete-Event System Simulation. *Second Edition, Prentice Hall.*

[5] El Sayed Y., Venkatesan R., & Sivakumar H. (1999) Fault Tolerance and Reliability Analyses of the Balanced Gamma Network. *Accepted for Publication in the Special Issue of International Journal of Parallel and Distributed Systems and Networks on ATM SwitchingNetworking Architectures and Performance.*

[6] Giacopelli J.N., Hickey J.J., Marcus W.S., Sincoskie W.D., & Littlewood M. (1991) Sunshine: A high-performance self-routing broadband packet switch architecture. *IEEE Journal on Selected Areas in Communications*, 9, 8, p. 1289-1298.

[7] Hui H. Y. (1990) Switching and traffic theory for integrated broadband networks. *Kluwer Academic Publishers.*

[8] Itoh A. (1991) A Fault-tolerant Switching Network for B-ISDN. *IEEE Journal of Selected Areas in Communications*, 9, 8, p. 1218-1226.

[9] Lindberg B. C. (1994) Digital Broadband Networks & Services. *McGraw-Hill Series on Computer Communications.*

[10] Matsumoto M. & Nishimura T. (1998) Mersenne Twister: A 623-dimensionally Equidistributed Uniform Pseudorandom Number Generator. *ACM Transactions on Modeling and Computer Simulation*, 8, 1, p. 3-30.

[11] Mckeown N., Izzard M., Mekkittikul A., Ellersick W., & Horowitz M. (1997) The Tiny Tera: A Packet Switch Core. *IEEE Micro (Jan./Feb.)*, p. 26-33.

[12] Oktug S.F. & Caglayan M.U. (1997) Design and Performance Evaluation of a Banyan Network Based Interconnection Structure for ATM Switches. *IEEE Journal on Selected Areas in Communications*, 15, 5, p. 807-816.

[13] Patavina A. & Bruzzi G. (1993) Analysis of Input and Output Queuing for Nonblocking ATM Switches. *IEEE/ACM Transactions on Networking*, 1, 3, p. 327-341.

[14] Scott S.L. & Sohi G.S. (1990) The Use of Feedback in Multiprocessors and Its Applications to Tree Saturation Control. *IEEE Transactions on Parallel and Distributed Systems*, 1, 4, p. 385-398.

[15] Sivakumar H. (1996) Performance, Fault Tolerance, and Reliability of MINs for broadband packet switch architecture. *M. Eng. Thesis, Memorial University of Newfoundland.*

[16] Stamoulis G.D., Anagnostou M.E., & Georgantas A.D. (1994) Traffic Source Models for ATM Networks: A Survey. *Computer Communications*, 17, 6, p. 428-438.

[17] Tobagi F.A. (1990) Fast Packet Switch Architectures for Broad-band Integrated Services Digital Networks. *Proceedings of IEEE*, 78, 1, p. 133-167.

[18] Tobagi F.A., Kwok T., & Chiussi F.M. (1991) Architecture, Performance, and Implementation of the Tandem-banyan fast Packet Switch. *IEEE Journal on Selected Areas in Communications*, 9, 8, p. 1173-1193.

[19] Venkatesan R., El-Sayed Y., & Sivakumar H. (1997) Pipelined Balanced Gamma Network for Broadband Communications Switch Fabrics. *Proceedings of CCECE'97*, St. John's, canada, 2, p. 680-683.

[20] Wong P.C. & Yeung M.S. (1992) Pipeline Banyan - A Parallel Fast Packet Switch Architecture. *Proceedings of IEEE ICC/SUPERCOMM'92*, Chicago, IL, p. 882-887.

[21] Wong P.C. & Yeung M.S. (1995) Design and Analysis of a Novel Fast Packet Switch – Pipeline Banyan. *IEEE/ACM Transactions on Networking*, 3, 1, p. 63-69.

PNNI And The Optimal Design Of High-Speed ATM Networks

Abdella Battou
Center for Computational Science,
U.S. Naval Research Laboratory. Washington D.C.
AND
Bilal Khan, Sean Mountcastle,
ITT Industries Systems Division,
at the Center for Computational Science,
U.S. Naval Research Laboratory. Washington D.C.
E-mail: {battou,bilal,mountcas}@cmf.nrl.navy.mil

Keywords: ATM Network Design, PNNI Simulation, PRouST

Edited by: Mohsen Guizani

Received: January 15, 1999 **Revised:** July 31, 1999 **Accepted:** August 25, 1999

In addition to being well-dimensioned and cost-effective, a high-speed ATM network must pass some performance and robustness tests. We propose an approach to ATM network topology design that is driven by the performance of its routing protocol, PNNI. Towards this end, we define performance indicators based on the time and traffic required for the protocol to first enter and subsequently return to the meta-stable state of global synchrony, *in which switch views are in concordance with physical reality. We argue that the benefits of high call admission rate and low setup latency are guaranteed by our indicators. We use the PNNI Routing and Simulation Toolkit (PRouST), to conduct simulations of PNNI networks, with the aim of discovering how topological characteristics such as the diameter, representation size, and geodesic structure of a network affect its performance.*

1 Introduction

The size of operational ATM clouds continues to grow at an increasing pace. Both in anticipation of this changing scale, and to insure smooth inter-operation of these networks, the ATM Private Network-Network Interface standard (PNNI) was recently adopted (ATM Forum 1996). PNNI defines a set of protocols for *hierarchical* networks of ATM switches, and is designed to provide efficient and effective routing. In the long term, however, the degree to which PNNI succeeds in this regard will depend crucially on two factors:

First, because PNNI does not mandate specific policies for call admission, route selection, or topology aggregation, these aspects of the protocol remain "implementation-specific". Clearly the degree to which PNNI meets the challenges posed by tomorrow's ATM networks will depend significantly on the success of *switch designers* in devising effective algorithms for the admission and routing of connections, and for the aggregation of topology information.

Second, *network designers* must have the tools and information necessary to design ATM network topologies that are (i) capable of meeting anticipated traffic demands, and (ii) *optimized for performance under PNNI.* In this paper, we shall not address the first of these two issues, that of dimensioning networks to satisfy known costs and traffic demands. Our investigations begin at the point where a network designer, having been given projected traffic profiles and switch/fiber specifications, has arrived at a set of candidate ATM network topologies which appear equally adequate. We argue that although two topologies may appear indistinguishable in terms of the mathematics of QoS requirements, the PNNI protocol exhibits significant differentiation in their performance. Understanding how the PNNI protocol affects network performance is a necessary first step to determining how the adoption of PNNI *should* affect ATM network design. In subsequent sections, we shall describe our simulation experiments and begin developing guidelines for ATM network topology design that take into consideration the specific nature of the PNNI protocol.

2 PNNI Performance Indicators

There are many candidate performance criteria for evaluating the relative merits of network topologies. Here we shall assume that topology design is motivated by increasing the ATM network's call admission rate and decreasing the average connection setup latency. Additionally, we desire that the background traffic due to the PNNI protocol itself, should not be excessively high.

Setup Latency. In the absence of crankback, setup latency within a peergroup is seen to be linearly correlated with the number of hops in the selected path (Niehaus et al. 1997) and thus may be estimated in the worst case by the network diameter. When crankbacks occur, each failed attempt at valid route selection contributes significant additional latency, required for backtracking to the entry border node, computing an alternate route and then re-traversing the peergroup along the new path. Reduction of setup latency thus requires minimizing the crankback frequency.

Crankbacks and Call Admission Rate. Recent results on PNNI aggregation schemes [2,4] indicate that ATM call admission rate and crankback frequency is directly proportional to the "distortion" present in switches' views of the network. In particular, the experimental data presented in (Awerbuch et al. 1998) confirms the intuition that when a switch has inaccurate (e.g. outdated) views of network topology and metric information, this increases the likelihood that calls entering the peergroup at that switch will be assigned sub-optimal routes. A larger discrepancy between a switch's local information and the underlying reality of the network's state, results in a larger fraction of calls originating at the switch being rejected en-route, hence undergoing crankbacks (and possibly even unwarranted rejection at the source).

Thus, beyond the problem of dimensioning, selecting topologies that will yield high ATM call admission rates and low average setup latency requires that one be able to characterize which topologies minimize the divergence in switches' views.

2.1 Our Approach

Local synchrony. We define *local synchrony* of a peergroup to be the state where every switch in the peergroup has knowledge of the same set of PNNI Topology State Elements (PTSEs). It follows from the logic of the PNNI NodePeer finite state machine, that if a peergroup reaches local synchrony, then all member switches agree about the topology metrics describing their peergroup. Within a PNNI peergroup, each member switch is responsible for originating and flooding accurate information about its internal state and the resource availabilities on its incident links. Thus, modulo any loss due to aggregation schemes, local synchrony may be interpreted as a state in which all members of the peergroup are in agreement not only amongst themselves, but also with the underlying reality of the peergroup's state.

Global synchrony. We define *global synchrony* of a connected ATM network to be the state where every peergroup at every level has reached local synchrony, and the PNNI network hierarchy has reached a unique apex. Admittedly, the notion of global synchrony is "artificial" in the sense that it may be rarely achieved in real dynamic networks where connections are constantly arriving and departing. However, in a simulated network this state is attainable, and we shall use it to probe the rate of PNNI information propagation.

When a switched virtual circuit (SVC) is established in an ATM network, bandwidth availability is altered for links that the SVC traverses. Assuming this change is significant, updated information is re-originated and flooded by each switch incident to the affected links. If the network had reached global synchrony prior to the SVC setup request, these re-originations cause the network to fall out of a state of global synchrony for a brief time, until the new information has reached every node. This naturally leads us to consider:

- **Resynchronization time:** Average time required for the network
  to return to global synchrony, after a single, isolated, random SVC setup.
- **Resynchronization traffic:** Average PNNI traffic required for the network
  to return to global synchrony, after a single, isolated, random SVC setup.

The basic "trial" involved in measuring the above **resynchronization parameters** is as follows: allow the network to reach global synchrony, then inject a connection request between randomly chosen source and destination nodes and measure the time required and bytes transmitted before the network returns to a state of global synchrony. By repeating this trial a large number of times, we obtain an average value, which we call the resynchronization time.

To illustrate the importance of resynchronization time, let us consider two extreme scenarios. First, consider a network where the average time between SVC requests (i.e. 1/ SVC arrival rate) is much higher than the network resynchronization time. In this situation, changes in bandwidth availability induced by an SVC setup will, on average, have time to flood to all other switches in the network prior to the arrival of the next SVC setup request. Thus, routing decisions for each SVC will, on average, be made according to completely accurate information at the originating switch. If an SVC setup is rejected or experiences unacceptably high latency, this undesirable behavior is attributable solely to inadequate dimensioning of the network, because there is no legitimate way to fulfill the request.

In contrast, consider a network where the average time between SVC requests is much lower than the network resynchronization time. In this situation, changes in bandwidth availability induced by an SVC setup have not yet propagated to many switches in the network by the time the next SVC setup request arrives. Thus, the routing decision for an SVC is likely to be made according to stale information. The extent to which the information is stale is determined by the extent to which network resynchronization time exceeds average inter-SVC arrival time. If an SVC setup is rejected or experiences unacceptably high latency, this undesirable behavior may be due to inadequate dimensioning of the network or it may be due to suboptimal routes and unwarranted rejections induced by inaccurate information at the switches.

Major changes in network topology, such as network partitioning due to link/node failure, or re-merging of components upon subsequent recovery, will cause the PNNI hierarchy to undergo severe restructuring. We define the **boot parameters** below to be indicators of the time and traffic required to reinstate consistent routing information after such catastrophic changes.

- **Boot time:** Time at which the network first reaches global synchrony.
- **Boot traffic:** PNNI traffic required for the network to reach
  global synchrony for the first time.

We take the parameters above as a worst-case estimate of the time and traffic resources required to return to global synchrony. By comparison, the resynchronization parameters described earlier aim to measure the same quantities for the average case, i.e. during normal (stable) operation of the network.

We contend that a network designer, given two topologies that are equivalent with regards to meeting anticipated QoS requirements, must take into consideration their resynchronization and boot times. In particular, reducing these two quantities increases the fraction of time that the network spends in a state of global synchrony. At the same time, the designer must keep a watchful eye on the Boot traffic and Resynchronization traffic to make sure that not too much of the network bandwidth is being expended by PNNI itself.

One way in which the designer might determine the values of the four parameters mentioned above is to take physical measurements of them on two live networks, each configured in the appropriate topology. But this would be difficult to do accurately because of the inherent problems in distributed measurement, and moreover it would completely defeat the intention of *design before implementation!* Alternately, one could simulate the candidate PNNI networks to determine both time and traffic for boot and resynchronization; we follow this latter approach here.

3 Experiments

3.1 The Simulation Environment

Our simulation experiments were carried out using the PNNI Routing and Simulation Toolkit (PRouST), which was developed by the Signaling Group at the Naval Research Laboratory's Center for Computational Sciences. PRouST is a faithful and complete implementation of version 1.0 of the PNNI standard and can be used both to simulate large networks of ATM switches as well as to emulate live ATM switch stacks. In particular, PRouST includes the Hello, NodePeer, and Election finite state machines, all relevant packet encoding and decoding libraries, routing database management, and full support for hierarchy. In addition, a "plug-in" interface for call admission, path selection and aggregation policies is provided. For inter-switch signaling PRouST makes use of the NRL Signaling Entity for ATM Networks (SEAN), which is a complete simulation/emulation library implementing version 4.0 of the ATM User-Network-Interface (UNI) standard. The fidelity of PRouST's PNNI implementation has been demonstrated extensively in live interoperability tests with commercial switches. Both PRouST and SEAN are written in C++ over the Component Architecture for Simulation of Network Objects (CASiNO) described in (Mountcastle et al. 1999), and both are available in the public domain.

In the simulations that follow, all network links operate at the OC3 rate. Link jitter varies uniformly between $+10\ \mu s$ and $-10\ \mu s$ for each transmitted message. PNNI messages that enter the switch control port experience a latency of 10 ms. The backplane of the switch routes data traffic on existing virtual circuits at OC48 rates. These figures, while artificial, are projections of current switch vendor specifications. We found that jitter did not noticeably alter the outcome of our simulations from one trial to the next. The variations were typically less than 1% and for boot and resynchronization time, and less than 5% for boot and resynchronization traffic. The values we have listed in our tables are mean values.

An outline of results presented: We seek to understand what factors influence resynchronization and boot parameters. To this end, we start by simulating single-peergroup networks with grid, chain, ring and star topologies; these results are presented in sections 3.2 and 3.3. We compare these very particular families of topologies with similar experiments using all possible topologies on 7 nodes–these results are described in sections 3.4-3.5. In addition, we simulate 100 randomly generated topologies on 20 nodes, the results of which are described in section 3.6. Finally, in sections 3.7 and 3.8 we address the impact of peergroup size and hierarchy, by simulating several different hierarchic configurations of linear networks.

3.2 Boot and Resynchronization Time

We start by investigating the characteristics of network topology that influence boot and resynchronization times. The NodePeer protocol floods PNNI Topology State Elements over a link whenever the switches incident to the link have discrepant databases. Thus, information will flow from each switch radially until it has reached every other switch in the peergroup. Given this, network boot time and worst-case resynchronization times should be linearly dependent on the diameter of the peergroup.

Simulation Results We simulated PNNI networks of increasing size, specifically, chains, grids rings, and star networks. The tables (1,2,3,4) show the results of the simulations, and are depicted in figures 1 and 2. The figures indicate that PNNI boot time grows linearly in network diameter; for each unit increase in diameter PNNI boot time increases by approximately 21.4 ms, while resynchronization times increase by 10.7 ms.

Chain Length	**Network Diameter**	**Boot Time** seconds	**Resynch. Time** seconds
2	1	30.0864	0.0108
4	3	30.1290	0.0323
6	5	30.1719	0.0537
8	7	30.2146	0.0755
10	9	30.2588	0.0962
20	19	30.4562	0.2035
30	29	30.6859	0.3104

Table 1: Chains—Boot/resynch. times

Grid Size	**Network Diameter**	**Boot Time** seconds	**Resynch. Time** seconds
2x2	2	30.1073	0.0212
3x3	4	30.1498	0.0428
4x4	6	30.1926	0.0636
6x6	10	30.2785	0.1067

Table 2: Grids—Boot/resynch. times

Star Size	**Network Diameter**	**Boot Time** seconds	**Resynch. Time** seconds
5	2	30.0872	0.02172
10	2	30.0886	0.02179
15	2	30.0894	0.02188
20	2	30.0896	0.02191
25	2	30.0909	0.02198

Table 4: Stars—Boot/resynch. times

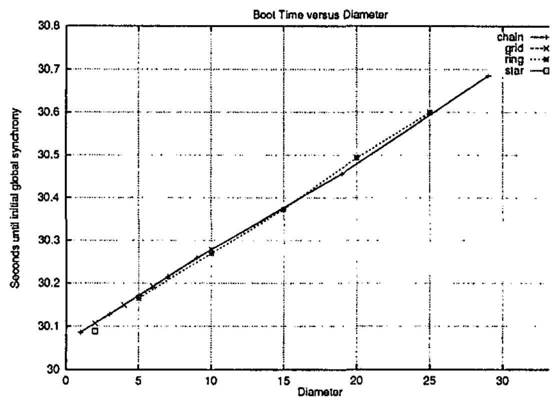


Figure 1: Boot time vs. network diameter

3.3 Boot and Resynchronization Traffic

To study PNNI boot and resynchronization traffic, we will introduce the notion of the representation size of a network: the minimum number of PNNI topology state elements required to fully describe its topology. In our subsequent discussion, we take the representation size of a network to be twice the number of links plus the number of switches.

Because the NodePeer protocol is responsible for transmission of the peergroup's current representation to each component switch, traffic required to reach *initial* global synchrony should be bounded below by a function proportional to the product of the representation size and the number of switches. For chains, grids and rings the number of nodes and the representation size are linearly related, so this product is quadratic in the representation size.

Resynchronization involves flooding updated information about links affected by the SVC, to all members of the peergroup. In the best-case, flooding takes place over a spanning tree and the total traffic required to resynchronize is proportional to the number of switches in the peergroup. In the worst case, when flooding is occurs over every edge in the network, traffic due to resynchronization is proportional to the number of links in the network. For this reason, we consider resynchronization traffic in PNNI networks as a function of representation size.

Simulation Results We interpret the traffic data collected from the simulations of the previous section. This data is shown in tables (5,6,7,8). The traffic required to reach initial global synchrony in each of these cases, grows super-linearly with the representation size, as can be seen in figures 3 and 4). Resynchronization traffic also manifests

Ring Length	**Network Diameter**	**Boot Time** seconds	**Resynch. Time** seconds
10	5	30.1654	0.0518
20	10	30.2694	0.1037
30	15	30.3732	0.1555
40	20	30.4932	0.2172
50	25	30.6002	0.2688

Table 3: Rings—Boot/resynch. times

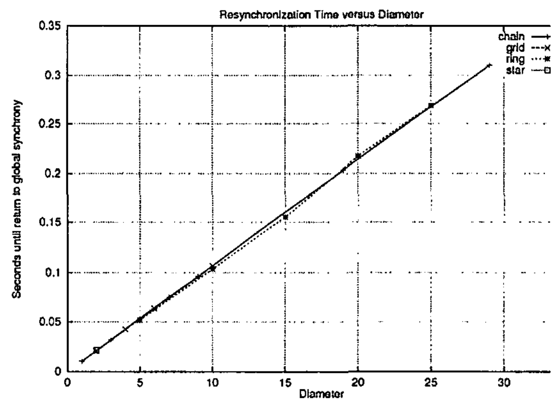


Figure 2: Resynch. times vs. network diameter

super-linear growth as a function of representation size.

Chain Length	Repres. Size	Boot Traffic bytes	Resynch. Traffic bytes
2	4	10292	2967
4	10	56028	13399
6	16	133976	30127
8	22	248124	53247
10	28	394476	82575
20	58	1629364	324487
30	88	3702444	719548

Table 5: Chains—Boot/resynch. traffic

Grid Size	Repres. Size	Boot Traffic bytes	Resynch. Traffic bytes
2x2	12	73284	14412
3x3	33	182232	43240
4x4	64	1674880	260640
6x6	156	3521512	816003

Table 6: Grids—Boot/resynch. traffic

Ring Length	Repres. Size	Boot Traffic bytes	Resynch. Traffic bytes
10	30	201812	2720
20	60	765388	5440
30	90	1379620	108160
40	120	2167876	816680
50	150	3353088	1286000

Table 7: Rings—Boot/resynch. traffic

3.4 General Tests I: All 7 Node Networks

In order confirm the validity of the above conclusions, we conducted the same experiments on the class of all (853 topologically distinct) 7 node connected graphs[1].

Simulation Results Because the 7 node networks all have relatively small diameter, there was little differentiation in their boot and resynchronization times. As the graphs in figures 5 and 6 indicate, boot and resynchronization times were clustered at discrete values spaced 10ms apart. We note that 10ms is the time required for processing of a single PNNI message at the control port of a switch.

[1]The software used to generate all non-isomorphic 7 node graphs was the NAUTY program developed by Brendan McKay.

Star Size	Repres. Size	Boot Traffic bytes	Resynch. Traffic bytes
5	16	30864	9144
10	31	129024	43976
15	46	292724	104776
20	61	526464	191624
25	76	820152	302496

Table 8: Stars—Boot/resynch. traffic

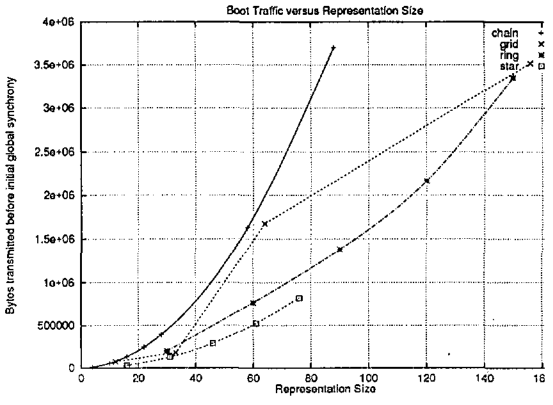


Figure 3: Boot traffic vs. representation size

While the plot does not immediately illustrate this, the distribution of the data points was not uniform over the cluster points; we have plotted the average as a function of diameter to emphasize this. Somewhat surprisingly, on average, resynchronization time seems to grows super-linearly with network diameter. More simulations need to be conducted over a larger class of graphs to determine the cause of this phenomenon. The PNNI boot traffic for these simulations is shown in figure 7, and is linear with an approximate growth rate of 6000 bytes per unit of representation.

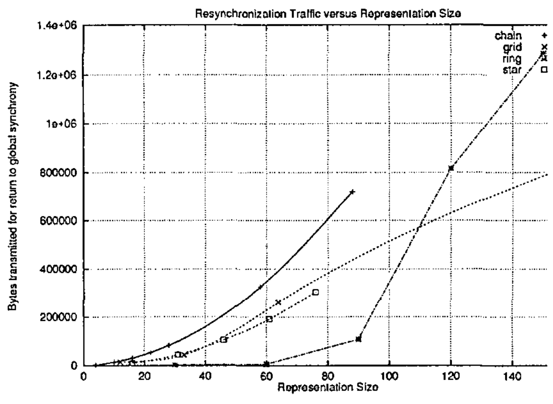


Figure 4: Resynch. traffic vs. representation size

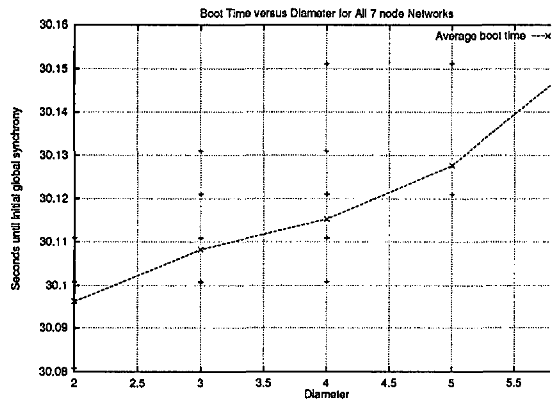


Figure 5: Boot times for 7 node networks

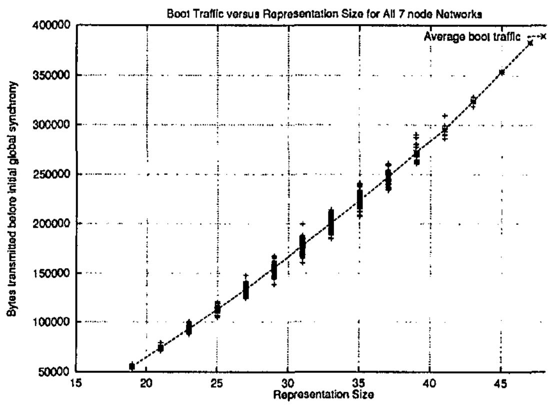


Figure 7: Boot traffic for 7 node networks

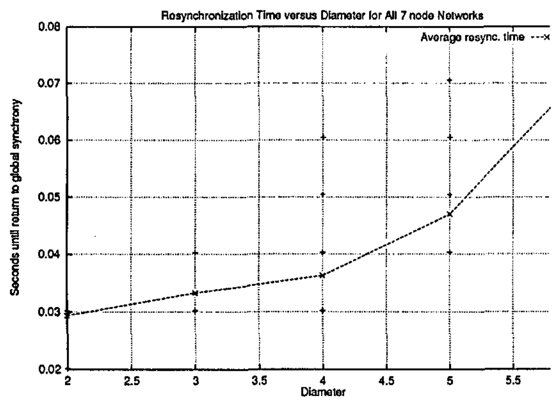


Figure 6: Resynch. times for 7 node networks

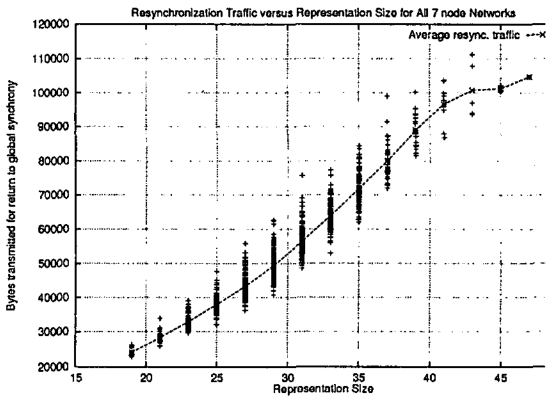


Figure 8: Resynch. traffic for 7 node networks

3.5 Network Geodesic Structure

The plot shown in figure 8 indicates that that over 7 node graphs of any fixed representation size, there is considerable variation in PNNI resynchronization traffic. We attempted to understand the cause for this differentiation by determining, for each representation size, which topologies achieved the highest and lowest resynchronization traffic. Figure 9 is a partial list of the best and worst performers (only those with representation sizes between 19 and 33 are shown).

We noted that the worst performers had a large number of redundant geodesics between pairs of switches. That is to say, topologies with the worst performance contained a multiplicity of shortest paths between nodes. Scrutiny of the simulations revealed that this multiplicity results in a redundant flooding in the PNNI NodePeer protocol. Figure 10 illustrates the redundant flooding of a PTSE along one edge when two nodes are connected by more than one shortest path. The fact that the worst performers in figure 9 contain many unchorded 4-cycles, whereas the best performers contained many triangles, supports this conclusion.

3.6 General Tests II: Randomly Generated 20 Node Networks

We also conducted measurements of boot and resynchronization parameters for randomly generated networks.

A random topology is generated as follows: 20 nodes are assigned random locations on a grid. Links are added via a random process that repeatedly generates a random node-pair and adds a link between them with probability that decays exponentially with the Euclidean distance between the two nodes (Waxman 1998). Links that will cause the degree of a node to exceed eight are rejected by the random process in order to keep the graph reasonably sparse. The process of adding links terminates when the graph is connected and every node has degree at least 2. In this manner, we generated 8000 random topologies on 20 nodes. These were then sorted into classes, based on the diameter of the generated network, and 10 networks were chosen randomly from each class.

Simulation Results The results of simulations of 100 random 20-node networks are shown in figures 11 and 12. The data indicates that for any given value of diameter, there is a significant variation in the resynchronization and boot times. We were not able to determine the topological structure property that is responsible for this variance, but

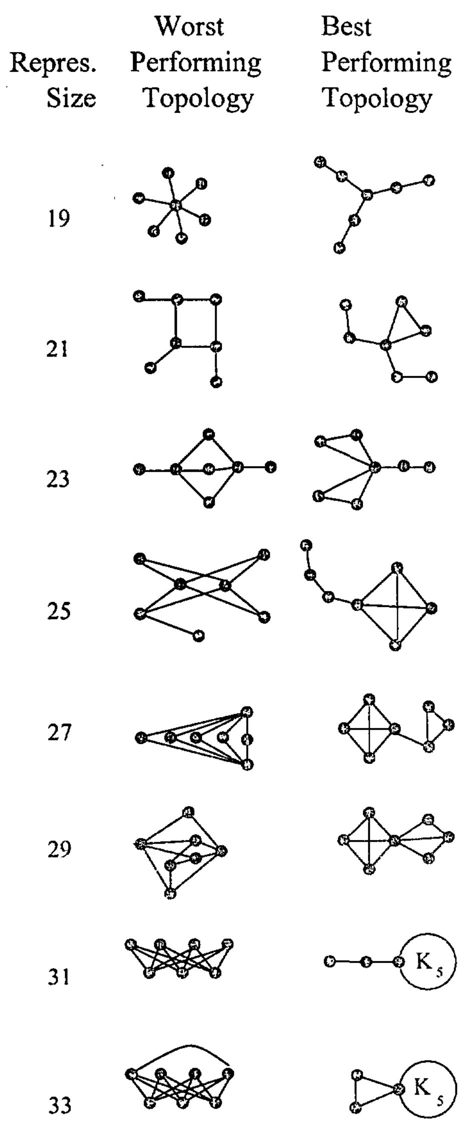


Figure 9: Best and worst 7 nodes topologies (in terms of boot traffic

hope to address the question in our future work. The results of previous sections indicate that if we are given two networks from the same restricted family (such as grids, for example) then diameter alone can serve to predict the approximate value of the resynchronization time. In the set of experiments presented in this section, we realize that, unfortunately, this simple estimation criterion does not hold as well over large mixed families of graphs (e.g. our random set). On the other hand, we remark that the average value of resynchronization and boot times (over the randomly chosen topologies) does increase linearly with diameter.

3.7 Peergroup Size

Another aspect of network design we consider is choice of peergroup size.

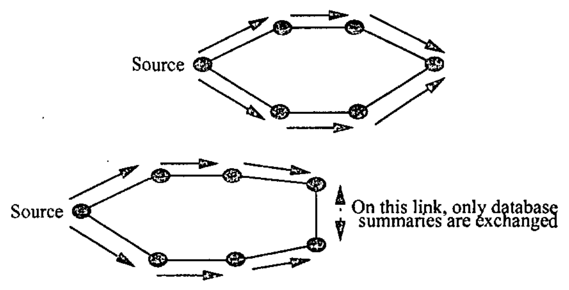


Figure 10: Effects of geodesic structure on traffic

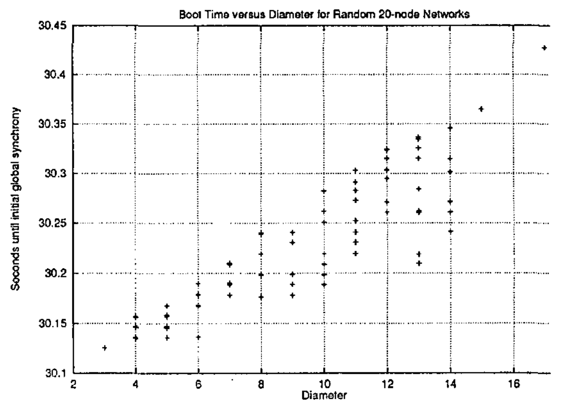


Figure 11: Boot times for 100 random 20-node networks

First, we note that partitioning a network into peergroups will result in more SVCs needing to be established between leaders at the next higher level. It also necessitates logical Hello finite state machines to stabilize over these SVCs and for the election process to conclude at the higher level. These two factors together are responsible for a discrete jump in the PNNI boot time when one moves from single peergroup to multiple peergroup configurations. As we begin to decrease the size of the peergroups, each requires less information to reach local synchrony, since detailed information about distant peergroups is being represented by a single logical node at the higher level. This causes boot traffic and resynchronization traffic to decrease. As we

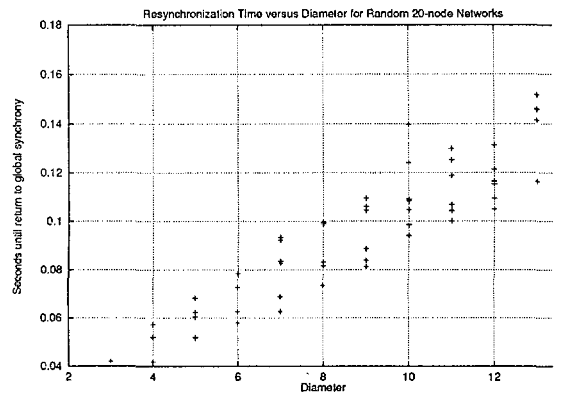


Figure 12: Resynch. times for 100 random 20-node networks

make peergroup size smaller still, an SVC setup on average traverses a larger number of peergroups, which in turn triggers re-aggregation of links at the higher level. Thus beyond a certain point, decreasing peergroup size causes an increase in resynchronization traffic and time.

Simulation Results We simulated chains of 16 and 32 switches, configured with peergroups of sizes 2,4 and 8 (the 16 node examples are illustrated in figures 13). The data from the simulations is presented in tables 9 and 10. In the 32 node chain, going from a single peergroup of 32 nodes to 4 peergroups of 8 nodes produces a jump in the PNNI boot time of over 59 seconds. On the other hand, boot traffic decreases by a factor of 3, and resynchronization traffic by a factor of 6.4. Reduction of the peergroup size from 4 to 2, causes an increase in resynchronization traffic by 30%. This happens because higher level links are re-aggregated and flooded downward in response to the change in resources at the lower level.

Topology name	16-16	16-4	16-2
# of PGs	1	4	8
PG size	16	4	2
Boot time	30.38s	90.26s	90.27s
Boot traffic	1034K	639K	587K
NNI boot traffic	0	1.5K	1.8K
Resync. traffic	0.30s	0.23s	0.24s
Resync. traffic	494K	156K	233K

Table 9: Varying peergroup size for a 16 node chain

Topology name	32-32	32-8	32-4	32-2
# of PGs	1	4	8	16
PG size	32	8	4	2
Boot time	30.73s	75.43s	75.39s	75.49s
Boot traffic	4217K	1401K	1121K	1420K
NNI boot traffic	0	3.1K	3.7K	3.9K
Resync. time	0.64s	0.46s	0.46s	0.51s
Resync. traffic	1981K	306K	689K	898K

Table 10: Varying peergroup size for a 32 node chain

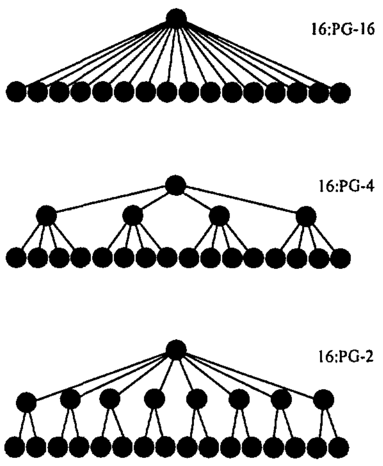


Figure 13: Examples of peergroups simulated.

3.8 Hierarchy

Finally, we consider how the presence of hierarchy affects our performance indicators.

First, we note that a network with many levels of hierarchy requires SVCs to be established between leaders in the same higher level peergroup. It also necessitates logical Hello finite state machines to stabilize over these SVCs and for the election process to conclude in each peergroup at each level. These two factors are the principal cause of the discrete jump seen in PNNI boot time and boot traffic as one considers networks with a greater number of levels. On the other hand, hierarchy localizes the side effects of network changes. In particular, it reduces the number of switches to which updated information must be flooded. Thus hierarchical addressing lowers both resynchronization time and traffic.

Simulation Results We simulated chains of 16 and 32 switches, configured with various hierarchical structure (The 16 node scenarios are depicted in figure 14). The data collected is presented in table 11. A 3-level configuration of the 32 switch chain boots in 75.4897 seconds, while the 5 level one requires 150.3220 seconds to reach initial global synchrony. The 5 level hierarchy has the advantage of improving both resynchronization time and traffic by a factor of 1.3 and 1.8 respectively.

4 Conclusion and Future Work

The simulations conducted using the PNNI Routing and Simulation Toolkit (PRouST) confirmed that topological characteristics such as the diameter, representation size, and geodesic structure do affect the boot and resynchronization times and traffic. These four indicators determine the discrepancy between switches' views of the network and its underlying physical state, and are thus critical to call admission rates and crankback frequency in ATM networks. Partitioning a network into peergroups reduces the

Topology name	16/3	16/5	32/3	32/5
Chain length	8	8	32	32
Hierarchy structure	16,8,1	16,8,4, 2,1	32,16,1	32,16, 8,4,1
Boot time	90.27s	120.32s	75.49s	150.32s
Boot traffic	587K	611K	1420K	1717K
NNI boot traffic	1.8K	3.1K	3.9K	6.8K
Resync. time	0.24s	0.16s	0.51s	0.38s
Resync. traffic	233K	215K	1191K	638K

Table 11: Simulation results for several hierarchy configurations.

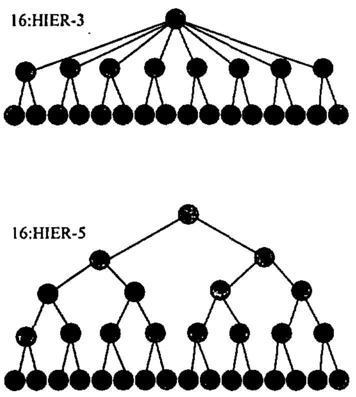


Figure 14: Two examples of hierarchic structures simulated.

boot and resynchronization traffic and introducing hierarchy results in improved resynchronization time and traffic, although it has a minor side-affect of increasing the boot time.

Peergroup size and hierarchy structure are two of the most importance choices a network designer must make. In our future research efforts will focus further on these two parameters. As we have shown in [section 3.7] there is an optimal value beyond which reduction of peergroup size results in increased resynchronization traffic, due to the re-aggregation and downward flooding of higher level links in response to changes at lower levels. We intend to precisely quantify both this optimal value, and the relative tradeoffs between peergroup size, hierarchy structure and resynchronization traffic and time.

5 Acknowledgements

We would like to thank David Talmage and Jack Marsh at the NRL Signaling Group for their contributions to PRouST, SEAN and CASiNO. Also we wish to acknowledge Sandeep Bhat for his assistance, particularly with the internals of the NodePeer finite state machine.

References

[1] ATM Forum. (1996) Private Network-Network Interface Specification, Version 1.0.

[2] Baruch Awerbuch, Yi Du, Bilal Khan, and Yuval Shavitt. (1998) Routing Through Networks with Hierarchical Topology Aggregation. *Journal of High Speed Networks*, vol. 7(1) p.57–73.

[3] W. C. Lee, Topology aggregation for hierarchical routing in ATM networks. *Computer Communications Review*, p. 82-92.

[4] S. Mountcastle, et al. (1999) CASiNO: A Component Architecture for Simulating Network Objects. *Proceedings of the 1999 Symposium on Performance Evaluation of Computer and Telecommunications Systems, July 11-15, 1999*. p. 261-272.

[5] D. Niehaus, et al. (1997) Performance Benchmarking of Signaling in ATM Networks. *IEEE Communications Magazine*, vol. 35 number 8, p. 134-142.

[6] Bernard M. Waxman. (1988) Routing of multipoint connections. *Journal on Selected Areas of Communications*, vol. 6 p. 1617-1622.

Performance Evaluation of A Scheduling Algorithm for Multiple Input-Queued ATM Switches

Ge Nong, Mounir Hamdi and Jogesh K. Muppala
Department of Computer Science
The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
E-mail: nong@ieee.org and {hamdi,muppala}@cs.ust.hk

Keywords: Input-queued ATM switch, analytical modeling and performance evaluation

Edited by: Mohsen Guizani

Received: December 25, 1998 **Revised:** September 16, 1999 **Accepted:** October 2, 1999

Performance analysis of a cell scheduling algorithm for an input-queued ATM switch is presented in this paper. The interconnection network of the ATM switch is internally nonblocking and each input port maintains a separate queue for each output port so as to reduce the head-of-line (HOL) blocking of conventional input queuing switches. Each input is allowed to send only one cell per time slot, and each output port is allowed to accept only one cell per time slot. The cells to be transmitted in a time slot are selected using a fast, fair and efficient scheduling algorithm called iSLIP *[10, 12, 13]. Using a tagged input queue approach, an approximate analytical model for evaluating the performance of the switch under IID Bernoulli traffic for different offered traffic loads is developed. The switch throughput, mean cell delay, and cell loss probability are computed from the analytical model. With the switch being under heavy traffic loads, the accuracy of the analytical model is verified using simulation.*

1 Introduction

The Asynchronous Transfer Mode (ATM) has been recommended by CCITT as an integrated transport technique for future broadband-ISDN. Consequently there is growing interest in identifying suitable switching architectures for ATM. Virtually all proposals of switching fabric architectures for ATM networks are based on self-routing structures in which each packet autonomously find its path through the interconnection network to the desired output port [3, 18]. This feature enables the design of switching fabrics characterized by a very high throughput, of the order of Gbits/sec.

These switches use various types of queueing strategies for cells: input queueing, dedicated internal queueing, shared internal queueing, or output queueing, each characterized by certain advantages and drawbacks [3]. The simplest approach among them is input queueing, where each input port maintains a FIFO queue of cells, and only the first cell in the queue is eligible for transmission during a given time slot. FIFO input queueing suffers from the well-known *head-of-line* (HOL) blocking, whereby when the cell at the head of the queue is blocked, all cells behind it in the queue are prevented from being transmitted, even though their destination output ports may be idle. It has been shown through analysis and simulation that HOL blocking limits the throughput of each input port to a maximum of 58.6% under uniform random traffic, and much lower for bursty traffic [7, 17].

Various approaches have been proposed to overcome the HOL problem: adopting a switch expansion, a windowing technique, or a channel grouping technique [3]. Using switch expansion, the internal switch bandwidth is expanded, using replication for example, so that it can transmit more than one cell to an output port in a single time slot. The windowing technique allows non-HOL cells to contend for the switch output ports. Given a window of depth w, if a cell in position i, $1 \leq i \leq w$, ($i = 1$ identifies the HOL position) is blocked owing to a conflict for the switch output port, a chance is given to the cell in position $i + 1$, until a search of depth w has been completed. Using channel grouping, the switch output ports are subdivided into groups of size R each and cells address groups, not single links of a group. Such arrangement is typical of a transit environment in which the switch interfaces a limited number of downstream switches with R links to each of them. Implementing channel groups in an ATM switch is a nontrivial task given the high rates of the links, since in each slot not only do we have to guarantee the absence of external conflicts, but we also need to allocate at transmission time each output port in a group to a specific input port addressing that group with its HOL cell.

Another approach that is receiving a lot of attention from the research community for reducing the HOL problem is for each input port to maintain a separate queue for cells destined to each output port [2, 8, 10, 11, 14]. During a single time slot, a maximum of one cell per input port can be transferred, and a maximum of one cell per output port can be received. The selection of the cell from an input to transmit during a time slot to an output is accomplished using a scheduling algorithm. Of particular interest to us in this paper are the iterative matching algorithms, namely, *parallel iterative matching* (PIM) [2] and iSLIP [10, 11]. The purpose of the iterative matching algorithm is to find a maximal matching between the inputs and outputs of the switch. It is shown that the number of iterations needed to find a maximal matching is N in the worst case, and $O(logN)$ in the average case. However it was shown through simulation that with as few as 4 iterations, the throughput of the switch exceeds 98% [16]. As a result, these switch architectures have received a lot of attention from the research community, and commercial switches based on this technique have already been built such as the DEC Systems AN2 switch [2] and the Tiny Tera [12].

Most of the performance studies of these switches appearing in the literature have been based on simulation. The only exception is our previous work on modeling these switches under the PIM scheduling algorithm. The main reason for that is the fact that mathematically analyzing the PIM and iSLIP based ATM switches is difficult. The inventors of PIM switches acknowledge this fact by saying, "In general, it is difficult to mathematically analyze the performance of a PIM switch, even for the simplest traffic models. The problem lies in the evolution and interdependence of the state of each arbiter and their dependence on arriving traffic" [2, 10].

Performance parameters such as throughput, mean cell delay, and cell loss probability are important to evaluate the performance of an ATM switch. For example, a cell loss probability requirement of 10^{-9} is not uncommon. As a result, estimating the rare cell loss probability by simulation is inefficient and sometimes impossible [9]. Consequently, analytical models are of great importance in solving these problems. Recently we have developed an analytical performance model of PIM switches [16]. However, it was shown that iSLIP scheduling is superior to PIM scheduling in terms of fairness, throughput and hardware cost [13]. As a result, we build on our work in [16] to accomodate the iSLIP scheduling algorithm.

In this paper we develop an analytical model of the iSLIP switch with finite input buffers using the *tagged queue* approach [1, 6]. Assuming an IID Bernoulli input traffic, with the cell destinations distributed uniformly over all the outputs, we compute interesting performance measures for the switch including throughput, mean cell delay, and cell loss probability as a function of the offered load and switch size. With the switch being under heavy traffic loads, the accuracy of our analytical model has been verified using simulation. (When traffic loads are light, cell loss events are so rare that estimating it by ordinary direct simulation techniques is practically prohibited. In case of this, efficient/fast simulation techniques such as *importance sampling* may need to be explored to accomplish this job [20].) The results from the analytical model match closely with the simulation results. This model will be extended in the future under more realistic traffic assumptions such as bursty and correlated traffic. This will render our mathematical analysis more complex. We consider the results of this paper as the foundation for solving the more difficult problem involving bursty and correlated traffic.

The remainder of this paper is organized as follows. Section 2 introduces the switch model as well as PIM and iSLIP scheduling algorithms. Section 3 develops the analytical model based on the tagged queuing approach. Equations for computing interesting performance measures including throughput, mean cell delay, and mean cell loss probability are derived in this section. Numerical results obtained from the analytical model are presented for switches of different sizes in Section 4, and compared with the results from simulation. Finally conclusions are presented in Section 5.

2 The Switch Model and iSLIP Scheduling

In this section, we present an overview of the ATM switch architecture and describe the iSLIP switch scheduling algorithm. Interested readers may refer to [10, 11, 12] for further details.

2.1 The Switch Model

The ATM switch under consideration is an $N \times N$ nonblocking switch, i. e., the N inputs are connected to the N outputs via a nonblocking interconnection network (e.g., crossbar switch). Each input queue of the switch is a random access buffer which can be viewed as N FIFO queues, each of which is used to store the cells destined for one of the N output ports. The architecture of this switch is shown in Figure 1. The first cell in each queue can be selected for transmission across the switch in each time slot, with the following constraints:

1. Only one cell from any of the N queues in an input port can be transmitted in each time slot.
2. Only one cell can be transmitted from any of the N input ports to an output port of the switch at any given time slot. In other words, at most one cell could received by a single output port.

As mentioned above, the reason for using a random-access buffer, instead of a conventional FIFO queue, is to avoid the HOL blocking. Two criteria must be considered for designing the switch: (i) the switch must route as many cells among the ones that arrived at its inputs as possible to the appropriate outputs to maximize the throughput, and (ii) the switch must solve the output contention problem (i.e., more than one cell can be destined for the same output port). As a result, the switch scheduling algorithm that decides, for each time slot, which inputs transmit their queued cells to which outputs is of paramount importance. Two

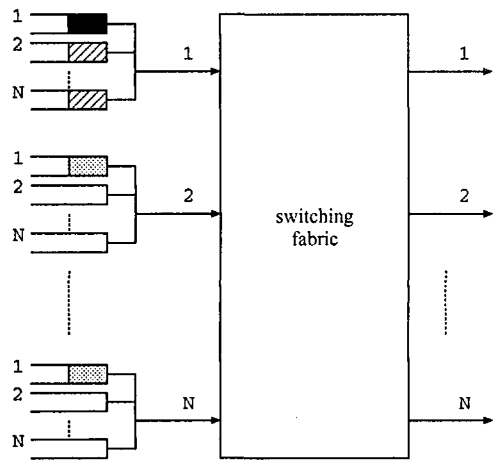


Figure 1: Architecture of the multiple input queues ATM switch ($N \times N$).

such effective algorithms include parallel iterative matching [2] and *i*SLIP [10, 12].

2.2 Parallel Iterative Matching (PIM) and *i*SLIP Algorithms

For switches used in high-performance ATM networks, the switch scheduling algorithm must be able to provide high throughput, low latency, and graceful degradation under heavy traffic loads. Anderson *et al.* [2] considered the architecture of a nonblocking switch with random-access buffers (as shown in Figure 1), and cast the switch scheduling problem as a bipartite matching problem of finding conflict-free pairing of inputs to outputs.The high throughput and low latency of an ATM switch dictates that the scheduling algorithm must be able to find a matching of as many conflict-free pairings as possible using as little time as possible. Note that a *maximum matching* is a matching with the maximum number of paired inputs and outputs; while *maximal matching* is one in which no unmatched input has a queued cell destined for an unmatched output (i.e., no parings can be trivially added). Unfortunately, all the conventional bipartite maximum matching algorithms have a high time complexity with regard to the time constraint imposed by 53-byte ATM cells and Gbps links. As a solution, Anderson *et al.* [2] proposed an algorithm, called parallel iterative matching, to find a maximal matching. This algorithm uses parallelism, randomness, and iteration to find a maximal matching between the inputs that have queued cells for transmission and the outputs that have queued cells (at the inputs) destined for them. Maximal matching is used to determine which inputs transmit cells over the nonblocking switch to which outputs in the next time slot. Specifically, their matching algorithm iterates the following three steps until a maximal matching is found or until a fixed number of iterations is performed:

1. *Request*: Each unmatched input sends a request to *every* output for which it has a queued cell.

2. *Grant*: If an unmatched output receives any requests, it grants to one by *randomly* selecting a request uniformly over all requests.

3. *Accept*: If an input receives grants, it accepts one by randomly selecting an output among those that granted to this input.

By considering only unmatched inputs and outputs, each iteration only considers connections not made by earlier iterations. Note that in step (2) above the independent output schedulers *randomly* select a request among contending requests. This has three effects: First, Anderson *et. al.* [2] show that each iteration will match or eliminate on average at 3/4 of the remaining possible connections and thus the algorithm will converge to a maximal match in $O(\log N)$ iterations. Second, it ensures that all requests will eventually be granted. Consequently, no input queue is starved. Third, it means that no memory or state is used to keep track of how recently a connection was made in the past.

McKeown [10] proposed an iterative algorithm named *i*SLIP which avoids the use of randomness while making the selections, and instead uses a varying priority based on state information that is maintained from one time slot to the next [10]. The *i*SLIP was shown to outperform PIM in terms of fairness, throughput and hardware cost [13]. The *i*SLIP algorithm iterates over the following three steps:

1. *Request*: Each unmatched input sends a request to every output for which it has a queued cell.

2. *Grant*: Each output that receives any request selects the one with the highest priority according to its *grant arbiter*. The output notifies each input whether or not its request was granted. The pointer to the highest priority element of the round-robin schedule is incremented (modulo N) to one location beyond the granted input if and only if the grant is accepted in step 3 of the firsı iteration.

3. *Accept*: Each input that receives any grants accepts the one with the highest priority according to its *accept arbiter*. The pointer to the highest priority element of the round-robin scheduler is incremented (modulo N) to one location beyond the accepted output.

Due to the complex behaviors involved in the arbitration processes, it is extremely difficult to mathematically analyze the *i*SLIP switches. Based on experience from our earlier research on modeling PIM switches and simulation results, we propose a variant of the *i*SLIP algorithm called 2PPIM (*2-Priorities PIM*) algorithm to model *i*SLIP. The 2PPIM algorithm closely approximates the *i*SLIP algorithm and we can mathmatically model it. As we can see from [10], *i*SLIP switches behave like PIM switches under low traffic load. This is because when the load is low, *the grant/accept arbiters* have a greater chance of being synchronized with one another. But when the traffic load is high, *the grant/accept arbiters* are desynchronized most of the time so as to outperform PIM. In view of this observation, we introduce the following scheduling algorithm to incorporate the effects of prioritization by the *grant/accept arbiters*.

Every input has a round-robin arbiter. Initially, the values of these arbiters are set to be the number of the corresponding input. For example, the arbiter of input i will has value i at time 0. At the end of every time slot, the values of these arbiters are incremented by 1 (modulo N). So at any time, no two arbiters will have the same value. Using the arbiters, a set of conflict-free matchings can be guaranteed. In the first iteration, for any input which has cells in the queue pointed to by its arbiter, both the input and its destination output will get matched. After the first iteration, the original PIM is applied to the unmatched inputs/outputs. That is, the queues of an input are separated into two classes by the round-robin arbiter: one high priority queue and $N-1$ low priority queues. The high priority queues at all the inputs will not have a common output destination, and hence there will be no conflict. For the low priority queues, random selections are made by the original PIM.

3 Queueing Model and Analysis of 2 Iterations PPIM

In this section, we model the ATM switch with 2PPIM scheduling using queueing theory and analyze the underlying Markov chain. Our method uses the concept of *tagged queues* in modeling the switch leading to a smaller state space. The concept of *tagged input queue* has been successfully used to evaluate the FIFO input-queued switch model [1, 6]. These switches involve a single stage of contention resolution. On the other hand, for the switch with PPIM scheduling, the contention resolution process consists of two stages. As observed from the algorithm descriptions of PPIM, a HOL cell in an input queue will contend for transmission not only with the HOL cells of the same input, but also the HOL cells destined for the same output. The corresponding model is more complicated than for the FIFO input-queued switch. We make the following assumptions in developing the PPIM switch model:

1. The switch operates synchronously.
2. Every input queue has the same buffer size, namely b_i.
3. Cells arrive at every input queue according to an IID Bernoulli process with a probability λ. The switch load is thus $N\lambda$.
4. New cells arrive only at the beginning of the time slots, and cells depart only at the end of the time slots.

Under the above assumptions, all the input queues will exhibit the same behavior when the system attains steady state. A queue at input i with output j as the destination is denoted by $Q(i,j)$. Figure 2 shows an example of the queueing model for the PPIM switch. In this example, the occupancy of $Q(1,1)$ is taken as the *tagged input queue* and the number of HOL cells at input 1 is represented by the *1st HOL input queue*. The *HOL input queue* is a virtual queue which does not exist in a real PPIM switch but is useful for our mathematical analysis.

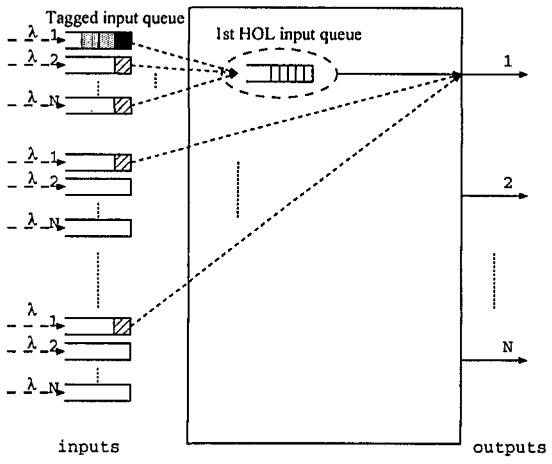


Figure 2: An example of the queueing model for the PIM switch.

3.1 Markov Model

We analyze the queueing model by considering the underlying Markov chain Z. The states of the Markov chain are sampled at the end of the time slots and can be expressed as a quartet (l,w,g,s), where l is the *length of tagged input queue Q(i,j)*, w is the number of *non-empty queues in input* i (including $Q(i,j)$ and Q_H, where Q_H is the high priority queue), g is the *priority of Q(i,j)* (0-high, 1-low), and s the *state of* Q_H(0 for empty and 1 for non-empty). The state-space of this four-dimensional Markov chain is

$$\{(0,0,0,0),(l,w,g,s) \quad | \quad 1 \le l \le b_i, 1 \le w \le N, \; g \in \{0,1\}, s \in \{0,1\}\}$$

and is ordered in lexicographic order, that is, (0,0,0,0), (1,1,0,1), (1,1,1,0),(1,1,1,1),(1,2,0,1),... $(b_i,N,1,1)$. The set of states {(1,1,0,1), (1,1,1,0),...(1,2,0,1), (1,2,1,0),...(1,N,1,1)} are labelled as states in level l of the Markov chain. This Markov chain has two important attributes: (1) when moving from state (l,w,g,s) to a state $(l+i,w',g',s')$, for $i \ge 1$, the chain must visit all intermediate levels at least once, and (2) it is a *Quasi Birth and Death (QBD)* process with block-partitioned form of transition probability matrix T which is given as follows:

$$T = \begin{bmatrix} A'_1 & A'_2 & 0 & \cdots & & & \\ A'_0 & A_1 & A_2 & 0 & \cdots & & \\ 0 & A_0 & A_1 & A_2 & 0 & \cdots & \\ 0 & 0 & A_0 & A_1 & A_2 & 0 & \cdots \\ \vdots & \vdots & \vdots & \cdots & & & \vdots \\ 0 & 0 & \cdots & 0 & A_0 & A_1 & A_2 \\ 0 & 0 & \cdots & 0 & 0 & S & B \end{bmatrix}$$

where $A'_1 + A'_2 e = 1$ and $A'_0 + A_1 e + A_2 e = (A_0 + A_1 + A_2)e = e$ with $e = [1,1,1,\ldots,1]^T$. Let $P_{blo,W_t(w'_i,g',s')|W_{t-1}(w_i,g,s)}$ denote the probability that the HOL cell of the tagged queue is blocked, and $P_{suc,W_t(w'_i,g',s')|W_{t-1}(w_i,g,s)}$ denote the probability that the HOL cell of the tagged queue is transmitted given that the state of the remaining HOL cells at the end of the last time slot is (w_i, g, s) and the state of remaining HOL cells at the end of the current time slot is (w'_i, g', s'). Let's define B and B_0 as:

$$B = \begin{bmatrix} P_{blo,W_t(1,0,1)|W_{t-1}(1,0,1)} & P_{blo,W_t(1,1,0)|W_{t-1}(1,0,1)} & \cdots & P_{blo,W_t(N,1,1)|W_{t-1}(1,0,1)} \\ P_{blo,W_t(1,0,1)|W_{t-1}(1,1,0)} & P_{blo,W_t(1,1,0)|W_{t-1}(1,1,0)} & \cdots & P_{blo,W_t(N,1,1)|W_{t-1}(1,1,0)} \\ P_{blo,W_t(1,0,1)|W_{t-1}(1,1,1)} & P_{blo,W_t(1,1,0)|W_{t-1}(1,1,1)} & \cdots & P_{blo,W_t(N,1,1)|W_{t-1}(1,1,1)} \\ \vdots & \vdots & \cdots & \vdots \\ P_{blo,W_t(1,0,1)|W_{t-1}(N,1,1)} & P_{blo,W_t(1,1,0)|W_{t-1}(N,1,1)} & \cdots & P_{blo,W_t(N,1,1)|W_{t-1}(N,1,1)} \end{bmatrix}$$

and

$$B_0 = [P_{blo,W_t(1,0,1)|W_{t-1}(0,0,0)},\ P_{blo,W_t(1,1,0)|W_{t-1}(0,0,0)},\ \cdots, P\ _{blo,W_t(N,1,1)|Wt-1(0,0,0)}].$$

Similarly, we define S as:

$$S = \begin{bmatrix} P_{suc,W_t(1,0,1)|W_{t-1}(1,0,1)} & P_{suc,W_t(1,1,0)|W_{t-1}(1,0,1)} & \cdots & P_{suc,W_t(N,1,1)|W_{t-1}(1,0,1)} \\ P_{suc,W_t(1,0,1)|W_{t-1}(1,1,0)} & P_{suc,W_t(1,1,0)|W_{t-1}(1,1,0)} & \cdots & P_{suc,W_t(N,1,1)|W_{t-1}(1,1,0)} \\ P_{suc,W_t(1,0,1)|W_{t-1}(1,1,1)} & P_{suc,W_t(1,1,0)|W_{t-1}(1,1,1)} & \cdots & P_{suc,W_t(N,1,1)|W_{t-1}(1,1,1)} \\ \vdots & \vdots & \cdots & \vdots \\ P_{suc,W_t(1,0,1)|W_{t-1}(N,1,1)} & P_{suc,W_t(1,1,0)|W_{t-1}(N,1,1)} & \cdots & P_{suc,W_t(N,1,1)|W_{t-1}(N,1,1)} \end{bmatrix}.$$

S_0 is the probability that the HOL cell of the *tagged input queue* gets matched given that the *tagged input queue* is empty at the end of last time slot. From the definitions of B_0, S_0, S, and B, we can show that:

$$S_0 + B_0 e = 1 \qquad (1)$$

$$S_c + B_c = e \qquad (2)$$

where $S_c = Se$, $B_c = Be$. As illustrated in the Appendix of this paper, Eq (1) and Eq (2) help us analyze the problem by simply focusing the computation on matrix B and vector B_0. By using the above equations and denoting $\bar{\lambda} = 1 - \lambda$, the element matrices in the transition probability matrix T can be computed as:

$$\begin{array}{ll} A'_0 = \bar{\lambda} S_c & A'_1 = \bar{\lambda} + \lambda S_0 \\ A'_2 = \lambda B_0 & A_0 = \bar{\lambda} S \\ A_1 = \lambda S + \bar{\lambda} B & A_2 = \lambda B \end{array}$$

The remaining subsections will cover the computation of the success and blocking probabilities, $P_{suc,W_t(w'_i,g',s')|W_{t-1}(w_i,g,s)}$, and $P_{blo,W_t(w'_i,g',s')|W_{t-1}(w_i,g,s)}$ respectively. Once these probabilities are computed, the transition probability matrix T can be constructed. Once the transition probability matrix is known, it is a routine matter to derive the steady state equations by utilizing the properties of Markov chains, and solving the equations to obtain the steady-state probability vector. Detailed procedures using the *matrix geometric approach* [15] are presented in the appendix of this paper.

3.2 Computing the Blocking and Success Probabilities

We now derive the equations for computing the blocking probability $P_{blo,W_t(w'_i,g',s')|W_{t-1}(w_i,g,s)}$ and the success probability $P_{suc,W_t(w'_i,g',s')|W_{t-1}(w_i,g,s)}$.

The transition of the state of the virtual HOL input queues from the state (w_i, g, s) to state (w'_i, g', s') is a two step process:

1. First, we account for the newly arriving HOL cells to the virtual HOL input queue.
2. Then, we consider the transition from the intermediate state to the final state after applying the 2PPIM algorithm.

This process is illustrated in Figure 3.

3.2.1 Arriving Cells at the Virtual HOL Queues

Let $K_t(k_i)$ denote the number of newly arriving HOL cells at the *virtual HOL input queue* (k_i new arrivals to the *virtual HOL input queue*), at the beginning of current time slot

$$\begin{tikzcd} w_i \arrow[r, "\text{Arriving HOL cells } k_i"'] & h_i \arrow[r, "\text{PIM algorithm to find maximal matching}"'] & w'_i \end{tikzcd}$$

Figure 3: Transition of the virtual HOL queues.

$Q(i,j)$	w'_i	Probability
blocked	h_i	$P_{blo\_0\vert H(h_i,g,s)}$
	$h_i - 1$	$P_{blo\_1\vert H(h_i,g,s)}$
successful	$h_i - 1$	$P_{suc\vert H(h_i,g,s)}$

Table 1: Transitions from (h_i, g, s) to (w'_i, g', s).

t. $W_{t-1}(w_i)$ denotes the numbers of remaining HOL cells at the *virtual HOL input queue* (w_i is the length of *virtual HOL input queue*), at the end of previous time slot $t-1$. Let $H_t(h_i) = K_t(k_i) + W_{t-1}(w_i)$. Define, $a_{K(k_i)|W(w_i)}$ = *Prob($K_t(k_i)|W_{t-1}(w_i)$)*. Let p_0 be the probability that a queue is empty in a time slot, and $p_1 = 1 - p_0$. A cell that arrives at $Q(i,j)$ when $Q(i,j)$ is empty, will observe that another queue is non-empty with probability p_1. If the current state is (l, w_i, g, s), $(N - w_i)$ queues of input i will be non-empty with probability p_1. Hence,

$$a_{K(k_i)|W(w_i)} = \begin{cases} \binom{N-1}{k_i-1} p_1^{k_i-1} p_0^{N-k_i} \\ 1 \leq k_i \leq N,\ w_i = 0 \\ \\ \binom{N-w_i}{k_i} \lambda^{k_i} (1-\lambda)^{N-(k_i+w_i)} \\ 0 \leq k_i \leq N - w_i,\ 1 \leq w_i \leq N. \end{cases}$$

3.2.2 Transition to $W_t(w'_i, g', s')$

For the HOL cell of the tagged input queue, its contention process can be split into two stages. In the first stage, the tagged input queue contends with other non-empty queues at the same input. If it succeeds in the first stage contention, it joins the second stage contention with all successful jth queues from other inputs.

Given $H_t(h_i)$ and $W_t(w'_i)$, the possible values of W_t in terms of H_t are:

$$w'_i = \begin{cases} h_i \\ h_i - 1, & for\ h_i > 0 \end{cases}$$

We define the following probabilities associated with the above transitions:

$P_{blo\_0|H(h_i,g,s)}$ = *Prob{the HOL cell at the tagged input queue gets blocked, and $W_t(w'_i) = H_t(h_i)$ given $H_t(h_i, g, s)$}*

$P_{blo\_1|H(h_i,g,s)}$ = *Prob{the HOL cell at the tagged input queue gets blocked, and $W_t(w'_i) = H_t(h_i - 1)$ given $H_t(h_i, g, s)$}*

$P_{suc|H(h_i,g,s)}$ = *Prob{the HOL cell at the tagged input queue gets transmitted, and $W_t(w'_i) = H_t(h_i - 1)$ given $H_t(h_i, g, s)$}*

Table 1 summarizes the transitions and their corresponding probabilities.

Given $d_i = w'_i - w_i$, the blocking probability $P_{blo,W_t(w'_i,g',s')|W_{t-1}(w_i,g,s)}$ is computed as:

$$P_{blo,W_t(w'_i,g',s')|W_{t-1}(w_i,g,s)} =$$

$$\begin{cases} 0,\ for\ g = 0\ or\ d_i < -1 \\ \\ a_{K(d_i+1)|W(w_i)} P_{blo\_1\_(g',s')|H(w'_i+1,g,s)} P_{l1} \\ +a_{K(d_i)|W(w_i)} P_{blo\_1\_(g',s')|H(w'_i,g,s)} (1 - P_{l1}), \\ for\ d_i \geq -1\ and\ g = 1\ and\ s = 1 \\ \\ \lambda a_{K(d_i)|W(w_i+1)} P_{blo\_1\_(g',s')|H(w'_i+1,g,1)} \\ +\bar{\lambda} a_{K(d_i)|W(w_i+1)} P_{blo\_0\_(g',s')|H(w'_i,g,s)} \\ +\bar{\lambda} a_{K(d_i+1)|W(w_i+1)} P_{blo\_1\_(g',s')|H(w'_i+1,g,s)} \\ \quad \cdot (d_i + 1 + l_0(w_i - 1))/w'_i \\ +\bar{\lambda} a_{K(d_i)|W(w_i+1)} P_{blo\_1\_(g',s')|H(w'_i,g,s)} \\ \quad \cdot (w_i - 1 - l_0(w_i - 1))/(w'_i - 1), \\ for\ d_i \geq -1\ and\ g = 1\ and\ s = 0 \end{cases} \tag{3}$$

in which

$$l_0(w) =$$

$$\begin{cases} 0, & for\ w = 0 \\ \sum_{u=1}^{w} \begin{pmatrix} w \\ u \end{pmatrix} P_{l1}^u (1 - P_{l1})^{w-u}, & for\ w > 0 \end{cases} \tag{4}$$

represents the number of input queues that contain only one buffered cell, and P_{l1} in Eq (4) is the probability that an input queue length is equal to 1 (there is only one buffered cell in this input queue) during a time slot, and is given by:

$$P_{l1} = (1 - \lambda)\pi_1 e / (1 - \pi_0)$$

To compute the state transition probabilities of the high priority queue, that is, from (g, s) to (g', s'), Figure 4 is useful. Figure 4(a) shows the transition probabilities for the priority of $Q(i,j)$, and Figure 4(b) gives the transition probabilities for the high priority queue of input i from

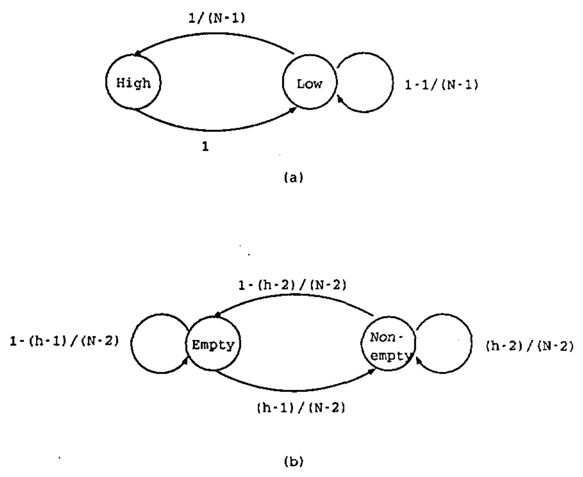


Figure 4: The state transition diagram for the high priority queue.

empty to non-empty and vice versa.

$$P_{blo\_0\_(g',s')|H(h,g,s)} = \begin{cases} 0,\ for\ s=1 \\ P_{blo\_0|H(h)}\frac{1}{N-1},\ for\ g'=0\ and\ s=0 \\ P_{blo\_0|H(h)}(1-\frac{1}{N-1})(1-\frac{h-1}{N-2}), \\ \quad for\ g'=1\ and\ s'=0 \\ P_{blo\_0|H(h)}(1-\frac{1}{N-1})(\frac{h-1}{N-2}), \\ \quad for\ g'=1\ and\ s'=1 \end{cases} \tag{5}$$

$$P_{blo\_1\_(g',s')|H(h,g,s)} = \begin{cases} \frac{1}{N-1},\ for\ s=1\ and\ g'=0 \\ (1-\frac{1}{N-1})(1-\frac{h-2}{N-2}), \\ \quad for\ s=1\ and\ g'=1\ and\ s'=0 \\ (1-\frac{1}{N-1})(\frac{h-2}{N-2}), \\ \quad for\ s=1\ and\ g'=1\ and\ s'=1 \\ P_{blo\_1|H(h)}\frac{1}{N-1},\ for\ s=0\ and\ g'=0 \\ P_{blo\_1|H(h)}(1-\frac{1}{N-1})(1-\frac{h-1}{N-2}), \\ \quad for\ s=0\ and\ g'=1\ and\ s'=0 \\ P_{blo\_1|H(h)}(1-\frac{1}{N-1})(\frac{h-1}{N-2}), \\ \quad for\ s=0\ and\ g'=1\ and\ s'=1 \end{cases} \tag{6}$$

For $W_{t-1}(w_i,g,s)=(0,0,0)$, the blocking probability $P_{blo,W_t(w_i',g',s')|W_{t-1}(0,0,0)} =$

$$\begin{cases} P_{r0}(w_i'+1,1,1)P_{blo\_1\_(g',s')|(w_i'+1,1,1)}P_{l1}' \\ +P_{r0}(w_i',1,1)P_{blo\_1\_(g',s')|(w_i',1,1)}(1-P_{l1}') \\ +P_{r0}(w_i',1,0)P_{blo\_0\_(g',s')|(w_i',1,0)} \\ +P_{r0}(w_i',1,0)P_{blo\_1\_(g',s')|(w_i',1,0)}(1-\frac{l_0(w_i'-1)}{w_i'-1}) \\ +P_{r0}(w_i'+1,1,0)P_{blo\_1\_(g',s')|(w_i'+1,1,0)}\frac{l_0(w_i')}{w_i'} \end{cases} \tag{7}$$

where

$P_{r0}(w_i,g,s) =$

$$\begin{cases} a_{K(w_i)|W(0)}/N,\ for\ g=0 \\ a_{K(w_i)|W(0)}(N-w_i)/N,\ for\ g=1\ and\ s=0 \\ a_{K(w_i)|W(0)}(w_i-1)/N,\ for\ g=1\ and\ s=1 \end{cases}$$

and the function P_{l1} in Eq(4) is replaced by P_{l1}':

$$P_{l1}' = (\lambda\pi_0 + (1-\lambda)\pi_1 e)/p_1$$

Finally, We can compute the probability $P_{suc,W_t(w_i',g',s')|W_{t-1}(w_i,g,s)}$ as follows:

$P_{suc,W_t(w_i',g',s')|W_{t-1}(w_i,g,s)} =$

$$\begin{cases} 0,\ for\ d_i<0\ or\ (g=1\ and\ s=1)\ or \\ \quad (g=0\ and\ g'=0) \\ a_{K(d_i)|W(w_i)}(1-\frac{w_i'-1}{N-1}) \\ ,for\ d_i>=0\ and\ s=0\ and\ g=0\ and \\ \quad g'=1\ and\ s'=0 \\ a_{K(d_i)|W(w_i)}(\frac{w_i'-1}{N-1}) \\ ,for\ d_i>=0\ and\ s=0\ and\ g=0\ and \\ \quad g'=1\ and\ s'=1 \\ \bar{\lambda}a_{K(d_i)|W(w_i+1)}P_{suc|H(w_i')}/(N-1) \\ ,\ for\ d_i>=0\ and\ s=0\ and \\ \quad g=1\ and\ g'=0 \\ \bar{\lambda}a_{K(d_i)|W(w_i+1)}P_{suc|H(w_i')}(1-\frac{1}{N-1})(1-\frac{w_i'-1}{N-2}) \\ ,\ for\ d_i>=0\ and\ s=0\ and\ g=1\ and \\ \quad g'=1\ and\ s'=0 \\ \bar{\lambda}a_{K(d_i)|W(w_i+1)}P_{suc|H(w_i')}(1-\frac{1}{N-1})(\frac{w_i'-1}{N-2}), \\ for\ d_i>=0\ and\ s=0\ and\ g=1\ and \\ \quad g'=1\ and\ s'=1 \end{cases} \tag{8}$$

3.2.3 Applying the 2PPIM Algorithm

We now compute the probabilities in Table 1 by considering each iteration of the 2PPIM scheduling algorithm. The state of the switch at the beginning of each iteration ϕ is characterized by the following parameters:

$n(\phi)$: the number of inputs/outputs remain unmatched;

$h(\phi)$: the number of outputs whose corresponding queue in input i is non-empty;

$r_f(\phi)$: the number of inputs whose outputs corresponding to their high priority queues remain unmatched and the corresponding queues in input i are non-empty (including the *tagged input queue*);

$r_e(\phi)$: the number of inputs whose outputs corresponding to their high priority queues remain unmatched and the corresponding queues in input i are empty;

$r_i(\phi)$: 1 if the output corresponding to the high priority queue of input i remains unmatched, and 0 if the output corresponding to the high priority queue of input i has been matched;

$r_j(\phi)$: 1 if the input whose output corresponding to its high priority queue is j remains unmatched, and 0 if the input whose output corresponding to its high priority queue is j has been matched;

At the end of the iteration the following parameters can be defined:

$$\begin{aligned}
m(\phi)i&=n(\phi) - n(\phi+1)\\
\Delta h_i(\phi)&=h_i(\phi) - h_i(\phi+1)\\
\Delta r_f(\phi)&=r_f(\phi) - r_f(\phi+1)\\
\Delta r_e(\phi)&=r_e(\phi) - r_e(\phi+1)\\
\Delta r_i(\phi)&=r_i(\phi) - r_i(\phi+1)\\
\Delta r_j(\phi)&=r_j(\phi) - r_j(\phi+1)
\end{aligned}$$

For the sake of simplicity, we do not mention the iteration number in the following discussion. If no iteration number is mentioned, then the current iteration ϕ is implied.

Let $x_i x_j$ represent the state of the matching process for input i and output j of the switch, where $x_i, x_j \in \{0, 1\}$ with 0 representing that the input/output is unmatched and 1 representing that the input/output is matched at the end of the current iteration. The possible states of the matching process are 00, 01, 10, and 11. However, the state 11 should explicitly consider if the tagged input queue $Q(i,j)$ at input i is matched. Thus the state 11 is split into two: 11_{suc} and 11_{blo} respectively. Given the current state of the switch $(n(\phi), h_i(\phi), r_f(\phi), r_e(\phi), r_i(\phi), r_j(\phi))$ and the current state of the matching process $x_i x_j$ the resulting state of the switch $(n(\phi+1), h_i(\phi+1), r_f(\phi+1), r_e(\phi+1), r_i(\phi+1), r_j(\phi+1))$ and the resulting state of the matching process $x_i' x_j'$ is controlled by the transition probabilities defined below:

$P_{blo\_00|00}$ = *Prob{at the end of current iteration, the HOL cell at the tagged input queue gets blocked, and both input i and output j remain unmatched given that both input i and output j were unmatched at the beginning of current iteration}*

$P_{blo\_01|00}$ = *Prob{at the end of current iteration, the HOL cell at the tagged input queue gets blocked, input i remains unmatched and output j gets matched given that both input i and output j were unmatched at the beginning of current iteration}*

$P_{blo\_1x|00}$ = *Prob{at the end of current iteration, the HOL cell at the tagged input queue gets blocked, input i gets matched given that both input i and output j were unmatched at the beginning of current iteration}*

$P_{blo\_01|01}$ = *Prob{at the end of current iteration, the HOL cell at the tagged input queue gets blocked, and input i remains unmatched given that input i was unmatched and output j was matched at the beginning of current iteration}*

$P_{blo\_11|01}$ = *Prob{at the end of current iteration, the HOL cell at the tagged input queue gets blocked, and input i gets matched given that input i was unmatched and output j was matched at the beginning of current iteration}*

$P_{suc|00}$ = *Prob{at the end of current iteration, the HOL cell at the tagged input queue $Q(i,j)$ gets matched with output j, given that input i and output j were unmatched at the beginning of current iteration}*

These probabilities are functions of the current state of the switch $(n(\phi), h_i(\phi), r_f(\phi), r_e(\phi), r_i(\phi), r_j(\phi))$. The transitions among the states of the matching process can be represented by the state transition diagram shown in Figure 5 with the state space $X = \{00, 01, 1x, 11_{suc}\}$ where $1x = 10 \cup 11_{blo}$.

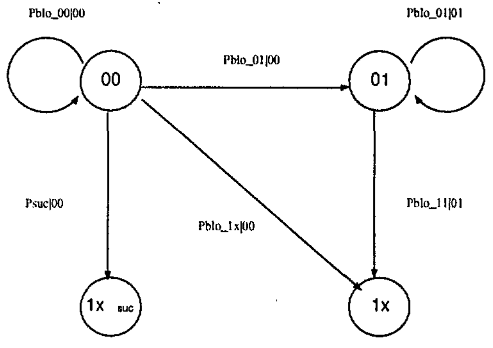


Figure 5: The matching process state transition diagram.

We derive equations for the transition probabilities next. Given $(n(\phi), h(\phi), r_f(\phi), r_e(\phi), r_i(\phi), r_j(\phi))$ and $(n(\phi+1), h(\phi+1), r_f(\phi+1), r_e(\phi+1), r_i(\phi+1), r_j(\phi+1))$, we compute the transition probabilities. In this paper, we present the equations for only two iterations. The extension of these equations to a larger number of iterations can be obtained using similar techniques used in our previous work [16] for analyzing the PIM scheduling algorithm with multiple iterations. The same equations are also applicable in this case because beyond two iterations, we apply the PIM algorithm iteratively. If the high priority queue Q_H is non-empty or $Q(i,j)$ is the high priority queue Q_H, the change to the state of $Q(i,j)$ is straightforward. Only those cases where $Q(i,j)$ is not the high priority queue and Q_H is empty are analyzed as follows.

First iteration In the first iteration only the high priority queues have a chance to be matched. Hence

$$P_{blo\_00|00} = \binom{h-1}{\Delta h}\binom{n-1-h}{m-\Delta h}p_1^m p_0^{n-1-m} \quad m \in [0, n-2],$$

$$P_{blo\_01|00} = \binom{h-1}{\Delta h-1}\binom{n-1-h}{m-\Delta h}p_1^m p_0^{n-1-m} \quad m \in [1, n-1],$$

$$P_{blo\_10|00} = 0 \quad and \quad P_{blo\_11|00} = 0.$$

Second iteration We define the following probabilities associated with the first stage of contention for a cell:

P_{sucl_r} = *Prob{the HOL cell of certain queue of input in set $\Re$ succeeds in the first stage contention}*

P_{sucl_nr} = *Prob{the HOL cell of certain queue of input not in set $\Re$ succeeds in the first stage contention}*

where the elements of set $\Re$ are the unmatched inputs whose outputs corresponding to their high priority queues are also unmatched. Here P_{sucl_r}, and P_{sucl_nr} are functions of p_0, and are computed as:

$$P_{sucl\_r} = \frac{1-p_0^{n-1}}{n-1}$$

$$P_{sucl\_nr} = \frac{1-p_0^n}{n}$$

Computing $P_{suc|00}$: Recalling that in the state transition diagram of the matching process, state 11_{suc} is an absorbing state, the transition probability $P_{suc|00}$ is computed without consideration on m.

$$P_{suc|00} = \frac{1}{h}\sum_{i=0}^{n-1-r_j}\sum_{j=0}^{min(i,r-r_j-r_i)}\frac{1}{i+1} \cdot\binom{r-r_j-r_i}{j}\binom{n-1-(r-r_i)}{i-j}P_{sucl\_r}^j \cdot(1-P_{sucl\_r})^{r-r_j-r_i-j}P_{sucl\_nr}^{i-j} \cdot(1-P_{sucl\_nr})^{n-1-(r-r_i)-(i-j)}$$

Computing $P_{blo\_0x|0x}$:

$$P_{blo\_0x|0x} = \frac{r_f}{h}\cdot P_{blo\_rf} + \frac{h-r_f}{h}\cdot P_{blo\_nrf}$$

where

$$P_{blo\_rf} = \sum_{i=0}^{n-2}\sum_{j=0}^{r-1-r_i}(1-\tfrac{1}{i+1})\binom{r-1-r_i}{j} \cdot\binom{n-1-(r-r_i)}{i-j}P_{sucl\_r}^j(1-P_{sucl\_r})^{r-1-r_i-j} \cdot P_{sucl\_nr}^{i-j}(1-P_{sucl\_nr})^{n-1-(r-r_i)-(i-j)}$$

and

$$P_{blo\_nrf} = \sum_{i=0}^{n-1}\sum_{j=0}^{r-r_i}(1-\tfrac{1}{i+1})\binom{r-r_i}{j} \cdot\binom{n-1-(r-r_i)}{i-j}P_{sucl\_r}^j(1-P_{sucl\_r})^{r-r_i-j}P_{sucl\_nr}^{i-j} \cdot(1-P_{sucl\_nr})^{n-1-(r-r_i)-(i-j)}$$

Computing $P_{blo\_1x|00}$:

$$P_{blo\_1x|00} = \frac{r_f-r_j}{h}(1-P_{blo\_rf}) + \frac{h-r_f-1+r_j}{h}\cdot(1-P_{blo\_nrf})$$

Computing $P_{blo\_11|01}$:

$$P_{blo\_11|00} = \frac{r_f}{h}(1-P_{blo\_rf}) + \frac{h-r_f}{h}\cdot(1-P_{blo\_nrf})$$

The states of the switch at the end of each iteration, $(n(\phi), h_i(\phi), r_f(\phi), r_e(\phi), r_i(\phi), r_j(\phi), x_i x_j)$, can be viewed as a weighted tree with the nodes of the tree corresponding to the switch states. The root of the tree is the initial state of the switch $(N, h_i, g, s, 00)$. All states in level ϕ of the tree correspond to the states of the switch at the end of the ϕ th iteration of the 2PPIM algorithm. Weights are assigned to the arcs between the states, and are equal to the transition probabilities $P_{blo\_x_i'x_j'|x_i x_j}$ or $P_{suc|x_i x_j}$.

Each state $(n(\phi), h_i(\phi), r_f(\phi), r_e(\phi), r_i(\phi), r_j(\phi), x_i x_j)$ is assigned a probability of $Pr(n(\phi), h_i(\phi), r_f(\phi), r_e(\phi), r_i(\phi), r_j(\phi), x_i x_j)$ equal to the product of the transition probabilities along the arcs from the root to the state. The probabilities $P_{blo\_00|H(h_i,r_f,r_e,r_i,r_j)}$, $P_{blo\_01|H(h_i,r_f,r_e,r_i,r_j)}$, $P_{blo\_1x|H(h_i,r_f,r_e,r_i,r_j)}$, and $P_{suc|H(h_i,r_f,r_e,r_i,r_j)}$ at the end of Φ iterations of the PIM algorithm can be computed as,

$$P_{blo\_x_i x_j|H(h_i,g,s)} = \sum_{n(\Phi),h_i(\Phi),r_f(\Phi),r_e(\Phi),r_i(\Phi),r_j(\Phi)} Pr(n(\Phi), h_i(\Phi), r_f(\Phi), r_e(\Phi), r_i(\Phi), r_j(\Phi), x_i x_j),$$

$$P_{suc|H(h_i,g,s)} = \sum_{n(\Phi),h_i(\Phi),r_f(\Phi),r_e(\Phi),r_i(\Phi),r_j(\Phi)} Pr(n(\Phi), h_i(\Phi), r_f(\Phi), r_e(\Phi), r_i(\Phi), r_j(\Phi), 11_{suc})$$

and

$$P_{blo\_0|H(h_i,g,s)} = P_{blo\_00|H(h_i,g,s)} + P_{blo\_01|H(h_i,g,s)},$$

$$P_{blo\_1|H(h_i,g,s)} = P_{blo\_1x|H(h_i,g,s)}.$$

3.3 Solving the Markov Chain

As can be seen from the above equations, p_0, π_0 and π_1 must be known in advance in order to compute the steady state probabilities. In the Appendix of this paper, we give the detail procedures to derive the computation formulas for these parameters. The steady state probabilities are given by:

$$\pi_0 = 1/(1+\alpha\sum_{l=1}^{b_i-1}\beta^{l-1}e + \alpha\beta^{b_i-2}\lambda B(I-B)^{-1}e)$$

$$\pi_l = \begin{cases} \pi_0\alpha\beta^{l-1} & ,for\ 0 < l < b_i \\ \pi_0\alpha\beta^{b_i-2}\lambda B(I-B)^{-1} & ,for\ l = b_i \end{cases}$$

When the queueing system attains equilibrium states, the following equation will hold

$$p_0 = (1-\lambda)\pi_0 \qquad (9)$$

The introduction of p_0 plays an essential role in solving of the Markov chain, but p_0 can't be derived directly from the known system parameters, such as the switch size, buffer size and traffic load. Instead of assuming p_0 as a known parameter [14], we prove a fixed point exits for Eq (9) so as to use the *fixed point* iterative method to obtain p_0 from the known system parameters.

Lemma 1 *A fixed point exists for Eq (9) in the interval [0,1].*

PROOF of Lemma 1:
Let the measure derived from the Markov chain be the probability p_0. We know that λ is a *constant.* According to the Rules (1) and (4) of THEOREM 2 in [19], a fixed point exists. In addition,

$$p_0 = (1-\lambda)\pi_0 \le \pi_0 \le 1$$

Thus, the fixed point must exist in interval [0, 1].

3.4 Computing the Performance Metrics

Once the steady state probabilities are known, then interesting performance parameters, such as throughput, mean queue length and mean cell loss probability can be computed directly by using the known parameters. Let ρ, $\bar{Q}$ and P_{loss} be throughput, mean queue length and mean cell loss probability respectively, then

$$\begin{aligned}\rho &= \lambda\pi_0 S_0 \\ &+ \sum_{l=1}^{b_i}\sum_{h=1}^{N}\sum_{g=0}^{N}\sum_{s=1-g}^{1}\pi_{(l,h,g,s)}P_{suc|W(h,g,s)} \\ \rho &= \lambda\pi_0(1-B_0e) \\ &+ \sum_{l=1}^{b_i}\sum_{h=1}^{N}\sum_{g=0}^{N}\sum_{s=1-g}^{1}\pi_{(l,h,g,s)}P_{suc|W(h,g,s)}\end{aligned}$$

$$\bar{Q} = \sum_{l=1}^{b_i} l\pi_l e \qquad P_{loss} = \lambda\pi_{b_i}e$$

Knowing the mean queue length $\bar{Q}$ and the mean queue throughput ρ, the mean cell delay $\bar{D}$ can be computed as follows:

$$\bar{D} = \bar{Q}/\rho$$

4 Numerical Results

Both mathematical analysis and simulation results are presented in this section in order to investigate the accuracy of the above queueing model. For cell loss probabilities, only which for the switch with heavy traffic loads are given, because that estimating a cell loss probability 10^{-9} for the switch under a small traffic load is practically prohibited. In all the experiments, the number of iterations in the 2PPIM algorithm is set to be 2. Figure 6 and 7 show the switch throughput as a function of switch size (16, 32 and 64) and offered load λ with buffer sizes of 5 cells and 10 cells, respectively. The curves show that both 2PPIM and iSLIP scheduling with 2 iterations are enough to get a high throughput > 90%.

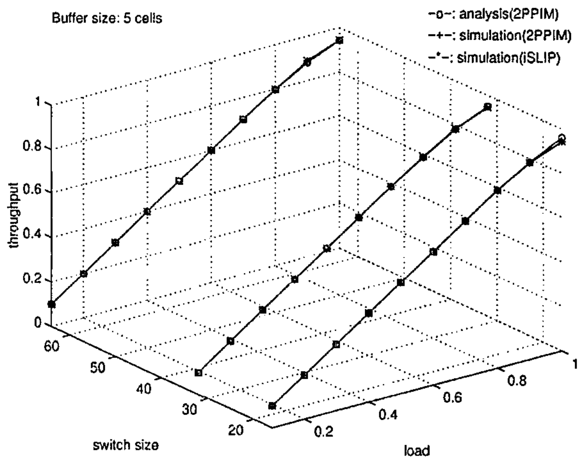


Figure 6: The throughput as a function of switch size and offered load, with a buffer size b_i=5.

Figure 8 and 9 show the mean cell delay as a function of switch size (16, 32 and 64) and offered load λ for buffer size b_i of 5 cells and 10 cells respectively. The figures indicate that the mean cell delay increases by increasing the switch size and the offered load. For 2 iterations 2PPIM and iSLIP scheduling, the mean cell delay increases dramatically when the offered load exceeds 80%, which indicates that iSLIP switches with 2 iterations iSLIP scheduling will be overloaded when the traffic load is greater than 80%. This phenomenon can also be observed in our previous paper [16] on PIM. Notice that when the traffic load is extremely low, such as 0.1, all curves cluster together. It is

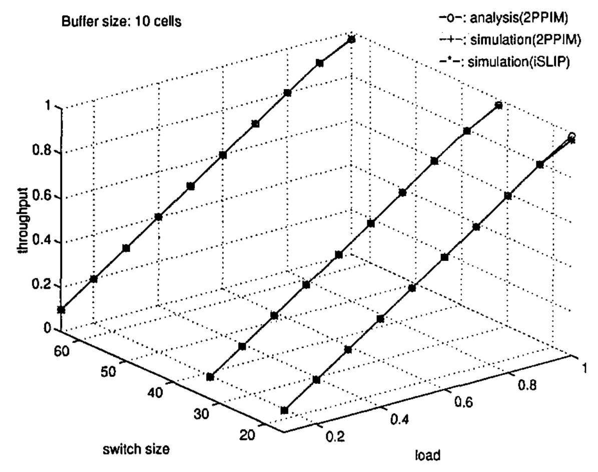


Figure 7: The throughput as a function of switch size and offered load, with a buffer size b_i=10.

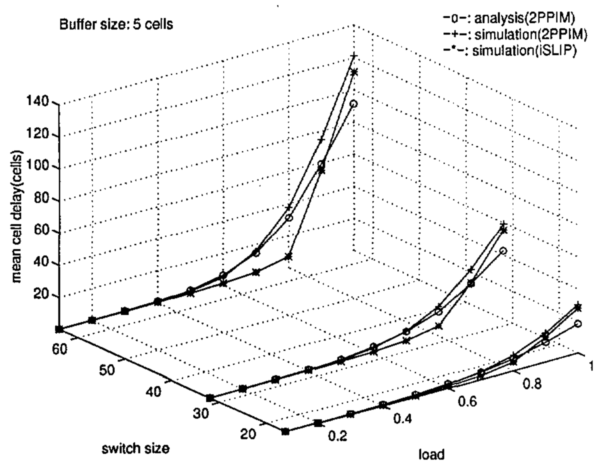


Figure 8: The mean cell delay as a function of switch size and offered load, with a buffer size b_i=5.

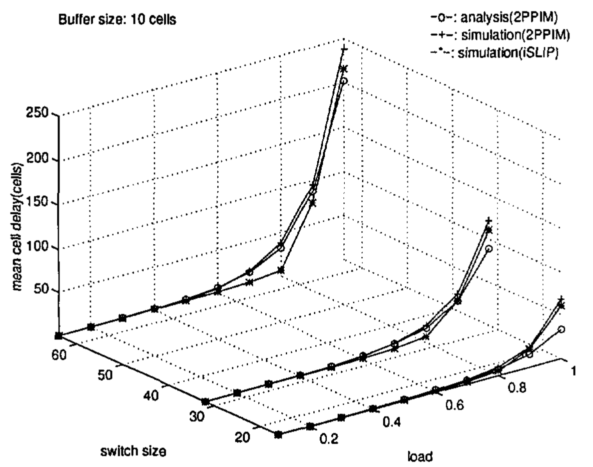


Figure 9: The mean cell delay as a function of switch size and offered load, with a buffer size b_i=10.

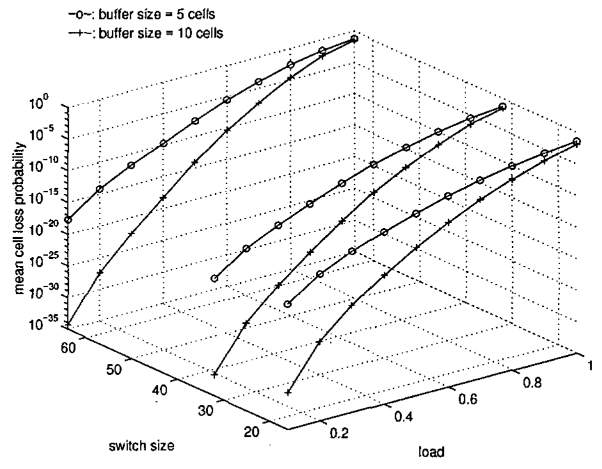


Figure 10: The mean cell loss probability as a function of switch size and offered load, with a buffer size b_i=5 and b_i=10.

not difficult to understand that, under low traffic load, both 2PPIM and iSLIP behave similarly to PIM so that there is almost no difference among them. But when the traffic load increases, the prioritization introduced by the *grant/accept arbiters* of iSLIP switches will play a significant role in the scheduling of the HOL cells. The simulation results for 2PPIM and iSLIP in Figure 8 and Figure 9 show that the biggest difference between 2PPIM and iSLIP occurs at traffic load of about 0.8. When the traffic load is extremely high, 2PPIM and iSLIP will behave similarly again. The bias between the analysis results and simulation results of 2PPIM and iSLIP can be reduced by expanding the dimension of the underlying Markov chain Z in section 3.1. Interested readers may refer to our previous paper [16] on PIM for more details.

In figure 10, the mean cell loss probabilities of 2PPIM switches with queue size of 5 cells and 10 cells are given as a function of offered load. It can be seen that, for a medium size 2PPIM switch with 2 iterations 2PPIM scheduling with traffic load less than 60%, a buffer size of 10 cells per queue is sufficient to guarantee a cell loss probability $< 10^{-9}$.

As mentioned in the introduction of this paper, we are extending these results to more realistic traffic patterns. This complicates the mathematical analysis, but would also lead to more practical results.

5 Conclusion

A queueing model for the performance analysis of a novel input access scheme ATM switch with iSLIP scheduling under random traffic has been presented. The queueing model provides a general method for analyzing such ATM switches to compute the throughput, mean queue length, mean cell delay, and mean cell loss probability given the switch size, queue buffer size, and offered load. The analytical results obtained from the queueing model matches the simulation results satisfactorily.

In order to make the original switch model tractable for analysis, a number of assumptions have been added. The most important is that the random traffic, that is, cells are arriving at each input queue according to IID Bernoulli process. In future research, we will try to apply this method to analyze the same kind of ATM switches with bursty and correlated traffic.

The contribution of this paper is two fold. First, we introduced a scheduling algorithm called 2PPIM to facilitate the analysis for *i*SLIP scheduling. Second, a theoretical analysis for various performance parameters including throughput, mean cell delay, and mean cell loss probability, of a ATM switch using a *i*SLIP scheduling scheme is presented by using 2PPIM scheduling as an approximation.

A Appendix: Computation of the Steady State Probabilities

Following the steps given in [6], we give the procedures to compute the steady state probabilities of the Markov chain. The method presented in [6] is based on the algorithmic approach given in [15].

Given that $\pi_{(l,w_i,w_o)}$ is the steady state probability of the state (l, w_i, w_o), where l is the length of the tagged queue, w_i is the length of the virtual HOL input queue, and w_o is the length of the virtual HOL output queue. The steady state probability vector of the Markov chain is given by $\Pi = [\pi_0, \pi_1, \pi_2, \dots, \pi_l, \dots, \pi_{b_i}]$ where every element $\pi_l = [\pi_{(l,1,1)}, \pi_{(l,1,2)}, \dots, \pi_{(l,N,N)}], l > 0$ is a row vector of size N^2, and π_0 is a scalar.

From the definition of the transition probability matrix, we know that:

$$\Pi T = \Pi \qquad (10)$$

By expanding Eq (10), we have:

$$\pi_0((1-\lambda) + \lambda S_0) + \pi_1(1-\lambda)S_c = \pi_0 \qquad (11)$$

$$\pi_0\lambda B_0 + \pi_1(\lambda S + (1-\lambda)B) + \pi_2(1-\lambda)S = \pi_1 \qquad (12)$$

$$\begin{aligned}&\pi_{i-1}\lambda B + \pi_i(\lambda S + (1-\lambda)B) + \\ &\pi_{i+1}(1-\lambda)S = \pi_i\ , \quad for\ 1 < i < b_i - 1\end{aligned} \qquad (13)$$

$$\pi_{b_i-2}\lambda B + \pi_{b_i-1}(\lambda S + (1-\lambda)B) + \pi_{b_i}S = \pi_{b_i-1} \qquad (14)$$

$$\pi_{b_i-1}\lambda B + \pi_{b_i}B = \pi_{b_i} \qquad (15)$$

Eq (11) can be re-arranged as:

$$\pi_0\lambda(1-S_0) = \pi_1(1-\lambda)S_c \qquad (16)$$

Multiplying Eq (12) by e we get:

$$\pi_1\lambda B_c = \pi_2(1-\lambda)S_c. \qquad (17)$$

Similarly, multiplying Eq (13) by e results in:

$$\pi_{i-1}\lambda B_c = \pi_i(1-\lambda)S_c, \qquad (18)$$

and the same operation on Eq (15) leads to:

$$\pi_{b_i-1}\lambda B_c = \pi_{b_i}S_c \qquad (19)$$

Let $\alpha = \lambda B_0(I - \lambda I_1 - (1-\lambda)B)^{-1}$ and $\beta = \lambda B((1-\lambda)(I-B))^{-1}$. After some algebra operations, we get:

$$\pi_0\lambda B_0 e = \pi_1(1-\lambda)(e - B_c) = \pi_1(I - \lambda I_1 - (1-\lambda)B)e \qquad (20)$$

So,

$$\pi_1 = \lambda B_0(I - \lambda I_1 - (1-\lambda)B)^{-1}\pi_0 = \pi_0\alpha \qquad (21)$$

and

$$\pi_l = \begin{cases} \pi_0\alpha\beta^{l-1}, & for\ 0 < l < b_i \\ \pi_0\alpha\beta^{b_i-2}\lambda B(I-B)^{-1}, & for\ l = b_i \end{cases}$$

Notice that $\pi_0 + \sum_{l=1}^{b_i} \pi_l e = 1$, we have:

$$\pi_0 = 1/(1 + \alpha\sum_{l=1}^{b_i-1}\beta^{l-1}e + \alpha\beta^{b_i-2}\lambda B(I-B)^{-1}e)$$

References

[1] P. Achille and B. Giacomo, Analysis of Input and Output Queueing for nonblocking ATM Switches, *ACM Trans. on Networking*, Vol. 1, No. 3, June 1993.

[2] T. E. Anderson, S. S. Owicki, J. B. Saxe, and C. P. Thacker, High-speed Switch Scheduling for Local-area Networks, *ACM Transactions on Computer Systems*, Vol. 11, No. 4, Nov. 1993, pp. 319-352.

[3] R. Y. Awdeh and H. T. Mouftah, Survey of ATM Switch Architectures, *Computer Networks and ISDN Systems*, Vol. 27, 1995, pp. 1567-1613.

[4] C.-J. Hou, C.-C. Han and W.-C. Chau, Priority-Based High-Speed Switch Scheduling for ATM Networks, *Proc. 20th Conf. on Local Computer Networks*, 1995, pp. 19-28.

[5] L. Jacob and A. Kumar, Saturated Throughput Analysis of an Input Queueing ATM switch with Multiclass Bursty Traffic. *IEEE Trans. on Communications*, Vol. 43, No. 2/3/4, Feb/Mar/Apr 1995, pp.757-761.

[6] Y.C. Jung and C. K. Un, Performance Analysis of Packet Switches with Input and Output Buffers, *Computer Networks and ISDN Systems*, Vol. 26, 1994, pp. 1559-1580.

[7] M. Karol, M. Hluchyj and S. Morgan, Input versus Output Queueing on A Space Division Packet Switch, *IEEE Trans. on Communications*, Vol. 35, Dec. 1987, pp. 1347-1356.

[8] R. O. LaMaire and D. N. Serpanos, Two-Dimentional Round-Robin Schedulers for Packet Switches with Multiple Input Queues, *IEEE/ACM Transactions on Networking*, Vol. 2, No. 5, Oct. 1994, pp. 471-481.

[9] M. J. Lee and D. S. Ahn, Cell Loss Analysis and Design Trade-Offs of Nonblocking ATM Switches, *IEEE/ACM Trans. on Networking*, Vol. 3, No. 2, April 1995, pp. 199-210.

[10] N. W. McKeown, *Scheduling Algorithms for Input-Queued Cell Switches*, Ph.D. thesis, University of California at Berkeley, 1994.

[11] N. Mckeown, P. Varaiya, and J. Walrand, Scheduling Cells in an Input-Queued Switch, *Electronics Letters*, Vol. 29, No. 25, 1994, pp. 2174-2175.

[12] N. McKeown, M. Izzard, A. Mekkittikul, W. Ellersick and M. Horowitz, Tiny Tera: a packet switch core, *IEEE Micro*, Vol.17, No.1, Jan.-Feb. 1997, pp. 26-33.

[13] N. McKeown and T. E. Anderson, A Quantitative Comparison of Iterative Scheduling Algorithms for Input-Queued Switches, *Computer Networks and ISDN Systems*, 30(1998), 2309-26.

[14] M. K. Mehmet-Ali, M. Youssefi, and H. T. Nguyen, The Performance Analysis and Implementation of An Input Access Scheme in A High-speed Packet Switch, *IEEE Trans. on Communications*, Vol. 42, No. 12, Dec. 1994, pp. 3189-3199.

[15] M. F. Neuts. *Matrix-Geometric Solutions in Stochastic Models*. Johns Hopkins University Press, 1981.

[16] G. Nong, J. K. Muppala and M. Hamdi, Analysis of Non-blocking ATM Switches with Multiple Input Queues, *IEEE/ACM Trans. on Networking*, Vol. 7, No. 1, Feb. 1999, pp.60-74.

[17] A. Pattavina and G. Bruzzi, Analysis of Input and Output Queueing for Nonblocking ATM Switches, *IEEE/ACM Trans. on Networking*, Vol. 1, No. 3, Jun. 1993, pp.314-328.

[18] R. Rooholamini and V. Cherkassky, Finding the Right ATM Switch for the Market, *IEEE Computer*, Vol. 27, No. 4, Apr. 1994, pp. 16-28.

[19] Mainkar V. and Trivedi K.S., Sufficient Conditions for Existence of A Fixed Point in Stochastic Reward Net-based Iterative Models, *IEEE Trans. on Software Engineering*, Vol. 22, No. 9, Sept. 1996, pp.640-53.

[20] J. Wang, KB. Letaief, M. Hamdi and X.R. Cao, Fast cell loss rate estimation of ATM switches using importance sampling, *GLOBECOM 97 Proceedings*, Vol.3, 1997, pp.1562-6.

On The Determination Of Absolute Network Performance

Marlin H. Mickle
Department of Electrical Engineering
348 Benedum Hall
University of Pittsburgh
Pittsburgh, PA 15261, U.S.A.
Phone: 412 / 624 - 9682, Fax: 412 / 624 - 8003
Email: mickle@ee.pitt.edu

Keywords: Networks, Performance, Capacity, Convex Set, High Performance, Network State

Edited by: Mohsen Guizani, Marcin Paprzycki

Received: April 21, 1999 **Revised:** July 22, 1999 **Accepted:** August 30, 1999

Any collection of networks can be termed an "internet" and can perform many useful functions. The analysis and characterization of the internet in the abstract is difficult at best and impossible at worst. The purpose of this paper is to provide a mathematical methodology for the analysis of the total performance of a collection of networks connected to form an internet. However, the internet in this case is captive and created to serve a single purpose, as is the case with numerous control problems such as the beam control of a linear accelerator. A captive internet has a theoretical capability at the physical level depending on the type of networks that have been interconnected. This capability, in terms of bits per second, i.e., data rates, represents a convex domain of feasibility in which the network can be characterized as a single state (point in a convex space) at any instant of time. The characterization or modeling of a captive internet in this fashion also provides a basis for analysis of the well-known internet under certain assumptions of protocols, priorities, etc.

1 Background

The state of network hardware and software makes the technology appealing for applications involving the control of large systems. Current potential applications may go as high as 80,000 nodes or DTEs on a single collection of networks. Such interconnections of networks are frequently termed captive networks, *i.e.*; all traffic is restricted to this particular interconnection.

The interconnection of multiple networks is covered in many current texts [1, 2, 3, 4, 5]. Captive networks are ideal to replace traditional cabling and complex wire harness. The network(s) serves as a bus with built-in access protocols.

The analysis in this type of problem can be performed on the basis of a flow or rate, which in this case is in terms of bits per second. Thus, any network capacity in bits per second is a source or resource that is to be distributed among nodes on the network. In general, the traffic generated by the nodes is predictable. In particular, in the example of dynamic system control, the bandwidth (in electrical engineering terms) defines a sampling rate for a sensor/computer/effector combination, which along with the precision specified, implies a rate in terms of bits per second. Allowing for some safety margin, the collection of rates specifies how many nodes can occupy a single network.

The specific applications such as real time control and high speed video (no compression) preclude the use of buffers and do not allow for delay tolerance or the dropping of packets. There is an entire class of systems for which network control is applicable requiring zero tolerance of delay.

This type of problem may be best suited for ATM networks, but the cost and availability of legacy LANs still gives a spectrum of potential networks for such applications. The sampling times and effector application times require essentially zero jitter placing restrictions on certain types of networks. However, by mathematical consideration of the rates involved, it is possible to design interconnection networks for system control using a variety of media access mechanisms.

In the following discussions, the bounded nature of the problem leads to the formulation. Frames and/or packets can not be delayed and can not be dropped. The situation in this case is analogous to incompressible fluid flow [6, 7, 8], *e.g.*, water.

2 Introduction

Consider a networked system consisting of m clusters where each cluster consists of n nodes or DTEs (processors). A cluster may be a single network or may be composed of multiple networks. Assume at this point, the cluster is on a single network.

WRT any network λ, there are three types of traffic from node i or to node j on network λ; (1) intra-network, I_{ij}, (2) outgoing inter-network traffic, T_{ijk}, and (3) incoming inter-network traffic, R_{ijk} to node i from node j on network k. The network has some bandwidth BW_λ. This bandwidth is the effective bandwidth based on typical utilization. For example, a 10Base2 network should have a bandwidth of 10 Mbps (Million bits per second), but many references suggest the actual bandwidth is on the order of 4 Mbps [9] due to collisions and other considerations.

The following relationship holds true by "the second law of thermodynamics." In the following, it is assumed that the network under consideration is network λ. There are m other networks or clusters on the "internet," and the networks other than λ are summed using the variable k with each of the k^{th} networks having n_k nodes capable of generating traffic to node i

$$\sum_{i=1}^{n_\lambda}\sum_{\substack{j=1\\ j\neq i}}^{n_\lambda} I_{ij} + \sum_{i=1}^{n_\lambda}\sum_{j=1}^{n_k}\sum_{\substack{k=1\\ k\neq\lambda}}^{m} T_{ijk} + \sum_{i=1}^{n_k}\sum_{j=1}^{n_\lambda}\sum_{\substack{k=1\\ k\neq\lambda}}^{m} R_{ijk} \leq BW_\lambda \quad (1)$$

I_{ij} is the intra-network traffic from node i to node j on the network, λ. T_{ijk} is the inter-network traffic from node i on network λ to node j on network, k. R_{ijk} is the inter-network traffic from node j on network k to node i on network, λ.

The I_{ij}, T_{ijk} and R_{ijk}, are all assumed to be in terms of bits per second as rates. At this point, the distributions in time are assumed to be such that the bandwidth of each network is utilized at some value less than specifications to incorporate a design safety factor.

The above inequality (1), represents a three-dimensional space bounded by a plane and the three coordinate limiting planes in the three dimensional space. The area satisfying the inequality is termed the domain of feasibility [10], and the area is a common part of the formulation of a linear programming problem [10].

The optimization of network performance without considering zero jitter is well-covered in the literature [11, 12, 13]. With m networks, the ability to operate the total internet with a guaranteed latency [14] for each network message, the intersection of the m domains of feasibility must not be null [10].

As a general formulation, it may be difficult to specify the traffic generated from each node. However, in a situation where the problem to be solved is a closed system, *i.e.*, a system to be controlled by computers acting on feedback and generating actuating signals, it is possible to identify the necessary traffic in terms of a rate and evaluate a given design on the basis of the above formulation.

The elements of the networks are nodes (DTEs), links (connecting wires or fibers), and various elements with topological index greater than one, which will route or direct the traffic on the internet. The individual nodes will be transmitting data as sensor measurements from the nodes and receiving data from a controlling source as actuators that implement the control scheme.

The formulation discussed here has two primary advantages. First, the linear programming formulation can be used to guarantee the solution space in terms of network traffic that will satisfy the rate constraints. The formulation represents a model that can be analyzed and applied as a distributed control system using classical analysis techniques [10].

3 An Example - Single Network Connection

As an example, consider the three networks shown in Figure 1, A, B, and C, interconnected by a token ring. The constraints resulting from messages exchanged within each network (intra-network) are shown in (2). In this case, the upper bound, BW_λ, would be the situation with only this type of traffic. As other traffic appears on the network, the operating point for I_{ij} would be less than BW_λ.

$$\sum_{i=1}^{n_\lambda}\sum_{\substack{j=1\\ j\neq i}}^{n_\lambda} I_{ij} \leq BW_\lambda, \qquad \lambda = 1,2,3 \qquad (2)$$

The incoming (received) messages are bounded as shown in (3).

$$\sum_{i-1}^{n_k}\sum_{j=1}^{n_\lambda}\sum_{\substack{k=1\\ k\neq\lambda}}^{m} R_{ijk} \leq BW_\lambda, \qquad \lambda = 1,2,3 \qquad (3)$$

The outgoing (transmitted) messages are also bounded as shown in (4).

$$\sum_{i=1}^{n_\lambda}\sum_{j=1}^{n_k}\sum_{\substack{k=1\\ k\neq\lambda}}^{m} T_{ijk} \leq BW_\lambda, \qquad \lambda = 1,2,3 \qquad (4)$$

In the following, the notation will be simplified by using R and T for the traffic of (3) and (4). The three networks are connected, in this example, by a ring, Network 4. Therefore:

$$R + T \leq BW_4 \qquad (5)$$

where

$$T = T_{12} + T_{13} + T_{23} + T_{21} + T_{31} + T_{32} \qquad (6)$$
$$R = R_{12} + R_{13} + R_{23} + R_{21} + R_{31} + R_{32} \qquad (7)$$

The formulation of the underlying problem is straightforward. There are several issues to deal with for a successful implementation.

(1) The clustering of groups of networks to isolate the most critical paths or links increases the accuracy of the result.

(2) The dimensionality of the result of (1) must be considered. This is not a particularly difficult problem in the design stage. However, if an on-line real-time implementation is required, it can be a computational problem. The problem is primarily one of distributing the necessary computation appropriately.

(3) The delays or latencies of the individual network elements must be determined, tested and evaluated. The operation on a real-time distributed control system requires the use of the elements in a somewhat different manner than a typical information network.

   (a) The information network allows buffering because the dynamics are not critical as they are in a discrete control or compensation scheme.

   (b) The information allows message classes providing for the ability to kill certain datagrams or frames after a certain amount of time has passed. While a certain amount of such messaging may be permitted on the distributed network, the fundamental control application must be satisfied within the domain of feasibility.

(4) The typical distributed control system uses different types of networks resulting in different protocols, frame sizes, etc. Thus, it is necessary to be able to calibrate the various parts of the model from existing system implementations. This is an area of current research that has motivated this paper.

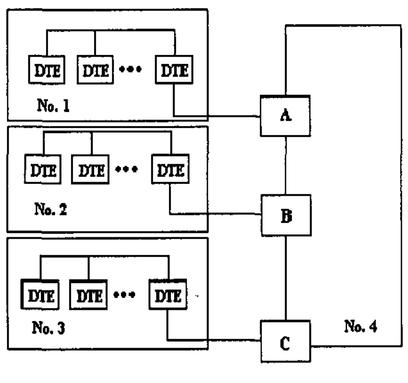


Figure 1: An example network.

As an example, consider the four networks of Figure 1. The total system constraints resulting from messages exchanged within each network (intra-network) are shown in (9), (10), (11) and (12):

$$I_1 + I_2 + I_3 = \text{Total intra-network traffic} \quad (8)$$

$$I_1 + T_{12} + T_{13} + R_{12} + R_{13} \leq BW_1 \quad (9)$$

$$I_2 + T_{21} + T_{23} + R_{21} + R_{23} \leq BW_2 \quad (10)$$

$$I_3 + T_{31} + T_{32} + R_{31} + R_{32} \leq BW_3 \quad (11)$$

Also, for Network 4:

$$T_{12} + T_{13} + T_{21} + T_{23} + T_{31} + T_{32} \leq BW_4 \quad (12)$$

The problem formulated to this point is a set of five constraints and nine dimensions. This is difficult to picture and draw on a two-dimensional paper. Consider the situation from Network 1 as connected to all other networks. T_{1n}, is all transmitted inter-network traffic, and R_{1n} is all received inter-network traffic on Network 1. This formulation is given mathematically in (13).

$$I + T_{1n} + R_{1n} \leq BW_1 \quad (13)$$

The situation with (13) is shown graphically in Figure 2.

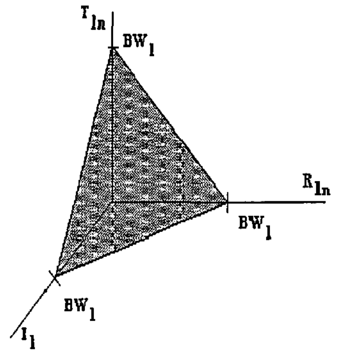


Figure 2: Constraints for Network 1 as a function of the remaining network.

In Figure 2, the solution must exist within the area indicated extending out from the origin, in the quadrant with all positive values, to the plane that intersects each axis at the bandwidth, BW_1. In linear programming terms, this is the domain of feasibility. Four obvious operating points (vertices) are: (1) No traffic, (2) $I_1 = BW_1$, (3) $R_{1n} = BW_1$, and (4) $T_{1n} = BW_1$. These demonstrate self-consistency.

Each network or cluster has a set of relationships as given in (9) through (12). If it is a network, the bandwidth is given by specifications. If it is a cluster, an analysis must be done on the cluster first. The analysis is similar to the network analysis that will be discussed later. The primary consideration in isolating a cluster (or multi-linked network) is the set of links that attach it to the remainder of the network. This is a problem of identifying the bridges in the spanning tree.

4 An Example - Multiple Network Connections

With the three networks of Figure 1, each has only one link to the remainder of the network. Consider the network of Figure 3, where Network 1 has two paths or links connecting it to the balance of the network.

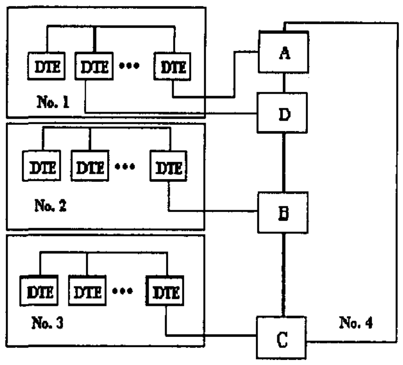


Figure 3: Network with 2 paths to the balance of the network.

The analysis of Network 1, in Figure 3, simply involves the two links to the remainder of the network (the outside world).

$$I_1 + T_{an} + T_{dn} + R_{an} + R_{dn} \leq BW_1 \qquad (14)$$

The concept of the optimum operating point can be seen from Figure 2. The portion of plane at the intersection of the three axes and the BW_1 point represents the maximum traffic that can occur on the network with no latency (delay) and no buffering. The portion of the plane that intersects with the positive quadrant is on the boundary of the domain of feasibility. Thus, the three points indicated earlier represent extremes in the indicated directions. For example, if $I = BW_1$, there can be no T or R traffic. If there must be some T or R traffic, then the I traffic must be reduced to accommodate the additional traffic. The possible reductions in I traffic while maintaining the maximum network traffic is a point lying on the plane indicated in Figure 2.

In general, the process of (9) and (14) is extended to include each externally connected node.

The plane of Figure 2, must satisfy the following simple relationship giving the locus of optimum operating points.

$$aI + bT + cR = BW_1 \qquad (15)$$

where $a, b, c \geq 0$, and $I, T, R \geq 0$.

There are three (3) primary assumptions under the strict operation as described above. These are:

(1) There is no buffering at any node.

(2) Delays (latencies) can not be tolerated.

(3) There are no zero class messages, packets, etc, that can be "dropped" under heavy traffic conditions.

There are also several implicit assumptions in the problem structure that will facilitate the type of system and traffic conditions as imposed above.

(1) Many terminal nodes have correlated transmit/receive traffic.

(2) The routing scheme is known *a priori* under ideal (no failure) conditions, *i.e.*, fixed routing using a successor matrix.

(3) Many terminal node transmission schedules are correlated.

5 The Application

Three major functions of the control type of application are (1) feedback loops [15], (2) system synchronization, and (3) multi-level control. In the feedback case, there is a sensor for measurement of the physical system which is fed to a computer which computes a control signal that is fed to an actuator to input to the physical system. This completes the loop around the physical system where the elements of the loop are interconnected by a network or networks.

Under certain circumstances, the sensors and effectors from different parts of the system must be coordinated in time. This coordination or synchronism requires signals that are likely from different networks.

In certain situations, the calculations within the closed loops first discussed contain parameters in the calculations that must be modified in a coordinated manner. This coordination requires measurement data at a supervisory level. This coordination requires information from various networks and/or clusters within the system providing data paths and certain desired topological considerations.

From a systems engineering standpoint, the topology is designed with the data rate requirements. These requirements are then translated into networks as in Figures 1 and 3, with more networks and interconnections.

The choice of number and type of network is simply to satisfy the specified system data rate and path requirements. The derivation within this paper is verified by mathematical construction. The analysis can be performed analytically and verified by simulation with known capacities, data traffic patterns and the topological design. These are ongoing projects in the Parallel Processing and Computer Networking Laboratory in the School of Engineering at the University of Pittsburgh.

6 Summary

A method of modeling a collection of networks, *i.e.*, an internet, has been presented based on data rates. The methodology is based on a captive internet topology. The primary difference in this case is the known traffic rates and the zero tolerance on jitter.

The application of the techniques of this paper to general network interconnections requires some rethinking of the current methods of analysis. However, the application of this methodology provides a powerful tool for network management.

References

[1] Stallings W. (1997) *Data and Computer Communications.* Fifth Edition, Upper Saddle River: Prentice Hall.

[2] Halsall F. (1996) *Data Communications, Computer Networks and Open Systems.* Reading: Addison-Wesley.

[3] Keshav S. (1997) *An Engineering Approach to Computer Networking.* Reading: Addison Wesley, p. 108.

[4] Tannenbaum A. S. (1988) *Computer Networks.* Upper Saddle River: Prentice Hall.

[5] Comer D. E. (1997) *Computer Networks and Internets.* Upper Saddle River: Prentice Hall.

[6] Gresho P. M. (1991) Incompressible Fluid Flow Dynamics: Some Fundamental Formulation Issues. *Annual Review of Fluid Mechanics,* 23, p. 413–453.

[7] Theodossiou V. M. & Sousa A. C. M. (1986) An Efficient Algorithm for Solving the Incompressible Fluid Flow Equations. *International Journal for Numerical Methods in Fluids,* 6, 8, p. 557–572.

[8] Hall C. A., Porsching T. A. & Mesina G. L. (1992) On a Network Method for Unsteady Incompressible Fluid Flow on Triangular Grids. *International Journal for Numerical Methods in Fluids,* 15, 12, p. 1383–1406.

[9] Walrand J. (1991) *Communication Networks: A First Course.* Boston: Irwin.

[10] Mickle M. H. & Sze T. W. (1972) *Optimization in Systems Engineering.* Scranton: Intext.

[11] Boorstyn P. R. & Frank H. (1977) Large-Scale Network Topological Optimization. *IEEE Transactions on Communications,* January 1977, p. 29–47.

[12] Floyd S. & Jacobsen V. (1995) Link-Sharing and Resource Management Models for Packet Networks. *IEEE Transactions on Networking,* 3, 4, p. 365–386.

[13] Boggs D., Mogul J. & Kent C. (1988) Measured Capacity of an Ethernet: Myths and Reality. *Proceedings of ACM SIGCOMM '88,* August 1988, p. 222–234.

[14] Wesel E. K. (1998) *Wireless Multimedia Communications.* Reading: Addison-Wesley.

[15] Zadeh L. A. & Desoer C. A. (1963) *Linear Systems Theory.* New York: McGraw-Hill.

The Brain As A Huygens Machine

B.E.P. Clement
Clement Neuronic Systems Ltd, 15 Everest Drive, Crickhowell, Powys, NP8 1DH, U.K.,
AND
P.V. Coveney
Department of Chemistry, University of Wales High Cross, Bangor, Gwynedd, LL57 2UW, U.K.,
now: Schlumberger Cambridge Research, Cambridge CB3 0HG, U.K.
AND
M. Jessel, retired (now deceased)
Laboratoire de mécanique et d'acoustique du CNRS, B.P. 71, F–13277 Marseille, Cedex 9, France
AND
P.J. Marcer
Aikido Enterprises, 53 Old Vicarage Green, Keynsham, Avon, BS18 2DH, U.K.

Keywords: brain, biological neuron, configuration space, dynamics, Huygens principle, information-processing, machine, mathematical framework, neural morphology, spin

Edited by: Anton P. Železnikar
Received: August 28, 1998 **Revised:** March 2, 1999 **Accepted:** March 25, 1999

A mathematical description of the physical world is required which can explain the brain's actual information-processing morphology and dynamics. On the basis of Huygens' principle we propose such a neural model in which each individual biological neuron is analogous to an artificial neural network of very great complexity. The model provides a mathematical framework in which to describe the information-processing properties of living systems.

1 Introduction

A theoretical framework is adopted following the line of the French School of Nicolas Bourbaki as previously advocated by one of us [1].This allows problems which have been treated directly and separately in the past to emerge as different manifestations of a general principle which in wavefield physics is known as Huygens' principle of secondary sources [2, 3].

In a recent paper, Huygens' principle is related to computability [4]. It is shown that thermodynamic machines or engines described in terms of Huygens' principle may simulate any form of behaviour utilising entropy production. The "category theoretical" formalisation of Huygens' principle [3] set out below may therefore be interpreted as specifying how the signals propagated in a medium change the mappings of any particular form of physical behaviour OP in order to control such a machine, or how such a machine may be controlled in order to compute a particular form of physical behaviour [4, 5].

Huygens' *Gedankenexperiment* for the propagation of a wavefield by the recursive formation of further wavefronts of secondary sources says [6] that the perturbation of the field F that passes through a surface S containing a wave source S_{or} is identical to the perturbation that is obtained by removing the source and substituting it by an appropriate system of secondary sources $S^{\mathrm{S}}_{\mathrm{or}}$, distributed on the surface S. It may be represented using category theory by the following categorical diagram,

$$\begin{tikzcd} F = F(0) \arrow[r, "\mathrm{OP}"] \arrow[d, "\mathrm{T}"'] & S_{\mathrm{or}} = \mathrm{OP}F \\ F(t) = \mathrm{T}F(0) \arrow[r, "\mathrm{OP}"] & S^{\mathrm{S}}_{\mathrm{or}} = (\mathrm{OP},\mathrm{T})F(0) \arrow[u] \end{tikzcd} \quad (1)$$

where $\mathrm{OP}F = S_{\mathrm{or}}$ is the particular physical behaviour that connects the sources to the field F, and $F(t) = \mathrm{T}F(0)$ describes the means by which the wave-field propagation takes place.

From the diagram

$$\begin{aligned} S^{\mathrm{S}}_{\mathrm{or}} = \mathrm{OPT}F(0) &= \mathrm{OPT}F(0) - \mathrm{T}S_{\mathrm{or}} + \mathrm{T}S_{\mathrm{or}} \\ &= \mathrm{OPT}F(0) - \mathrm{TOP}F(0) + \mathrm{T}S_{\mathrm{or}} \\ &= (\mathrm{OP},\mathrm{T})F(0) + \mathrm{T}S_{\mathrm{or}} \end{aligned}$$

where $(\mathrm{OP},\mathrm{T}) = \mathrm{OPT} - \mathrm{TOP}$ is the Lie product or commutator and T is a suitable weighting operator. But the syllogistic form of Huygens' principle stated above asserts that in order to correctly describe a particular wavefield phenomenon through the operator OP, T must be chosen to be a Heaviside-like operator so that $\mathrm{T}S_{\mathrm{or}} = 0$ inside S for example. Such a Heaviside-like operator T appropriate to the Lie product (OP,T) is equivalent to the singular Green's function (Schwartz distribution) that permits the corresponding description of the same phenomenon by means of an integral formula [2]. For example if $F(r,t)$ is a field of particles at location r such that

$$F(r,t) = \iint G(r,t,r',t')S(r',t')dr'dt'$$

where $G(r,t,r',t')$ is the Green's function, then it has been shown by one of us [7] that

$$G(r,t,r',t') = \tfrac{1}{2}c \cdot h\Big(t - t' - \tfrac{|r-r'|}{c}\Big) \quad \text{(one dimension)}$$

$$G(r,t,r',t') = \frac{c \cdot h\Big(t - t' - \frac{|r-r'|}{c}\Big)}{2\pi\sqrt{c^2(t-t')^2 - |r-r'|^2}} \quad \text{(two dimensions)}$$

$$G(r,t,r',t') = \frac{\delta\Big(t - t' - \frac{|r-r'|}{c}\Big)}{4\pi|r-r'|} \quad \text{(three dimensions)}$$

where $h(x)$ is the Heaviside step function, $\delta(x)$ is the Dirac delta function, and c is the velocity of propagation of signals. The categorical diagram (1) shows, therefore, that a description in terms of Lie products is appropriate to any form of wavefield behaviour [5]. In quantum physics, for example, this description is interpreted as arising from the behaviour of the field bosons in question. Moreover, (1) implies that the inverse transformation $F = (\mathsf{OP})^{-1}S_{\mathsf{or}}$ may exist and is then calculable.

2 Application of the Huygens model to a biological neuron

The categorical diagram (2) below shows that a Huygens machine working by entropy production may consist of a hole which opens and closes according to some physical mechanism in say a biological membrane and through which molecules or ions can pass.

Let S be the entropy density and σ^S be the entropy production which can create or destroy states in the molecular or ionic field; then (1) says that

$$\begin{tikzcd} S(0) \arrow[r, "\mathsf{OP}"] \arrow[d, "\mathsf{T}_1"'] & \sigma^S \arrow[d] \\ S(t) = \mathsf{T}_1 S(0) \arrow[r, "\mathsf{OP}"] & \mathsf{T}_2\sigma^S = S(0)\frac{\partial \mathsf{T}_1}{\partial t} + J^S \cdot \nabla \mathsf{T}_1 \end{tikzcd} \quad (2)$$

Here $\mathsf{T}_2\sigma^S$ is the entropy production required to obtain the new behaviour $S(t)$ at the time t and may be calculated in view of the fact that the entropy in an isolated system is governed by the continuity equation

$$\frac{\partial S}{\partial t} + \nabla \cdot J^S = \sigma^S = \mathsf{OP}S(0)$$

where $J^S = Sv$ is the flux of the entropy over the surface of the hole [4].

The recent work of Zurek [8, 9] demonstrates that entropy provides a suitable "information metric" by which to describe any form of algorithmic behaviour via an entropic system working far from equilibrium. This entropic computational gate working far from equilibrium inside a closed system and requiring a source of free energy may be conceptualised as the traditional Maxwell demon [9].

Consider now a much more complex thermodynamic machine consisting of the membrane of the neural surface on which is located an electric field of charge provided by a system of such holes or computational gates. Diagram (1) says that these holes constitute the set of Huygens' secondary sources needed to control the machine so as to compute the physical behaviour of the system of charges which are themselves a system of secondary sources. Conversely, the system of charges may be used to control the system of holes. In either case, the mathematical description of the machine may be interpreted in terms of a manifold M which is the abstract control surface contained within the structure of a Lie algebra of the formal infinitesimal transformations of a pseudo-group Q [10, 11]. In other words, the vectorfields on M represent in this case the formal solutions to a set of partial differential equations which define the infinitesimal transformations of Q and which the neuron may be said to be simulating by means of its system of either computational gates or charges (both constitute a wavefield). But the electric field of charges on the neural surface is exactly equivalent to a "Restricted Coulomb Energy Neural Net (RCENN) [12] modelled by means of the expression $\sum_i Q_i V_i$ where Q_i and V_i are respectively the charges and potentials of the field. The Huygens model, however, uses the actual charges as its components, thereby providing the equivalent of a cortical map on the neural surface. This suggests that biological neurons are probably utilising an 'ultimate' form of evolutionary miniaturisation to achieve their computational effectiveness. Moreover, the occurrence of Heaviside-like operators in the Huygens model suggests a possible explanation for the 'all or nothing' rather than the '0,1' character of neural firing (see also Appendix B).

3 The computational degrees of freedom: spin

What does the Huygens model have to say about the conditions or neural morphology necessary to ensure that such a cortical map on the neural surface is computable and about the nature of its computational degrees of freedom?

In order to answer this question, some consideration must be given to the possibility of a spin associated with each charge on the neural surface. As Feynman has shown, each such spin can be envisaged as constituting a computational switch in a configuration of switches —or digital computer—working by quantum mechanical means [13]. And ultimately each spin itself defines a ray space of rotations in a Hilbert space appropriate to quantum mechanical

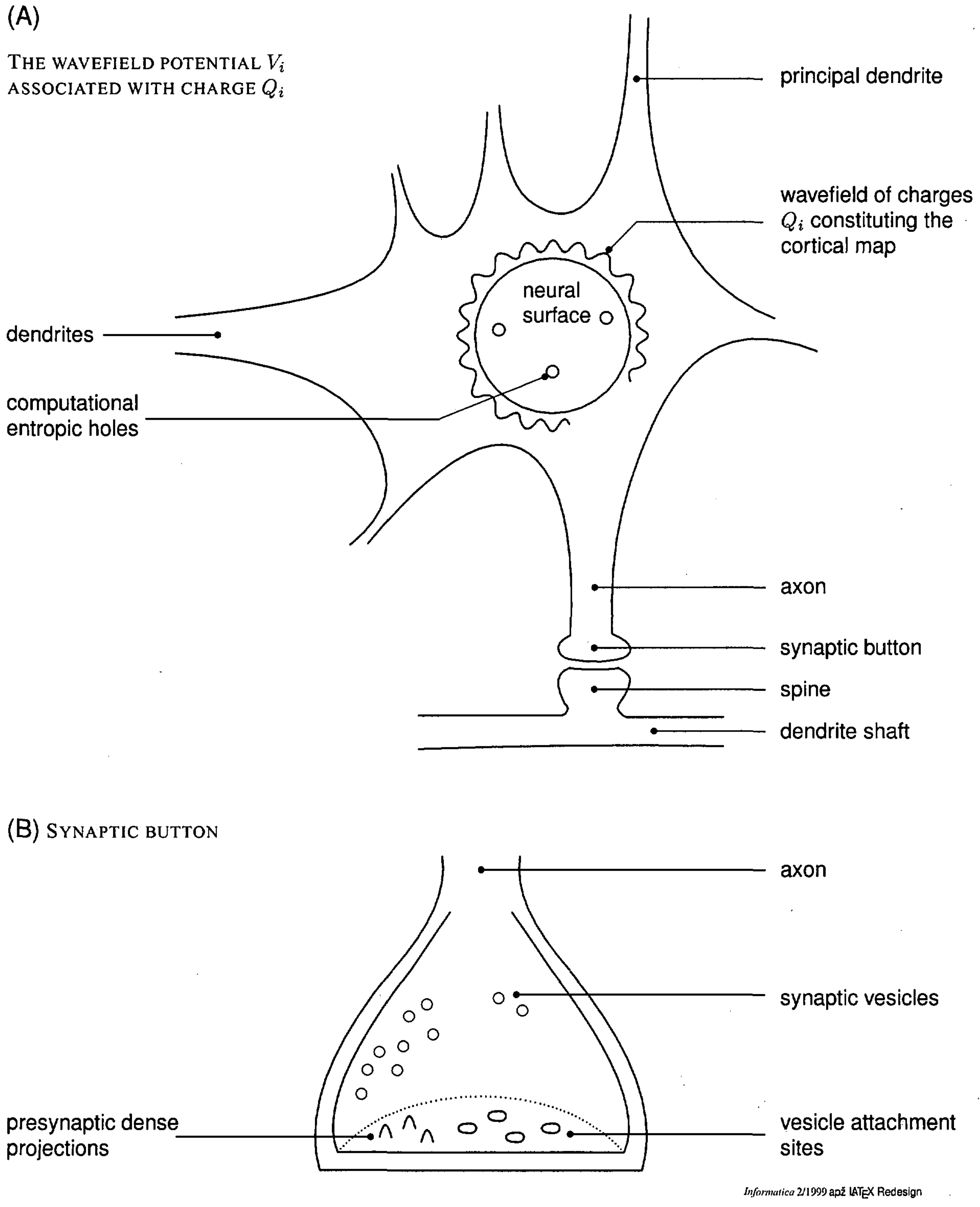

(A)
THE WAVEFIELD POTENTIAL V_i ASSOCIATED WITH CHARGE Q_i
principal dendrite
wavefield of charges Q_i constituting the cortical map
neural surface
dendrites
computational entropic holes
axon
synaptic button
spine
dendrite shaft
(B) SYNAPTIC BUTTON
axon
synaptic vesicles
presynaptic dense projections
vesicle attachment sites
Informatica 2/1999 apž LaTeX Redesign

Figure 1: Neural morphology (taken from Eccles [15])

computation, as described by Deutsch [14]. This implies that each individual charge with spin is itself potentially a universal quantum computer. Such cortical maps are therefore also potentially computer universal [14]. Of course, it may be that the charges on neural surfaces are due to ions without spin in the quantum mechanical sense. Nevertheless, the hole-occupancy of the membrane may be cast into an occupation number representation. In either case, Appendix A indicates that a basis is thus provided for the incorporation of error correction encoding into the Huygens model, with the concomitant evolution of the morphology required to simulate digital processes.

That quantum mechanical considerations may indeed be involved in the control taking place on the neural surface is suggested by the observation that the firing of a neuron causes the release at the synapse of a single synaptic vesicle probabilistically [15] (see Fig. 1). Such a vesicle releases transmitter molecules through the synaptic membrane on a scale of dimensions that is subject to quantum effects. These points are discussed further in Appendix A.

4 Neural morphologies of the Huygens model

Suppose it is required that the simulation on the neural surface by means of the cortical map represent some external behaviour. If the field of charges constituting the cortical map is to simulate a particular form of behaviour in accordance with the Huygens model then the neural morphology must be determined from the following considerations. The solution to the set of partial differential equations which define the infinitesimal transformations Q for the simulation taking place can be computed if this simulation is determined by the Jacobi identity or one of its higher forms, for example

$$\begin{aligned}&(((X;Y),Z),W)+(((Y,X),W),Z)+\\&(((Z,W,),X),Y)+(((W,Z),Y),X)=0\end{aligned}\qquad(3)$$

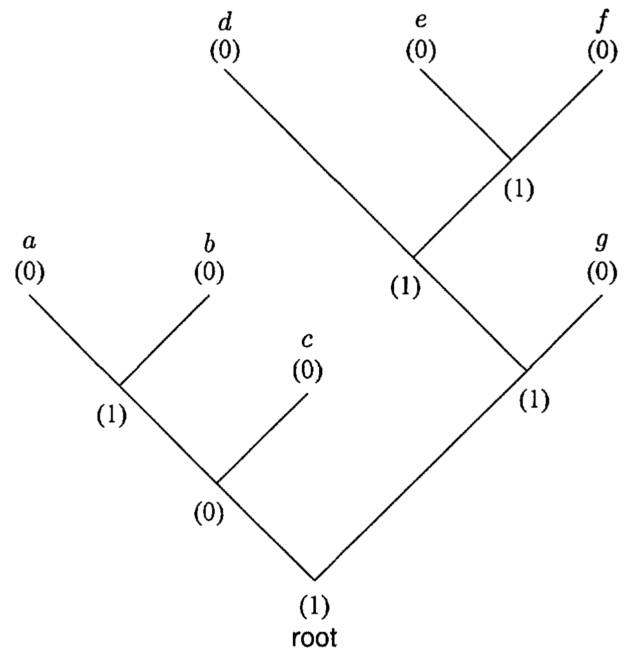
where $(a,b)=ab-ba$ is the Lie product or commutator, and X,Y,Z,W, etc., are operators by means of which the simulation may be described [10, 11].

Now the Lie commutator in the Huygens model is the exact analogue of the Sejnowski covariant estimator used for computing neural net behaviours [16]. Gutfreund has shown that its use ensures that the solution to a finite set of inequalities or ordering relations (equivalent to the operators $X,Y,\ldots$, above) appropriate to the net can always be found, while the use of other estimators leads in general to biased or inaccurate results [17].

But as Hoffman [18] has pointed out, (3) corresponds to a "tree structure" in Lukasiewicz's theory of parentheses (see Figs. 2 and 3). Such tree structures thus define the Huygens morphologies required to produce the desired simulations. The tree structure indicating how signals should be propagated to give a particular prescribed mapping or behaviour should therefore correspond, in the case of the neural surface, to the actual dendrite morphology of synaptic spines. (In more complex cases the neural surface corresponding to the manifold M is segmented into submanifolds each with its own morphology or tree structure).

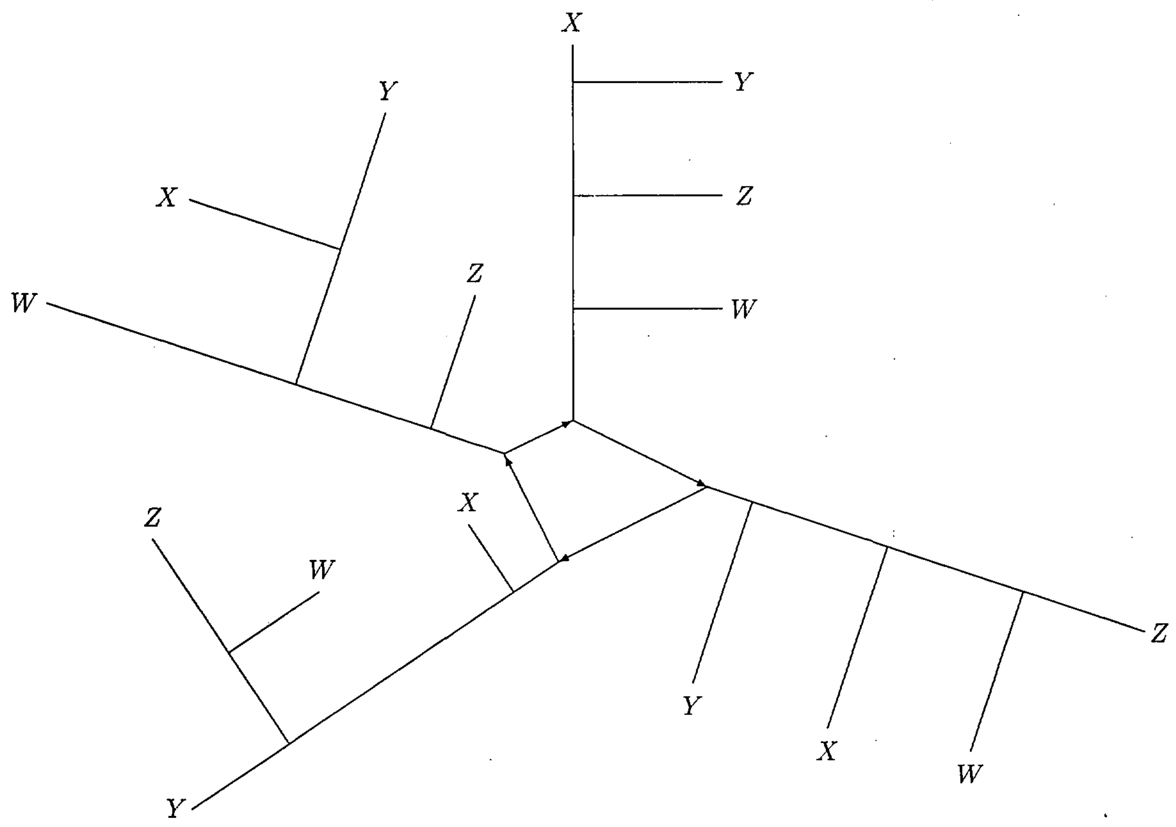
It might therefore be said that a particular morphology 'programs' the cortical map using the configuration of computational gates or entropic holes as its hardware. The model suggests that dendrite morphologies built up as a result of experience should make it possible, through the various combinations of possible vertices or spines, to reproduce simulations of actual experiences along with other variants. These variants may be classed as predictions or fictions, etc.

The Huygens model may also be envisaged as operating on another level where now it is the actual cortices such as the visual cortex which correspond to the mathematical manifold M. The morphology still of course involves vertices or Lie group germs [18] (i.e., Lie products in the Jacobi identity or its higher forms) but where each of these is now interpreted as a neuron rather than a dendritic spine. In this case, therefore, it is the morphology of the neurons themselves which determines the actual cortical maps or simulations.



Informatica 2/1999 apž LaTeX Redesign

Figure 1: Taken from W.C. Hoffman "The Neuron as a Lie Group Germ and a Lie Product" [18]. A Tree representing $((a\cdot b)\cdot c)\cdot((d\cdot(e\cdot f))\cdot g)$ according to Lukasiewicz's theory of parantheses (after Berge [28]).



Informatica 2/1999 apž LATEX Redesign

Figure 2: The tree representing the Jacobi identity higher form, Eq. 3.

5 How the morphologies may be linked

We have described how a system of m charges on the neural surface defined by a system of n entropic holes or computational gates corresponds to an $m \longrightarrow n$ mapping relation or to its inverse $n \longrightarrow m$. Thus in order for the two morphologies to simultaneously model a particular physical property correctly, the dendrite morphology of that property must be the inversion of the corresponding neuron morphology. In this case the two levels must be linked [13, 19] one-to-one by means of a set of unit wires, as indeed they are by actual axons which link the neural surfaces to synaptic spines across a synaptic junction. In this case a particular set of neural cortical maps constitutes in effect the computable secondary sources of a particular property of an actual cortical map and conversely. Such an inversion ensures that a mapping or simulation on a cortex is equivalent to an actual external behaviour being simulated.

In such an inversion all the operators $X, Y, \ldots$ in the Jacobi identities or higher forms correspond to the infinitesimal transformations $\{Li\}$ of a Lie algebra L which form a Lie group G_L such that

$$(L_i, L_j) = \sum_k c_{ij}^k L_k \qquad (4)$$

where the c_{ij}^k are complex constants of the information processing dynamics [10]. Thus the action of any particular inversion will allow the behaviour expressible by its morphology to be linearized and expressed in terms of dynamical invariants. Such invariants are undoubtedly the basis of perception and cognition. Indeed, by analogy with the more usual quantum-mechanical description [20], the right hand side of (4) describes the actual firing behaviour of the assembly of axons required to bring about a particular inversion. In such cases (where the dynamics of a system is expressible in terms of a Lie group G_L); Fatmi and Resconi [21] have shown that according to this model the biological system of neurons will constitute a unified, multiply-ordered, parallel, non-linear analogue computer capable of simulating translation, rotation, Euclidean movement, affine, gauge and other topological transformations. Moreover its description will be expressible in Lagrangian form, ensuring that it corresponds to a possible physical system.

6 The configuration space of the model

Since the two morphologies (dendritic and neuronal) are linked one-to-one by a set of axons, the corresponding set of synaptic mechanisms can be interpreted in the Huygens model as defining the weighted measures of its configuration space. Since T is a suitable weighting operator chosen to give the prescribed behaviour, in an isolated system where for $t = 0$ S_{or} is constant, there is a conservation of sources and, therefore, from diagram (1)

$$S_{\text{or}}^{\text{S}} = \int_S (\text{OP}, \text{T}) F(0) dS = S_{\text{or}} = \text{constant} \qquad (5)$$

(OP, T) being the measure in the configuration space for the particular phenomenon in question. As for the configuration spaces for learning and generalisation in "hidden layered neural net" (HLNN) models [22], it is shown below that these map on to the Huygens model; thus the configuration of the latter model provides the means by which to implement input/output maps [5].

In HLNN configuration space [22], the fractional volume P occupied by configurations which implement a specific behaviour f may be written

$$P(f) = \int dW \rho(W) H_f(W),$$

given a prior density $\rho(W)$ such that

$$\int dW \rho(W) = 1;$$

regions corresponding to the implementation of a specific map f are selected by the Heaviside-like operator

$$h_f(W) = \begin{cases} 1 \text{ if } f_W = f \\ 0 \text{ if } f_W \neq f \end{cases}$$

since every point W selects a specific net which implements the map f_w.

In both HLNN and Huygens models, the theoretical formalism employs Heaviside-like operators to segment the configuration space in order to study a particular simulated property or functional capability. Therefore, in such a configuration space, long-term memory consists of supervised learning in relation to the synaptic weights already established by the totality of experience arriving at the senses, and/or as determined by internal simulations within the domain of that experience, which might be described as 'thought'.

The neuron as amplifier/transducer in the above Huygens model also exemplifies the neural net criterion of Hopfield and Tank that there be a one-to-one relation between the number of amplifiers (one) and possible solutions (one) as in the 'flip-flop' and the 'n-flop'. Thus the computation taking place on the neural surface may correspond exactly to the task assignment problem of Hopfield and Tank [23]. In this case the open entropic holes are the configuration of internal amplifiers in the +1 state, and the inputs are the potentials V_i appropriate to the system of charges Q_i, where the V_i are a consequence of the actions taking place at the dendritic spines. In other words, each neuron once again contains within itself the germ or mechanism for a solution which is that of an NP complete problem. But since it is known that once one NP complete problem can be solved, so potentially can any other, each neuron may be classified as a universal computer of the NP complete type.

Furthermore, since each neuron is an amplifier of this type it may be configured into an optimizing circuit which is a matrix of such amplifiers [23]. In each row and column of this matrix, there will be inhibitory connections which provide the constraint that only one amplifier or neuron in any given row and column can be in the +1 state. That is, such a matrix provides the architecture for a system of concurrent computation by individual computers of the NP complete type. Thus the processes taking place on the neural surface in the Huygens model are replicated on yet another level.

7 Discussion

If biological brains are implemented in accordance with the Huygens model we have proposed, as the above descriptions and the extensive investigations [11, 18, 24, 25, 26] of Hoffman indicate to be the case, then the many advantages of Huygens' architecture begin to become apparent.

Brains consist of a hardware implementation of neural net methodologies on a scale such as neural surface cortical maps which it will be very hard to emulate and probably impossible to surpass. This implementation gives immense power; in the human brain it consists of a system of 10^{11} linked highly complex neural nets or cortical maps $n \longrightarrow m$ where n is the number of entropic holes and m is the number of charges or ions on the neural surface, even neglecting the quantum-mechanical computational possibilities (see Appendix A).

All forms of signal, whether visual, auditory, tactile, etc., would then be ultimately encoded in the machine in a common representation, the 'reduced Coulomb energy representation', without the need for any internal formal knowledge representation. The representation is implicit in the Huygens model in the way that the common technology of the machine is implemented. It is indeed a knowledge representation since the external behaviour in the form, say, of an auditory or visual wavefield can be replicated exactly by the restricted Coulomb energy wavefields. Such an exact replication is possible by using the totality of the information arriving at the sensors of the machine in the form of both the amplitude and the phase of the incoming waves. That this can be done and does actually occur is shown by Heiligenberg's recent study of coding and processing of electrosensory information in *Gymnotiform* fish [27]. For navigation and other purposes, the fish utilises the restricted Coulomb energy wavefield produced by its electric discharge organ, which returns from the surroundings onto a set of receptors located on the fish's body surface. Thus, this set of receptors corresponds to the manifold M in the Huygens model as described above.

8 Conclusion

Many properties of the Huygens model proposed here appear to be consistent with the brain's neurophysiological morphology. The mathematical transformational properties of this model may thus be used with some confidence to described the information-processing characteristics of living systems.

Appendix A

In the paper relating Huygens' principle to computability [4], it is shown that under appropriate conditions the entropic holes of the biological membrane of the neural surface may simulate the behaviour of the more usual digital hardware computational gates or 0,1 switches.

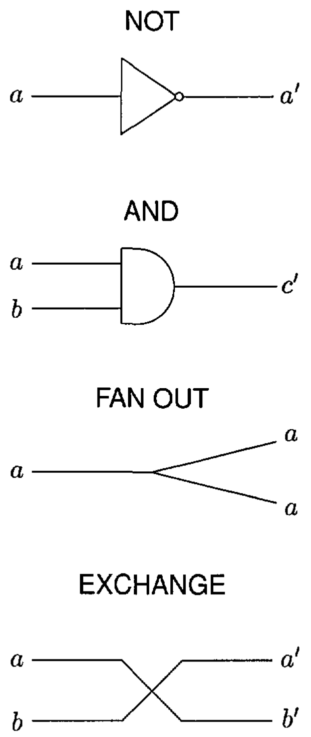
A new morphological requirement must be imposed if the brain is to establish the control necessary for such conditions. Since this morphological requirement is additional to those already described, it must therefore operate on levels of scale above and below those already utilised.

On the lower level of scale the charges of the restricted Coulomb energy representation must act like switches in accordance with the Feynman model for a set of charges with spin working as a digital computer by quantum mechanical means [13]. Then according to the Huygens model, the morphology of the brain at the higher level of scale must constitute an inversion of Feynman's model.

Simplistically this suggests that the brain be morphologically divided into two separate sub-brains or supercortices, the left and right brains, corresponding to each spin state 0 and 1 and joined one to one by unit wires at this new higher level (using axons as before). This is indeed the case [15]. In fact Feynman [13] shows that the four primitive logical elements (see Fig. 4) in addition to unit wires, essential to any primitive computing machine are NOT, AND, FANOUT and EXCHANGE. NOT and AND (or just the one element NAND = NOT AND) suffice in the usual 'classical' analysis which assumes there to be two standard voltages representing the local 1 and 0.

However in the Huygens model these can only be provided by local dynamical invariants, so that FAN OUT corresponding to the Jacobi identity (or one of its higher forms) and the linked morphologies of Sect. 5 are required. This leaves EXCHANGE, as constituted by the Lie products (a, b) themselves, as the only possible representations in the Huygens model for the actual Huygens morphology that can be implemented at the level of the supercortices or left and right brains, if simulation of digital computation within the Huygens machine is to be possible [4].

This 'closure' of the Huygens model with the basic Lie products as the fundamental units of the model at both the lowest and highest levels of its information processing morphology not only corresponds with the morphology observed in the human brain, but ensures that (a) the whole brain itself is a complete cortical map and (b) each



a	a'
0	1
1	0

a	b	c'
0	0	0
0	1	0
1	0	0
1	1	1

Informatica 2/1999 apž LaTeX Redesign

Figure 3: The four primitive logical elements.

and every Lie product or germ is potentially covariant and calculable.

We believe therefore that it ought to be possible to test experimentally the notion that the two supercortices are essential for the proper exercise of the human brain's logical faculties as our model requires. In any event, a consequence of the one-to-one mappings at this new, higher level of the supercortices would be the appearance at the preexisting levels of principal dendrites, one per neuron where required, each having dendritic spines without synaptic connections (since these are now effectively in use in the higher level morphology). This is exactly the observed morphology of the human brain.

In the Huygens model, digital computation seen in terms of spin corresponds to operations $\times$ such that [29]

$$(B, A_1 \times A_2) = (B, A_1) \times A_2 + A_1 \times (B, A_2) \quad \text{and}$$
$$2A_1 \times A_2 = A_1A_2 + A_2A_1 = \{A_1, A_2\}$$

where the Clifford product or anticommutator $\{,\}$ (the quantization products of which in quantum theory are fermions) leads to Clifford algebras in the same way that the Lie product or commutator $(,)$ leads to Lie algebras. The immediate consequence of this is that brains (where the charges on the neural surface have spin) would behave like spin glasses on an Ising-like model [30]. Sourlas [30] has shown, this additional property would provide the information processing morphology of the Huygens model with the potential of an internal error correcting code which, under the right conditions, constitutes the first known codes to saturate Shannon's well known cost performance bounds for the transmission of information. And this would ensure that in the Huygens model optimally efficient Gray codes—which are simply connected—are brought into use, rather than those of ordinary binary arithmetic (used in conventional digital computers) which are multiply connected and therefore can be very inefficient [31].

Similarly, this aspect of the switching properties of neural firings will be analogous to fermionic behaviour involving the Pauli exclusion principle. In this case the logical operations will satisfy Wittgenstein's principle [32], namely that the sense of the logical proposition which each firing determines can only be given once, and there is necessarily only one proposition for each fact that answers to it. The consequences are therefore that the neural firing activity appropriate to any such inversion will be minimised, and that the restricted Coulomb energy representation on each neural surface is to be regarded as constituting a unique fact or labelling of such activity. This is essential since in classical computational machines the sets of input and output states must be labelled in some canonical way [14].

A specific case of what would happen on the neural surface in the Huygens model for the above morphologies is given in Appendix B with respect to the simulation of a solution to a boundary value problem. This is in accordance with Bowden's [33] identification of Huygens' principle with Kron's method of 'tearing', the solution of a boundary value problem by its division into the solution of appropriate sub-problems and a problem on the intersection network by means of which the subproblems overlap [34]. It is chosen to illustrate in detail the behaviours that are possible on a neural surface in the Huygens model, because Kron's method is a well established and well validated if little known problem solving methodology that has been recently employed for concurrent information processing on transputer arrays [35].

Finally it seems reasonable to postulate a yet higher morphology in which the charges are universal quantum computers in accordance with the Deutsch model [14]. These would require the existence of weighting operators W such that

$$R(F(f)) = W(R(f)) \qquad (6)$$

for each vector f in the Hilbert space; R is a rotation in the Hilbert space and represents the simplest possible way of obtaining the associated self-adjoint linear operator F. Eq. (6) follows from the categorical diagram

$$\begin{tikzcd}
f \arrow[r, "\mathsf{OP=R}"] \arrow[d, "F"'] & R(f) \arrow[d, "W"] \\
F(f) \arrow[r, "\mathsf{OP=R}"] & R(F(f)) = W(R(f))
\end{tikzcd}$$

for the Huygens model (which would commute under the conditions of the previous morphology for the simulation of digital computation), and thus results in a similar linearization to that of Eqs. (3) and (4) in the main text. That is, the spin associated with the charge emulates the ray space of the quantum mechanical rotations in Hilbert space. And the inversion of this ray space morphology now occurs in the form of the synaptic vesicles themselves which are an actual morphological dual of the ray space; these are released probabilistically at the synaptic junctions following the firing of a neuron [15]. It has a further morphological requirement that the principal dendrites postulated previously must become grouped into 'columns' or spin states.

That these additional morphologies are actually observed provides further evidence that the Huygens model may indeed furnish a basis on which to describe the information processing taking place in brains. Their existence also leads to a truly astonishing assessment of the potential computational powers of the human brain since each charge on the neural surface is then potentially a universal quantum computer 'programmed' or controlled by means of the synaptic vesicles. It appears to have reached the limits of what is conceivably possible and leaves in no doubt the statement that human brains are the most complex organised systems ever devised and, perhaps, that it may be possible to devise. We can only gaze in awe at Nature's evolutionary triumph of technology.

Appendix B

Bowden's identification of Huygens' principle with Kron's method [34] of 'tearing' (the concurrent solution of a boundary value problem by its division into the solution of appropriate subproblems together with a problem on the intersection network by means of which the sub-problems overlap) centres on the commutative diagram corresponding to Huygens principle [4]. Thus,

$$\begin{tikzcd}
x \arrow[r, "Z"] & b - CY \\
\begin{vmatrix} x \\ y \end{vmatrix} \arrow[u, "s=[IO]"] \arrow[r, "\begin{vmatrix} Z & C \\ C' & Y \end{vmatrix}"'] & \begin{vmatrix} b \\ c \end{vmatrix} \arrow[u, "{[IO] - C(C'Z^{-1}C - C)^{-1}[C'Z^{-1} - I]}"']
\end{tikzcd}$$

where Z is the block diagonal system matrix of the boundary value problem;

Y	is the intersection network system matrix;
C'	is the partitioned connection matrix $[C1, \ldots, C'k]$ and C is the corresponding covector;
x	is the partitioned vector of subsystem solutions;
y	is the intersection network solution vector;
b	is the partitioned vector of subsystem boundary conditions;
c	is the vector of intersection network boundary conditions.

If it is required to solve for the ith system only, then s will take the special form $[0, ...0, 1, 0...0]$ where 1 is in the ith position only and the intersection vector is a holographic field of secondary sources containing all the information from the boundary conditions.

Thus we may identify each s in the special form with the opening or closing of an entropic hole on the neural surface; the intersection vector in the form of a holographic field containing all the information from the boundary conditions is stored on the neural surface as a dynamic field of charges Q_i subject to potentials V_i. In fact, if we consider the electrostatic equation

$$\mathrm{div}\, D = \rho$$

where the charges ρ are contained within a surface defined by s, then the secondary sources necessary to reproduce the field outside the surface are given from Huygens principle by

$$\begin{aligned} S^{\mathrm{S}}_{\mathrm{or}} &= (\mathrm{div}, s)D \\ &= \mathrm{div}(sD) - s\,\mathrm{div}(D) \\ &= D\,\mathrm{div}(s) \end{aligned}$$

and

$$\begin{tikzcd} D \arrow[r, "\mathrm{div}"] \arrow[d, "s"'] & \rho \arrow[d, "s'=s+(\mathrm{div},s)\mathrm{div}^{-1}"] \\ sD \arrow[r, "\mathrm{div}"] & D\,\mathrm{div}(s) \end{tikzcd}$$

which gives the field D in the region outside the surface in terms of the secondary sources on the surface which in their turn are calculated from the original sources ρ inside. The similarity with the earlier commutative diagram for Kron's method is clear.

Moreover, since the submatrices Z_i in a higher dimension are themselves of the form

$$\begin{vmatrix} Z & C \\ C' & Y \end{vmatrix}$$

where Z is block diagonal, this recursive behaviour extends naturally to the solution of multi-dimensional systems, in the manner that the Huygens morphology of the brain requires. Then the firing of a neuron corresponds mathematically to a function of the type s defined with respect to a larger boundary value problem relating to a cortical map on a cortex.

References

[1] JESSEL, M. 1985. Sound recording and reproduction. Impact 35:2/3:91–109.

[2] JESSEL, M. 1988. From Huygens' principle to Huygens' machines. *In* M. Jessel, Ed. Proc. France-Japan Meeting on Acoustical Information Processing. 5/9/1988. CNRS–LMA. Marseille, France.

[3] RESCONI, G. & M. JESSEL. 1986. A general system logical theory. Intern. J. of General Systems 12:159–182.

[4] COVENEY, P.V., M. JESSEL & P.J. MARCER. 1990. Huygens' Principle and Computability. Speculations in Science and Technology. In press.

[5] FATMI, H., M. JESSEL, P. MARCER & G. RESCONI. 1990. Theory of cybernetic and intelligent machines based on Lie commutators. Int. J. of General Systems 16:123–164.

[6] JESSEL, M. 1954. Une formulation analytique du principe de Huygens. Comptes Rendus 239:1599–1601.

[7] JESSEL, M. 1973. Acoustique theorique: propagation et holophonie. Masson. Paris.

[8] ZUREK, W.H. 1989. Thermodynamic cost of computation, algorithmic complexity and the information metric.Nature 341:6238:119–124.

[9] ZUREK, W.H. 1989. Algorithmic randomness and physical entropy. Physical Review A40:8:4731–4751.

[10] COHN, P.M. 1957. Lie Groups. Cambridge University Press.

[11] HOFFMAN, W.C.. 1970. Higher visual perception as prolongation of the bases Lie transformational group. Math. Biosci. 6:437–471.

[12] COOPER, L. ET AL. 1990. Systéme d'apprentissage basé sue des réseaux de neurones multiples. Translated from English and republished in the Inter. Conf. Neural Networks, Biological Computer or Electronic Brains (ICNNBCEB). Lyon 6–8 March, 1990.

[13] FEYNMAN, R. 1986. Quantum mechanical computers. Foundations of Physics 16:6:507–531.

[14] DEUTSCH, D. 1985. Quantum theory, the Church–Turing principle and the universal quantum computer. Proc. Roy. Soc. Lond. A400:97–117.

[15] ECCLES, J. 1986. Do mental events cause neural events analogously to the probability fields of quantum mechanics? Proc. Roy. Soc. Lond. B227:411–428.

[16] SEJNOWSKI, T.J. 1990. Neurobiological models of synaptic plasticity. Proc. Lyon Conf. (ICNNBCEB) March 6–8, 1990. In press.

[17] GUTFREUND, H. 1990. Learning in neural networks as a problem in statistical physics. Proc. Lyon Conf. (ICNNBCEB) March 6–8, 1990. In press.

[18] HOFFMAN, W.C. 1968. The neuron as a Lie group germ and Lie product. Quart. J. Appl. Math. 25:4:423–440.

[19] FREDKIN, E. & T. TOFFOLI. 1982. Conservative logic. Intern. J. Theor. Physics 21:219–253.

[20] DIRAC, P.A.M. 1947. Principles of Quantum Mechanics. Oxford Univ. Press.

[21] FATMI, H. & G. RESCONI. 1988. A new computing principle. Nuovo Cimento, 101B:2:239–242.

[22] SOLLA, S.A. 1990. Supervised learning and generalization. Proc. Lyon Conf. (ICCNBCEB) March 6–8, 1990. In press.

[23] TANK, D.W. & J.J. HOPFIELD. 1987. Collective computation in neuronlike circuits. Scientific Amer. (Dec 1987):62–70.

[24] HOFFMAN, W.C. 1966. The Lie algebra of visual perception. J. Math. Psych. 3:65–98.

[25] HOFFMAN, W.C. 1980. Subjective geometry and geometric psychology. Math. Modelling 1:349–387.

[26] HOFFMAN, W.C. 1989. The visual cortex is a contact bundle. Appl. Math. and Comp. 32:137–167.

[27] HEILIGENBERG, W. 1989. Coding and processing of electrosensory information in gymnotic fish. J. Exp. Biol. 146:255–275.

[28] BERGE, C. 1962. The Theory of Graphs and Its Applications. Wiley & Sons. New York.

[29] HERMANN, R. 1970. Lie algebras and quantum mechanics. W.A. Benjamin. New York.

[30] SOURLAS, N. Spin-glass models as error-correcting codes. Nature 339:693-695.

[31] CLEMENT, B.E.P. Spin-glass in a whirl. Nature 340:514.

[32] WITTGENSTEIN, L. 1975. Philosophical Remarks. Oxford Univ. Press.

[33] BOWDEN, K. 1990. On general physical system theories. Intern. J. of General Systems 18:61–79.

[34] KRON, G. 1963. Diakoptics: The Piecewise Solution of Large-Scale Systems. McDonald. London.

[35] BOWDEN, K. 1990. Kron's method of tearing on a transputer array. British Computer Society Journal 33:5:453–459.

Electroencephalographic (EEG) Correlates Of Some Activities Which May Alter Consciousness: The Transcendental Meditation Technique, Musicogenic States, Microwave Resonance Relaxation, Healer/Healee Interaction, And Alertness/Drowsiness

D. Raković[1], M. Tomašević[2], E. Jovanov[1,3], V. Radivojević[4], P. Šuković[1], Ž. Martinović[4] , M. Car[5], D. Radenović[1], Z. Jovanović-Ignjatić[6], and L. Škarić[1]
[1]Faculty of Electrical Engineering, PO Box 35-54, Belgrade, Yugoslavia
E-mail: rakovic@buef31.etf.bg.ac.yu
[2]"Vinča" Institute of Nuclear Sciences, PO Box 522, Belgrade, Yugoslavia
[3]Institute "M. Pupin", Computer Systems Dept., PO Box 15, Belgrade, Yugoslavia
[4]Institute for Mental Health, Dept. of Clinical Neurophysiology, Belgrade, Yugoslavia
[5]Institute for Biological Research, Center for Multidisciplinary Studies, Belgrade, Yugoslavia
[6]Private Medical Practice "LAV", Belgrade, Yugoslavia

Keywords: EEG correlates, consciousness, altered states, transcendental meditation, musicogenic states, microwave resonance relaxation, healer/healee interaction, alert wakefulness, drowsy wakefulness

Edited by: A.P. Železnikar

Received: March 23, 1999 **Revised:** August 19, 1999 **Accepted:** September 7, 1999

A key problem of finding the most complete and useful theory of consciousness may revolve around how to empirically determine different styles or states of consciousness and how to incorporate these within a single paradigm. This was our motivation to start examination of EEG correlates of some activities or substates of consciousness which occur spontaneously or are induced artificially. Our investigations demonstrated more or less characteristic features in 25 subjects practicing the transcendental meditation program (increased β power in prefrontal region, increased θ power in left frontal and right temporal regions, increased α power in both temporal regions, and correlation between increased α power and decreased correlation dimension), 6 subjects with 4 types of spiritual music provided to induce musicogenic states (with significant changes in only 3 cases out of 24, where increased θ and α power was observed in only those subjects who have described their musical experiences as very pleasant), 28 subjects of relaxation induced by microwave resonance therapy applied to corresponding acupuncture points (with slightly decreased EEG power in all frequency bands, especially in the left central region, which can be ascribed to higher activation of the stimulated left circulatory part of the acupuncture system; it should be also noted that persons not previously subjected to this treatment responded stronger, presumably as a consequence of the more imbalanced acupuncture system), 5 healer/healee, noncontact interactions (with increase in the maximum mean coherence of their EEG patterns in the α band observed only in short 4 s time intervals), and 30 subjects for monitoring alertness/drowsiness level (with implemented automatic procedure of the neural network classifier to assess the correlation between EEG power spectrum fluctuations related to changes of vigilance level, demonstrating linear separability of the states of alert wakefulness and drowsy wakefulness, allowing very fast data processing and possible real time applications in clinical practice). New technologies applied to the EEG may permit rapid and reproducible identification of different styles or states of consciousness. Such a tool might be useful in evaluating the effectiveness of different techniques for stress reduction and for altering expressions of consciousness.

1 Introduction

The key problem of any future theory of consciousness is how to incorporate altered states of consciousness [1-3] (REM sleep, meditation, hypnosis, psychedelic drug influence, some psychopathological states, near-death experiences, ...) within a new paradigm. It should be pointed out that purely biochemical mechanisms of the extended reticular-thalamic activating system (serving as a selector and amplifier of the conscious content out of many other currently processed nonamplified contextual unconscious contents) are not accelerated up to several orders of magnitude, as the subjective time sense is dilated in altered states of consciousness - in respect to the normal awake state. According to the biophysical model for altered states of consciousness (proposed by one of us, D. R. [4-6]), the electromagnetic (EM) component of ultralowfrequency (ULF) brainwaves, related to "subjective" reference frame of consciousness, enables perfect fitting with narrowed-down limits of conscious capacity in normal awake states and very extended limits in altered states of consciousness - due to the biophysical relativistic mechanism of dilated subjective time base. In this model, consciousness is subtle internal display in the form of EM component of the "optical" microwave (MW)/ULF modulated quantum holographic neural network-like acupuncture ionic system, in which a complete information (both conscious and unconscious) is permanently coded from brain's neural networks, as a spatio-temporal pattern resulting from changes of the electrosynaptic interconnections in the neural networks of the brain.

Then, according to this model, altered states of consciousness are a consequence of partial displacement of the MW/ULF ionic acupuncture system outside the body (when the embedded EM component of ULF brainwaves is propagating through this weakly ionized structured gaseous medium of low-dielectric relative permittivity, $\varepsilon_r \approx 1$), while normal states of consciousness (alert state, non-REM sleep, ...) are achieved when there are no such displacements (when brainwaves are propagating only through the structured brain tissue of high-dielectric relative permittivity, $\varepsilon_r \gg 1$)! The displaced ionic structure in this model must have a form of weakly ionized gaseous "optical" MW/ULF neural network, for continually inflowing ULF information from the brain's neural networks to be "subjectively" recognized. Also, such ionic neural network may behave as an optical sensor, which can even perceive an environment extrasensory, as reported by reanimated persons [3].

The afore-mentioned relativistic mechanism also enables the dream-like mixing of the normally conscious and unconscious contents in altered states, due to the relativistic Doppler mapping of EM component of the "objective" ULF brainwaves power spectrum on the zero-degenerate frequency "subjective" one - revealing their psychological significance: in dreams one has continuous access and more efficient "subjective" integration of normally conscious and unconscious contents, giving rise to integration and growth of human personality (otherwise divided into conscious and unconscious associative "ego states"), which also results in alleviation of emotional conflicts. Therefore, meditation, as a prolonged altered state of consciousness, enables more efficient "subjective" integration of human personality, this being its major role in the growth of human personality.

Even most peculiar spatio-temporal transpersonal interactions are predicted in transitional states of interchange of normal and altered states of consciousness (when brainwaves traverse from high-dielectric ($\varepsilon_r \gg 1$) to low-dielectric ($\varepsilon_r \approx 1$) state or vice versa, the relative velocity $v = c_0/\sqrt{\varepsilon_r}$ of "subjective" reference frame being therefore subjected to abrupt change in short transitional period $\tau \sim 0.1$ s, with "subjective frame" acceleration $\sim c_0/\tau \sim 10^9$ m/s^2) - due to the relativistic generation of so-called wormholes in highly noninertial "subjective" reference frame - fully equivalent, according to Einstein's Principle of equivalence, to extremely strong gravitational fields where generation of wormholes (or Einstein-Rosen space-time bridges, whose entrance and exit could be in very distant space-time points) is theoretically predicted [7]. It should be pointed out that apart from the EM field, the displaced part of ionic acupuncture system (in the form of MW/ULF ionic neural network, having the "optical" sensory function), must also be tunneled in such (acausal) interactions of consciousness with distant events in space-time!

This was our motivation to start examination [8] of EEG correlates of transcendental meditation, musicogenic states, microwave resonance relaxation, healer/healee interaction, and alertness/drowsiness, as some of relatively easily reproducible altered states of consciousness.

2 Method

Electroencephalographs were recorded in electromagnetically shielded room by a MEDELEC 1A97 EEG machine, with lower and upper band-pass filter limits set at 0.5 Hz and 30 Hz, respectively. Ag/AgCl electrodes with impedance less than 5 kΩ were placed at 16 locations (F7, F8, T3, T4, T5, T6, Fp1, Fp2, F3, F4, C3, C4, P3, P4, O1, O2) according to the International 10-20 system with average reference. The EEG outputs were digitized with 12-bit precision at a sampling rate of 128 Hz per channel using A/D converter Data Translation 2801.

The length of each EEG-trace was 60 s (7680 points). Time-varying EEG spectra (spectrograms) with 0.5 Hz resolution were calculated by the MATLAB program using a 256-point FFT algorithm performed on 2 s Hamming-windowed, half-overlapped epochs. An array of EEG partial power spectra for each subject and each derivation was computed by integration by the trapezoidal rule of the spectrogram over the five frequency bands: δ (1-4 Hz), θ (4-8 Hz), α (8-13 Hz), β_1 (13-18 Hz), and β_2 (18-25 Hz).

2.1 Transcendental meditation

In transcendental meditation (TM), normally practiced for periods of twenty minutes twice a day, the subject is instructed to sit quietly with eyes closed and is then taught to repeat a certain type of sound or "mantra" according to a particular definite set of instructions. The mantras are a set of short speech sounds, meaningless in themselves, preserved by an ancient vedic tradition and assigned to individuals by the instructor on the basis of a set of objective rules which is trained to apply. They are chosen so as to "resonate" with the structure of an individual nervous system.

Subjectively, the meditator usually reports an immediate sense of bodily quiet and relaxation. An important feature of the subjective experience of the TM technique is the "expansion" of consciousness [9]. As the mantra is experienced in successively finer stages, subjects report that the spatial extent of conscious self-awareness, which ordinarily seems to be localized in the area of head and upper body, undergoes a progressive expansion.

Wallace, Orme-Johnson, and Farrow [10], among others, have reported physiological changes during the practice of the TM technique that are consistent with these predictions, such as reduced oxygen uptake, reduced CO_2 elimination, constant respiratory quotient, reduced respiratory minute ventilation, reduced respiratory frequency, reduced hearth rate, increased basal skin resistance, and EEG changes indicative of alertness. Wallace was led to propose that the TM technique produces a fourth major state of consciousness in which the mind remains alert while mental activity reaches a least excited state.

The study was carried out on 25 healthy adult volunteers who had been practicing the TM technique from 0.2 to 25 years, with a mean of 7.2 years. There were 15 males and 10 females, whose ages ranged from 17 to 68 years with a mean age of 32 years. All subjects were free of any medication. Prior to the experiment, subjects were informed verbally about all aspects of the experimental procedure.

The experiment was conducted in a soundproof room, dimly lit for observation. Subjects were seated comfortably. Each recording session was divided into two sequential periods: (1) relaxing 5 min with eyes closed and (2) meditating 15 min. During those periods two random samples, one minute each, were recorded for every subject. The EEG record was stored on a hard disk.

2.2 Musicogenic states

The study of the perception of music is a paramount example of multidisciplinary research, in which musicians, psychologists, neurobiologists, physicists, and engineers must communicate and work together. This study comprises three broad problem areas [11]: (a) perception of musical tones; (b) interpretation of acoustical information relevant to music; and (c) emotional response to musical messages. In the past two decades, a considerable mutual integration of these three problem areas has taken place, due to the progress in the understanding of general human brain functions, and the recognition that in the conscious state even the simplest perceptual events are bound to trigger operations that involve the brain as a whole.

It should be also pointed out that one of the most profound consequences of the evolution of human brain functions (and human consciousness itself!) has been the emergence of systematic postponing of behavioral goals and rearrangement of behavioral priorities. This led to conflicts between cortical functions and those of the limbic system: while in animals the limbic system is mostly activated by environmental and somatic input, in humans it can also respond to internally evoked images displayed on the cortex during the process of thinking. As motivation and emotion are integral manifestations of limbic function (assuring that all cortical processes are carried out so as to be of maximum benefit of organism, through the extended reticular-thalamic activating system [12]), in humans they can be triggered with no relationship to the instantaneous state of the environment. It is along this line that we should seek a lead toward understanding the human emotional response to music (and to art in general!), when the messages therein seem to be of no obvious survival value [11]. It might even be that deep artistic experiences of spectators could have strong spiritual note through spontaneous spectator's mental addressing on the masterpiece (exciting him in transitional state of consciousness), and through it on the illuminating idea related to the artist in the moment of masterpiece creation [5].

This motivated us to address our question on similar possible neurobiological origin of musicogenic altered states of consciousness, induced by deep spiritual music of different cultures [13], and its possible EEG correlates. The analogous more frequently used physical mechanisms for sound-induced altered states of consciousness is an introspective repeating of a certain type of sound or "mantra", which is chosen so as to "resonate" with the structure of an individual nervous system [9]. The sound resonances within the human lobe would be then achieved through a formation of standing sound waves, with principal harmonic (of $\sim$ 1000 Hz) having its maximal amplitude in the centre of the lobe cavity, i.e. around the region of limbic system - therefore inducing the local stimulation of thalamic formation through some mechano-chemical receptors (to be still specified therein).

The study was carried out on 6 healthy adult volunteers. There were one male and five females, whose ages ranged from 18 to 29 years with a mean age of 25 years. All subjects were free of any medication. Prior to the experiment, subjects were informed verbally about all aspects of the experimental procedure.

The four types of spiritual music were provided to the subjects during experiments: (1) The Indian Bhajan in Sanskrit, (2) The Byzantine Easter liturgy in Greek, (3) The Maronit Song in Arabian, and (4) Mozart's Requiem in Latin.

The experiment was conducted in a soundproof room, with only one music a day. Each recording session was divided into three sequential periods: (1) relaxing 5 min with eyes closed; (2) listening of the music 10 min; and (3) after listening, 5 min. During those periods three samples, one minute each, were recorded for every subject. The EEG record was stored on a hard disk.

2.3 Microwave resonance relaxation

The application of microwave resonance therapy (MRT) in biomedicine is a new trend [14,15], revealing sharply-resonant characteristic eigenfrequencies and sensory responses of the disordered human organism, when applied upon acupuncture points. Such quantum-like coherent characteristics of MRT inspired Sit'ko and coworkers to propose a quantum physics of alive [15,16], considering biological structures as a macroscopic quantum systems with nonlocal selfconsistent macroscopic quantum potentials, giving rise to nonlinear coherent EM MW long-range maser-like excitations of biological nonlinear absorption medium with the cells as active centers. This quantum picture can be more simply visualized [17] considering acupuncture system as a dynamic structure assembled at the locations of maximums of the interference three-dimensional (*3D*) standing waves resulted in reflections from the skin of the nonlinear coherent EM MW Fröhlich excitations [18] of strongly polarized molecular subunits in the cell membranes and cytoplasmatic proteins, supported also by other investigations which suggest that gating and assembly of gap junctions (of higher density at acupuncture points and meridians [19]) is a dynamic process sensitive to electrochemical potential of the cell [20] - which might be therefore stimulated at spatio-temporal maximums of MW EM field of the organism [17].

In the above context the explanation for the efficiency of the MRT, as a noninvasive biophysical medical treatment, should be sought [17]: some disorders in the organism (related to local changes of dielectric properties of tissues and organs) give rise to deformation in the standing wave structure of the MW EM field of the organism, which influences corresponding changes in the spatio-temporal structure of the acupuncture system, and consequently resonance frequencies of its meridians, resulting in some disease. During MRT therapy, applying the MW source at a corresponding acupuncture point the excited meridian of patient's acupuncture system relaxes to the previous healthy condition, while reaching its normal resonance frequency responses upon the wide spectrum MW source - and following physiological mechanisms of acupuncture regulation (via nervous and endocrine systems [21]) the organism biochemically overcomes the disease.

The MRT was applied by the wide spectrum apparatus POROG-3, and the measurement of frequency was achieved by the narrow spectrum apparatus AMRT-01, adjusted manually. The frequency range of the POROG-3 is 52-78 GHz. Up to 10 mW low-power microwave generators, of the output power density of 0.2-5 μW/cm^2 - much lower than biologically limited 10 mW/cm^2 during 8 hours, as prescribed by USA National Standards, or 10 μW/cm^2 during 8 hours, as prescribed by Russian and Ukrainian National Standards [22] - are power supplied by the 220±22 V/50 Hz a.c. or the autonomous 4.5 V d.c. The output power density as well as the seance duration significantly influence the MW absorbed dose and corresponding MRT bioeffects, which can be biostimulative for low-level therapeutically recommended doses of typical 20 min daily MRT treatments (causing local temperature increases up to 38°C, with maximally fast bioeffect), and biodepressive and even biodestructive for much higher doses (causing much higher and harmful local temperature increases) [22]. MRT generator was applied on acupuncture points in following order: Du 20, and the left-side points Li 4, Pc 6, H 7, and Ap 55, which resulted in relaxation, similarly to the parasympathicus effect [21]. The choice of the acupuncture points for the relaxation seance was achieved on the basis of well known principles of acupuncture stimulation, characteristics of the chosen points, and the therapeutist experience.

The study was carried out on 28 healthy adult volunteers (13 males and 15 females).

The experiment was conducted in a soundproof room, dimly lit for observation. Subjects were laid comfortably. Each session was divided into three sequential periods: (1) relaxing 5 min with eyes closed; (2) MRT 20 min; and (3) relaxing 5 min with eyes closed. During the first and third periods two random samples, one minute each, were recorded for every subjects. The EEG records were stored on a hard disk.

2.4 Healer/Healee interaction

The transitional states of consciousness [5,6] are presumably the basis of most transpersonal phenomena [23] - being really described by seldom practitioners as not subjected to spatio-temporal limitations [24-26] - providing also explanation for their transitional nature and poor reproducibility: they elapse only $\sim$ 0.1 s, and spontaneous conditions for them are achieved only every 1.5-2 hours, with periodicity of ultradian rhythms which govern the interchange of normal and altered states of consciousness [27]. However, it should be noted that the non-low-dielectric barriers in interaction with the low-dielectric displaced part of ionic acupuncture system are helping in overcoming themselves in such induced transitional states - quite opposite to normal experience in usual mechanical interactions - enabling even their deliberate control and prolongation [24,25]!

It should be also pointed out that the ionic nature of the acupuncture system suggests the possibility that ions in air (prana, qi, pneuma!?) can be physiologically effective [28], just through the acupuncture ionic system and biophysical mechanisms that lie in the basis of acupuncture regulation [21] (out of them, the positive ions have an catabolic influence (yang) and the negative ones an anabolic influence (yin) [4,5,17]). So, qi (sometimes erroneously referred as a new kind of biological energy, bioenergy) can be related to ions flowing through the ionic channels of the acupuncture system in the form of MW/ULF ionic currents, with informational content coded in spatio-frequency form of currents and EM fields. It should be pointed out that a lot of experimental phenomena related to external qi gong treatment [29] can be reconciled with the ionic nature of qi. Hence, it seems that the healing process can be related with the transfer of ions between the healer and healee, and/or transfer of the MW/ULF EM information patterns responsible for normal functioning of acupuncture system and overall health [4,5,17]. Also, even distant displacements of healer's ionic structure in remote diagnosis and healing [25] could be expected in transitional states of consciousness.

This was our motivation to start examination [30] of

EEG correlates of the healer/healee interactions, as presumably most intriguing and relatively easily reproducible transpersonal phenomena [29]. This paper presents preliminary results obtained during five healing sessions of one healer. Obtained results will be used to set-up framework of future research.

The study was carried out on one healer and five healthy adult volunteers. There were 3 males and 2 females, whose ages ranged from 24 to 30 years with a mean age of 26 years. All subjects were free of any medication. Prior to the experiment, subjects were informed verbally about all aspects of the experimental procedure.

The experiment was conducted in a soundproof room. Healer and healee were in relaxed state with eyes closed. Subjects had no physical contact. Each recording session was divided into three sequential periods: (1) before the healing session (2 min with eyes closed, healer had no activity); (2) during the healing session (3 min); and (3) after the healing session (2 min). During those periods EEGs of healer and healee were recorded and stored on a hard disk.

2.5 Alertness/Drowsiness

Two interesting and to great extent related problems are automatic scoring of sleep stages, and detection of EEG segments of reduced vigilance level during awake stages.

The same standards for the visual classifying of sleep stages have been in use over 30 years [31,32]. These standards suffer from severe limitations, and therefore automatic procedures that implement them cannot give satisfying results. Hence, despite the advance of computer technology over the past decades, automatic scoring of sleep stages is far from being solved till now [33]. Man/machine agreement of even the best methods is limited by poor interscorer agreement obtained by comparing results of visual inspection of different electroencephalographers.

The situation is even more vague with the detection of reduced vigilance level segments from EEG recordings. Different approaches have been used to classify the degree of alertness during the awake stages, from subjective ratings, performance tests and neurophysiological measures. Subjective ratings are unreliable because of the fact that feelings such as alertness or drowsiness are poorly defined. Methods based on monitoring performance level like measuring reaction time to some stimuli, cannot give satisfying results as such measurement itself influences the subject's vigilance. It is evident, therefore, that it is the EEG on which the vigilance level estimation should be based. However, there are no universally accepted standards for visual classification of vigilance level during the awake stages, based on EEG traces. This is in contrast to the fact that the changes in EEG frequency bands, and occurrence of patterns assigned to changes of vigilance level, are assumed to be known. Different authors have used different approaches to combine these known characteristic changes in order to form meaningful rules, so further efforts should be obviously directed toward modeling underlying biological processes [33].

An effort in this work is done toward finding underlying functional relationship between power spectrum fluctuations related to changes of vigilance level (not used predetermined relationships) in order to estimate vigilance level. This is done by means of neural network classifier. Employing neural network classifier as a structure with modifiable parameters is of benefit for the following reasons: (a) underlying relationships which are assumed to exist, are not known, and are to be found; (b) by supplying the neural network with training sets obtained from recordings on single subjects, the network "learns" individual patterns characteristic for lower vigilance; and (c) the method can be adjusted to correspond to the results obtained by visual inspection of different experts. This allows interaction between electroencephalographers and machine, that will lead to better understanding of underlying principles and therefore to more efficient standards.

Electroencephalograms of 30 healthy young subjects were recorded. Subjects were aged 20-28 (median: 25), 22 males and 8 females, and have passed neurological screening. Uniformly aged subjects were chosen because EEG changes with age, and universal rules for automatic detection of vigilance level should require much bigger experimental group. Recordings were performed between 2-4 p.m. Subjects were not sleep deprived, nor had any deviations from their usual circadian cycle, and they took no drugs.

Recordings lasted from 15 to 30 minutes, depending on subject's level of drowsiness i.e. frequency of occurrence of low vigilance segments on corresponding EEG traces. Subjects were not allowed to fall asleep (i.e. further from stage I of slow wave sleep - drowsiness). We required at least two minutes of low vigilance level in total EEG time. EEG signals were analyzed off-line, and epochs without artifacts, characteristic for full wakefulness and for lower vigilance were cut by experienced electroencephalographer and pasted to form two-minute long segments used, one as a representative of normal fully awake state (alert wakefulness), and another one as a representative of drowsy wakefulness.

3 Results

3.1 Transcendental meditation

The representative examples of spectral arrays in our subjects (N°s 1 and 12) with slow α activity and θ burst during meditation, respectively, are shown in Fig. 1(a, b).

The comparison of the medians of partial EEG power for one derivation was performed using Wilcoxon matched pairs test. Fig. 2 shows the spatial distribution of the changes (z-scores) over the whole head. Shaded areas indicate the fields that have significant power increase. The primary sources of differences were the left frontal region (F3, $z = 3.24, p = 0.001$) in θ band, right temporal region (T4, $z = 2.65, p = 0.008$) in α_1 band, left temporal

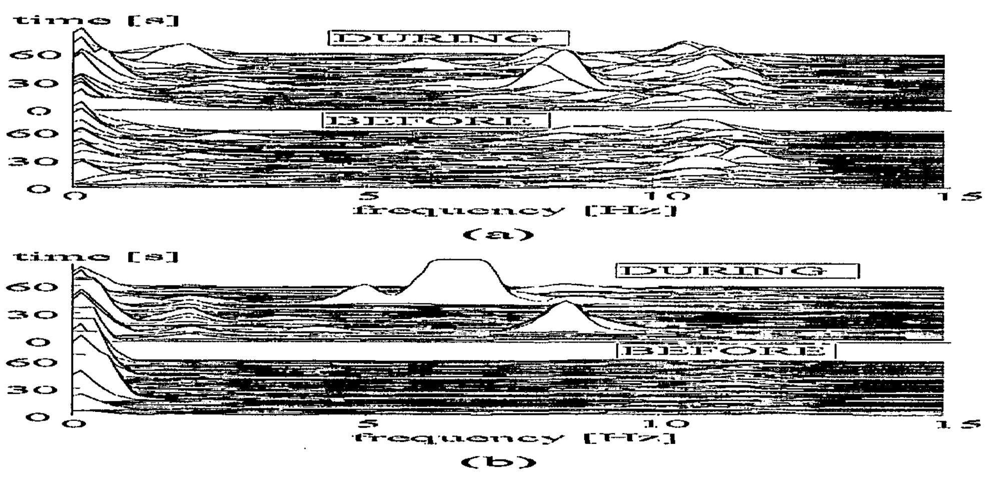


Figure 1: (a) Compressed power spectral arrays of the EEG for subject 1 obtained from the electrode P3 before and during meditation, showing the slow α waves at 8 Hz; (b) Compressed power spectral arrays of the EEG for subject 12 obtained from the electrode F3 before and during meditation, showing a θ burst at 6 Hz.

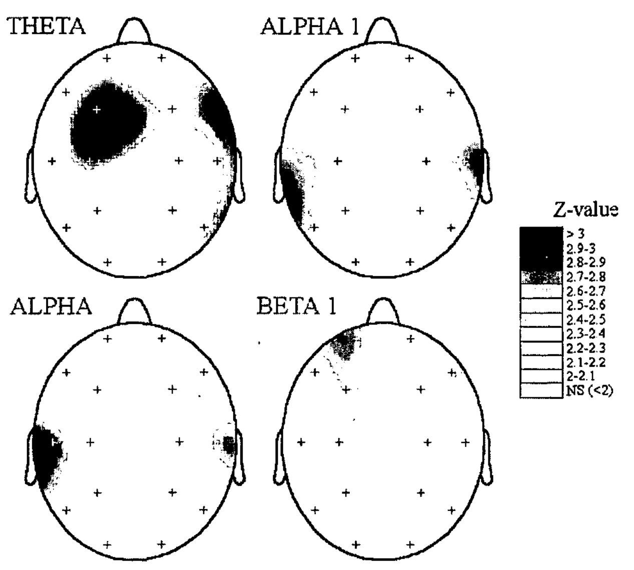


Figure 2: The topographic mapping of significant ($p < 0.05$) z-values of partial EEG power for each frequency band for the meditation study: θ (4-8 Hz), α_1 (8-10.5 Hz), α (8-13 Hz), and β_1 (13-18 Hz); Wilcoxon matched pairs test; inverse distance method.

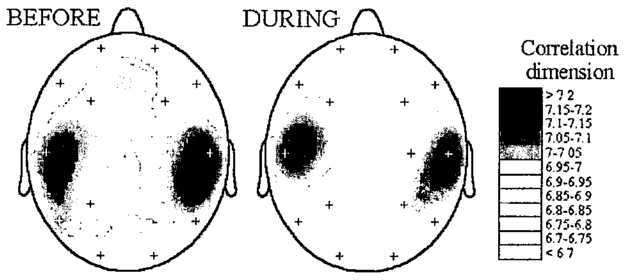


Figure 3: Spatial distribution of the correlation dimension before and during meditation.

region (T3, $z = 2.73, p = 0.006$) in whole α band, and left prefrontal region (Fp1, $z = 2.59, p = 0.01$) in β_1 band.

Spatial distributions of medians of correlation dimension over the whole head before and during meditation are given in Fig. 3. Although there are no statistically significant changes, a decrease in correlation dimension in right frontal and left parietal regions is obvious. In Fig. 4 the z-scores obtained by comparison of the medians of correlation dimension of each channel of every subject, prior and during meditation is given. Wilcoxon matched pairs test was applied.

The present study confirms previous reports [10,34-37]. The meditators as a group displayed a significant increase of θ activity ($z = 2.00, p = 0.046$) over the whole head. In particular, out of 25 meditators, 10 (40%) significantly increased their θ activity during meditation, and 4 (16%) significantly decreased. Analysis of each of the 16 derivations separately showed that the prominent θ wave activity is present in the frontal, central, and right temporal regions at frequency of 8 Hz (channel F3, $z = 3.91, p = 0.0001$). The patterns of θ frequencies fluctuated. Observed hypersynchronous θ bursts were similar to the θ bursts occurring during phases of emotional excitation [38,39].

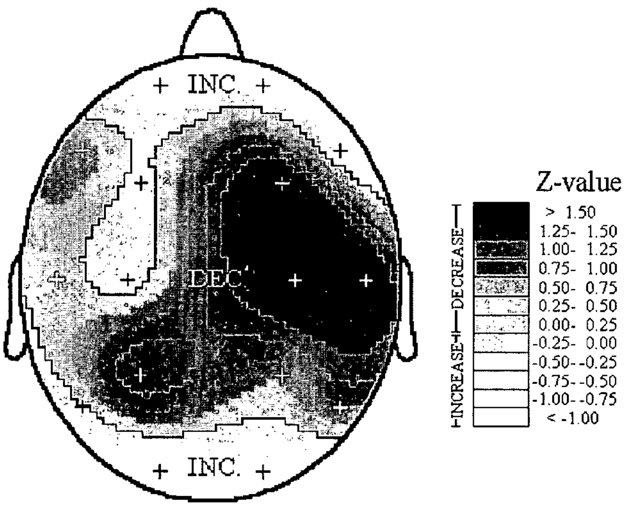


Figure 4: The topographic mapping of z-values of correlation dimension for meditation study; Wilcoxon matched pairs test.

The consistent changes in the other frequency bands were not observed in meditator group as a whole. Out of 25 subjects, 12 (48%) significantly increased their slow α activity (8-10.5 Hz) during meditation, and 4 (16%) significantly decreased. The prominent slow α activity occurred in right frontal, central and temporal regions. During TM, there was a significant increase of α activity in 13 (52%) experimental subjects and significant decrease in 7 (28%), most frequently in temporal region (T3-T4). The increase of slow α activity during the TM technique is apparently due to the nature of the technique, which according to adherents involves the increasingly abstract experience of quieter levels of mental activity, attained without concentration or procedures of controlling the mind [40]. Increased orderly functioning of the frontal and central regions of the brain may be correlated with this improvement in mental abilities, especially since these brain areas are known to be responsible for such activities as sensory-motor integration, memory, cognition, concentration, judgment, and volition [36]. Those changes may not necessarily occur in long-term meditators. The subjects also showed a significant increase of prefrontal β activity.

Many of the previously published papers have reported physiological changes during meditation that seem to characterize substates of wakefulness [9,38,41-43]. Those changes have been interpreted as a support for the fourth major state of consciousness, the restful alertness state, being a combination of restfulness (increase in α and θ activity) and alertness (increase in β activity). This is in accordance with the Ellias and Grossberg model of neuron [44], which predicts that higher input to brain neural network increases frequency and decreases amplitude of oscillations. In this case, appearance of significant θ component and the α rhythm slowing may be the result of deprivation of the sensory input. On the other hand, increased β power could be a consequence of the increased mental activity.

3.2 Musicogenic states

In Table 1, the results of the Wilcoxon matched pairs test for medians of EEG power of all 16 channels, prior and during the listening of music.

In most cases, during the listening of music, the EEG power decrease is observed in various frequency bands. In the three cases (out of 24), a significant power increase in θ and α bands is registered, in accordance with an intense aesthetic experience in these cases; the two most prominent spectral arrays before, during, and after the listening of music are shown in Fig. 5(a, b).

3.3 Microwave resonance relaxation

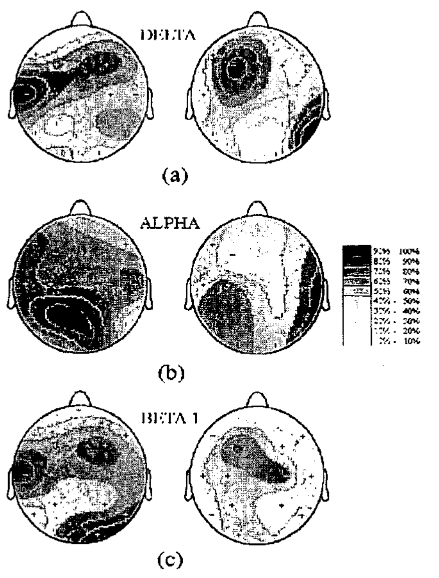


Figure 6: The topographic mappings of the number of subjects (in %) of the group 1 (left) and the group 2 (right), having the significant EEG power changes in the: (a) δ band (1-4 Hz), (b) α band (8-13 Hz), and (c) β_1 band (13-18 Hz), during MRT study. The gradual percentage changes are presented in various degrees of shading, as designated in the insert.

The subjects were classified in two groups: the group 1 (11 subjects) not previously subjected to the MRT treatment, and the group 2 (17 subjects) being subjected to the MRT in the past two years.

Both groups of subjects have significant changes in the EEG power over the whole head in the α and β_1 frequency bands, with observation that a percentage of subjects with minor reactions is much less in the group 1. The group 2 has also the significant EEG power changes over the whole head in a δ frequency range.

	SUBJ. 1			SUBJ. 2			SUBJ. 3			SUBJ. 4			SUBJ. 5			SUBJ. 6		
MUSIC	θ	α	β_1	θ	α	β_1	θ	α	β_1	θ	α	β_1	θ	α	β_1	θ	α	β_1
1	-	-	0	0	-	0	-	+	0	+	-	0	-	-	-	0	-	-
2	0	-	-	-	-	-	-	-	-	0	0	0	-	-	-	x	x	x
3	x	x	x	+	+	-	-	-	-	0	-	0	0	-	-	-	-	-
4	0	-	-	0	-	-	-	-	-	x	x	x	0	-	0	x	x	x

+ sign. increase, - sign. decrease, 0 no sign. changes, x not recorded

Table 1: The EEG power changes during the listening of music.

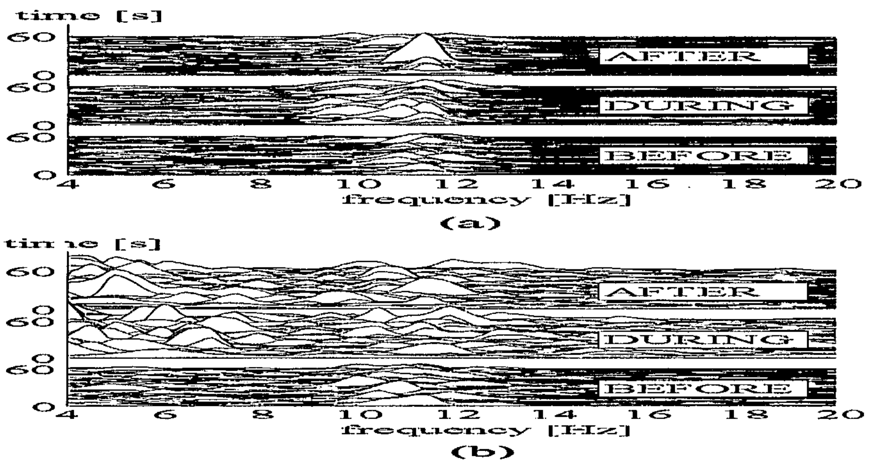


Figure 5: (a) The spectrogram with the observed EEG power increase in the α band and the appearance of slower α frequencies during the listening of music 1 in channel P3 of subject 3; (b) The spectrogram with the observed high EEG power increase in the θ band during the listening of music 1 in channel T6 of subject 4.

Within the whole frequency range (1-30 Hz), 37 channels in the first group and 22 channels in the second group with the power changes in more than 50% subjects are notified. The changes are evident in δ, α, and β_1 frequency bands. The most prominent power changes in all frequency bands are observed in the channels 3 (T3), 11 (C3), and 12 (C4) of the group 1, and in the channels 6 (T6), 9 (F3), 11 (C3), and 13 (P3) of the group 2. The channels 3 (T3), 11 (C3), and 15 (O1) of the group 1, and the channels 6 (T6) and 11 (C3) of the group 2 have power changes in more than 50% subjects in 4 out of 5 frequency bands. In both groups of subjects, a decrease in the EEG power is more frequently observed than an increase. As an illustration, in Fig. 6(a-c) the topographic mappings of the number of subjects (in %) having the significant EEG power changes in the δ, α, and β_1 frequency bands, for the two groups of subjects are presented.

The changes in coherency are not too significant. Regarding the complete group of subjects, in δ frequency range the changes are observed in the backward temporal (T5 and T6), parietal (P3 and P4), and occipital (O1 and O2) regions in 25%, 25%, and 42% of subjects, respectively. The changes in frontal region (F3 and F4) within α frequency band are observed in 29% of subjects, and in occipital region (O1 and O2) within β_1 frequency band in 25% of subjects. Most prominent changes, over the whole frequency interval (1-30 Hz), are registered in occipital region (O1 and O2). A decrease in the coherency is generally observed.

The examples of spectral arrays in our subjects (N°s 7 and 18), showing the decrease in the EEG power after MRT treatment, are shown in Fig. 7(a, b).

3.4 Healer/Healee interaction

The healer's EEG at the beginning of each of five sessions exhibited similar properties (Fig. 8). The specific pattern of this slow fluctuation is not due to the cortical activity, but to intensive neuro-vegetative reaction. This reaction is accompanied with the changes of skin resistivity, which is the greatest at the parietal and temporal brain sites.

The healer's power spectrum exhibited some changes in θ region, the greatest being at frontal brain sites (Fig. 9(a)). The healee's power spectrum during the session exhibited changes in θ region at frontal brain sites too (Fig. 9(b)).

The Man-Whitney U-test was used to analyse coherence spectral arrays. Significant changes in 3 out of 5 experiments were observed. Changes were most pronounced at channel T3 (both healer's and healee's). However, due to their small values, medians of coherence time series (30-50%) were not taken into consideration. Increase of coherence occurred only in short intervals. Epochs of 4 s, before and during the session were used to estimate maximum coherence value in α and θ bands. Results are summarized

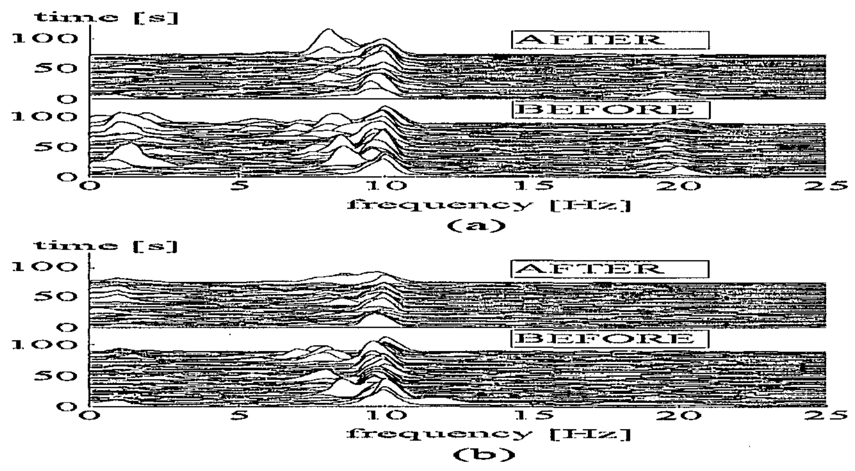


Figure 7: (a) Compressed power spectral arrays of the EEG for subject 7 obtained from the electrode C3 before and after MRT treatment, showing the decrease in the EEG power in all frequency bands; (b) Compressed power spectral arrays of the EEG for subject 18 obtained from the electrode T3 before and after MRT treatment, showing the decrease in the EEG power in α band.

	θ BAND			α BAND		
SUBJ.	BEFORE	DURING	AFTER	BEFORE	DURING	AFTER
1	73% (T3-T3)	82% (F4-F4)	75% (T3-T3)	70% (O2-O2)	78% (F3-F3)	76% (P4-P4)
2	73% (P4-P4)	70% (O2-O2)	69% (P4-P4)	78% (T4-T4)	84% (O1-O1)	83% (T3-T3)
3	74% (F4-F4)	72% (P3-P3)	81% (T3-T3)	79% (O2-O2)	92% (O1-O1)	84% (O2-O2)
4	71% (F4-F4)	78% (F3-F3)	71% (F3-F3)	81% (F3-F3)	82% (O2-O2)	77% (P4-P4)
5	77% (F4-F4)	81% (P3-P3)	74% (O2-O2)	80% (T3-T3)	74% (F3-F3)	84% (O1-O1)
mean	74.6%	76.6%	74.0%	77.6%	82.0%	80.8%

Table 2: Maximum mean coherence in θ and α bands for 4 s time epochs, before, during, and after the healing session.

in Table 2. Increase of maximum coherence in α band was 6-13% for subjects 1, 2, and 3, while 4.4% for the whole group.

An example of synchronized EEG signals of healer and healee 3 during the treatment, as well as the corresponding phase diagrams and coherence diagrams are shown in Fig. 10. The phase difference at frequencies with maximum coherence is 180°.

3.5 Alertness/Drowsiness

Single layer perception neural network was used for vigilance level assessment. Training and test input vectors were made from one two-minute long segment of alert wakefulness and one of drowsy wakefulness. The state of alert wakefulness was assigned the value of 1, and the state of drowsy wakefulness was assigned the value of 0. These values were supplied to the network as desired output values during the training session.

Segments of 30 s of both recording were used to construct training sets. The network was tested on the rest 90 s of each recording. To achieve high processing speed it was necessary to make input vectors of low dimensionality. Also, it was necessary to make the components of input vectors to be easy to compute. Thus, the power spectrum characteristics were used to form input vectors. Spectral analysis is the most important and most common technique in EEG time series analysis [45]. Due to great inter-individual variations in total power, the relative values were computed, i.e. the power in each frequency band was divided by the total power in all bands.

The slowing of dominant α frequency and widening of α peak are assumed to be the most important signs of drowsiness. However, it is not possible to see these events through changes of relative power in α band. Therefore, α band was divided into two bands, α_1 (7.5-9.5 Hz) and α_2 (9.75-12.5 Hz). These boundaries are carefully chosen so that values of relative power in these bands carry the same amount of information, as does the shift of the dominant α frequency, and widening of α peak.

Boundaries of frequency bands used were as follows: 0.5-3.25 Hz (δ), 3.5-7.25 (θ), 7.5-9.5 Hz (α_1), 9.75-12.5 Hz (α_2), 12.75-18 Hz (β_1), and 18.25-25 Hz (β_2).

Since it is shown in earlier studies [46] that significant differences in EEG patterns between hemispheres during the drowsiness do not occur, power spectrum was computed from the one hemisphere (right).

Slight vigilance fluctuations can occur in time periods as short as few seconds. Therefore, short time epochs were used to compute power spectrum characteristics. The use

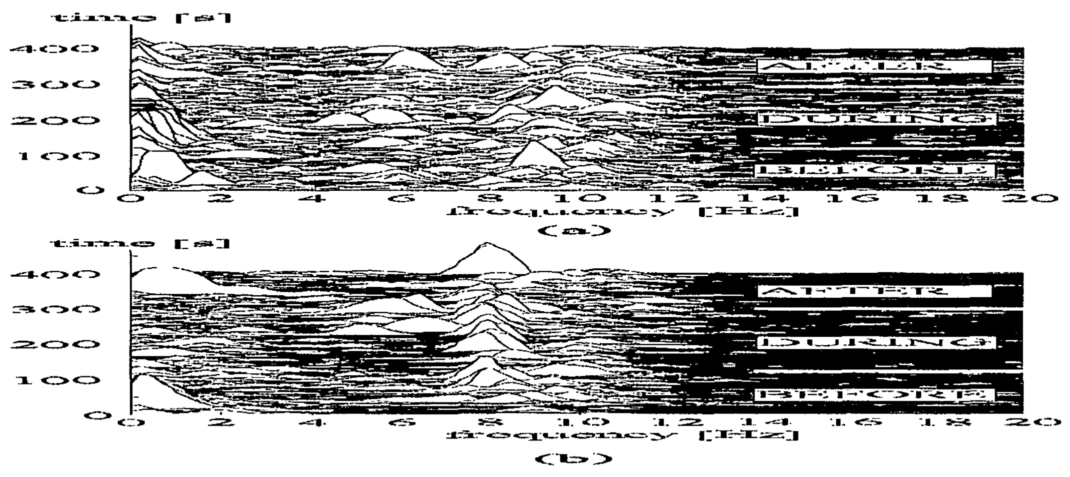


Figure 9: Spectrogram (channel T3) of (a) healer and (b) healee; increase in power is most pronounced in θ band during and after the healing session.

FREQUENCY BAND	α_2	α_1	θ	θ	β_1	β_1	β_1	β_2	β_2	β_2	θ/α
ELECTRODE POSITION	O2	O2	O2	F4	F4	C4	O2	F4	C4	T6	O2+F4+F8+C4+T4

Table 3: Combinations of power spectrum values and electrode positions used to form input vectors.

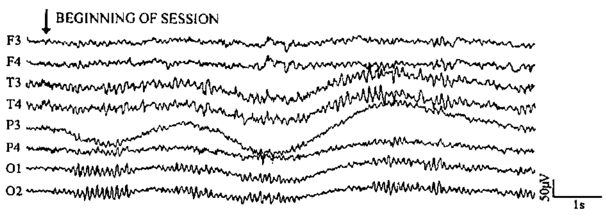


Figure 8: An example of healer's neuro-vegetative reaction at the beginning of session.

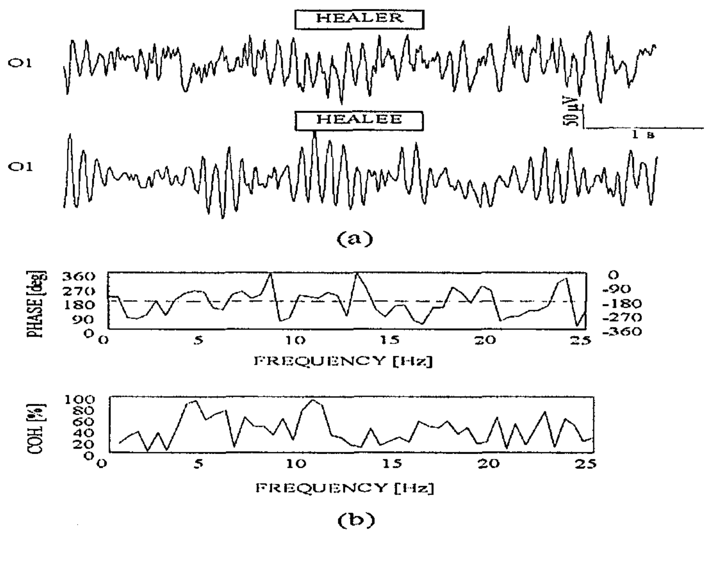


Figure 10: An example of short-time EEG synchronization with maximum coherence; channel O1 of healer and subject 3; plots of (a) original signal and (b) phase and coherence diagrams.

of 4 s long epochs provided a good compromise between time and frequency resolution. Frequency resolution was 0.25 Hz, which was good enough to successfully divide spectrum into bands. In order to improve time resolution, epoch overlapping of 2 s was used. Thus, 28 epochs were used for training, and 84 epochs were used for testing.

In order to find the power spectrum characteristics that best reflect the expert's knowledge used for visual classification results from previous studies were consulted as well [46-52]. Characteristics that in combination with perceptron neural net proved to give the most satisfactory results are shown in Table 3. These values were used to form input vectors for both training and testing. Input vectors during the training phase were supplied to the network along with the desired values of output. During training cycle, perceptron neural network assigned the weighting values to each component of the input vectors. These weights were used

	SUBJECT				
	1	2	3	4	5
TRAINING PHASE					
N^0 of vectors	28	28	28	28	28
N^0 of vectors successfully learned	28	28	28	28	28
Error rate (%)	0.00	0.00	0.00	0.00	0.00
TESTING PHASE					
N^0 of vectors	84	84	84	84	84
N^0 of vectors successfully learned	79	81	83	81	84
Error rate (%)	5.95	3.57	1.19	3.57	0.00

Table 4: Results of perceptron classification of vigilance level, for five typical subjects.

in testing phase, in order to estimate the level of vigilance. Training network on each subject enabled the network to "learn" individual fluctuations in EEG spectrum.

Numerical results were computed on a standard PC. Procedures for data manipulation and neural network implementation were developed in MATLAB 4.2 environment [53]. Training and testing of the network took a few seconds of CPU time on PC 486/100 MHz.

Results of the procedure applied for five typical subjects are summarized in Table 4.

4 Conclusions

Our investigations demonstrated that new technologies applied to the EEG may permit rapid and reproducible identification of different styles or states of consciousness. Such a tool might be useful in evaluating the effectiveness of different techniques for stress reduction and for altering expressions of consciousness.

4.1 Transcendental meditation

The EEG correlates estimated before and during meditation clearly distinguish the following characteristics: (1) Meditators as a group showed a significant increase of θ activity in meditation, dominantly in left frontal (F3) and right temporal (T4) regions at frequency of 8 Hz; (2) Slow α (α_1) and whole α activity increased significantly in temporal region (T3-T4); (3) β activity increased significantly in prefrontal region (Fp1-Fp2); and (4) correlation dimension decreased not significantly, but there was significant correlation between decreasing of the correlation dimension and increasing of the partial EEG power in α_1 and α band. Appearance of significant θ component and the α rhythm slowing may be the result of deprivation of the sensory input, while the increased β power could be a consequence of the increased mental activity - supporting the previous interpretations of meditation as a fourth major state of consciousness, the restful alertness state.

4.2 Musicogenic states

According to our pilot study with six subjects and four types of spiritual music, it might be concluded that the EEG power changes during their listening are quite individual. In the three cases where significant raise of power (i.e. relaxation) in θ and α bands is observed, the subjects have described their musical experiences as very pleasant - in contrast to the cases with drop in EEG power and unpleasant musical experiences. The most prominent changes were observed in subjects 3 and 4 upon the influence of the music 1 (Bhajan, the Indian spiritual music, sang in Sanskrit); and somewhat less in subject 2 upon the influence of the music 3 (the Maronit spiritual music, sang in Arabian). Concerning the coherence increase, it seems not to be correlated with the aesthetic experience as, for instance, all subjects described their experience of music 2 as "slightly unpleasant", while their coherence even being increased. In spite of the observed particular EEG changes upon some types of spiritual music, it might be that more conclusive results could be achieved only in the case of more careful choice of subjects, regarding their musical affinities and/or education.

4.3 Microwave resonance relaxation

Upon the MRT application, a decrease in the EEG power is observed in most subjects of the both groups - the first group not previously subjected to the MRT treatment (11 subjects), and the second group being subjected to the MRT in the past two years (17 subjects). The EEG power over the whole head is significantly changed in α and β_1 frequency bands in the both groups of subjects. The subjects of the group 1 had about 70% more MRT responding channels (reacting to MRT in over 50% subjects in all frequency bands) than the subjects of the group 2, especially in the α and β_1 frequency bands, which might be ascribed to the energetically more dysbalanced acupuncture system of subjects in group not previously under the MRT treatment, therefore revealing more relative changes towards the energetically normal acupuncture state. It should be added that in both groups of subjects the dominant power changes within the EEG channels in all frequency bands (with the exception of β_1 band in the group 2) are registered in the left hemisphere, which might be ascribed to higher activation of the left circulatory part of the acupuncture system, as a consequence of the systematic MRT stimulation of the left-side acupuncture points Li 4, Pc 6, H 7, and Ap 55.

4.4 Healer/Healee interaction

The main aim of our pilot experiment made on one healer and five healees was to establish experimental set-up for larger experimental group as well as methods for quantitative analysis. Due to small experimental group and non-stationarity of EEG signals, results obtained by dynamic analysis of power and coherence time series were not statistically significant. If the healer/healee interaction is related with the transfer of the EM/ionic information patterns, then it is to be established during short time intervals. This is suggested by the increase of the maximum mean coherence of the whole group in α band (4.6%), obtained by statistical analysis of the coherence of 4 s long time epochs. Due to the nature of the underlying phenomenon it is necessary to perform data acquisition with the higher sampling rate ($>$ 256 Hz). Quantification of the observed changes in δ and subdelta band requires more elaborated signal processing procedures.

4.5 Alertness/Drowsiness

This research has shown that automatic detection of fast fluctuations (order of few seconds) of the vigilance level is possible. Moreover, phase space created by variables listed in Table 3 has the property of linear separability of the states of "alert wakefulness" and "drowsy wakefulness". Therefore, it is possible to construct hyperplane that separate these classes. In this work, this separation is done by the perceptron neural network. Moreover, the simplicity of implemented procedure allows very fast data processing and near real-time processing, which is of great importance in clinical use. Potential application of the developed procedure is in the area of psychophysiology (cognitive testing during EEG recording, and similar procedures), long-term monitoring of vigilance in clinical medicine (epileptology), and automatic elimination of low vigilance segments from EEG record when necessary.

References

[1] In C. Tart, ed., *Altered States of Consciousness* (Academic, New York, 1972).

[2] K. Jaspers, *Allgemeine Psychopathologie* (Springer, Berlin, 1953).

[3] R. A. Moody, jr., *Life after Life* (Bantam, New York, 1975).

[4] D. Raković, Neural networks, brainwaves, and ionic structures: Acupuncture vs. altered states of consciousness, *Acup. & Electro-Therap. Res., Int. J.* 16 (1991), pp. 88-99; D. Raković, Neural networks vs. brainwaves: A model for dream-like states of consciousness, *Proc. 14th Ann. Conf. IEEE/EMBS* (1992), pp. 2651-2652; D. Raković, Traditional psychology and healing: Biophysical model of altered states of consciousness as a clue, in B. D. Sharma, ed., *Holistic Health, Healing and Astrosciences* (Shirdi Sai Baba Centre for Spiritual Awakening and Holistic Health, Yammu, India, in press); D. Raković and Z. Jovanović-Ignjatić, Microwave resonance therapy and acupuncture: New prospects for traditional medicine, *14th Symp. Acup. & Electro-Therap.*, October 22-25, 1998, New York.

[5] D. Raković, Brainwaves, neural networks, and ionic structures: Biophysical model for altered states of consciousness, in D. Raković and Dj. Koruga, eds., *Consciousness: Scientific Challenge of the 21st Century* (ECPD, Belgrade, 1995), pp. 291-316; D. Raković, Hierarchical neural networks and brainwaves: Towards a theory of consciousness, in Lj. Rakić, G. Kostopoulos, D. Raković, and Dj. Koruga, eds., *Brain and Consciousness, Proc. ECPD Workshop* (ECPD, Belgrade, 1997), pp. 189-204; D. Raković, *Fundamentals of Biophysics* (Grosknjiga, Belgrade, 1994, 1995), Chs. 5 and 6, in Serbian.

[6] D. Raković, Prospects for conscious brain-like computers: Biophysical arguments, *Informatica* 21 (1997), pp. 507-516; D. Raković, Consciousness and quantum collapse: Biophysics versus relativity, *The Noetic J.* 1 (1997), pp. 34-41; D. Raković and M. Dugić, Consciousness mediated quantum gravitational collapse via generated wormholes: From macroscopic biophysical to microscopic quantum arguments, in P. P. Wang, ed., *Proc. Joint Conf. Inform. Sci./Vol. 2: 3rd Int. Conf. Comput. Intell. & Neurosci.*, G. Georgiou, ed. (Assoc. for Intelligent Machinery, RTP, NC, 1998), pp. 265-268.

[7] K. S. Thorne, *Black Holes and Time Warps: Einstein's Outrageous Legacy* (Picador, London, 1994), Ch. 14, and references therein.

[8] D. Radenović, D. Raković, Z. Jovanović-Ignjatić, M. Tomašević, V. Radivojević, and E. Jovanov, Relaxation induced by microwave resonance therapy: EEG correlates, in Lj. Rakić, G. Kostopoulos, D. Raković, and Dj. Koruga, eds., *Brain and Consciousness, Proc. ECPD Symp.* (ECPD, Belgrade, 1997), pp. 213-218; P. Šuković, V. Radivojević, Ž. Martinović, D. Raković, and E. Jovanov, A novel neural network approach to estimation of vigilance level from EEG power spectrum, *ibid.*, pp. 251-254; M. Tomašević, D. Raković, E. Jovanov, V. Radivojević, and M. Car, EEG correlates of transcendental meditation, *ibid.*, pp. 255-262; L. Škarić, M. Tomašević, D. Raković, E. Jovanov, V. Radivojević, P. Šuković, M. Car, and D. Radenović, EEG correlates of musicogenic states of consciousness, *ibid.*, pp. 263-268; M. Tomašević, E. Jovanov, D. Raković, P. Šuković, S. Stanojlović, and M. Car, EEG correlates of healer/healee states of consciousness, *ibid.*, pp. 269-274.

[9] T. Nader, *Human Physiology: Expression of Veda and the Vedic Literature* (Maharishi Vedic Univ., Vlodrop, The Netherlands, 1995).

[10] R. K. Wallace, *The Physiology of Consciousness* (Maharishi Int. Univ., Fairfield, USA, 1993); R. K. Wallace, Physiological effects of transcendental meditation, *Science* 167 (1970), pp. 1751-1754; D. W. Orme-Johnson and J. T. Farrow, eds., *Scientific Research of the Transcendental Meditation Program - Collected Papers*, Vol. 1 (MERU Press, Seelisberg, Switzerland, 1977).

[11] J. G. Roederer, Physical and neuropsychological foundations of music: The basic questions, in M. Clynes, ed., *Music, Mind, and Brain: The Neuropsychology of Music* (Plenum, New York, 1982), and references therein.

[12] B. J. Baars, *A Cognitive Theory of Consciousness* (Cambridge Univ., Cambridge MA, 1988).

[13] G. Rouget, *La Musique et la Transe* (Gallimard, Paris, 1980); R. Jourdain, *Music, the Brain, and Ecstasy* (Avon Books, New York, 1997).

[14] N. D. Devyatkov and O. V. Betskii, eds., *Biological Aspects of Low Intensity Millimeter Waves* (Seven Plus, Moscow, 1994).

[15] S. P. Sit'ko and L. N. Mkrtchian, *Introduction to Quantum Medicine* (Pattern, Kiev, 1994).

[16] S. P. Sit'ko, Ye. A. Andreyev, and I. S. Dobronravova, The whole as a result of self-organization, *J. Biol. Phys.* 16 (1988), pp. 71-73; S. P. Sit'ko and V. V. Gizhko, Towards a quantum physics of the living state, *J. Biol. Phys.* 18 (1991), pp. 1-10; S. P. Sit'ko, Quantum physics of the alive: Medical aspects, in Lj. Rakić, G. Kostopoulos, D. Raković, and Dj. Koruga, eds., *Brain & Consciousness, Proceedings of ECPD Workshop* (ECPD, Belgrade, 1997).

[17] D. Raković, Biophysical basis of traditional medicine and traditional psychology, *Serb. J. Acup.* 1 (1998), pp. 6-12, in Serbian; Z. Jovanović-Ignjatić and D. Raković, A review of current research on microwave resonance therapy: Novel opportunities in medical treatment, *Acup. & Electro-Therap. Res., Int. J.*, submitted.

[18] H. Fröhlich, Long-range coherence and energy storage in biological systems, *Int. J. Quantum Chem.* 2 (1968), pp. 641-649; H. Fröhlich, Theoretical physics and biology, in H. Fröhlich, ed., *Biological Coherence and Response to External Stimuli* (Springer, New York, 1988).

[19] V. F. Mashansky, Yu. V. Markov, V. H. Shpunt, S. E. Li, and A. S. Mirkin, Topography of gap junctions in human skin and its possible role in non-nervous transition of information, *Archive of Anatomy, Histology, and Embryology*, 84, 3 (1983), pp. 53-60, in Russian; D. Djordjević, Epithelial conception of genesis of the system of vital meridians and their acupuncture points, *Serb. J. Acup.* 1 (1998), pp. 17-22, in Serbian.

[20] E. R. Kandel, S. A. Siegelbaum, and J. H. Schwartz, Synaptic transmission, in E. R. Kandel, J. H. Schwartz, and T. M. Jessell, eds., *Principles of Neural Science* (Elsevier, New York, 1991), Ch. 9; M. V. L. Benett, L. C. Barrio, T. A. Bargiello, D. C. Spray, E. Hertzberg, and J. C. Sáez, Gap junctions: New tools, new answers, new questions, *Neuron* 6 (1991), pp. 305-320.

[21] Y. Omura, *Acupuncture Medicine* (Japan Publ. Inc., Tokyo, 1982); F. G. Portnov, *Electropuncture Reflexotherapeutics* (Zinatne, Riga, 1982), in Russian; A. I. Škokljev, *Acupuncturology* (ICS, Belgrade, 1976), in Serbian.

[22] In S. P. Sit'ko, ed., *Miscellany of Methodological Recommendations and Regulations in Microwave Resonance Therapy (MRT)* (Vidguk, Kiev, 1992), in Russian.

[23] R. G. Jahn, The persistent paradox of psychic phenomena: An engineering perspective, *Proc. IEEE* 70 (1982), pp. 136-170.

[24] R. Monroe, *Journeys Out of the Body* (Doubleday, Garden City, NY, 1971).

[25] K. C. Markides, *Fire in the Heart. Healers, Sages and Mystics* (Paragon House, New York, 1990).

[26] V. P. Kaznacheev and A. V. Trofimov, *Cosmic Consciousness of Humanity, Problems of New Cosmogony* (Elendis - Progress, Tomsk, Russia, 1992).

[27] R. Broughton, Human consciousness and sleep/waking rhythms, in B. B. Wolman and M. Ullman, eds., *Handbook of States of Consciousness* (Van Nostrand Reinhold, New York, 1986).

[28] A. P. Krueger, Preliminary consideration of the biological significance of air ions, *Scientia* 104 (1969), pp. 1-17.

[29] Y. Omura, T. L. Lin, L. Debreceni, B. M. Losco, S. Freed, T. Muteki, and C. H. Lin, Unique changes found on the qi gong (chi gong) master's and patient's body during qi gong treatment: Their relationships to certain meridians & acupuncture points and the recreation of therapeutic qi gong states by children & adults, *Acupuncture & Electro-Therap. Res., Int. J.* 14 (1989), pp. 61-89.

[30] E. Jovanov, D. Raković, V. Radivojević, Đ. Kušić, P. Šuković, and M. Car, Evaluation of state of consciousness using software support for monitoring spatiotemporal EEG changes, *ISCA Conf. Comp. Med.*, Lafayette, 1995; E. Jovanov, D. Raković, V. Radivojević, and D. Kušić, Band power envelope analysis - A new method in quantitative EEG, *Proc. 17th Ann. Int. Conf. IEEE/EMBS* (1995).

[31] A. Rechtschaffen and A. Kales, eds., *A Manual of Standardized Terminology, Techniques and Scoring System for Sleep Stages of Human Subjects* (U. S. Public Health Service, U. S. Government Printing Office, Washington DC, 1968).

[32] N. Ilanković, V. Ilanković, and M. Jašović-Gašić, *Sleep Disorder - Diagnostics and Curing* (Cibif, Belgrade, 1995), in Serbian.

[33] J. Hasan, Past and future of computer-assisted sleep analysis and drowsiness assessment, *J. Clin. Neurophysiol.* 13 (1996), pp. 295-313.

[34] J. P. Banquet, Spectral Analyses of the EEG in meditation, *EEG Clin. Neurophysiol.* 35 (1973), pp. 143-151.

[35] P. H. Levine, J. R. Hebert, C. T. Haynes, and U. Strobel, EEG coherence during the transcendental meditation technique, in D. W. Orme-Johnson and J. T. Farrow, eds., *Scientific Research of the Transcendental Meditation Program - Collected Papers*, Vol. 1 (MERU Press, Seelisberg, Switzerland, 1977), pp. 187-207.

[36] D. J. Kras, The transcendental meditation technique and EEG alpha activity, *ibid.*, pp. 173-181.

[37] A. M. Rouzeré, K. Badawi, and R. Hartmann, *High Amplitude Fronto-central Alpha and Theta Activity During the Transcendental Meditation Technique* (Department of Neurophysiology, MERU, Seelisberg, Switzerland, 1979).

[38] J. P. Banquet and M. Sailhan, EEG analysis of spontaneous and induced states of consciousness, in D. W. Orme-Johnson and J. T. Farrow, eds., *Scientific Research of the Transcendental Meditation Program - Collected Papers*, Vol. 1 (MERU Press, Seelisberg, Switzerland, 1977), pp. 165-172, in English, and *Revue d'Electroencephalographie et Neurophysiol. Clinique* 4 (1974), pp. 445-453, in French.

[39] A. C. Mundy-Castle, The electroencephalogram and mental activity, *EEG Clin. Neurophysiol.* 9 (1957), pp. 643-655.

[40] J. Forem, *Transcendental Meditation: Maharishi Mahesh Yogi and the Science of Creative Intelligence* (Dutton, New York, NY, 1973).

[41] N. Alexander and W. E. Larimore, *Distinguishing Between Transcendental Meditation and Sleep According to Electrophysiological Criteria* (Department of Psychology and Social Relations, Harvard University, Cambridge, MA, and The Analytic Sciences Corporation, Reading, MA, 1981).

[42] J. M. Davidson, The physiology of meditation and mystical states of consciousness, in D. H. Shapiro and R. N. Walsh, eds., *Meditation: Classic and Contemporary Perspectives* (ADLINE Publishing Company, New York, 1984), pp. 376-395.

[43] N. N. Lyubimov and S. N. Lyubimov, Dual reactivity of cerebrum during application of the special form of psychological training - transcendental meditation, *ECPD Workshop Brain and Consciousness '98*, June 24-25, Belgrade.

[44] S. A. Ellias and S. Grossberg, Pattern information, contrast control, and oscillations in the short-term memory of shunting on-center of surround networks, *Biological Cybernetics* 20 (1975), pp. 69-98.

[45] E. Basar, *EEG Brain Dynamics* (Elsevier, Amsterdam, 1980).

[46] K. P. Wright, P. Badio, and A. Wauquier, Topographical and temporal patterns of brain activity during the transition from wakefulness to sleep, *Sleep* 18 (1995), pp. 880-889.

[47] T. Jung, S. Makeig, M. Stensmo, and T. J. Sejnowski, Estimating alertness from the EEG power spectrum, *IEEE Trans. BME* 44 (1997).

[48] M. Matoušek and I. Petersén, A method assessing alertness fluctuations from EEG spectra, *EEG Clin. Neurophysiol.* 55 (1983), pp. 108-113.

[49] A. Belyavin and N. Wright, Changes in electrical activity of the brain with the vigilance, *EEG Clin. Neurophysiol.* 66 (1987), pp. 137-144.

[50] M. Nakamura, T. Sugi, A. Ikeda, R. Kakigi, and H. Shibasaki, Clinical application of automatic integrative interpretation of awake background EEG: Quantitative interpretation, report making, and detection of artifacts and reduced vigilance level, *EEG Clin. Neurophysiol.* 98 (1996), pp. 103-112.

[51] B. Streitberg, J. Roehmel, W. M. Herrmann, and S. Kubicki, COMSTAT rule for vigilance classification based on spontaneous EEG activity, *Neuropsychobiology* 17 (1987), pp. 105-117.

[52] C. Cajochen, D. P. Brunner, K. Kräuchi, P. Graw, and A. Wirz-Justice, Power density in theta/alpha frequencies of the waking EEG progressively increases during sustained wakefulness, *Sleep* 18 (1995), pp. 890-894.

[53] *MATLAB 4.0 User's Guide* (The MathWorks Inc., Natick, USA, 1994).

Floating-Point Arithmetic And The IEEE-754 Standard, I: Number-System Design

Amos R. Omondi
School of Informatics and Engineering
Flinders University
Bedford Park, SA 5042
Australia

Keywords: computer arithmetic, floating-point, IEEE-754 standard

Edited by: Mohsen Guizani

Received: February 7, 1999 **Revised:** July 24, 1999 **Accepted:** September 10, 1999

For many years almost every computer manufacturer had its own system for floating-point computation, a situation that made software portability a difficult task with different machine architectures and implementations. Moreover, not all of these systems were devised to facilitate error diagnosis or arithmetic with minimal errors. The situation changed considerably with IEEE's introduction of a standard system (IEEE-754) that has now been adopted by almost all computer manufacturers and which solves most of the problem associated with previous floating-point systems. Nevertheless, the rationale for the decisions made in formulation of the standard are not always well-understood. The objective of this paper is to discuss the issues involved in the design of a floating-point system and, in particular, to explain and justify those made in the IEEE standard. A companion paper deals with implementation issues.

1 Introduction

With the introduction and near-universal adoption of the IEEE Standard on Floating-Point Arithmetic (IEEE-754), floating-point computation has changed dramatically in the last decade. Prior to the introduction of that standard, almost every computer manufacturer had its own standard, a situation that made software portability a rather difficult task. Furthermore, not all such systems were devised with an eye to computation with minimal errors; nor was error-diagnosis always a primary factor in their formulation. The IEEE-754 standard is a carefully thought-out standard that deals with such matters and is now likely to be the only floating-point system that most computer users will encounter. Nevertheless, the decisions made in the formulation of the standard are not always appreciated, and this paper is an attempt to explain them in a simple and straightforward manner; previous discussions on the standard will be found in [8, 9, 10, 13, 15, 16, 17, 19]. (We hope that this paper will provide adequate background for and interest in the well-written and more formal tutorial by Goldberg [13].) It is also our intention to address floating-point-system design issues in a manner that will enable those who wish to implement something other than the exact IEEE-754 — this is typical, for example, in the design of special-purpose processors — to make informed decisions. Moreover, there is still a need, for the time being at least, to understand and implement floating-point systems other than IEEE-754 [18]. This is because the large investments in software that were made in earlier OEM standards has meant that even with the wide acceptance of the IEEE standard, there continue to be software and hardware systems that implement other floating-point systems: for example, the DEC Alpha architecture has both IEEE and DEC standards, implementations of some IBM architectures have both IEEE and IBM standards, and the NEC SX machines have IEEE, IBM, and Cray standards.

In what follows, we shall proceed to examine each of the main factors that need to be taken into account in devising a floating-point system, and in each case we shall show that the corresponding decision made in the IEEE-754 standard is a good one. All this is done in the next section of the paper. The third section of the paper summarises the IEEE-754 standard and also discusses some issues that are not addressed in the second section. The fourth section of the paper is a summary. A companion paper deals with implementation aspects of the standard [28].

For what follows, the reader should recall that in a floating-point system of a *radix* (or *base*) R, a number N is represented by a pair $\langle E, S\rangle$, where E is a signed integer and S is a signed fixed-point number, such that $N = S \times R^E$. R is not explicitly represented, since it is fixed for a given system, and it is a power of two, this being the most convenient for computer representation. E is the *exponent*; S is the *significand*, which is also sometimes referred to as the *fraction* (when it is of a magnitude less than one), *coefficient*, or *mantissa*. The number of digits used for the significand is the *precision*; for greater "accuracy", some systems allow intermediate (and, occasionally, final) results to be computed in an *extended precision* that exceeds the normal. The three parameters R, the range of S, and the range of E may therefore be used to characterize a floating-point number system, and much of the following discussions will deal with the factors that affect the choice of appropriate values for these parameters and the representation of these values; in particular, we aim to clearly justify the decisions made in formulating the IEEE-754 standard.

It should also be recalled that the main motivations for floating-point number systems (relative to fixed-point ones) are the large range of representable numbers and the automatic calculation of scale factors during arithmetic operations.

2 Issues in floating-point-system design

In this section we shall discuss the main issues in the design of a floating-point number system: the representation of numbers in a standard form that ensures each number is uniquely represented; how to carry out the basic arithmetic operations; what to do when the result of an arithmetic operation is too large or too small to be represented; and methods for reducing, with minimal error, each intermediate result to a form that can fit the machine requirements. In each case, we shall give a general discussion that concludes with a justification of the decision taken in the design of IEEE-754.

2.1 Normalization

In order to have as much "representational accuracy" as possible, it is desirable in floating-point computation to work with significands in which as many "non-significant" digits as possible have been eliminated — for example, to have 0.3142×10^1 instead of 0.0003142×10^4. Since the number of representable digits within a machine is fixed, the deletion of non-significant digits, in favour of retaining significant ones, is, loosely speaking, a way of ensuring "maximal accuracy" in representation. Accordingly, a significand is said to be *normalized* (or *standardized*), if it is within the permitted range for significands and has a leading (non-sign) digit that is significant; otherwise, it is said to be *unnormalized* (or *non-standardized*). In the latter case, we may further distinguish between a *supernormal* (or *superstandard*) significand, which is one that is unnormalized because its magnitude is too large, and a *subnormal* (or *substandard*) significand, which is one that is unnormalized because its magnitude is too small. A supernormal significand will have a magnitude exceeding that of the largest normalized one, and a subnormal significand will have a magnitude smaller than that of the smallest normalized significand. It will be apparent that if only normalized representations are permitted, then normalization ensures that each representable number, with the possible exception of zero, has a unique representation. In IEEE-754, normalized non-zero significands lie in the range $[1, 2)$. But the standard also permits unnormalized representations in exceptional circumstances (Section 2.3); in such cases, the significands are subnormal ones, with magnitudes in the range $(0, 1)$, and the representations are referred to as *denormalized.* The representation for zero may be regarded as being normalized or as unnormalized, depending on the circumstances, although it may be sensible to view it as the limiting form of denormalized numbers. The representation of zero is discussed in more detail in Section 2.4.2.

When non-zero radix-2 significands are normalized, the leading (non-sign) bit in each case is always known (to be 1) and an extra bit of precision may be gained, and the number of representable numbers thus doubled, if this bit is not explicit. The implicit bit is usually referred to as the *hidden bit*; and in IEEE-754, this will be the bit to the left of the radix-point in the complete significand. In general, bit-hiding is not always possible: The use of high radices in computers almost always involves the representation of each high-radix digit with its pure binary equivalent, or some variant of that,[1] and normalizing an unnormalized radix-R significand involves shifting it by $\log_2 R$ bit positions for each change in the exponent. But high-radix pure-binary significands can have several leading non-significant bits (not digits !) even when normalized: for example, with radix-16, all of the positive significands that start with bits $0001, 0010, 0011, \ldots, 0111$ are normalized. We therefore see here one advantage of having a small radix, as in IEEE-754.

Unless otherwise specified, we shall henceforth assume that the operands of all arithmetic operations are normalized and that the results of all operations are to be normalized. This means that there will be exactly one permissible representation for each non-zero number. Among other things, this uniqueness of representation simplifies the task of comparing numbers and also ensures that the best scale factor is chosen for each number. Other benefits of normalization include uniformity in the hardware implementation, as well as the facilitation of error analysis and a reduction in certain errors; see Table 1 for an example. It should, however, be noted that insisting on normalization does have at least one disadvantage — the loss of the ability to represent some very small numbers. This is because normalizing a subnormal significand involves shifting it left and decrementing the exponent, once for each shift, and if the magnitude of the subnormal significand is very small to start with, the exponent can go below its permitted lower bound. For example, with a radix R and an exponent lower bound of e_{min} for normalized operands, any number that has an unnormalized representation with a significand of k (where $k \geq 1$) 0s and an exponent not greater than $e_{min} - k$ cannot have a normalized representation. Hence — and examples of this will be given in Section 2.3 — there are exceptional situations, involving numbers of a very small magnitude, in which it may be worthwhile to abandon the requirement that all operands and results be normalized. This is so in

[1]For this reason, we shall in many cases simply use "bits", instead of the more general "digits", irrespective of the radix at hand.

IEEE-754.

2.2 Basic arithmetic operations

We shall now loot at the procedures for carrying out the basic arithmetic operations, i.e. addition, subtraction, multiplication, and division. The procedure for addition or subtraction is as follows. First, if necessary, the significand of one of the operands is adjusted to ensure that the two exponents are equal and the radix points therefore aligned; the significands are then added or subtracted.[2] The first step consists of shifting the significand of one of the operands by a number of digits equal to the magnitude of the difference in the exponents; no explicit change is required of the corresponding exponent, as it is implicitly taken to be equal to the other exponent. Usually, it is the significand of the operand with the smaller exponent that is shifted; the shift is therefore to the right. Were the other operand to be shifted, this would be a left-shift, and, as the significance of digits increases towards the left-hand end, the most significant digits could be lost (assuming a significand adder of a practical width). The other two operations are relatively easier to carry out: multiplication is carried out by multiplying the significands and adding the exponents, and division is carried out by dividing the significands and subtracting the exponents. All this may be summarized thus

- Addition: $\langle E_1, S_1\rangle + \langle E_2, S_2\rangle =$

$$\begin{cases} \langle E_1, S_1 + S_2R^{-e}\rangle & \text{if } e = E_1 - E_2 \geq 0 \\ \langle E_2, S_1R^{e} + S_2\rangle & \text{otherwise} \end{cases}$$

- Subtraction: $\langle E_1, S_1\rangle - \langle E_2, S_2\rangle =$

$$\begin{cases} \langle E_1, S_1 - S_2R^{-e}\rangle & \text{if } e = E_1 - E_2 \geq 0 \\ \langle E_2, S_1R^{e} - S_2\rangle & \text{otherwise} \end{cases}$$

- Multiplication:

$$\langle E_1, S_1\rangle \times \langle E_2, S_2\rangle = \langle E_1 + E_2, S_1 \times S_2\rangle$$

- Division: (if $S_2 \neq 0$)

$$\langle E_1, S_1\rangle \div \langle E_2, S_2\rangle = \langle E_1 - E_2, S_1 \div S_2\rangle$$

where $\langle E_1, S_1\rangle$ and $\langle E_2, S_2\rangle$ are the two operands, and R is the radix.

If results are to be normalized, then, assuming the intermediate result is not zero, each of the above sequences of steps may be followed by one or more shifts, with a corresponding adjustment of the exponent. (A significand that is zero must be treated as a special case; otherwise, there is no bound on the shift-distance.) In the addition operation, the result-normalization step will consist of either a one-digit right-shift of the significand and an increment by unity of the exponent, or of multiple-digit left-shifts of the significand and a reduction of the exponent by unity for each digit shifted. The one-digit shift takes place whenever the result of the significand-addition is too large; this excessive result is in the fixed-point operation on the significands, but not necessarily in the floating-point operation being carried out, and the supernormal significand can sometimes be normalized by shifting the overflow digit back into the significand proper and adjusting the exponent. The multiple-digit shift takes place whenever a cancellation of leading significant digits during the addition results in a subnormal significand; and the left-shifts will eliminate the leading 0-digits. In multiplication and division, if normalization is necessary, then it will be of a one-digit left-shift for multiplication and of a one-digit righ-shift for division. The shifts prior to the main arithmetic operation are usually referred to as *pre-arithmetic shifts*, *alignment shifts*, or simply *preshifts*; those after the operation are known as *post-arithmetic shifts, normalization shifts*, *post-normalization shifts* (a popular but misleading terminology), or just *post-shifts*.

$0{\cdot}00001 \times 10^{3}$	$0{\cdot}10000 \times 10^{-1}$
$\underline{0{\cdot}08000} \times 10^{-1}$	$\underline{0{\cdot}80000} \times 10^{-2}$
$\downarrow$ align (four-place shift)	$\downarrow$ align (one-place shift)
$0{\cdot}00001 \times 10^{3}$	$0{\cdot}10000 \times 10^{-1}$
$\underline{0{\cdot}00000} \times 10^{3}$	$\underline{0{\cdot}08000} \times 10^{-1}$
$0{\cdot}00001 \times 10^{3}$	$0{\cdot}18000 \times 10^{-1}$
(a)	**(b)**

Table 1: Effect of normalization on error

We stated above that normalization can help in the reduction of certain errors. We are now in a position to appreciate this. As an example of this, suppose that we add 0.00001×10^{3} and 0.08000×10^{-1}, with a precision of five, and that digits shifted out during alignment are not retained. Then the addition is as shown in Table 1(a). The error there, relative to the true result, is approximately 44%. On the other hand, if the operands are first normalized, then the addition is as shown in Table 1(b); there is no error in that case.

2.3 Overflow, underflow, and indeterminate results

The range of a floating-point number system may be depicted roughly as shown in Figure 1. The shaded areas and zero delimit the range of the representable numbers. Y_{min} and X_{min} are the numbers of the smallest representable magnitudes and correspond to the smallest representable magnitude in the significand and the smallest (i.e. most negative and of the largest magnitude) repre-

[2] Because of the way in which subtraction is usually carried out within a computer — that is, by negating the subtrahend and adding to the minuend — we shall, unless otherwise specified, henceforth use "addition" to refer to both true addition and true subtraction.

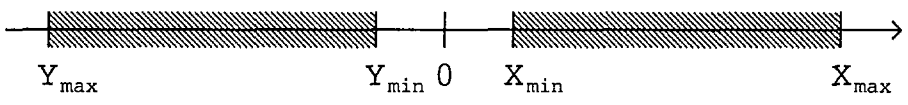


Figure 1: Range of representable floating-point numbers

sentable exponents; Y_{max} and X_{max} are the numbers of the largest representable magnitudes and correspond to the largest representable signficand and exponents. Usually $Y_{min} = -X_{min}$ and $Y_{max} = -X_{max}$; this may not always be the case, but it is desirable. Unless otherwise specified, we shall assume in what follows that the range delimited corresponds to normalized representations.

Exponent spill is said to have occurred whenever some computation leads to an exponent that is too large or too small for exponents permitted in final results: *Exponent overflow* (or just *overflow*) occurs when the exponent of a result is larger than the largest permissible exponent; it usually occurs because an attempt is made to represent a number whose magnitude exceeds that of X_{max} or Y_{max}. *Exponent underflow* (or just *underflow*) occurs when the exponent is smaller than the smallest permissible exponent; it usually occurs because an attempt is made to represent a number whose magnitude is smaller than that of X_{min} or Y_{min}. Sometimes exponent spill is only temporary, and recovery can be made in a subsequent normalization step. Checks for exponent spill should therefore be carried out only after normalization and rounding (the reduction of an intermediate result to the final precision), as such temporary spills can occur during both the primary operation and during rounding.

When overflow or underflow occurs, either the proceedings may be brought to a halt, and the situation indicated to the user, via, say, an exception, or an attempt may be made to continue in some reasonable fashion. The latter course is preferable as there are instances when it is still possible to get meaningful results despite some intermediate overflow or underflow. Other than those cases that can be resolved by re-normalization, this course usually consists of dealing with the situation as follows. When underflow occurs, there are at least two straightforward ways to proceed: one is to set the result to zero or to Y_{min} or X_{min} (according to whether the result is negative or positive); and the other is to abandon the requirement for normalization, shift the significand, and adjust the exponent accordingly, so as to obtain an unnormalized significand with the smallest possible exponent. The former is usually referred to as *flushing to zero* (or *sudden underflow*) and is the solution that was used in almost all computers before the introduction of IEEE-754; the latter is known as *gradual underflow*, and representations that are so obtained are said to be *denormalized*. (In effect, sudden underflow replaces all numbers whose magnitudes are smaller than that of with zero X_{min} or Y_{min} , whereas gradual overflow replaces them with representations of numbers that lie between X_{min} and zero or between Y_{min} and zero. It may be argued that when underflow leads to a result of zero, then, since the exact result is not a true zero, the replacement should be with plus or minus zero, according to the sign of the underflowing result, should be used, as the sign may be informative to the user. This suggests the use of sign-and-magnitude or diminished-radix complement notation to represent the significand; in IEEE-754 the former notation has been adopted.) Similarly, when overflow occurs, the result may be set to *minus infinity* ($-\infty$) or *plus infinity* ($+\infty$), or to Y_{max} or X_{max}, according to whether the intermediate result is negative or positive. The choice of which methods to use in handling underflow and overflow should also take into account the method used to round intermediate results. In IEEE-754, overflow is dealt with by either setting the result to $+\infty$ or $-\infty$ or to X_{max} or Y_{max}, according to the method used for rounding, or by having a trap that allows the programmer/software to take other action. And for underflow, the result if either 0, a denormalized number, or X_{min} or Y_{min}.

In the above, *infinity* (∞) is just a unique machine-recognizable pattern that is interpreted as representing the number with the properties that we usually associate with the corresponding abstract number; for all finite real x, these include

$$\begin{aligned} -\infty &< x \\ x &< \infty \\ \infty + x &= \infty \\ \infty - x &= \infty \\ \infty \times x &= \infty, \quad x \neq 0 \\ x \div 0 &= \infty \end{aligned}$$

Operations such as $0 \times \infty$, $\infty - \infty$, and $0 \div 0$ need further consideration, as it is not immediately clear what the result should be: the result of $0 \times \infty$ could reasonably be taken to be 0, or ∞, or indeed any number; that of of $\infty - \infty$ to be 0 or ∞; and that of $0 \div 0$ to be 0, 1, or ∞. A sensible solution, used as early as the CDC 6600 computer, is to produce an *indefinite* (or *undefined*) "value" for such operations that do not have a mathematically defined result. Here, *indefinite* is another unique pattern that is interpreted as not representing any number. Such a pattern is more usually referred to as *Not-a-Number* (*NaN*) and can be helpful in transmitting diagnostic information. In IEEE-754, $0 \times \infty =$ NaN, $\infty - \infty =$ NaN, $-\infty + \infty =$ NaN, $\infty \times \infty = \infty$, $\infty \div \infty =$ NaN, $0 \div 0 =$ NaN, $x \div 0$ has a result of $+\infty$ or $-\infty$, according to the sign of x and the sign of 0, and any operation in which one of the operands is NaN produces a result that is NaN.

Gradual underflow can be more difficult to implement correctly than sudden overflow, but it generally yields more sensible results: Consider, for example, the operations in $(x \div y) \times y$, and suppose that performing $x \div y$ results in an underflow (if normalized) but that the value of $(x \div y) \times y$ can be represented normally. Then flushing to zero the result of $x \div y$ will yield a final result of zero, whereas gradual underflow will yield the correct result, x. Flushing to a particular value can lead also to inconsistencies in the results: Consider, for example, the multiplication $x \times y \times z$, and suppose that the value of $x \times y$ is out of range but the values of $y \times z$ and $x \times y \times z$ are in range. Then it is not the case that $(x \times y) \times z = x \times (y \times z)$, which is not surprising, since it should be expected that setting some intermediate result to an arbitrary value may lead to larger errors[3]. Nevertheless, there are cases where flushing to zero is quite reasonable. Suppose, for example, that we are evaluating the power series $1 + x + x^2 + x^3 + \cdots$ with terms of decreasing magnitude. Then flushing to zero the values of very small terms may still yield a meaningful result. In general, flushing to zero is not problematic when adding values of small magnitude to relatively much larger values. Further discussions of underflow, overflow, and exception-handling in general in IEEE-754 will be found in [10, 12, 14].

2.4 Representation format

Choosing a representation format means deciding on how to allocate the available bits between the exponent and the significand, selecting a notation for the significand, selecting a notation for the exponent, and selecting a radix. The three standard notations (radix complement, diminished-radix complement, and sign-and-magnitude) used for the representation of signed fixed-point numbers give rise to a variety of possible formats for the machine representation of numbers in floating-point form, since the values of both the significand and the exponent can be positive or negative. Thus we may have both the significand and exponent represented in sign-and-magnitude notation, one represented in sign-and-magnitude notation and the other in diminished-radix complement or radix complement, one represented in diminished-radix complement and the other in diminished-radix complement or radix complement, and so forth. For the bit allocation, the number of bits available is usually at least the word length of the machine, half that, or twice that, and the main trade-off is between the exponent range and the precision.

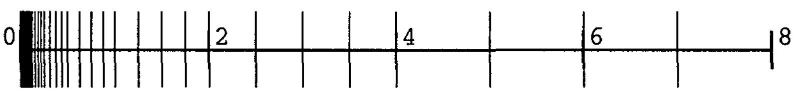
Since a given number of bits permits only a finite number of distinct patterns, any floating-point number system can represent only a finite number of the real numbers. Assuming significands in the ranges $[Y_{max}, Y_{min}]$ on the negative side and $[X_{min}, X_{max}]$ on the positive side, a radix R, and an exponent range $[E_{min}, E_{max}]$, the range of representable numbers consists of zero and (at discrete points) the intervals $[Y_{max}R^{-E_{max}}, Y_{min}R^{E_{min}}]$ and $[X_{min}R^{E_{min}}, X_{max}R^{E_{max}}]$. Those numbers are not evenly distributed within the range but are between successive powers of R: the density of the distributions changes with each unit reduction in the power of the radix. The main effect of trading off bit allocations between the significand and the exponent is to change the range and distribution of the representable numbers; varying the radix also has a similar effect. Figure 2 shows positive portions of some distributions, assuming the significands are fractional and both significand and exponent are in sign-and-magnitude notation.

[3]In general, the associative law does not hold in floating-point arithmetic, but there are instances where it would hold with gradual underflow but not with sudden overflow.

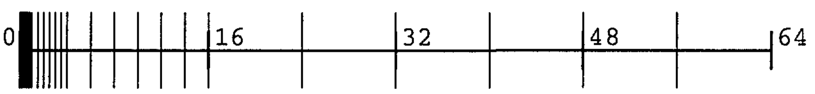
2.4.1 Radix

In principle, there are three independent radices in a floating-point arithmetic system: R_E, the radix used for the exponent arithmetic; R_S, the radix used for the significand arithmetic; and R, the main radix of the representation (as defined in Section 1). There is little advantage in using a radix larger than two for R_E, as exponents are relatively small, and the only operations performed on them are addition and subtraction; therefore, R_E is always taken to be two. Also, normalization and other shifting operations dictate that R_S should be some multiple of R and that R should be a power of two, so that shifts of the significand can be of an integral number of bits. R_S is, however, a purely implementation parameter that will not be here considered further. The rest of what follows therefore deals with just R, and we shall conclude that there is now little reason to have it larger than two.

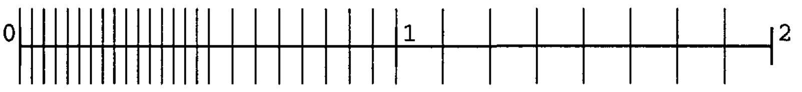
There are two main advantages in using a large radix R. The first is that an increased range of representable numbers is obtained: since, for a given placement of the radix point, the range is determined solely by the radix and the minimum and maximum permissible exponents, for a given number of exponent bits, larger and smaller numbers can be represented when a larger radix is employed (see Figure 2). This also means that a large radix can be used to counterbalance a small exponent-range, which may be a consequence of not having enough bits. The second advantage of a large radix lies in the fact that although R is a representational parameter, it affects the implementation in terms of the speed that can be achieved, especially in the execution of floating-point addition and subtraction. This is a consequence of the fact that fewer alignment and normalization shifts are required as the radix increases: one study has shown that as the radix is varied from two to sixteen, the probability that no alignment shifting is required increases from 0.3 to 0.5, the probability that no normalization shifting is required increases from 0.6 to 0.8, and the probability of overflow in the addition of significands decreases from 0.2 to 0.06 [30]. Nevertheless, the trend in the design of arithmetic units has been to make the time spent on shifting independent of the operands, and for such implementations a large radix does not offer any benefit in this respect. The only case where shifting time might



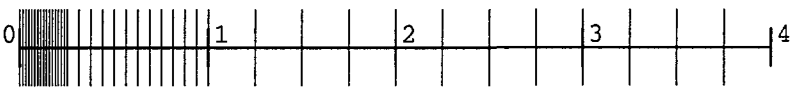
(i) $R = 2$, 3-bit exponent, 4-bit significand



(ii) $R = 4$, 3-bit exponent, 4-bit significand



(iii) $R = 2$, 2-bit exponent, 5-bit significand



(iv) $R = 4$, 2-bit exponent, 5-bit significand

Figure 2: Distribution of floating-point numbers

still be considered important is in the design of digit-serial units, in which the shifting is completely serial.

The above benefits of a large radix are, however, not obtained for free: First, for a given precision, increasing the radix also leads to a loss in "accuracy", as, in general, fewer significant *bits* of the significand are retained during normalization and rounding. This is apparent if we consider the fact that with a radix greater than two, even normalized significands can have several leading non-significant bits, and a unit change in the exponent requires shifting the significand by several bits. Second, as there is only a finite number of bits available and, therefore, only a finite number of numbers can be represented, the increased range that is obtained with a large radix means that there is also an increase in the sparseness within the range; that is, there is a greater separation between every pair of adjacent representable numbers (see Figure 2). The consequences of this is that certain errors get larger with increases in the radix.

In summary, a large radix gives the advantages of a large range and high speed in the implementation, but the price to be paid for these may be low "accuracy" and large errors (see also Section 2.5). One study has concluded that for "accuracy", radix-2 is the best for normalized operands, and radix-4 is the best for unnormalized operands. The majority of machines that have been built to date have employed a radix of two. Of greater radices, sixteen has been used more frequently than any other, although a few machines have had a radix of eight, and at least one machine has employed a very large radix — the radix-256 Los Alamos MANIAC II. Nevertheless, advances in technology have been such that improvements in performance and reductions in the costs of hardware make it doubtful that there should now be any advantages to using a radix larger than two. More importantly, two is the radix that has been chosen in IEEE-754, and nowadays this is what is implemented in almost all machines.

2.4.2 Exponent representation

There are two main decisions to be made regarding exponent representation: one is the number of bits to be allocated; the other is the notation to be used for representation. (One could also include the radix for exponent computations as a third, but, as has been indicated above, this is almost always taken to be two.) The number of bits allocated determines the range. A wide exponent field is therefore preferable in this respect, but as this involves a trade-off with the number of bits allocated for the significand — typically, both exponent and significand reside in one data unit (e.g. a word) — and for most applications a relatively small number of exponent bits will give an adequate range, the number of bits allocated to the exponent typically ranges from eight to sixteen. (The number of bits allocated to the exponent is discussed further in Section 2.4.5.)

Although we have stated above that, in principle, any one of the three standard notations for representing signed fixed-point numbers could be used for the exponent, in practice it is not just simply a matter of picking any one of the three: a closer examination of the special position of the number zero leads to the conclusion that a slightly different notation is desirable. Take any representation for the number zero. It is not clear whether the significand should be viewed as normalized or not; moreover, mathematically, any representation $\langle E, 0\rangle$ evaluates to zero. It may therefore be said that zero does not have a unique representation. Now, suppose that the addition of some number X and zero is to be carried out. If X has the smaller exponent, a loss of significant digits can occur (at the less significant end) when its significand is shifted during alignment. It is, however, possible to prevent this from happening if it can be guaranteed that the exponent used to represent zero is never larger than that used to represent *any* number X. This is easily achieved by using the smallest possible exponent — that is, a negative one of the largest permissible magnitude — for the representation of zero. There is another reason for picking the smallest possible exponent: Sometimes computational errors lead to a result with a significand that should be, but is only close to, zero. Now, for a given significand, a negative exponent represents a number that is closer to zero than one with a positive exponent. Therefore, a representation with the smallest possible exponent may be said to represent a number closer to to zero and is therefore a better representation of zero than all other representations. This choice of exponent for zero also has, as a consequence, the advantage of ensuring that, excluding the role of the sign bit, there is now a unique representation for zero, and this representation may be taken to be "normalized".

From an engineering point of view, it is convenient if (plus) zero is represented by a string of all 0s, as this allows easy detection. (It should, however, be noted that when the significand has a hidden bit, an explicit significand pattern of all 0s can no longer be relied on to indicate a representation of zero, and the reservation of a special exponent, as indicated above, or some other similar measure, must be used for this.) This can be achieved, in conjunction with the preceding constraint on exponents, by storing a *biased exponent* (or *characteristic*), instead of the true exponent; for uniformity and ease of implementation, this is done for every number. Thus if the true exponent, E, satisfies the relation $-N \leq E \leq M$, then the biased exponent, E^*, is made to satisfy the relation $0 \leq E^* \leq N + M$; that is, $E^* = E + N$, and, therefore, $\langle 0, 0\rangle$ can be used as the unique representation for zero. (The term *exponent spill* will now refer to exponents outside the new range, and when an exponent is biased by adding N, we say that it is in *excess (XS)-N* notation.) Thus in IEEE-754, zero is represented with a biased exponent of 0 and a signficand of 0.

Another advantage of biasing exponents is that with formats in which the ordering is of the form $\langle$*number–sign*, *exponent*, *rest–of–significand*$\rangle$, the ordering of patterns used to represent floating-point numbers is similar to that of the patterns used to represent fixed-point numbers; and so some fixed-point instructions, such as those of tests for sign, test for zero, and comparisons can be used directly with the floating-point numbers.

The last decision that has to be made about exponents is how to partition the possible exponents between negative and positive: The number of exponents that can be represented in binary is even and therefore (with the zero exponent excluded) there will either be one more negative exponent than there are positive exponents or vice versa. Consequently the reciprocals of some representable numbers will not be representable: the reciprocals of numbers represented with the "extra" exponents will overflow or underflow. In IEEE-754 the decision has been made to have more positive exponents than negative ones, on the grounds that underflow is usually less of a problem than overflow.

2.4.3 Significand representation

Given that addition and subtraction are the most common arithmetic operations, and these are easiest to implement with radix complement notation, that notation would appear to be the best notation for the significand. Nevertheless, there are a number of factors that suggest another choice. First, as has been pointed out (Section 2.3), for one good method of dealing with underflow, it makes sense to have both plus zero and minus zero; this is possible only with sign-and-magnitude and diminished-radix complement notations. Second, the lack of symmetry in the range of numbers representable in radix complement notation is something of an inconvenience: it is not the case that the negation of every representable number is also representable. Third, the four operations $A+B$, $-A+B$, $A+(-B)$, and $-A+(-B)$ cannot be carried out with *equal* ease in radix complement but can be in sign-and-magnitude and diminished-radix complement notations. Fourth, some *rounding* procedures, which are used to reduce intermediate results to final precision, are easier to implement or give more consistent results with sign-and-magnitude than with diminished-radix complement or radix complement notations.

The above considerations, together with the fact that multiplication and division of operands of all signs are easier to implement with sign-and-magnitude notation than with the other two standard notations, suggest that this is the best choice, and it is this that has been selected for IEEE-754. Of course, addition and subtraction are more problematic in this notation, but there is no reason why the arithmetic operations have to be carried out in the notation of representation: a number of computers use sign-and-magnitude for representation but *in essence* convert to and from radix complement or (less commonly) diminished-radix complement to implement these two operations.

2.4.4 Location of radix point

There are three options for the location of the radix point in the significand: at the left-hand end, i.e. between the sign bit and the rest of the significand proper; at the right-hand end, just after the least significant bit of the significand; and anywhere in between these two extremes. The first placement results in a significand whose magnitude does not exceed unity and is therefore a proper fraction; this was the most common choice prior to the introduction of IEEE-754. The second placement gives an integral significand and was employed in many Control Data Corporation machines. And the third placement gives an improper fraction and is now the most common with the adoption of IEEE-754. Assuming a fixed exponent range, one immediate effect of chosing one of the three is that the range is affected: in going from the first case to the second, the ends of the range of representable numbers are multiplied by a factor of R^p, where R is the radix and p is the precision used; however, the gaps between adjacent representable numbers also increase, as do the errors in representation.

As far as the basic operations are concerned, the location of the radix point does not matter for addition and subtraction, as the radix points must still be aligned, and the sole effect of relocating them is to scale the operands and results by some power of the radix used. The situation is, however, different for multiplication and division, and we shall examine these cases in detail. In what follows, we shall assume that the significands are of a radix R in sign-and-magnitude notation and with p digits used for the magnitude. We shall also assume that neither of the operands involved is zero. Thus with a radix point at the right-hand end of normalized significands, the magnitudes of the significands will lie between R^{p-1} and $R^p - 1$, and with the radix point at the left-hand end, they will lie between R^{-1} and $1 - R^{-p}$. Let the two operands to be multiplied or divided be $X \triangleq S_x R^{e_x}$ and $Y \triangleq S_y R^{e_y}$.

Application of the basic multiplication procedure to X and Y yields $XY = S_x S_y R^{e_x + e_y}$. For integral operands, we have $R^{p-1} \leq |S_x| < R^p$, $R^{p-1} \leq |S_y| < R^p$, and, therefore, $R^{2p-2} \leq |S_x S_y| < R^{2p}$. This product is $2p$ digits wide and, for representation within p digits, has to be shifted down by p digits (i.e. divided by R^p) and the associated exponent increased accordingly. Therefore, the result significand will be $S_x S_y R^{-p}$ (where $R^{p-2} \leq |S_x S_y R^{-p}| < R^p$), and the exponent will be $e_x + e_y + p$. The new significand product is representable in p digits, but it can be subnormal and may therefore require a normalization shift. If, on the other hand, the significands are fractional, then $R^{-1} \leq |S_x| < 1$ and $R^{-1} \leq |S_y| < 1$ yield $R^{-2} \leq |S_x S_y| < 1$. This significand product is already within the range representable in p digits, although it too may require a normalization shift — when it is subnormal. We therefore conclude that more complex exponent-arithmetic will be required the farther the radix point is from the left-hand end of the significand.

The situation for division is similar. Application of the basic division procedure to the two operands above yields $X \div Y = (S_x \div S_y) R^{e_x - e_y}$. For integral operands, we have $R^{-1} \leq |S_x \div S_y| < R$. To get the quotient back into the permitted range, it is shifted up by p digits (i.e. multiplied by R^p) and the exponent reduced accordingly, to yield the significand $(S_x \div S_y)R^p$ (and $R^{p-1} \leq |(S_x \div S_y)R^p| < R^{p+1}$), with the exponent $e_x - e_y - p$. This new significand may require a normalization shift, since it can be supernormal. Alternatively, the former shift could be of $p - 1$ digits, in which case normalization will be required for subnormal results. Note that since $S_x \div S_y$ is not necessarily integral, we are here assuming that the quotient-significand may be held in non-integral form. (This is not problematic: With basic division algorithms, if the dividend is always twice as wide as the divisor, we can ensure that the quotient is always integral if S_x is scaled by R^p prior to the division. In this case, X is now represented by a significand of $\widehat{S}_x \triangleq S_x R^p$, and its exponent is accordingly reduced to $e_x - p$. Then $R^{2p-1} \leq |\widehat{S}_x| < R^{2p}$ and $R^{p-1} \leq |S_y| < R^p$ yield a significand that satisfies the condition $R^{p-1} < |\widehat{S}_x \div S_y| < R^{p+1}$, with the exponent $e_x - e_y - p$. A normalization shift may be required, since the quotient can be subnormal or supernormal. It is, however, possible to avoid the supernormal case by initially scaling by R^{p-1} instead of R^p.) With fractional operands, $R^{-1} \leq |S_x| < 1$ and $R^{-1} \leq |S_y| < 1$ yield $R^{-1} < |S_x \div S_y| < R$. This may require a normalization shift, since it can be supernormal. It is, however, possible to ensure a fractional result if it can be guaranteed that the magnitude of the dividend is always smaller than that of the divisor. Since the significands are normalized, to ensure that the latter condition holds, it is sufficient to shift the dividend down by one digit, to $\widehat{S}_x \triangleq S_x R^{-1}$, and decrement the exponent by unity. Then $R^{-2} \leq |\widehat{S}_x| < R^{-1}$ and $R^{-1} \leq |S_y| < 1$ yield $R^{-2} < |S_x \div S_y| < 1$. The result is now fractional but will require a normalization shift if it is subnormal. Again, we see from this discussion that the exponent arithmetic required is more complex the farther the radix point is from the left-hand end of the significand.

From the preceding discussion, we may conclude that, at least with respect to exponent arithmetic, placing the radix point near the left-hand end is preferable to other choices. To this should be added the fact that having significands whose magnitudes exceed unity means that some representable numbers may not have representable reciprocals, whereas they might otherwise. Nevertheless, placing the radix at the least significant end does have some advantages. One is that little effort is required to convert between fixed-point representations and floating-point representations and, therefore, the hardware that is employed for operations with the one type of number can also be easily employed with the other. Another is that a direct correspondence can be made between bit positions in all relevant arithmetic-and-logic unit registers and bit positions on the data buses in the machine. But these advantages do not outweigh those for a left-hand placement. Until the introduction of IEEE-754, almost all computers had the radix point at the left-hand end of the significand, which was therefore fractional. IEEE-754 deviates slightly from this: by allowing both normalized significands (which have the representation $1 \cdot xxx \dots$) and, in exceptional cases, denormalized significands (which have the representation $0 \cdot xxx \dots$), the range of representable numbers is greatly increased, but the radix point is still very near the left end.

2.5 Range, precision, and accuracy

The *representational accuracy* (or *effective precision*) of a floating-point representation is the degree to which it is a faithful representation of the corresponding number and can be measured by the number of "correct" significant significand bits that it contains.[4] Evidently, the representational accuracy will vary according to the number represented, the radix, and the precision, with the last playing the largest role. Nevertheless, for a floating-point number *system* with normalized significands (hence unique representations), we can reasonably bound the accuracy: The minimum accuracy is determined by the precision and the radix, and the maximum accuracy is determined by the precision alone. The range, on the other hand, is determined primarily by the radix and the width of the exponent field; precision plays little part in determining the range, and its effect, with respect to a given range of representable numbers, is primarily on the density of the numbers within that range (see Figure 2) — specifically, the density will increase with the precision.

A floating-point number system with radix R, exponent range $[E_{min}, E_{max}]$, and significand range $[-a, b]$ has a range that is approximately $[-aR^{E_{min}}, bR^{E_{max}}]$. If the precision is p bits, then the maximum accuracy is $p-2$ bits, and minimum accuracy is $p-L-1$ bits, where $L \triangleq \log_2 R$ is the digit size. The measurement of minimum accuracy is obtained by the following reasoning. A normalized significand of p bits can have $L-1$ leading 0s; one bit is used for the sign bit; and one bit is dubious as a result of rounding. This gives $p-(L-1)-1-1 = p-L-1$ bits of accuracy. The figure for maximum accuracy is arrived at by assuming that there are no leading 0s, i.e. that all bits are significant. Thus, as stated above, all other things being equal, a large radix implies a larger variation in the accuracy and lower accuracy on average. Radix-2 is therefore especially advantageous, as it gives the best possible maximum accuracy and with no variation between maximum and minimum; furthermore, an additional bit of accuracy can be gained by having a hidden bit. Other than some desirable level of accuracy, the other factor that plays a significant role in determining the precision is the fact that alignment and normalization shifts must be in digit-sized units: for a given number of bits, the selected precision should allow for one sign bit and an integral number of digits.

Let us now briefly consider the options for a few formats. For a given number of bits, a good format is one that minimizes the number of unused bits, gives an adequate range, and also gives the highest possible accuracy. For a 32-bit format, fewer than seven bits in the exponent will not give an adequate range for many scientific applications, as the largest number that can be represented with six exponent bits (assuming fractional or near-fractional significands) is approximately 10^{19}. Suppose the radix is two. Then all possible partionings of the available bits between the exponent and significand will leave no unused bits. Therefore, if accuracy is of the utmost importance, seven or eight bits should be allocated to the exponent and the remainder allocated to the significand. If, on the other hand, it is the range that is of concern (possibly because there is to be only one format), then eleven or twelve bits for the exponent will still leave enough for a reasonable accuracy (about six decimal digits) in the significand. Similar considerations suggest that if accuracy is the main consideration, then the number of bits allocated to the exponent when the radix is four, eight, or sixteen should be seven in all three cases; and if range is the main consideration, then the number of bits allocated to the exponent should be eleven for radix-4 and radix-16 and ten for radix-8.

With a 64-bit format obtaining both sufficient range and accuracy is easy. Indeed, it should be possible — and this has been the case in many machines — to have it as the only format in a floating-point system. This suggests that the exponent should be at least eleven bits wide; for the upper bound, sixteen bits are probably more than enough for most applications. An examination of the possible formats within this range of exponent-bit allocations leads to the following conclusions: To get both an adequate range and high accuracy, the number of exponent bits should be eleven or twelve for radix-2 and radix-8 and eleven for radix-4 and radix-16. If accuracy must be traded for range, then the number of exponent bits should be fifteen or sixteen for radix-2, fourteen or fifteen for radix-4, and fifteen for radix-8 and radix-16. However, more than eleven bits for the exponent is quite wasteful with radix-8 and radix-16, and one would hope that the larger radix compensates for not having more than that number of exponent bits.

Table 2 shows the values of various parameters of floating-point number systems that have been implemented in some machines; italics indicate those that are important, for the reasons given above in the introduction. All figures given for the precision include the sign bit; for the entries marked with an asterisk, the figures for precision include a hidden bit; a double asterisk is used to indicate a radix point at the right end of the significand; and in the last column, $10^{3.01}$ has been used as an approximation of 2^{10} and the final exponent then rounded to the closest integer. The

[4]Some authors have used *precision* alone for this; others have used *precision* to refer to the value R^{-p}, for a radix-R system with p significand digits. R^{-p}, which is *dependent on p*, is the weight of the least significant digit of the significand and, therefore, the smallest difference between two representable numbers. The reader should therefore be aware that here terms such as *accuracy* and *precision* do not necessarily have the same meaning as in other contexts — e.g. conventional usage or in numerical analysis — and that IEEE-754 defines *precision* as we have done above.

reader is invited to evaluate these formats on the basis of the criteria established in the preceding discussions.

The IEEE-754 standard specifies four formats, as shown in Table 2. The two basic formats are the first (*single precision*) and the third (*double precision*); the other two are corresponding *extended-precision* formats that may be used to compute intermediate results — i.e. before rounding to one of the two basic formats — with greater accuracy than would otherwise be the case. For the 32-bit format, accuracy was considered to be the most significant factor, and, indeed, this dictated the choice of two for the radix. For the 64-bit format, the main consideration was the range and the requirement that the product of any two 32-bit numbers be representable in the 64-bit format.

2.6 Rounding and representation error

In this section we shall discuss the choice of rounding methods; this is important in ensuring that machine errors are kept to the minimum possible. The most evident machine error is that which that arises from the limitations of finitude: since computers (and registers) are finite objects, the representation of any number is necessarily finite, and some loss of information will be incurred when a number requires more representational facilities than the machine at hand permits: for example, the decimal the representation of the decimal 1/3 as $0.33\ldots3$, or the binary representation of the decimal 0.1 in a given number of digits. These errors are, accordingly, known as *representation errors* or, for reasons that will become apparent shortly, *round-off* (or *rounding*) *errors*.

Since, in what follows, we shall compute upper bounds on certain errors, the first order of business is to determine exactly what is to be measured. At first glance it might appear that if the correct (computed) intermediate result is X and the final result is $\widehat{X}$, then it should suffice to calculate $\widehat{X} - X$ as the error. This, the *absolute error* is, however, not always useful: in general, the significance of an error is relative to the magnitude of the actual result; for example, a difference of 0.1 between X and $\widehat{X}$ has greater import if X is 0.1 than if it is 10^6. Therefore, in what follows, we shall also consider the *relative error*, which, provided $X \neq 0$, is defined to be $(\widehat{X} - X)/X$; if $X = 0$, then the relative error is undefined. The relationship between $X, \widehat{X}$, and the relative error, ε_X, in the representation of X can be seen more clearly in the identity $\widehat{X} = X(1 + \varepsilon_X)$.

Frequently, an intermediate result is computed in more than p significand digits, but the machine requires that the final result be represented in exactly p digits. For example, a format may permit only p-digit significands, but, in general, the multiplication of two p-digit significands yields a $2p$-digit result. Likewise, pre-arithmetic shifting in addition and subtraction might mean that (assuming extended-precision arithmetic) the arithmetic operation is carried out on significands that are wider than p digits. Lastly, it may simply be the case that an operation is carried out for which an exact representation of the result requires an infinite number of digits, a decimal example being the result of dividing one and three. In all of these cases, the intermediate result must be reduced to fit within the given significand width. Most of what follows is concerned with *rounding methods*, which are procedures for achieving this reduction in width to p digits (excluding the sign digit), and their effects on the error. Rounding may be carried out (on the operands and intermediate results) before, during, or after the arithmetic operations; but it is generally better to leave it until after the operation. Normalization should take place before rounding, and not after, as rounding before normalization can remove digits that could otherwise end up in the significand and so generally leads to larger errors.

The bounds on errors will be calculated as follows. Suppose the rounding to p significand digits of $\langle S, E\rangle$, which represents SR^E, yields $\langle S_p, E\rangle$, which represents S_pR^E. Then absolute error has a magnitude $|(S_p - S)R^E|$, and the magnitude of the relative error is $|(S_p - S)R^E|/|SR^E| = |(S_p - S)/S|$. If b_a is the smallest upper bound on the absolute error in the significands and b_s is the largest lower bound on S, then the relative error is bounded from above by b_a/b_s. For normalized radix-R significands with k significant digits to the left of the radix point, $b_s = R^{k-1}$, and, therefore, the magnitude of the absolute error will be bounded from above by b_aR^E and that of the relative error is bounded by b_a/R^{k-1}. For most of the common rounding methods, b_a is R^{-p} or $R^{-p}/2$, and, therefore the bound on the magnitude of the absolute error is R^{E-p} or $R^{E-p}/2$, and the bound on the magnitude of the relative error is R^{1-p-k} or $R^{1-p-k}/2$. The term R^{-p}, which is the weight of the least significant digit of the significand, appears sufficiently often in this type of analysis that a special name has been coined for it: *ulp*, which stands for 'unit in the last place'. The relative error corresponding to an absolute error of half an ulp can vary by a *wobble*, a factor of up to R. The upper bound on the wobble is known as the *machine epsilon*.

We shall now examine a number of rounding methods. A rounding method will be said to be *biased* if it has a tendency to round in one direction. Let S_p be the result of rounding S to p significand digits to the right of the radix point and k to the left. If the expected value of S_p is smaller than S, then we say that the rounding method is *downward biased*; and if the expected value of S_p is larger than S, then we say that the rounding method is *upward biased*. In looking at the bias of a rounding method, we shall consider three cases: when the significands are evenly distributed among the positive and negative ones, when the significands are evenly distributed among only the positive ones, and when the significands are evenly distributed among only the negative ones. (Finer analyses are of course possible, but these lead to similar conclusions as those from the three groupings that we have chosen, so we shall not here consider them.) A good rounding method should be unbiased in all three cases, and this is what "always unbiased" will mean in what follows.

The simplest method for rounding is simply to drop all

Machine	**Radix**	**Exponent width (notation)**	**Precision-bits (notation) incl. sign**	**Min. acc.**	**Max. +ve no.**
DEC PDP-8*	2	12 (2's compl)	25 (S&M)	23	10^{616}
DEC PDP-11*	2	8 (XS-128)	25 (S&M)	23	10^{38}
Burroughs B-5500	8	7 (S&M)	39 (S&M)	35	10^{56}
IBM 360/370	16	7 (XS-64)	25 (S&M)	20	10^{75}
		7 (XS-64)	54 (S&M)	49	10^{75}
IBM Stretch	2	11 (S&M)	49 (S&M)	47	10^{308}
CDC 6000**	2	11 (XS-1024)	49 (1's compl)	47	10^{322}
Ferranti Atlas	8	8 (2's compl)	40 (R's compl)	36	10^{114}
Manchester MU5	16	11 (2's compl)	21 (R's compl)	16	10^{1225}
		11 (2's compl)	53 (2's compl)	48	10^{1225}
Cray-1	2	15 (XS-4096)	49 (S&M)	47	10^{4931}
CDC Cyber 205	2	8 (2's compl)	24 (2's compl)	22	10^{38}
		16 (2's compl)	48 (2's compl)	46	10^{9682}
HEP	16	7 (XS-64)	57 (S&M)	52	10^{140}
*DEC VAX-11**	2	8 (XS-128)	25 (S&M)	23	10^{38}
		8 (XS-128)	57 (S&M)	55	10^{38}
		11 (XS-1024)	54 (S&M)	52	10^{308}
		15 (XS-16384)	112 (S&M)	110	10^{4931}
*IEEE Std**	2	8 (XS-127)	25 (S&M)	23	10^{38}
		≥ 11	≥ 33	≥ 31	$>10^{308}$
		11 (XS-1023)	54	52	10^{308}
		≥ 15	≥ 65	≥ 63	$>10^{4931}$

Table 2: Parameters of some floating-point formats

digits beyond the number required. This is usually referred to as *chopping*, *truncation*, or *rounding-towards-zero* (since the magnitude of the significand is decreased in that direction). Thus, for example, with a precision of three, 0.3141 would be chopped to 0.314, and 0.2718 would be chopped to 0.271. For a radix of R and p digits used for the significand, the significance of the last digit retained is R^{-p}. Therefore, the magnitude of the absolute error in chopping cannot exceed R^{E-p}, and the magnitude of the relative error cannot exceed $R^{-p}/R^{k-1} = R^{1-p-k}$. Chopping is downward biased if the significands are in radix complement notation. With numbers in diminished-radix complement and sign-and-magnitude notations, chopping is biased only for a set of numbers of the same sign: the bias is upward for a set of negative numbers and downward for a set of positive numbers. Chopping is one of the four rounding methods specified in IEEE-754.

A better method of rounding is *round-to-nearest-value*; this is also usually referred to — not very informatively — as *unbiased rounding* or *symmetric rounding*. The procedure is as follows. If the digits to be discarded represent a value of magnitude greater than $R^{-p}/2$, then the magnitude of the value represented by the retained digits is increased by unity, by incrementing or decrementing (according to the sign of the represented number) in the least significant digit position; if those digits represent a value that is less than $R^{-p}/2$, then chopping takes place; otherwise — that is, if the value represented by the digits to be discarded is exactly $R^{-p}/2$ and is therefore equally close to two values of precision p — a fair choice is made between the two equally close values. Here, *fair* means that digit p in the set of all possible roundings is incremented or decremented equally often, and, therefore, all possible values have an equal chance of being selected. An example of a fair selection rule, known as *round-to-nearest-even*, is to select the even one — that is, the one with a least significant bit of 0 — of the two equally close values; *round-to-nearest-odd*, also known as *R*-rounding*, is an analogous rule.[5] (The latter was implemented in the Burroughs Scientific Processor, one of the few early machines to implement round-to-nearest.) Thus, for example, if the p is three, then 0.3141 would be rounded to 0.314, 0.2718 to 0.272, 0.2345 to 0.234 or 0.235, −0.2718 to −0.272, and −0.3141 to −0.314. It is evident that if all significands are equally probable, then the increment at digit p will be performed in exactly half of all cases. The bounds for errors are obtained as follows. In the first of the three rounding cases, we have a reduction of at least $R^{-p}/2$ (the discarded digits) and an increase of R^{-p} (adding unity to the magnitude at digit p) in the magnitude of the significand; this is a net increase of at most $R^{-p}/2$. In the second case, chopping causes a magnitude reduction of at most $R^{-p}/2$. And in the third case, assuming a fair selection rule, there is an increase or decrease by $R^{-p}/2$ equally often. Therefore, the magnitude of the absolute error with

[5]Knuth suggests round-to-nearest-even if $R/2$ is odd and round-to-nearest-odd if $R/2$ is even [21]; this will reduce error in the common operation of division by 2. Keir suggests round-to-nearest-even if R is divisible by 2 but not by 4 and round-to-nearest-odd if R is divisible by 4 [20].

round-to-nearest is bounded by $R^{E-p}/2$ and that of the relative error by $R^{1-p-k}/2$. Round-to-nearest is always unbiased and is therefore a great improvement on chopping. It is, however, relatively difficult to implement because of the required addition/subtraction and the efforts required to deal the boundary case. Nevertheless, of the four rounding methods specified in IEEE-754, round-to-nearest is the default method.

As a solution to the problem of boundary-detection in round-to-nearest, many computers, such as those in the DEC VAX-11 range, adopted a simpler variant instead. This is the rounding method that is routinely taught in primary-school mathematics — *half-adjust rounding* (or *round-to-nearest-up*). Here, the boundary case is simply dealt with by incrementing the magnitude at digit p. Thus in the last example above, 0.2345 would always be rounded to 0.235, and never to 0.234. The method is easier to implement than round-to-nearest, as it is sufficient to examine only digit $p + 1$ of the result to be rounded in order to determine what action is to be taken. Indeed, it is not even strictly necessary to examine digit $p + 1$: it is sufficient always to increment at digit $p + 1$ and then chop. The magnitudes of the absolute and relative errors in half-adjust rounding are still bounded by $R^{E-p}/2$ and $R^{1-p-k}/2$, respectively, as in round-to-nearest, but the method is not always unbiased. If the significands are equally distributed among the positive and negative ones, then it is unbiased. Otherwise, it is downward biased for a set of all-negative significands and upward biased for an all-positive set; this is what we should expect, since the boundary case is always settled in the direction that increases the magnitude.

A third form of rounding is to chop and then force a 1 (or a 0 for a negative significand in diminished-radix complement notation) into the least significant bit position. This is known as *force-1 rounding*, *jamming*, *making-odd*, or, more popularly, as *von Neumann rounding* and is almost as fast as chopping. Assuming that all significands occur with equal probability, the largest error with this method occurs when the least significant bit retained is 1 but should be 0 (or 0 instead of a 1 in diminished-radix complement notation). Then the magnitude of the absolute error is bounded by R^{E-p}, and that of the relative error is bounded by R^{1-p-k}. The biasing for jamming is similar to that for half-adjust rounding.

Jamming can be improved to *conditional force-1 rounding* (or *conditional jamming*), in which the force-1 operation is carried out only when it is known that none of the digits to be discarded is significant; that is, if not all the bits to be discarded are 0s (or 1s for a negative significand in diminished-radix complement notation). The speed of an implementation of conditional jamming will therefore be limited primarily by the all-0s/all-1s detection phase. With this rounding method, the magnitude of the absolute error is bounded by R^{E-p}, and that of the relative is bounded by R^{1-p-k}. The method may therefore appear to be no better than chopping, but this is not so: it is always unbiased, whereas chopping is not.

ROM-rounding is a rounding method that is qualitatively good and not as costly, in hardware or operation time, to implement as round-to-nearest. The method is as follows. Let $d_1 d_2 \ldots d_p d_{p+1} \ldots$ be the intermediate significand. To round this to p digits, the low-order $m + 1$ digits $d_{p-m+2} d_{p-m+1} \ldots d_{p+1}$, where m is reasonably small, are used to address a lookup table (implemented as, say, a programmable logic array or read-only memory) whose output is a sequence of m digits that is equivalent to half-adjust rounding the $m + 1$ address digits to m digits. (This is done only if half-adjust rounding those digits would not produce significand overflow — for example, if for positive radix-2 significands, it is not the case that all the bits are 1s.) The output digits are then appended to digits $d_1 d_2 \ldots d_{p-m+1}$ to obtain a p-digit result. Chopping occurs in those cases where half-adjust rounding would lead to rounding overflow, which is not easily accommodated in implementing this procedure. For this reason, the error bounds for ROM-rounding are as bad as those for chopping, although, on the whole, the method is better than chopping. The biasing is similar to that for chopping.

Table 3 shows the result produced by each of the above rounding procedures when a normalized 5-bit radix-2 fractional significand in sign-and-magnitude notation is rounded to three bits. The entries marked with an asterisk are the results of renormalizing supernormal significands. (The values in Table 4 are calculated from the supernormal values.) A ROM addressed with three bits, and yielding a 2-bit output, is assumed for ROM-rounding. Table 4 gives the absolute errors corresponding to the entries in Table 3. The row labelled 'Max' gives the largest error-magnitude in each column; the row labelled 'Sum+' gives the sum of errors for the positive significands; and the row labelled 'Sum–' gives the sum of errors for the negative significands. Table 4 confirms a number of the comments made above.

The measure that we have used above to compare the various rounding methods is the *maximum relative representation error*; some casual remarks have also been made about bias. Finer analyses usually look at the magnitude of the *average relative representation error*, estimates of the *average bias*, and the results of statistical tests. As far as the rounding methods above are concerned, investigations of this sort have confirmed the conclusions already reached above. A classification by the magnitudes of the bounds on relative errors shows that the ordering in terms of decreasing quality is: round-to-nearest and half-adjust rounding, ROM-rounding, force-1 rounding, conditional force-1 rounding, and chopping. (Of course, a precise comparison based on the magnitude of errors requires finer analyses, such as those discussed above.) In terms of implementation aspects, both round-to-nearest and half-adjust rounding require an addition, which may involve extra expense in hardware and a loss of performance, since carry-propagation is implied. Furthermore, temporary unnormalization and, as a result, overflow in the significand, and, consequently, possibly in the exponent, can occur because of the addition.

Parameter	Format			
	single	single extended	double	double extended
format-width	32	≥ 43	64	≥ 79
precision	24	≥ 32	53	≥ 64
accuracy	22	≥ 30	51	≥ 62
exponent-width	8	≥ 11	11	≥ 15
max-exponent	128	≥ 1024	1024	≥ 16384
min-exponent	-127	≤ -1023	-1023	≤ 16383
exponent-bias	127		1023	
accuracy	22	≥ 30	51	≥ 62

Table 5: IEEE Standard 754 floating-point formats

Round-to-nearest also requires special actions to be taken in the boundary case. Force-1 rounding may be viewed as better than round-to-nearest and half-adjust roundings in that addition is not required in the implementation, and, therefore, overflow cannot occur; however, the conditional case requires special handling of the bits to be discarded, and this may involve additional hardware and time.

If a machine is to give consistently good results, it is important that an unbiased method be used. For unbiased methods, the absolute-error function is symmetric around zero, and if a large set of numbers is used to produce a small set of results, an unbiased method increases the likelihood that errors will, to some extent, cancel out each other. This narrows the choice of a good rounding method to conditional force-1 and round-to-nearest, both of which have been shown to be unbiased in cases where the others are not.
Of the two, the latter has smaller error bounds, and despite the difficulties mentioned above, round-to-nearest-even is the default method in IEEE-754. In the past, implementations of round-to-nearest were not considered worthwhile, but with the continued drop in the costs and continued rise in the performance of hardware, this view is now hard to justify; however, other, faster, methods may still be found implemented in some special-purpose computers. The reader will find detailed discussions of rounding and related error analysis in [20, 23, 25, 32, 33, 34].

3 A summary of IEEE-754

In this section we shall give a summary of various aspects of the IEEE-754 standard; more detailed descriptions can be found in the published literature. The aspects covered are formats, rounding methods, operations, and exceptions.

3.1 Formats and values

The IEEE standard specifies four floating-point number formats. The two basic formats are *single precision* and *double precision*, which are shown in Figure 3. There is also an *extended* format corresponding to each of the two. The extended precision formats are intended to be used in situations where it is desirable to compute with a higher precision than that used for the final result. The bit allocations and other characteristics of the formats are summarized in Table 5.

The radix is two for all four formats. The significands are in sign-and-magnitude notation and are improper fractions with a hidden bit when normalized: the magnitudes of the normalized significands are in the range [1,2), and those of denormalized ones lie in the range (0,1). The exponent values shown on the fifth and sixth lines of the table are unbiased. The extra value — that is, the one without a negation — in each exponent range has been put on the positive side so that the reciprocal of the numbers of the smallest magnitudes do not overflow; reciprocals of the largest magnitudes will underflow, but underflow is considered to be less likely to happen than overflow. It is required that any conforming implementation support the single format and recommended that one of the following three combinations be supported: (i) single and single-extended; (ii) single, double, and double-extended; (iii) all four formats.

The exponents are biased, and two of them may not be used for normal numbers. The latter two are reserved for special cases: the smallest possible exponent is reserved for $-0, +0$, and denormalized numbers; and the largest possible exponent is reserved for $-\infty, +\infty$, and NaNs. Thus, after biasing, the interpretation of the combinations of the various fields is as follows:

- A maximal exponent and a non-zero significand represents NaN. The value in the significand may be set according to the machine implementation, thus giving different NaNs.
- A maximal exponent and a zero significand represents $-\infty$ or $+\infty$, according to the sign bit. For every finite number x, $-\infty < x < +\infty$.
- A minimal exponent and a non-zero significand represents a denormalized number.
- A minimal exponent and zero significand represents -0 or $+0$, according to the sign bit.
- The remaining exponent values and significands represent normal numbers.

The uses of the special values are as follows. The result of an overflowing operation or of division of a real number by zero is $+\infty$ or $-\infty$, according to the signs of the dividend and divisor. The standard calls for two types of NaN: *signalling* and *quiet*; the signalling NaN is intended to provide values for uninitialized variables and arithmetic-like enhancements not included in the standard, and the quiet NaN is intended to provide retrospective diagnostic information for invalid or unavailable operands and results.

3.2 Rounding

The standard requires that rounding the result of an operation with normalized operands be the same as that obtained by rounding an intermediate result of infinite precision. Round-to-nearest-even is specified as the default

Interm. Result	Rounding method						
	Chop	Nearest even	Nrst. odd	Half-adjust	Force1	Cond. force1	ROM
1.1111	1.11	1.10*	1.10*	1.10*	1.11	1.11	1.11
1.1110	1.11	1.10*	1.11	1.10*	1.11	1.11	1.11
1.1101	1.11	1.11	1.11	1.11	1.11	1.11	1.11
1.1100	1.11	1.11	1.11	1.11	1.11	1.11	1.11
1.1011	1.10	1.11	1.11	1.11	1.11	1.11	1.11
1.1010	1.10	1.10	1.11	1.11	1.11	1.11	1.11
1.1001	1.10	1.10	1.10	1.10	1.11	1.11	1.10
1.1000	1.10	1.10	1.10	1.10	1.11	1.10	1.10
0.1000	0.10	0.10	0.10	0.10	0.11	0.10	0.10
0.1001	0.10	0.10	0.10	0.10	0.11	0.11	0.11
0.1010	0.10	0.10	0.11	0.11	0.11	0.11	0.11
0.1011	0.10	0.10	0.11	0.11	0.11	0.11	0.11
0.1100	0.11	0.11	0.11	0.11	0.11	0.11	0.11
0.1101	0.11	0.11	0.11	0.11	0.11	0.11	0.11
0.1110	0.11	0.10*	0.11	0.10*	0.11	0.11	0.11
0.1111	0.11	0.10*	0.10*	0.10*	0.11	0.11	0.11

Table 3: Examples of roundings

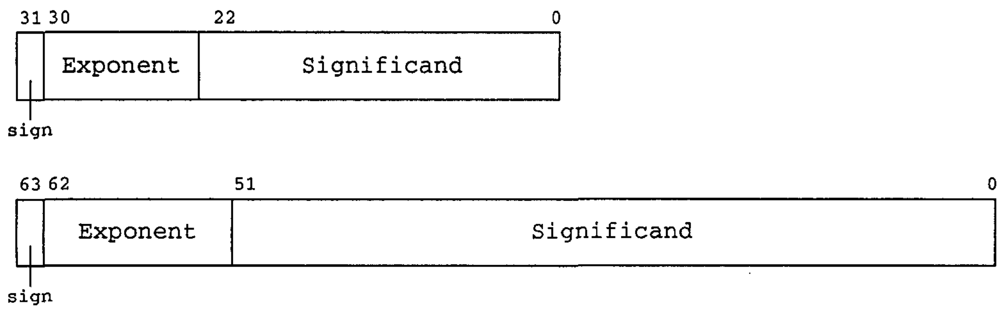


Figure 3: IEEE Standard single and double formats

rounding method in any conforming implementation; and any intermediate result of a magnitude greater than or equal to $2^e(2 - 2^{-p})$, where e is the largest permissible normal exponent and p is the precision, is rounded to $-\infty$ or $+\infty$, according to its sign. An implementation of the standard is also required to provide the following three user-selectable rounding methods:

- *round-toward-plus-infinity*, in which the result should be the value closest to and no less than the result of rounding the corresponding infinitely precise result
- *round-toward-minus-infinity*, in which the result should be the value closest to and no greater than the result of rounding the corresponding infinitely precise result
- *round toward zero* (chopping), in which the result should be the value closest to and no greater in magnitude than the result of rounding the corresponding infinitely precise result

The first two user-selectable methods are intended to support Interval Arithmetic.[6]

3.3 Operations

The operations that any conforming implementation must support are *compare, add, subtract, multiply, divide, remainder, square root, round-to-integer* in floating-point format, conversions between the different supported floating-point points, conversions between supported floating-point formats and integer formats, and conversions between binary and decimal.

The *add, subtract, multiply, divide*, and *remainder* operations are to be supported for any two operands of the same format, and the result of each must be represented in a format that is at least as wide as that of the operands; it is further recommended that the operations be supported for operands of differing formats. The result of *remainder* is defined whenever the divisor is not zero, irrespective of the rounding method used. The *square root* is defined for -0 and all positive operands and in all supported formats; all square roots are to be positive, except for $\sqrt{-0}$, which is -0.

Conversion between the different floating-point formats

[6]In *Interval Arithmetic*, each number is represented by two machine numbers that specify an interval in which the represented number lies. Arithmetic is performed on the interval delimiters. The result of each arithmetic operation is an interval obtained by carrying out the operation twice and then rounding down in one case and up in the other.

Interm. Result	Rounding method						
	Chop	Nearest even	Nrst. odd	Half-adjust	Force1	Cond. force1	ROM
1.1111	$\frac{3}{16}$	$-\frac{1}{16}$	$-\frac{1}{16}$	$-\frac{1}{16}$	$\frac{3}{16}$	$\frac{3}{16}$	$\frac{3}{16}$
1.1110	$\frac{2}{16}$	$-\frac{2}{16}$	$\frac{2}{16}$	$-\frac{2}{16}$	$\frac{2}{16}$	$\frac{2}{16}$	$\frac{2}{16}$
1.1101	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$
1.1100	0	0	0	0	0	0	0
1.1011	$\frac{3}{16}$	$-\frac{1}{16}$	$-\frac{1}{16}$	$-\frac{1}{16}$	$-\frac{1}{16}$	$-\frac{1}{16}$	$-\frac{1}{16}$
1.1010	$\frac{2}{16}$	$\frac{2}{16}$	$-\frac{2}{16}$	$-\frac{2}{16}$	$-\frac{2}{16}$	$-\frac{2}{16}$	$-\frac{2}{16}$
1.1001	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$-\frac{3}{16}$	$-\frac{3}{16}$	$\frac{1}{16}$
1.1000	0	0	0	0	$-\frac{4}{16}$	0	0
0.1000	0	0	0	0	$\frac{4}{16}$	0	0
0.1001	$-\frac{1}{16}$	$-\frac{1}{16}$	$-\frac{1}{16}$	$-\frac{1}{16}$	$\frac{3}{16}$	$\frac{3}{16}$	$-\frac{1}{16}$
0.1010	$-\frac{2}{16}$	$-\frac{2}{16}$	$\frac{2}{16}$	$\frac{2}{16}$	$\frac{2}{16}$	$\frac{2}{16}$	$\frac{2}{16}$
0.1011	$-\frac{3}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$
0.1100	0	0	0	0	0	0	0
0.1101	$-\frac{1}{16}$	$-\frac{1}{16}$	$-\frac{1}{16}$	$-\frac{1}{16}$	$-\frac{1}{16}$	$-\frac{1}{16}$	$-\frac{1}{16}$
0.1110	$-\frac{2}{16}$	$\frac{2}{16}$	$-\frac{2}{16}$	$\frac{2}{16}$	$-\frac{2}{16}$	$-\frac{2}{16}$	$-\frac{2}{16}$
0.1111	$-\frac{3}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$-\frac{3}{16}$	$-\frac{3}{16}$	$-\frac{3}{16}$
Max	$\frac{3}{16}$	$\frac{2}{16}$	$\frac{2}{16}$	$\frac{2}{16}$	$\frac{4}{16}$	$\frac{3}{16}$	$\frac{3}{16}$
Sum+	$-\frac{3}{8}$	0	0	$\frac{1}{4}$	$\frac{1}{4}$	0	$\frac{1}{4}$
Sum−	$\frac{3}{8}$	0	0	$-\frac{1}{4}$	$-\frac{1}{4}$	0	$-\frac{1}{4}$

Table 4: Errors in roundings

is to be effected by one of the four rounding methods, as is the conversion between floating-point and integral numbers in the same format. Conversion between decimal strings (in at least one format) and floating-point numbers (in all supported formats) are subject to some constraints on the magnitude of the values involved: for decimal to binary, the numbers must be in the range $[-(10^9-1)10^{99}, -(10^9-1)10^{-99}]$ or $[(10^9-1)10^{-99}, (10^9-1)10^{99}]$ for the single format and $[-(10^{17}-1)10^{999}, -(10^{17}-1)10^{-999}]$ or $[(10^{17}-1)10^{-999}, (10^{17}-1)10^{999}]$ for the double format; for binary to decimal conversion, the numbers must be in the range $[-(10^9-1)10^{53}, -(10^9-1)10^{-53}]$ or $[(10^9-1)10^{-53}, (10^9-1)10^{53}]$ for the single format and $[-(10^{17}-1)10^{340}, -(10^{17}-1)10^{-340}]$ or $[(10^{17}-1)10^{-340}, (10^{17}-1)10^{340}]$ for the double format.

3.4 Exceptions

There are five types of exception that must be *signalled* when detected, where *signalling* means setting an appropriate status flag, entering a trap routine, or both. Where a flag is set, the user must be able to test and restore the value of any flag. The fives types of exception are:

- *Invalid operation*, which is signalled if an operand is not valid for the operation to be performed. The invalid operations are any operation on a signalling NaN, addition or subtraction of infinities, multiplication of zero by an infinity, division of an infinity by an infinity, square root of a negative number, remainder in a case where the dividend is an infinity or the divisor is zero, comparison of unordered operands, and a conversion from floating-point to integer if the result cannot be represented in the format at hand. The result in each case is a quiet NaN.

- *Division by zero*, which is signalled if the dividend is a finite non-zero number and the divisor is zero. The result is plus or minus infinity, according to the sign of the dividend.

- *Overflow*, which is signalled if the magnitude of a result exceeds the largest finite number representable in the format of the operation. In the absence of a trap, the final result is determined by the rounding mode: the result with round-to-nearest is plus or minus infinity, according to the sign of the intermediate result; for round-toward-zero, the result is plus or minus the largest number representable in that format, according to the sign of the intermediate result; for round-toward-minus-infinity, the result is minus infinity for negative numbers, and for positive numbers it is the largest number representable in that format; and for round-toward-plus infinity, the result is plus infinity for positive numbers, and for negative numbers it is the most negative number representable in the format.

- *Underflow*, which is signalled if the magnitude of a result is too small for representation in the chosen format. The result delivered may be zero, a denormalized number, or plus or minus the number of the smallest representable magnitude.

- *Inexact result*, which is signalled if the rounded result of an operation is not exact or if overflow occurs but there is no overflow trap.

For both trapped overflow and trapped underflow, the result delivered for all operations, except conversions, is equivalent to that obtained by multiplying the infinitely precise result by 2^α, where α is 192 for the single format, 1536 for the double format, and $3 \times 2^{n-2}$ for an extended format with n exponent bits, and then rounding. The choice of α is such that the result is near the middle of the exponent range and can therefore be used in further operations, but with a smaller likelihood of further exceptions arising. The result for a trapped conversion is a result in a wider format or a quiet NaN if the wider format is not possible.

4 Summary

We have discussed a number of issues in the design of floating-point number systems and shown that the now widely-accepted IEEE-754 standard on floating-point arithmetic is the best system that has been formulated to date. It is our hope that this discussion will be useful to those seeking to understand the rationale for the design of IEEE-754, as well as to those seeking to design or understand other floating-point systems.

A summary of the main points discussed is as follows. Where possible, significands of floating-point representations should be normalized in order to retain as many significant bits as possible; underflow and overflow should be dealt with in a manner that allows computation to proceed, where possible, and which minimizes computation errors; thirty-two bit formats should have seven or eight bits for the exponent, and sixty-four bit formats should have eleven or twelve bits for the exponent; the exponent in a floating-point representation should be in biased two's complement notation; the significand should in radix-2 for the best accuracy, have a radix point close to the left-hand end, be in sign-and-magnitude notation, and have a hidden bit; the rounding method should be round-to-nearest; and indeterminate results should be represented with a pattern that provides diagnostic information.

Acknowledgements: We are indebted to the anonymous referees, whose comments have resulted in several improvements on the original manuscript.

Bibliography

1. Brent, R.P. 1973. On the precision attainable with various floating-point number systems. *IEEE Transactions on Computers*, vol. C-22, no. 6, pp 601–607.
2. Boehlender, G. 1990. What do we need beyond IEEE arithmetic? In: *Computer Arithmetic and Self Validating Numerical Methods* (Academic Press) pp 1–32.
3. Brown, W.S. 1981. A simple but realistic model of floating-point computation. *ACM Transactions on Mathematical Software*, vol. 7, pp 445–480.
4. Burks, A.W., H.H. Goldstine, and J. Von Neumann. 1946. Preliminary Discussion of the Logical Design of an Electronic Computing Instrument. Report, Institute for Advanced Study, Princeton, New Jersey. Reproduced in: E.E. Swartzlander, Ed., *Computer Design Development: Principal Papers*, Hayden Book, Rochelle Park, New Jersey, pp 221–259.
5. Campbell, S.G. 1962. Floating-point operation. In: W. Buchholz, Ed., *Planning a Computer System: Project Stretch*, McGraw-Hill, New York, pp 92–121.
6. Clenshaw, C.W. and F.W. Olver. 1984. Beyond floating point. *Journal of the ACM* 31:319–329.
7. Cody, W.J. 1973. Static and dynamic numerical characteristics of floating-point arithmetic. *IEEE Transactions on Computers*, vol. C-22, no. 6, pp 598–601.
8. Cody, W.J. 1981. Analysis of proposals for the floating-point standard. IEEE *Computer*, March, pp 63–69.
9. Coonen, J.T. 1980. An implementation guide to the proposed standard for floating-point arithmetic. *IEEE Computer*, January, pp 68–79.
10. Coonen, J. T. 1981. Underflow and denormalized numbers. IEEE *Computer*, March, pp 75–87.
11. Dahlquist, G. and A. Björck. 1974. *Numerical Methods*. Prentice-Hall, Englewood Cliffs, New Jersey.
12. Demmel, J. 1984. Underflow and the reliability of numerical software. *SIAM Journal of Scientific Statistical Computation*, 5(4):887–919.
13. Goldberg, D. 1991. What every computer scientist should know about floating-point arithmetic. *ACM Computing Surveys*, vol. 23, no. 1, pp 5–48.
14. Hauser, J.R. 1996. Handling floating-point exceptions in numeric programs. *ACM Transactions on Programming Languages and Systems*, 18(2):139–174.
15. Hough, D. 1981. Applications of the proposed IEEE 754 standard for floating-point arithmetic. IEEE *Computer*, March, pp 70–74.
16. IEEE. 1981. A proposed standard for floating-point arithmetic. IEEE *Computer*, March, pp 51–62.
17. IEEE. 1985. *IEEE Standard For Binary Floating-Point Arithmetic*. IEEE Press, New York.

18. International Standards Organization. *Information Technology — Language Independent Arithmetic — Part I: Integer and floating-point arithmetic.*

19. Kahan, W. 1995. Lecture notes on the status of IEEE standard 754 for binary floating-point arithmetic. Department of Electrical Engineering and Computer Science, University of California, Berkley, USA.

20. Keir, R.A. 1975. Programmer-controlled roundoff and the selection of a stable roundoff rule. In: *Proceedings, 3rd IEEE Symposium on Computer Arithmetic*, pp 73–75.

21. Knuth, D.E. 1967. *The Art of Computer Programming*, vol. II. Addison-Wesley, Reading, Massachusetts. (Chapter 4)

22. Kuck, D.J., D.S. Parker, and A.H. Sameh. 1975. ROM-rounding: a new rounding scheme. In: *Proceedings, 3rd IEEE Symposium on Computer Arithmetic*, pp 67–72.

23. Kuck, D.J., D.S. Parker, and A.H. Sameh. 1977. Analysis of rounding methods in floating-point arithmetic. *IEEE Transactions on Computers*, vol. C-26, no. 7, pp 643–650.

24. Liddiard, L.A. 1978. Required scientific floating-point arithmetic. In: *Proceedings, 4th IEEE Symposium on Computer Arithmetic*, pp 56–62.

25. McKeenan, W.M. 1967. Representation error for real numbers in binary computer arithmetic. *IEEE Transactions on Computers*, vol. EC-16, pp 682–683.

26. Marasa, J.D. and D.W. Matula. 1973. A simulative study of correlated error propagation. *IEEE Transactions on Computers*, vol. C-22, no. 6, pp 587–597.

27. Omondi, A.R. 1994. *Computer Arithmetic Systems: Algorithms, Architecture, and Implementation.* Prentice-Hall International, UK.

28. Omondi, A.R. 1998. Floating-point number systems and the IEEE-754 standard, II: implementation. (To be submitted for publication.)

29. Sterbenz, P.H. 1974. *Floating-Point Computation.* Prentice-Hall, Englewood Cliffs, New Jersey.

30. Sweeney, D.W. 1965. An analysis of floating-point addition. *IBM Systems Journal*, vol. 4, pp 31–42.

31. Thornton, J.E. 1970. *Design of a Computer: The Control Data 6600.* Scott, Foresman and Company, Glenview, Illinois. (Chapter 7)

32. Tsao, N. 1974. On the distribution of significant digits and roundoff errors. *Communications of the ACM*, vol. 17, no. 5, pp 269–271.

33. Wilkinson, J.H. 1963. *Rounding Errors in Algebraic Processes.* Prentice-Hall, Englewood Cliffs, New Jersey.

34. Yohe, J.M. 1973. Roundings in floating-point arithmetic. *IEEE Transactions on Computers*, vol. C-22, no. 6, pp 577–586.

35. Yokoo, H. 1992. Overflow/underflow-free floating-point number representations with self-delimiting variable-length exponent field. *IEEE Transactions on Computers*, vol. 41, no. 8, pp 1033–1039.

36. Vignes, J. 1996. A stochastic approach to the analysis of round-off-error propagation – a survey of the CESTAC method. In: *Proceedings of the 2nd Real Numbers and Computer Conference*, pp 233–251.

Conceptual Interactive Learning Tools Based on Computer Simulators

Damjan Zazula, Bogdan Viher, Dean Korošec, Enis Avdičauševič, Mitja Lenič, Božidar Potočnik
University of Maribor, Faculty of Electrical Engineering and Computer Science
Smetanova 17, 2000 Maribor, Slovenia
Phone: + 386 62 220 7480, Fax: + 386 62 211 178
E-mail: zazula@uni-mb.si

Keywords: modelling and simulation, conceptual learning, client-server, Java applets, electromyography

Edited by: Rudi Murn

Received: December 3, 1998 **Revised:** July 19, 1999 **Accepted:** August 18, 1999

An existent computer-based simulator may serve double purpose: simulating the basic phenomenon that was conceptually modelled by the simulator, and offering its inherent teaching characteristics at the same time. In this paper, we describe shortly the basic concepts of a simulator of electromyographic signals (EMGs), which was, first, written for a single-user environment in C++ and later ported to Java and interactive network environment. At this stage it became natural that the Java application was upgraded by the features of a teaching and learning tool. Such an integration was possible in a client-server approach providing the network users with a compact facility for generating the EMG components and signals, and learning about the electro-physiological phenomena in parallel.

1 Introduction

The world-wide accessibility of the Internet applications has recently opened a new prospective of the computer-supported processing and dissemination of information. Interaction has become a basic rule moving frontiers out of the local computer, out of the local community. At the same time, the philosophy of security and protection drew the major attention. New concepts have been developed, one of the most outstanding being the interactive applications in Java [6], and the corresponding tools and approaches, for example the client-server connections, three-tier architecture, and applets.

The simulator of electromyographic signals (EMGs) was under the name of *EmgSim* [3] developed in the System Software Laboratory at the Faculty of Electrical Engineering and Computer Science in Maribor. Initially, it was written in C++ for the Windows environment. Later on, we ported it into Java as an Internet application, built in the client-server mechanism, and even added some features being characteristic for the computer-assisted educational tools. Thus, we developed an integrated environment that may contribute at different levels:

- it generates artificial EMG signals whose building blocks are known in details; therefore, it plays a role of a reference to the EMG decomposition techniques [4, 5, 1],
- it enables experimenting with the effects of different electro-physiological parameters on the needle- and surface-recorded EMGs [7], and
- it also runs as a teaching/learning tool of the electro-physiology of muscles.

The approach will be presented briefly in this paper. Section 2 describes conceptual model of the *EmgSim* simulator, Section 3 speaks about the transformation into an Internet application, whilst the modifications introducing the educational elements are revealed in Section 4.

2 Simulations of electromyographic signals

Contractions of muscles are accompanied by electrical activities that may be measured as an EMG, invasively by needle electrodes or non-invasively by surface electrodes. Physiologically, muscles consist of the so called fibres. Each fibre is innervated by a motoneuron transmitting the triggering electric pulses that cause the contraction of a fibre. In fact, one nerve innervates several fibres that are, consequently, contracted at the same time and acting as a primary unit of force – called a motor unit (MU). Several MUs then build up the whole muscle.

Looking at this structure from the electrical point of view, the activation potentials flow down from the motoneuron. It triggers the corresponding single fibres, which affects them with a charge spreading from the innervation zone towards the tendons. This travelling charge is measured as a single-fibre potential (SFP). All the SFPs of one MU sum up into a motor-unit potential (MUP). Several MUPs are finally superimposed and, thus, form the observed EMG (Figure 1)[7].

Our simulator conceptually follows the electro-physiology. Therefore, it begins first with building up a muscle. Several parameters may be specified in this stage [3], like the distribution of single fibres in the MUs, the

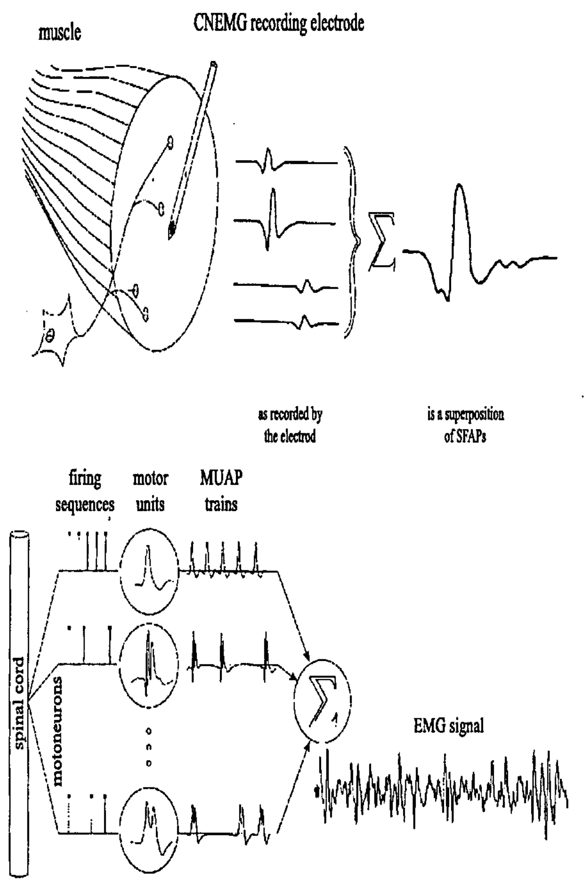

muscle
CNEMG recording electrode
as recorded by the electrod
is a superposition of SFAPs
firing sequences
motor units
MUAP trains
spinal cord
motoneurons
EMG signal

Figure 1: The physiology of a muscle and sources of electrical activity – top, a model of the muscle – bottom.

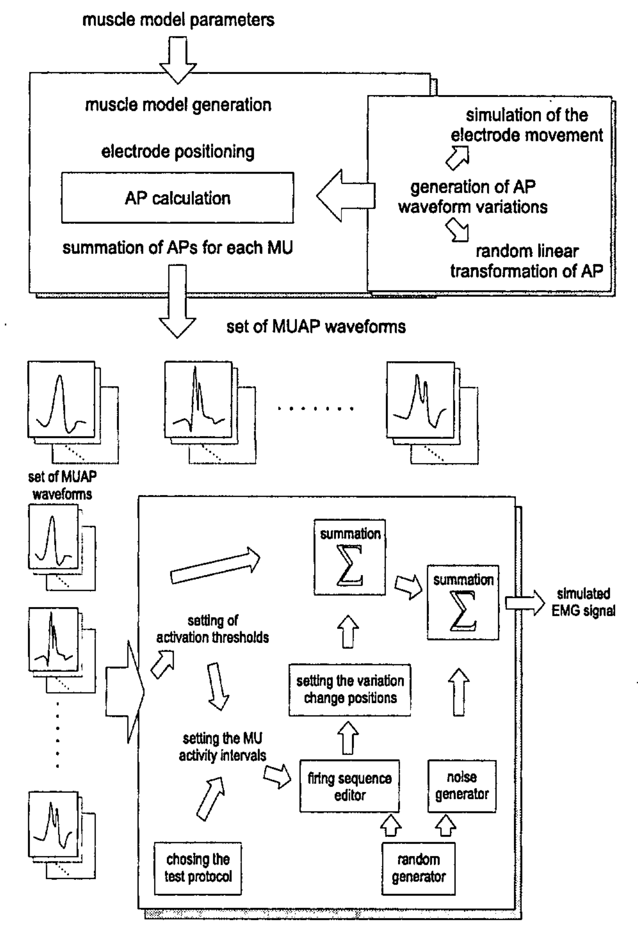

muscle model parameters
muscle model generation
electrode positioning
AP calculation
summation of APs for each MU
simulation of the electrode movement
generation of AP waveform variations
random linear transformation of AP
set of MUAP waveforms
set of MUAP waveforms
summation
summation
simulated EMG signal
setting of activation thresholds
setting the variation change positions
setting the MU activity intervals
firing sequence editor
noise generator
chosing the test protocol
random generator

Figure 2: Principle of operation of the EMG simulator.

distribution of MUs in the muscle, the features of the fibres (their diameters), the triggering instants, etc. The principle of operation is depicted in Figure 2.

3 The EMG simulator as an application in Java

We have stated multiple advantages of today's interactive network applications. The most important in our case are flexibility and accessibility supported by intrinsic levels of control, security and protection. This is exactly what was needed for our EMG simulator to become widely available application on the Internet. Actually, the client-server link seems ideal for this purpose. On the server site, the modelling and simulation part runs with all the calculation routines, data bases, control, security and protection. Clients enter the server from their machines on a platform-independent principle (see Figure 3). If they have access rights, they may send in their requests, i.e., they may run the EMG simulator at different stages. Only the user-interface code is transferred to their site, where it is interpreted and supports the windows and menu environment.

However, as all the computation is done within the server, there is a lot of data that must be transmitted to the users in certain stages of simulation. This could degrade the simulator performance and its responsiveness, that's why a special data communication protocol was developed between the server and the clients.

The characteristics of our Java realisation, as well as the client-server communication, will be described in the following paragraphs.

3.1 Realisation of the server and client

The client software that runs on a user's machine in Java would need more time to run the simulation. Therefore, we divided the simulator in two parts: the client written in Java for user interface, and the server written in C++ (native code) for the calculation of the simulation. The client performs simple calculations and provides user interface. The server performs all the time critical calculations.

The client-server architecture has been used to speed up the simulation. The basic problem is communication over the network. If all the clients were connected to the same instance of the server, an internal server mechanism would have to keep trace and status of every action of individual clients. This would make the server code rather complicated and vulnerable. Therefore, it was worthwhile to sacrifice a part of the server's performance by starting a new

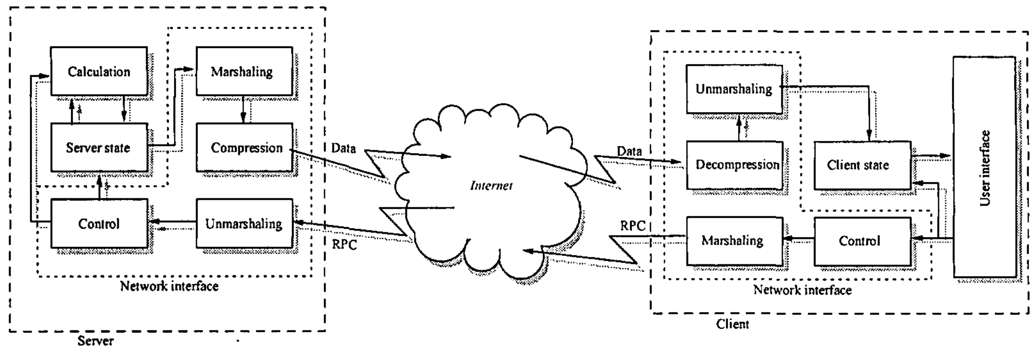


Figure 3: The simulator structure in Java: a client-server connection.

server instance for every new client, and even avoiding the solution with threads for the same reason.

The simplest solution for the communication would be remote procedure calls (RPC), where there would be no need to track the status of simulation, i. e., the stage of its execution. The client would inherently employ the remote functions as demanded in the successive steps by the user. However, Java programming does not support RPCs, it is based on remote method invocation (RMI), which lacks tools for standard serialisation of data and actions.

As mentioned, our solution to the server was starting new instances for every new client. The server incorporates special communication routines – deamons that use standard inputs (STDI) and outputs (STDO). The STDO transfer of data is activated by detection of the ENTER code. One inconvenient behaviour of the deamon is that, after the transfer is finished, terminates the connection. The action is natural in the multi-client environment, however, in our solution it is disturbing. A client and its instance of the server should stay connected and on-line all the time. To circumvent the problem of the communication channel termination, all the server messages are sent out by flushing.

On the other hand , clients call the server functions by simple strings of the following form:

<function name> <arguments>.

As the clients are Java applets, they run under multithreading principle. The server on the other side, which is a C program, is not able to synchronize with the order of precedence of incoming messages. Therefore, the clients communication is synchronized with the server using an interval monitor structure which prevents sending a message before the previous one has been completed.

To reduce the amount of network communication, the client and the server must have a synchronized copy of the state of their common actions. Another problem is data format. For example, Java uses IEEE 745 for representing floating point arithmetics. The server could also run on a Motorola, MIPS, SPARC or Intel-based machine that use different floating-point representations. Therefore, the data format for the network should be in a machine independent format. That is done by marshaling/unmarshaling of the data sent or received over the network (see Figure 3).

Hence, the data is serialised and transfered in either direction in strings containing also all the necessary information about the original data structures (tables, vectors, etc.) and types (integers, floating point values, etc.).

The amount of data sent from a client to the server is rather low. On the other hand, in the stage generating the EMG singal the server have to send out about 3 M bytes of data. It is, therefore, of crucial importance that the transfer is coded. We implemented run-time length code compression based on nibbles. The compression rate achieved is 38%, on overage.

Figure 4 depicts class-hierarchy diagram considering the client-server interconnection. Solid lines in the figure show the hierarchical dependence, whereas dotted lines indicate the information flow. Only the part depicting the client can acctually be shown in the form of class hierarchy. The server has been coded in C++, so the class hierarchy is not applicable in the same way. Nevertheless, the user interface in Figure 4 consists of two separated branches: one deals with user interaction, the other with data visualisation. The MUDialog box stands for the parameter input on the motor units, and the SPShapeDialog box accepts parameters on the action potentials. On the other hand, the EMGPanel box deals with the visualisation of the muscle structure, while the MUAPsPanel takes care of displaying the motor-unit action potentials, i.e., the EMG signals.

4 Upgrading to educational tool

Once having a conceptual model in the form of a simulator, it is just a step ahead to upgrade it for educational purposes. Although a reversed order may seem more normal, i.e., to build a teaching tool and then to include the simulation sessions as explanations, we haven't found any drawbacks in our approach. Moreover, it offers a two-level construction that, on the first level, acts as an extended help for the users of our simulator, on the next level, however, it enables self-evaluation. The latter is done in a typical learning cycle: a piece of learning material is provided, with explanations, demonstrations, graphics, diagrams, etc., then questions about the topic are asked. The answers are scored automatically and the results direct the user either over to the next topic or suggest repetition of some previous chapters (potentially, even some additional pages for basic comprehension may be inserted).

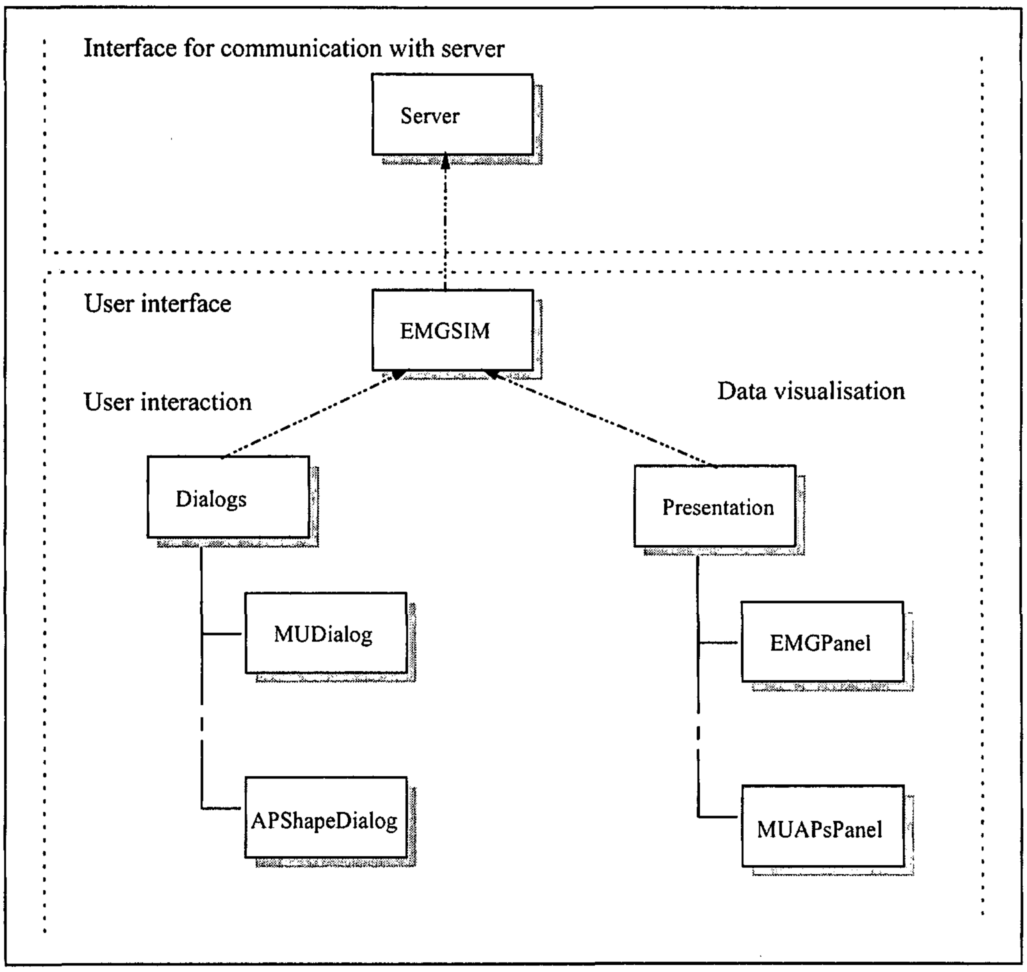


Figure 4: Class hierarchy: the client-server connection.

The advantage of having chosen the simulator as a driving engine for the learning subroutines certainly is reactivation of the same learning modules at any stage of the simulation run, as depicted in Figure 5.

In order to extend the Internet EMG simulator as in Figure 3, special program exits were inserted into the code of Java realisation. Their role can be compared to masked procedure calls: the application runs with all of its functions even if program exits are not activated. When activated, they trigger certain parallel actions which are understood as auxiliary to the main operation of the application.

The most important and instructive notions and steps in the simulation are elaborated additionally in separate pages of explanations an teaching material. The links are realised in terms of program exists. Each program exit has a corresponging parameter containing the address of an HTML document which will be displayed upon a call initiated by the user pressing a button to enter the teaching/learning procedure.

There is another parameter that has special meaning. With this parameter we can switch between two types of the program exits. The first type is browser independent and it is implemented with an applet context. As we know, Java 1.1 is still not fully supported by browsers. Therefore we decided to implement additional program exits bound to Netscape's Internet browser. In this way, we can use appletviewer which supports Java 1.1 to run our applet, and the Netscape's browser to view educational documents. However, similar solution would be possible with Internet Explorer as well.

Every such a program exit calls the indicated HTML document that may contain any necessary educational/explanation contents, supported by all the available relevant information, either locally or over the Internet connections. This may include texts, voice, static images, tables, graphs, or movies.

Afterwards, a self-evaluation procedure is entered by a special button. It consists of three stages:

- questions on the topics presented,
- answers typed in the pre-prepared forms, and
- an automatic evaluation of the answers.

4.1 Concept of the question-answer engine

The first two stages are more or less straightforward. The questions are sequentially numbered and insertet at the end of the HTML pages that are activated through the simulator

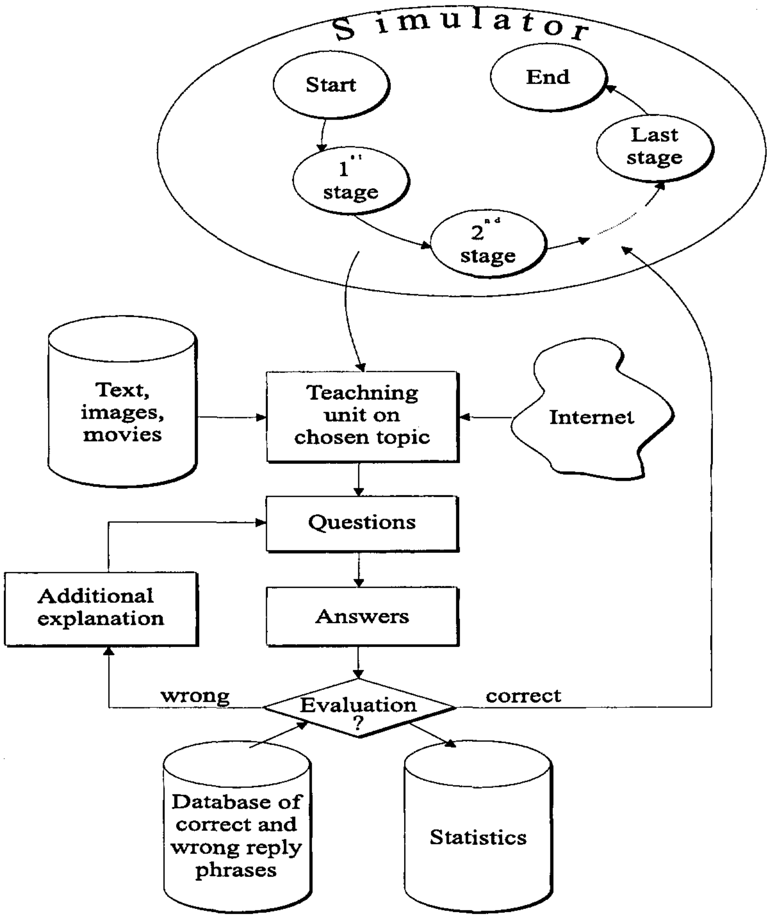


Figure 5: Block diagram of the simulator upgrading to a teaching/learning tool.

program exits. Dialog windows followning the questions receive the typed in answers. The entered answer links to an evaluation CGI (Common Gateway Interface) script directly from HTML. Such scripts enable execution of applications being called from the HTML documents. For the time being, our question-answer engine classifies answers as false or correct only. This is done with the help of a database containing sets of the words and phrases of possible correct and false answers in the following format:

<sequential number of the question>:< elemnts of possibly correct answers>:< elements of possibly wrong answers>.

The computer-assisted evaluation of the answers is, however, not an easy task. If we wanted to have it universal and complete, we would need a thorough base of grammatical, syntactical and semantic rules for every teaching/learning language. At the present stage of development, we have simplified these demands considerably.

The answers are parsed to the individual words. The words and word groups are, afterwards, compared to two lists of phrases. These lists are linked with the corresponding question. One list contains typical correct words and phrases, the other typical wrong words and phrases. According to the score achieved in both lists, every answer is treated either correct or wrong.

For each question, the user/teacher has to prepare and enter a target path (the document which contains the question), a question, a list of correct answers, a list of wrong answers, and a path to the next question in chain. This is realised by another CGI script. The target path is an absolute path in the file system, the path of the next document in chain is relative to the HTTP server. A blank field for the next question terminates the chain. After having been submitted, a new record is written to the database and the target document is generated. Each question has unique ID (a sequential number) which links it to corresponding record in the database. The generated document contains the question and a field for answer. Submitted answer is evaluated and a new document is generated, containing the evaluation score and a link to the next question in chain, although the entered answer may have been incorrect.

5 Conclusion

The described solution could mean a universal approach to upgradings of network versions of simulators. The necessary change in the simulator code is minimum, actually it means only inserting a simple routine to load HTML documents for additional explanation and activating the question-answer engine. The latter runs quite independent of the application and, therefore, can be developed and treated stand alone once for all the applications using it.

Acknowledgement

This work was supported by an HP Labs donation of workstations and by a grant of the Slovenian Ministry of Science and Technology, Contact No. L2-8834, to the Centre for Interactive Network Applications at the Faculty of Electrical Engineering and Computer Science in Maribor.

References

[1] Korošec D., Martinez C., Zazula D., "Parametric modelling of EMG signals," *IEEE EMBS*, Amsterdam 1996, 2 pp.

[2] Roeleveld K., *Surface motor unit potentials; the building stones of surface electromyography*, PhD Thesis, Katholieke Universiteit Nijmegen, The Netherlands, 1994.

[3] Viher B., Korošec D., Zazula D., "Computer simulator of electromyographic signal," *CBMS '96*, Ann Arbor 1996, pp. 47-52.

[4] Zazula D., Korže D., Šoštarič A., Korošec D., "Study of methods for decomposition of superimposed signals with application to electromyograms," in: Pedotti et al. (Eds.), *Neuroprosthetics*, Springer Berlin 1996, pp. 377-389.

[5] Zazula D., Korošec D., Šoštarič A., "Parametric decomposition of the SEMG," *Proceedings of the SENIAM Workshop*, Nijmegen, The Netherlands 1998.

[6] Horstmann C. S., Cornell G., *Core Java Volume II - Advanced Features*, Sun Microsystems Press, USA, 1998.

[7] Stalberg E., Trontelj J. V., *Single Fiber Electromyography*, Raven Press, New York, 1979.

Intelligent Agents On The Internet And Web: Applications And Prospects

San Murugesan
WebISM (Web-based Information Systems and Methodologies) Research Group
Department of Computing and Information Systems
Faculty of Informatics, Science and Technology
University of Western Sydney Macarthur
Campbelltown NSW 2560, Australia
eMail: s.murugesan@uws.edu.au

Keywords: inteligent agents, Internet

Edited by: Matjaz Gams

Received: September 12, 1998 **Revised:** May 17, 1999 **Accepted:** June 5, 1999

Intelligent agents have significant potential for a wide range of Internet and Web-based applications. They offer a new way of designing and implementing intelligent/software systems. An intelligent agent (IA) is a self-contained, autonomous software module that could perform certain tasks on behalf of its users. It could also interact with other intelligent agents and/or human in performing its task(s). There is growing interest in using intelligent software agent for a variety of tasks in diverse range of applications: personal assistants, intelligent user interfaces, managing electronic mail, navigating and retrieving information from the Internet and databases, scheduling meetings and manufacturing operations, electronic business, online shopping, negotiating for resources, decision making, design and telecommunications. This paper gives a brief introduction to intelligent agents, outlines applications of intelligent agents on the Internet and Web, and highlights their prospects.

1 Introduction

In the last few years, intelligent agent has emerged as an important and widely used artificial intelligence technology. There have been rapid developments in the area of intelligent agents and the level of activity and interest in this area is growing. The agent- based approach offers a new suite of powerful techniques, and metaphors to analyse, design and implement intelligent systems of varying complexity and sophistication. An intelligent agent (IA) is a self-contained, autonomous software module that could perform certain tasks on behalf of its users. It could also interact with other intelligent agents and/or human in performing its task(s). It is claimed as an important and exciting new paradigm in software to emerge in the 1990s and is currently a very popular area of investigation and development.

The advent and widespread use of the Internet, Intranets and extranets and Web have opened vast opportunities for intelligent agents and, hence, industry, academia and media are now showing great interest in development and applications of intelligent agents. Intelligent agents are being used in an increasingly wide variety of applications: personal assistants, intelligent user interfaces, managing electronic mail, navigating and retrieving information from the Internet and databases, scheduling meetings and manufacturing operations, electronic business, online shopping, negotiating for resources, decision making, design and telecommunications.

This paper provides an overview of intelligent agents on the Internet and Web. With a brief introduction to intelligent agents, it outlines a range of applications of intelligent agents on the Internet and Web environments and highlights their prospects. It also provides references to more detailed treatment on specific applications of agents.

2 Intelligent Agents

As Nwana (1996) notes software (intelligent) agent is now used as an umbrella term, meta-term or a class to represent a range of software with different characteristics and abilities. And hence, there are many definitions of an agent. They include:

- "Autonomous agents are computational systems that inhabit some complex dynamic environment, sense and act autonomously in this environment, and by doing so realise a set of goals or tasks for which they are designed" [Maes 1995, page 108].

- "Intelligent agents are software entities that carry out some set of operations on behalf of a user or another program with some degree of independence or autonomy, and in so doing, employ some knowledge or representation of the user's goals or desires. It can be described in terms of three dimensions of agency (degree of autonomy and authority), intelligence (degree of reasoning, and learned behaviour) and mobility (degree to which agents themselves travels through a network)" IBM - http://activist.gpl.ibm.com: 81/WhitePaper/ptc2.htm

- An agent is as an encapsulated problem solving entity, which exhibits the following properties [Wooldridge and Jennings 1995]. Autonomy: agents perform majority of their tasks without the direct intervention of humans or other agents, and they have control over their own actions and their own internal state. Social ability: agents interact, when they deem appropriate, with other artificial agents and humans in order to accomplish their tasks and to help others. This requires that agents have, as a minimum, a means by which they can communicate their requirements to others and an internal mechanism for deciding what and when social interactions are appropriate, generating requests and judging incoming requests. Proactiveness: agents take the initiative to do certain tasks where appropriate. Responsiveness: agents perceive their environment and respond in a timely fashion to changes occurring in the environment.

2.1 Attributes of an Agent

The primary behavioural attributes of an agent are autonomy, cooperation and learning [Nwana 1996]. An agent possesses one or more of these attributes.

Autonomy: It "refers to the principle that agents can operate on their own without the need for human guidance. Agents have individual internal states and goals, and they act in such a manner as to meet its goals on behalf of its user. A key element of their autonomy is their proactiveness, i.e. their ability to 'take the initiative' rather than acting simply in response to their environment [Wooldridge and Jennings 1995]."

Cooperation: Cooperation with other agents is paramount: it is the reason for having multiple agents in the first place in contrast to having just one. In order to cooperate, agents need to possess social ability, i.e. "the ability to interact with other agents and possibly with humans via some communication language [Wooldridge and Jennings, 1995]".

Learning - For agent systems to be truly 'smart', they would have to learn as they react and/or interact with their external environment and interface with users. A key attribute of an intelligent system is its ability to learn to understand the user's preferences and behaviour, to cope with new situations it may face and to improve its performance over time.

For further discussion on agent properties see Franklin and Graesser (1996).

2.2 Classification of Intelligent Agents

Agents could be classified in a variety of ways: by their characteristics, by the underlying technology, by the function they perform (what they do), or by a combination of them. Agent classification is, however, not unique, as some of the features of agents used for classification are not mutually exclusive.

Static and Mobile Agents Based on mobility of the agents, agents could be classified as static (stationary) agents, or mobile agents. A stationary agent resides in one computer system and executes its functions/tasks and interacts with the outside world by exchanging messages in a manner similar to the conventional software. A mobile agent can physically move through the network to other computer systems, execute the designated functions and process required information at a different computer and return to the home computer with acquired/processed information. Rather than transmitting messages to receive information required, they visit other computer systems on a network to gather information (see Morreale 1998, Ohsuga 1997).

Interface Agents

Interface agents which mainly assists the users:
* reduce the interface complexity and enhance the ease of use in the presence of greater functionality.
* provide "smart" user interfaces that can detect when a user is having difficulty and "coach" the user around the problem.
* observe and learn the user's preferences and habits, and try to perform automatically what the user routinely does.

An interface agent can observe the actions taken by the user, learn new capabilities dynamically, suggest a course of action to the user, provide context-sensitive help, adapt the interface automatically to the user's personalised requirements, or automate tasks which would otherwise require tedious sequences of manual operations. To do these, it may use learning and reasoning techniques developed in other areas of artificial intelligence and may adopt the principles of good human-computer interaction.

Collaborative Agents and Multi-Agent System (MAS)

A collaborative agent interacts and cooperates with other agents (and/or humans) to perform the tasks on behalf of a user. A multi-agent system (MAS) consists of two or more semi- autonomous agents that interact, collaborate and work together to perform a set of tasks or goals. These agents have, among others, problem solving, communication, and coordination abilities. Collaborative agents may be a homogeneous or heterogeneous group of agents and they may have similar or differing goals, and knowledge representation facilities. They heavily draw upon on the principles of artificial intelligence, sociology, organisational theory, animal behaviour, economics, and distributed systems. Agents in a MAS have ability to solve problems, and to communicate, coordinate and negotiate with other agents. For further information on MAS see Sycara (1998).

Information/Internet Agents

In the networked world, an enormous amount of information is available from a wide variety of sources. Information/Internet agents can filter inexhaustible amount of information available on the Internet, passing on to the user only those information which the user is interested in.

They can seek for and retrieve information required and process the information obtained. They aid in the tedious task of retrieving the relevant information from networked resources.

Learning Agents

A learning agent uses machine learning techniques to 'learn' as they react and/or interact with external environments and users. It tries to improve its performance with time and to learn its users' preferences and interests to better assist them. It could learn: 1) by observing and imitating the user, 2) from (corrective) feedback from the users, 3) by following explicit instructions from the user, and/or 4) by asking other agents for advice.

Emotional Agents

These agents can depict emotions and convey their internal state to the user (Bates, 1994). Emotions can help animate faceless software agents to cartoon-like characters, making them more live like -"serving with smile." They provide an illusion of life in convincing ways to make audience believe the agent is real. They are also known as believable agents, synthetic characters (Elliot and Brzezinski, 1998) and anthropoids (Huhns and Singh, 1998). Examples of emotional agents include: Creatures (www.cyberlife.co.uk) (Grand and Cliff, 1998), real-time, interactive, self-animating creatures called Woggles (Oz Project at Carnegie-Melon University, USA)

The Persona project at Microsoft is developing the technologies required to produce conversational assistants - life like animated characters that interact with user in a natural spoken dialogue.

Socionics is a new term that combines the disciplines of sociology and informatics. One of the aims of research work in this area is to develop intelligent computer technologies by using /adopting paradigms from sociology. It is envisaged that advances in socionics might lead to more human like agent environment and formation of better and more capable multi-agent societies. For more information on a new interdisciplinary work has that begun recently in this area in Germany see: http://www.tu-harburg.de/tbg/SPP/Start_Spp.html

For a detailed overview on intelligent agents see Bradshaw 1997, Chorafas 1997, Cockayne and Zyda 1997, Comm ACM 1994, Harrison and Caglayan 1997, Huhns and Singh 1998c, Nwana 1996, Nwana and Azarmi 1997, Green, and Jennings and Wooldridge 1998.

3 Applications of Intelligent Agents on the Internet And Web

There is enormous interest in deploying intelligent agents on the Internet, intranets and extranets and Web for a variety of applications ranging from personal to entrepreneurial: intelligent personal assistance (Huhns and Singh 1998b, Millman 1998), mail management, information retrieval and presentation, planning and scheduling (Sen 1997), personal service, electronic commerce and on-line shopping, collaborative decision making, enterprise management (AI Magazine 1997, IEEE Internet Computing 1998).

3.1 Agent-Based Information Retrieval

Information Explosion: In the last few years, there has been a major revolution in information generation and dissemination, resulting in information explosion. Inexhaustible amount of information is now available from an increasing number of heterogeneous, distributed information sources. Information can now be made available on the Internet very easily and at a very low cost with minimal effort. Further, most of the information is dynamic - it may updated or modified at anytime. In addition, many persons encounter a large volume of email. As observed by John Naisbit, "we are drowning in information, but starved of knowledge."

Growing problems of using the Internet and Web to retrieve information of interest to a user include:
* Inexhaustible pool of information
- Difficulty of retrieving relevant information
- More time spent on searching for information.
- Information overkill - infostress
* Dynamic nature of the information on the Internet
- Sudden appearance and disappearance of information.
* Distributed, heterogeneous nature of information and information services.

To manage these problems, we need tools to search, retrieve, filter and present relevant information, reactively and proactively. Intelligent software agents offer promising solutions to the current (threat of) information overkill on the Internet and the problem of information retrieval (IR). They have potential to mitigate the complexity of information retrieval and management by providing locus of intelligence.

Agents could provide intelligent IR interfaces, or perform mediated searching and brokering, clustering and categorisation, summarisation and presentation. Agent based approaches make IR systems more scalable, flexible, extensible and interoperable.

Interfaces: An intelligent agent-based interface to an information retrieval system provides an easy-to-use, personalised, adaptive interface. It can act as an agent for a user, and learn about its users and their preferences (Krulwich and Burkey 1997).

Mediated Searches and Information Brokers: Currently many search tools/engines are available for searching information available on the Internet and Web. However, each has its own query structure, interface and nomenclature. Gathering relevant information from multiple, distributed, heterogeneous information sources on the Internet and Web poses difficult problems, which include:
* Finding appropriate sources of information relevant to the query and suitable search tool(s).
* Formulating queries in the 'terms' that the information sources can understand

* Interpreting retrieved information
* Collating the information received from several sources into a coherent, concise information.

For details on agents for information gathering agents see Ambite and Knoblock (1997a and 1997 b).

Information brokering and mediated search agents can provide a uniform interface for general queries and send requests to the most appropriate resource(s) and search tool(s) – Examples: Ahoy and Metacrawler.

Information Filtering Agents: They find information of specific interest to a user from different information sources [Belkin and Croft 1992, Moukass 1997, Moukas and Maes 1998]. Unlike the Web search engines that provide URLs of information sources, information filtering agents:
* gather recent articles about a chosen topic from various sources (eg. Web pages, news feeds, etc.),
* filter this information based on the personal preferences of the user, and
* present the filtered information to the user.

Examples of information filtering agents:
* NewsHound: the personal news service by San Jose Mercury News
* ZDNet Personal View: the personalized computing news by Ziff-Davis
* NewsPage Direct: the personal newspaper

Categorization and Presentation: It would be better if agents analyse the information, and then structure it in a manner more useful and convenient to a user. Further, it could also derive 'metadata from information'.

3.2 Recommender Systems

To cope with an insurmountably large array of choices people have in making a selection, such as buying a CD, a book, or an airline ticket, people turn into some kind of automatic recommendation services, which over time adapts to its user's interests and preferences. There is now a growing potential for on-line automatic recommender services.

Recommendation services falls into three classes: 1) content-based - those that recommend items based on some analysis of their content, 2) collaborative - those that recommend items based on recommendations of other users, exploiting similarities between the users' interests, and 3) a hybrid approach that make of use of both the content-based and collaborative approaches, thereby taking advantages of shared interest among users without losing the benefits of representation provided by content analysis.

3.3 Agents in Electronic Commerce

One of the objectives of agent-mediated electronic commerce is to help users in all aspects of online shopping. Software agents on the Internet could help users in a number of ways [Doorenbos 97, Andreoli 1997, Gutman, et al., 1998]:
* Helping the user decide which products to buy (e.g.: by listing what products of a certain type are available).
* Finding specifications and reviews of them.
* Making recommendations
* Comparison shopping to find the best price for the desired product.
* Watching for and intimating user about special offers and discounts.

Already a few comparison-shopping agents are in use. Examples include BargainFinder and ShopBot (www.cs.washington.edu/research/shopbot http://bf.cstar.ac.com/). They scour the Internet on the user's behalf, visiting all the on-line 'virtual' shops and give the lowest price, an ordered list of prices and the vendors, and other related information on the products that the user is interested.

BargainFinder is a virtual shopping agent for the Web developed by Anderson Consulting to query pricing and availability of user specified music CDs. It uses the parallel search architecture used in meta search engines such as MetaCrawler. It submits user's product query in parallel to a group of on-line vendors by filling out the form at each site. It parses the query results after filtering out the header, trailer, and advertisements to find each vendor's price for the required product and prepares a summary to the user combining the filtered results from each vendor.

Comparison shopping agents offer a number of advantages:
* They provide a unified interface for dealing with cyberstores.
* There is no need for the user to navigate to different stores, and to deal with separate user interfaces
* They find the best price and availability through comparison shopping, thus relieving the user to search for a product at each online store.
* Further, they eventually trigger a price war among sellers potentially making prices almost same everywhere.

Comparison shopping agents do, however, have limitations:
* Currently, most agents ignore other factors such as service, quality.
* It is harder for agents to compare products with varied configurations such as computer, travel packages, etc. However, they can work well for standard goods such as books, CDs and airfares.

The roles of agents have expanded beyound tasks such as comparison-shopping - agents on the Internet can interact with each other and take part in procurement activities. The agents may issue request for goods, bid to supply the required goods, evaluate the bids, place an order and, after receipt of goods, make payments. Many multiagent applications in electronic business are emerging (see Guttman et al 1998).

3.4 Agent Services

Service agents provide a specified service to a user based on information made available to them. They include:

* Announcement Agents: Remind users of important occasions (eg. birthday, anniversary, Valentines Day and the like) that are customised for personal needs.
* Book Agents: Track newly released books to notify users books that match each user's reading interests.
* Business Information Monitoring Agents: Monitor the exchange of information on the Internet relating to services, products, industry and companies, and reports findings as a report.
* Classified Agents: Search a database of classifieds daily to find a user-specified item, and notify the user via email. An example of classified agent is AdHound. It searches databases of classifieds in areas specified by the user and it finds a match, it sends an email giving the relevant classified ads.
* Job Agents: Serve as virtual recruiters to find employees who match with the employer job profiles.
* Direct Mail Agents: Enable personalised direct mail advertising that matches the user's stated personal background, activities, lifestyle and liking.
* Financial Service Agents: Deliver email messages containing prices and financial news for a personalised portfolio of securities and mutual funds.
* Food and Wine Agents: Remember each user's previous purchases and tasting notes and make customised presentations of the inventory during the next visit.
* Entertainment Agents: Find communities with similar interests to those of the user, and recommend albums, movies, etc. based on group evaluations.

3.5 Agents for Business Operations

Intelligent agents also find applications in Intranets of organisations and business enterprises. They include:
* Meeting scheduling agents: Design and develop efficient meeting scheduling (Sen 1997).
* Collaborative customisation agents: intranet agents that automate workflow processes in business units.
* Process automation agents: Intranet agents that automate industrial processes.
* Database agents: Intranet agents that provide agent services for users of enterprise databases.
* Resource brooking agents: Agents that perform resource allocation in client/server architectures.

3.6 Entertainment Software Agents

Autonomous software agents have significant potential in areas such as entertainment and in online chat. Interactive entertainment products based on anthropoids or synthetic characters are emerging. These agents can depict emotions and convey their internal state to the user. These cartoon-like characters provide an illusion of life in convincing ways to make audience believe the agent is real.

Some of the examples include: Creatures (www.cyberlife.co.uk) (Grand and Cliff, 1998), real-time, interactive, self-animating creatures called Woggles (Oz project at Carnegie-Melon University, USA).

There is growing interest in incorporating agents in interfaces to make interactions more natural and easier. Examples include: Extempo's Web-situated virtual bar based on Barbara Hayes-Roth's work at Stanford University on virtual bartender (www.extempo.com/webBar/) and virtual personalities (www.vperson.com/). Another example is the agent Jullia who participates on online chat rooms, providing directions to users and taking part in online discussions. Agents have potential to play influencing roles in marketing, education and training (see PPP and Lewis 1998).

4 Further Work

Though in the last few years agent-based systems have advanced and there have many successful agent-based applications, agent technology is still young and many problems and research challenges still remain. Further, as we move to deploy agents on the Internet for critical applications, and as our dependence on agents continue to grow, issues and requirements such as reliability, real-time performance, openness, security and trust become paramount importance. To fully benefit from agent technologies, further developments are needed in the following major areas:
* Information access, filtering, etc.
* Learning
* Multiagent cooperation, coordination and negotiation
* Real-time performance issues and considerations
* Testing, verification and validation
* Trustworthiness, security
* Ethical issues and considerations
* Mobile agents - execution considerations in remote host(s)
* Integration of agent-based systems with other systems

For further information a roadmap on agent research and development see Jennings, Sycara and Wolldridge (1998).

5 Prospects

Intelligent agents on the Internet and Web will grow dramatically in the next few years as new agent standards and architectures and powerful development tools emerge. They would make the Internet and the corporate intranet and extranet environments easier to use and would enable a variety of new applications and services. The following quotes aptly highlight the prospects of intelligent agents.

"Agents are here to stay not least because of their diversity, their wide range of applicability and the broad spectrum of companies investing in them. As we move further and further into the information age, any information-based organisation, which does not invest in agent technology, may be committing commercial hara-kit." - Nawana, 1996

"One of the most interesting entrepreneurial activity for the next several years is likely to be discovering new and unique ways of using the Internet [and Intelligent Agents] for business [and personal] advantage." - Yourdon, 1997

"Agents will be the most important computing paradigm in the next 10 years. By the year 2000, every significant application will have some form of agent enablement." - BIS Strategic Decisions

"In the future, it [agent] is going to be the only way to search the Internet, because no matter how much better the Internet is going to be organised, it can't keep pace with information." - Bob Johnson of Dataquest

As we move further into the information age which is supported by more complex and sophisticated networked applications, there will be a steady growth in agent applications and in the agent market. Realising this many organisations are investing in agent technology and many players - computer manufacturers, software houses and many start-up companies - have entered into this highly competitive field. The stakes are high and the future belongs to those who are prepared. The winners would be those who have vision, are innovative and are prepared to realise those visions.

References

1. AI Magazine, Summer 1997, Special Issue on Intelligent Agents on the Internet.
2. Ambite, J.L., and Knoblock, (1997 b), Agents for Information Gathering, in Bradshaw, J. M (ed.,) Software Agents, Menlo Park, CA: AAAI/MIT Press, 1997
3. Ambite, J.L., and Knoblock, (1997 b), Agents for Information Gathering, IEEE Expert, Sep/Oct 1997, pp 2-4.
4. Andreoli, J., Xpect: A Framework for Electronic Commerce, IEEE Internet Computing, Vol. 1, No. 3, July -August 1997, pp 40 - 49.
5. Baclace, P.E., Competitive Agents for Information Filtering, Communications of the ACM, 35 (12), 1992, page 50.
6. Bates, J., The Role of Emotions in Believable Agents, Comm of the ACM, Vol 37, July 1994, pp 122-125.
7. Belkin, N.J., and Croft, W.B., Information Filtering and Information Retrieval: Two Sides of the Same Coin? Communications of the ACM, 35, Dec 1992, pp 29-38.
8. Bradshaw, J. M (ed.,)., 1997, Software Agents, Menlo Park, CA: AAAI/MIT Press, 1997
9. BT 1996, (British Telecom) Technology Journal, Special issue on Intelligent Software Systems, Vol. 14, No. 4, Oct 1996.
10. Chorafas, D.N., Agent Technology Handbook, McGraw Hill, June 1997.
11. Cockayne, W.R., and M. Zyda, Mobile Agents: Explanations and Examples, Manning Publishers/Prentice-Hall, 1997.
12. Comm. of the ACM 1994, Special Issue on Intelligent Agents, July 1994.
13. Doorenbose, R.B., et al., A Scalable Comparison-Shopping Agent for the World-Wide Web, Proceedings of the Agents'97 Conference, ACM, 1997.
14. Elliot C and J. Brzezinski, Autonomous Agents as Synthetic Characters, AI Magazine, Summer 1998, pp 13-30.
15. Franklin, S. & Graesser, A. (1996), "Is it an Agent, or just a Program - A Taxonomy for Autonomous Agents", Proceedings the Third International Workshop on Agent Theories, Architectures, and Languages, Springer-Verlag. http://www.msci.memphis.edu/ franklin/AgentProg.html
16. Goldberg, D., et al., Using Collaborative Filtering to Weave an Information Tapestry, Communications of the ACM, 35 (12), 1992, pp 61-70.
17. Grand, S., and D. Cliff, Creatures: Entertainment Software Agents with Artificial Life, Agents and Multi-Agent Systems, Vol.1, No.1, 1998, pp 39-57.
18. Green, S., et al., Software Agents: A Review. Available on the World Wide Web at: http://www.cs.tcd.ie/research_group/aig/iag/
19. Gudivada, et al., Information Retrieval on the World Wide Web. IEEE Internet Computing, Vol. 1, September-October 1997, pp 58-68.
20. Guttman, R.H., A.G. Moukas and Pattie Maes, Agent-Mediated Electronic Commerce: A Survey, Knowledge Engineering Review, June 1998. Also available at: http://ecommerce.media.mit.edu/pagers/ker98.pdf
21. Harrison, C.G., and A. Caglayan, Agent Source Book, John Wiley and Sons, June 1997.
22. http://www.alumini.caltech.edu/ croft/research/agent/etiquette
23. Huhns, M.N., and M.P. Singh (1998a), Anthropoid Agents, Internet Computing, J an-Feb. 1998, pp 94-95.
24. Huhns, M.N., and M.P. Singh (1998b), Personal Assistants, Internet Computing, Sep - Oct. 1998, pp 90-92
25. Huhns, M.N., and M.P. Singh (1998c), Readings in Agents, Morgan Kaufmann Publishers Inc., 1998.
26. IEEE Internet Computing, Vol. 1, No. 3, July -August 1997, Special Issue on Internet-Based Agents.
27. Jennings, N.R., and M.J. Wooldridge, Agent Technology, Foundations, Applications and Markets, Springer Computer Science, 1998.
28. Jennings, N.R., Katia Sycara, M. Wolldridge, A Roadmap of Agent Research and Development, Autonomous Agents and Multi-Agent Systems, Vol.1, No.1, 1998, pp7-38.
29. Krulwich, B., and C. Burkey, The InfoFinder Agent: Learning User Interests through Heuristics Phrase Extraction, IEEE Expert, Sep/Oct 1997, pp22-27.
30. Krulwich, B., Industry Report: Automating the Internet - Agents as User Surrogates, IEEE Internet Computing, Vol. 1, No. 3, July -August 1997, pp 34 - 39.
31. Lewis, M., Designing for Human-Agent Interaction, AI Magazine, Summer 1998, pp 67-78.
32. Mayes, P., Intelligent Software, Scientific American, 273, September 1995.
33. Millman, H., Agents at Your Service, Inforworld, 20(7), 16 Feb 1998, pp 77-80.
34. Morreale, P., Agents on the Move, IEEE Spectrum, April 1998, pp 34-41.
35. Moukas, A., Amalthaea: Information Discovery and

Filtering Agents Using Multiagent Evolving Eco Systems, Journal of Applied Intelligence, Vol 11 (5), pp 437-457, 1997.
36. Moukas, A., and P. Maes, Amalthaea, An Evolving Multi_Agent Information Filtering and Discovery Systems for the WWW, Autonomous Agents and Multi-Agent Systems, Vol 1, No 1, 1998, pp 59- 88.
37. Murugesan, S and D. Oleary (Eds.), Proc. of 1999 AAAI Spring Symposium on Intelligent Agents on Cyberspace, Stanford, Ca, USA. http://btwebsh.macarthur.ues.edu.au/san/iac/
38. Murugesan. S., and Christi XXX (eds), Proc. of International Workshop on Intelligent Agents on the Internet and Web, World Congress on Expert Systems, Mexico, 1998. http://btwebsh.macarthur.ues.edu.au/san/iaiw/
39. Nwana, H.S., 1966 Software Agents: An Overview, Knowledge Engineering Review, Vol. 11, No. 3, pp 1-40, Sep 1996. http://www.cs.umbc.edu/agents/introduction/ao/
40. Nwana, H.S., and N. Azarmi, Soft Agents and Soft Computing: Concepts and Applications, Lecture Notes in Artificial Intelligence, Springer-Verlag, 1997.
41. Ohsuga, A., Plangent: An Approach to making Mobile Agents Intelligent, IEEE Internet Computing, Vol. 1, No. 3, July -August 1997, pp 50-57.
42. PPP: Personalised Plan-Based Presenter, German Research Center for Artificial Intelligence (DFKI); http://www.dfki.unisb.de/ jmueller/ppp/persona/index.html
43. Sen, S., Developing an Automated Distributed Meeting Scheduler, IEEE Expert, July/August, 1997, pp 41-45.
44. Sycara, K.P., Multiagent Systems, AI magazine, Summer 1998, p 79.
45. Wooldridge, J., and N. R. Jennings, Intelligent Agents: Theory and Practice, The Knowledge Engineering Review, 1995, 10 (2), pp 115-152.
46. Yourdon, Ed., 1997, Riding on the Internet Tiger, Yourdon Press, 1997.

URLs of Some Useful Web Pages
Multiagent Systems: WWW.cs.wustl.edu/ mas/links.htm
The Agent Society: www.agent.org
Agents List and Archive: http://www.cs.umbs.edu/agentslist/
AgentWeb: http://www.cs.umbc.edu/agents/
The Foundation for Intelligent Physical Agents, FIPA: http://drogo.cselt.stet.it/fipa

THE MINISTRY OF SCIENCE AND TECHNOLOGY OF THE REPUBLIC OF SLOVENIA

Address: Slovenska 50, 1000 Ljubljana, Tel.: +386 61 1311 107, Fax: +386 61 1324 140.
WWW:http://www.mzt.si
Minister: Lojze Marinček, Ph.D.

The Ministry also includes:

The Standards and Metrology Institute of the Republic of Slovenia
Address: Kotnikova 6, 61000 Ljubljana, Tel.: +386 61 1312 322, Fax: +386 61 314 882.

Slovenian Intellectual Property Office
Address: Kotnikova 6, 61000 Ljubljana, Tel.: +386 61 1312 322, Fax: +386 61 318 983.

Office of the Slovenian National Commission for UNESCO
Address: Slovenska 50, 1000 Ljubljana, Tel.: +386 61 1311 107, Fax: +386 61 302 951.

Scientific, Research and Development Potential:

The Ministry of Science and Technology is responsible for the R&D policy in Slovenia, and for controlling the government R&D budget in compliance with the National Research Program and Law on Research Activities in Slovenia. The Ministry finances or co-finance research projects through public bidding, while it directly finance some fixed cost of the national research institutes.

According to the statistics, based on OECD (Frascati) standards, national expenditures on R&D raised from 1,6 % of GDP in 1994 to 1,71 % in 1995. Table 2 shows an income of R&D organisation in million USD.

Objectives of R&D policy in Slovenia:

- maintaining the high level and quality of scientific technological research activities;
- stimulation and support to collaboration between research organisations and business, public, and other sectors;

Total investments in R&D (% of GDP)	1,71
Number of R&D Organisations	297
Total number of employees in R&D	12.416
Number of researchers	6.094
Number of Ph.D.	2.155
Number of M.Sc.	1.527

Table 1: Some R&D indicators for 1995

	Ph.D.			M.Sc.		
	1993	1994	1995	1993	1994	1995
Bus. Ent.	51	93	102	196	327	330
Gov. Inst.	482	574	568	395	471	463
Priv. np Org.	10	14	24	12	25	23
High. Edu.	1022	1307	1461	426	772	711
TOTAL	1565	1988	2155	1029	1595	1527

Table 2: Number of employees with Ph.D. and M.Sc.

- stimulating and supporting of scientific and research disciplines that are relevant to Slovenian national authenticity;
- co-financing and tax exemption to enterprises engaged in technical development and other applied research projects;
- support to human resources development with emphasis on young researchers; involvement in international research and development projects;
- transfer of knowledge, technology and research achievements into all spheres of Slovenian society.

Table source: Slovene Statistical Office.

	Basic Research		Applied Research		Exp. Devel.		Total	
	1994	1995	1994	1995	1994	1995	1994	1995
Business Enterprises	6,6	9,7	48,8	62,4	45,8	49,6	101,3	121,7
Government Institutes	22,4	18,6	13,7	14,3	9.9	6,7	46,1	39,6
Private non-profit Organisations	0,3	0,7	0,9	0,8	0,2	0,2	1,4	1,7
Higher Education	17,4	24,4	13,7	17,4	8,0	5,7	39,1	47,5
TOTAL	46,9	53,4	77,1	94,9	63.9	62,2	187,9	210,5

Table 3: Incomes of R&D organisations by sectors in 1995 (in million USD)

JOŽEF STEFAN INSTITUTE

Jožef Stefan (1835-1893) was one of the most prominent physicists of the 19th century. Born to Slovene parents, he obtained his Ph.D. at Vienna University, where he was later Director of the Physics Institute, Vice-President of the Vienna Academy of Sciences and a member of several scientific institutions in Europe. Stefan explored many areas in hydrodynamics, optics, acoustics, electricity, magnetism and the kinetic theory of gases. Among other things, he originated the law that the total radiation from a black body is proportional to the 4th power of its absolute temperature, known as the Stefan–Boltzmann law.

The Jožef Stefan Institute (JSI) is the leading independent scientific research institution in Slovenia, covering a broad spectrum of fundamental and applied research in the fields of physics, chemistry and biochemistry, electronics and information science, nuclear science technology, energy research and environmental science.

The Jožef Stefan Institute (JSI) is a research organisation for pure and applied research in the natural sciences and technology. Both are closely interconnected in research departments composed of different task teams. Emphasis in basic research is given to the development and education of young scientists, while applied research and development serve for the transfer of advanced knowledge, contributing to the development of the national economy and society in general.

At present the Institute, with a total of about 700 staff, has 500 researchers, about 250 of whom are postgraduates, over 200 of whom have doctorates (Ph.D.), and around 150 of whom have permanent professorships or temporary teaching assignments at the Universities.

In view of its activities and status, the JSI plays the role of a national institute, complementing the role of the universities and bridging the gap between basic science and applications.

Research at the JSI includes the following major fields: physics; chemistry; electronics, informatics and computer sciences; biochemistry; ecology; reactor technology; applied mathematics. Most of the activities are more or less closely connected to information sciences, in particular computer sciences, artificial intelligence, language and speech technologies, computer-aided design, computer architectures, biocybernetics and robotics, computer automation and control, professional electronics, digital communications and networks, and applied mathematics.

The Institute is located in Ljubljana, the capital of the independent state of **Slovenia** (or S♡nia). The capital today is considered a crossroad between East, West and Mediterranean Europe, offering excellent productive capabilities and solid business opportunities, with strong international connections. Ljubljana is connected to important centers such as Prague, Budapest, Vienna, Zagreb, Milan, Rome, Monaco, Nice, Bern and Munich, all within a radius of 600 km.

In the last year on the site of the Jožef Stefan Institute, the Technology park "Ljubljana" has been proposed as part of the national strategy for technological development to foster synergies between research and industry, to promote joint ventures between university bodies, research institutes and innovative industry, to act as an incubator for high-tech initiatives and to accelerate the development cycle of innovative products.

At the present time, part of the Institute is being reorganized into several high-tech units supported by and connected within the Technology park at the Jožef Stefan Institute, established as the beginning of a regional Technology park "Ljubljana". The project is being developed at a particularly historical moment, characterized by the process of state reorganisation, privatisation and private initiative. The national Technology Park will take the form of a shareholding company and will host an independent venture-capital institution.

The promoters and operational entities of the project are the Republic of Slovenia, Ministry of Science and Technology and the Jožef Stefan Institute. The framework of the operation also includes the University of Ljubljana, the National Institute of Chemistry, the Institute for Electronics and Vacuum Technology and the Institute for Materials and Construction Research among others. In addition, the project is supported by the Ministry of Economic Relations and Development, the National Chamber of Economy and the City of Ljubljana.

Jožef Stefan Institute
Jamova 39, 61000 Ljubljana, Slovenia
Tel.:+386 61 1773 900, Fax.:+386 61 219 385
Tlx.:31 296 JOSTIN SI
WWW: http://www.ijs.si
E-mail: matjaz.gams@ijs.si
Contact person for the Park: Iztok Lesjak, M.Sc.
Public relations: Natalija Polenec

INFORMATICA

AN INTERNATIONAL JOURNAL OF COMPUTING AND INFORMATICS

INVITATION, COOPERATION

Submissions and Refereeing

Please submit three copies of the manuscript with good copies of the figures and photographs to one of the editors from the Editorial Board or to the Contact Person. At least two referees outside the author's country will examine it, and they are invited to make as many remarks as possible directly on the manuscript, from typing errors to global philosophical disagreements. The chosen editor will send the author copies with remarks. If the paper is accepted, the editor will also send copies to the Contact Person. The Executive Board will inform the author that the paper has been accepted, in which case it will be published within one year of receipt of e-mails with the text in Informatica LaTeX format and figures in `.eps` format. The original figures can also be sent on separate sheets. Style and examples of papers can be obtained by e-mail from the Contact Person or from FTP or WWW (see the last page of Informatica).

Opinions, news, calls for conferences, calls for papers, etc. should be sent directly to the Contact Person.

QUESTIONNAIRE

☐ Send Informatica free of charge

☐ Yes, we subscribe

Please, complete the order form and send it to Dr. Rudi Murn, Informatica, Institut Jožef Stefan, Jamova 39, 61111 Ljubljana, Slovenia.

Since 1977, Informatica has been a major Slovenian scientific journal of computing and informatics, including telecommunications, automation and other related areas. In its 16th year (more than five years ago) it became truly international, although it still remains connected to Central Europe. The basic aim of Informatica is to impose intellectual values (science, engineering) in a distributed organisation.

Informatica is a journal primarily covering the European computer science and informatics community - scientific and educational as well as technical, commercial and industrial. Its basic aim is to enhance communications between different European structures on the basis of equal rights and international refereeing. It publishes scientific papers accepted by at least two referees outside the author's country. In addition, it contains information about conferences, opinions, critical examinations of existing publications and news. Finally, major practical achievements and innovations in the computer and information industry are presented through commercial publications as well as through independent evaluations.

Editing and refereeing are distributed. Each editor can conduct the refereeing process by appointing two new referees or referees from the Board of Referees or Editorial Board. Referees should not be from the author's country. If new referees are appointed, their names will appear in the Refereeing Board.

Informatica is free of charge for major scientific, educational and governmental institutions. Others should subscribe (see the last page of Informatica).

ORDER FORM – INFORMATICA

Name: ....

Title and Profession (optional): ....

....

Home Address and Telephone (optional): ....

....

Office Address and Telephone (optional): ....

....

E-mail Address (optional): ....

Signature and Date: ....

Informatica WWW:

http://ai.ijs.si/informatica/
http://orca.st.usm.edu/informatica/

Referees:

Witold Abramowicz, David Abramson, Adel Adi, Kenneth Aizawa, Suad Alagić, Alan Aliu, Richard Amoroso, John Anderson, Hans-Jurgen Appelrath, Grzegorz Bartoszewicz, Catriel Beeri, Daniel Beech, Fevzi Belli, Istvan Berkeley, Azer Bestavros, Balaji Bharadwaj, Jacek Blazewicz, Laszlo Boeszoermenyi, Damjan Bojadžijev, Jeff Bone, Ivan Bratko, Jerzy Brzezinski, Marian Bubak, Leslie Burkholder, Frada Burstein, Wojciech Buszkowski, Rajkumar Bvyya, Netiva Caftori, Jason Ceddia, Ryszard Choras, Wojciech Cellary, Wojciech Chybowski, Andrzej Ciepielewski, Vic Ciesielski, David Cliff, Travis Craig, Noel Craske, Matthew Crocker, Tadeusz Czachorski, Milan Češka, Honghua Dai, Andrej Dobnikar, Sait Dogru, Georg Dorfner, Ludoslaw Drelichowski, Matija Drobnič, Maciej Drozdowski, Marek Druzdzel, Jozo Dujmović, Pavol Ďuriš, Hesham El-Rewini, Pierre Flener, Wojciech Fliegner, Vladimir A. Fomichov, Terrence Forgarty, Hans Fraaije, Hugo de Garis, Eugeniusz Gatnar, James Geller, Michael Georgiopolus, Jan Goliński, Janusz Gorski, Georg Gottlob, David Green, Herbert Groiss, Inman Harvey, Elke Hochmueller, Rod Howell, Tomáš Hruška, Alexey Ippa, Ryszard Jakubowski, Piotr Jedrzejowicz, Eric Johnson, Polina Jordanova, Djani Juričič, Sabhash Kak, Li-Shan Kang, Roland Kaschek, Jan Kniat, Stavros Kokkotos, Kevin Korb, Gilad Koren, Henryk Krawczyk, Ben Kroese, Zbyszko Krolikowski, Benjamin Kuipers, Matjaž Kukar, Aarre Laakso, Phil Laplante, Bud Lawson, Ulrike Leopold-Wildburger, Joseph Y-T. Leung, Alexander Linkevich, Raymond Lister, Doug Locke, Peter Lockeman, Matija Lokar, Jason Lowder, Andrzej Małachowski, Bernardo Magnini, Peter Marcer, Andrzej Marciniak, Witold Marciszewski, Vladimir Marik, Jacek Martinek, Tomasz Maruszewski, Florian Matthes, Timothy Menzies, Dieter Merkl, Zbigniew Michalewicz, Roland Mittermeir, Madhav Moganti, Reinhard Moller, Tadeusz Morzy, Daniel Mossé, John Mueller, Hari Narayanan, Rance Necaise, Elzbieta Niedzielska, Marian Niedq'zwiedziński, Jaroslav Nieplocha, Jerzy Nogieć, Stefano Nolfi, Franc Novak, Antoni Nowakowski, Adam Nowicki, Tadeusz Nowicki, Hubert Österle, Wojciech Olejniczak, Jerzy Olszewski, Cherry Owen, Mieczyslaw Owoc, Tadeusz Pankowski, Mitja Peruš, Warren Persons, Stephen Pike, Niki Pissinou, Ullin Place, Gustav Pomberger, James Pomykalski, Gary Preckshot, Dejan Rakovič, Cveta Razdevšek Pučko, Ke Qiu, Michael Quinn, Gerald Quirchmayer, Luc de Raedt, Ewaryst Rafajlowicz, Sita Ramakrishnan, Wolf Rauch, Peter Rechenberg, Felix Redmill, David Robertson, Marko Robnik, Ingrid Russel, A.S.M. Sajeev, Bo Sanden, Vivek Sarin, Iztok Savnik, Walter Schempp, Wolfgang Schreiner, Guenter Schmidt, Heinz Schmidt, Denis Sever, Zhongzhi Shi, William Spears, Hartmut Stadtler, Olivero Stock, Janusz Stokłosa, Przemysław Stpiczyński, Andrej Stritar, Maciej Stroinski, Tomasz Szmuc, Zdzislaw Szyjewski, Jure Šilc, Metod Škarja, Jiři Šlechta, Zahir Tari, Jurij Tasič, Piotr Teczynski, Stephanie Teufel, Ken Tindell, A Min Tjoa, Wieslaw Traczyk, Roman Trobec, Marek Tudruj, Andrej Ule, Amjad Umar, Andrzej Urbanski, Marko Uršič, Tadeusz Usowicz, Elisabeth Valentine, Kanonkluk Vanapipat, Alexander P. Vazhenin, Zygmunt Vetulani, Olivier de Vel, John Weckert, Gerhard Widmer, Stefan Wrobel, Stanislaw Wrycza, Janusz Zalewski, Damir Zazula, Yanchun Zhang, Zonling Zhou, Robert Zorc, Anton P. Železnikar

EDITORIAL BOARDS, PUBLISHING COUNCIL

Informatica is a journal primarily covering the European computer science and informatics community; scientific and educational as well as technical, commercial and industrial. Its basic aim is to enhance communications between different European structures on the basis of equal rights and international refereeing. It publishes scientific papers accepted by at least two referees outside the author's country. In addition, it contains information about conferences, opinions, critical examinations of existing publications and news. Finally, major practical achievements and innovations in the computer and information industry are presented through commercial publications as well as through independent evaluations.

Editing and refereeing are distributed. Each editor from the Editorial Board can conduct the refereeing process by appointing two new referees or referees from the Board of Referees or Editorial Board. Referees should not be from the author's country. If new referees are appointed, their names will appear in the list of referees. Each paper bears the name of the editor who appointed the referees. Each editor can propose new members for the Editorial Board or referees. Editors and referees inactive for a longer period can be automatically replaced. Changes in the Editorial Board are confirmed by the Executive Editors.

The coordination necessary is made through the Executive Editors who examine the reviews, sort the accepted articles and maintain appropriate international distribution. The Executive Board is appointed by the Society Informatika. Informatica is partially supported by the Slovenian Ministry of Science and Technology.

Each author is guaranteed to receive the reviews of his article. When accepted, publication in Informatica is guaranteed in less than one year after the Executive Editors receive the corrected version of the article.

Volume 23 Number 3 September 1999 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

Introduction		303
Supporting All Service Classes in ATM: A Novel Traffic Control Framework	T. Marosits, S. Molnár G. Fodor	305
Service Specific Information Based Resource Allocation for Multimedia Applications	I. Cselényi R. Szabó	317
Fault-Tolerant ATM Switch Using Logical Neighborhood Network	A. Rayhan, F. Elguibaly A. Almulhem	325
Issues on Gigabit Switching using 3-Stage Clos Networks	F.K. Liotopoulos	335
Performance Analysis of Pipelined Multistage Interconnection Networks	R. Venkatesan et al.	347
PNNI and the Optimal Design of High-speed ATM networks	A. Battou, B. Khan S. Mountcastle	359
Performance Evaluation of A Scheduling Algorithm for Multiple Input-Queued ATM Switches	G. Nong, M. Hamdi J.K. Muppala	369
On the Determination of Absolute Network Performance	M.H. Mickle	383
The Brain as a Huygens Machine	B.E.P. Clement et al.	389
Electroencephalographic (EEG) correlates of some activities which may alter consciousness	D. Raković et al.	399
Floating-point arithmetic and the IEEE-754 standard, I: number-system design	A.R. Omondi	413
Conceptual Interactive Learning Tools Based on Computer Simulators	D. Zazula, B. Viher D. Korošec	431
Intelligent Agents On The Internet And Web: Applications And Prospects	S. Murugesan	437
Reports and Announcements		445