Title Quality of Service Study for Single and Semi layer Interconnection Networks with Relaxing Blocking

ELEFTHERIOS STERGIOU
Department of Informatics and Telecommunication Technology
Technological Educational Institute of Epirus
Kostaki, Artas, 47100
GREECE
ster@art.forthnet.gr http://myweb.teleinfom.teiep.gr/stergiou//index.html

Abstract: - Performance evaluation, using both analytical and simulation models, of multistage priority interconnection networks with a single layer or more layers is presented. The configurations of the networks under study apply a conflict drop resolution strategy. Our analytical models are based on a more realistic assumption. A new analysis is given and is verified by simulation results. In single network configuration, it is shown quantitatively that the high priority packets are serviced better than the low priority ones. The proposed architecture's performance is subsequently analysed under the uniform traffic condition, considering various offered loads, buffer-lengths, and network sizes. We demonstrate and quantify the improvements in the performance of single or semi-layer multistage fabrics stemming from the introduction of priorities in terms of throughput and packet delay. These performance measures can be valuable assets for designers of parallel multiprocessor systems and networks in order to minimize the overall deployment costs and deliver efficient systems.

Key-Words: - multistage interconnection networks; analytical model; performance analysis; relaxing blocking; multilayer MINs; throughput; drop blocking

1 Introduction

Multistage interconnection networks (MINs) have been proposed by many research groups for interconnecting multiple processors [1]. Also, MINs have been identified as efficient interconnection networks for communication structures such as ATM switches, terabit routers, and Ethernet switches [2]–[4].

To evaluate a MIN's performance some general methods have been used, which are mainly classified into three major groups. The first group of approaches includes analytical models. Most of these models are based on Markov models. The second group applies graphical on Petri-nets methods, which are considered rather complicated. Finally the third group of scientific evaluation methods employs simulation to estimate network performance. In this paper it was decided to develop an analytical method as it is known that although these methods take more time in the development stage they give results in a very short time. Accurate performance estimation before network implementation is essential, since it allows network designers to adapt network designs and tune operational parameters to the specific requirements of the system under implementation, thus allowing the building of efficient systems, cost reduction, and minimization of rollout times.

A typical MIN can be considered as a network used to interconnect a group of N inputs to a group of M outputs using several stages of small size switching elements (SEs) connected by link states. To specify a MIN its topology, routing algorithm, switching strategy, and flow control mechanism, among others, must be given. A MIN with the banyan property is specified by the fact that there is exactly one unique path from each source (input) to each sink (output). Banyan MINs are multistage self-routing switching fabrics. Thus, each SE of the s^{th} stage can decide which output port a packet should be routed to, depending on the corresponding s^{th} bit of the destination address. An $(N \times N)$ MIN can be constructed by $s = \log_c N$ stages of $(c \times c)$ SEs, where c is the degree of the SEs. A typical SE is illustrated in "Fig. 1". At each stage there are exactly N/c SEs, and consequently the total number of SEs of a MIN is $(N/c) \cdot \log_c N$. Thus, there are $O \cdot (N \cdot \log_c N)$ interconnections among all stages, as opposed to the crossbar network, which requires $O(N^2)$ links.

1.1 Prior related surveys

Performance evaluation work on the networks has also been done quite extensively [5]–[8]. However, the older of these evaluation models are commonly

based on an unrealistic assumption that an independent request is generated to replace the previously blocked and still un-serviced request, and so on. This assumption helps researchers to simplify the theoretical model, but the simplification will result in discrepancies in predicting network performance.

Turner [9] makes a thorough study related to the multicast feature in Clos networks, considering the Clos networks as a subclass of MINs. Petri nets [10]–[12] have also been used as modelling methods either to complement Markov chains or as self-contained approaches. There are many studies of uniform load traffic on MINs' inputs in the literature, such as [13], [14]. Hot-spot traffic performance was also examined by Jurczyk [15] in a clever manner.

1.2 This work

In this paper we propose a novel two-level priority scheme for performing routing within the MIN. This approach takes into account the queue lengths of the MIN switching elements, prioritizing packets in SEs having greater queue lengths. The rationale behind this approach is that by using two parallel queues which are serviced by one server, the probability of high priority packets increases, and thus online applications are given the chance to show much better behavior. This is expected to increase network performance, while fairness between packets is also promoted. The performance of the proposed scheme is also evaluated and compared against that of single-priority and single-layered MINs. This work aspires to put the spotlight on a ripe and important area for future performance estimation research.

Organization. The remainder of this paper is organized as follows: Section 2 describes the network operations and priority schemes. Assumptions and basic definitions for the analytical model are also presented in Section 2. In Section 3 the analysis and performance evaluation methodology for finite-buffered MINs with priority schema is presented. Section 4 presents the results of our performance analysis, which has been conducted through simulation experiments for single and semi-layer MINs. In Section 5, model verification and semi-layer MIN performance results are presented, while Section 6 provides the concluding remarks and outlines some directions for future work.

2 Network operations, assumptions, and definitions

2.1 General view

A typical configuration of a banyan type $N \times N$ MIN –proposed by Patel [18] – is shown in "Fig. 1".

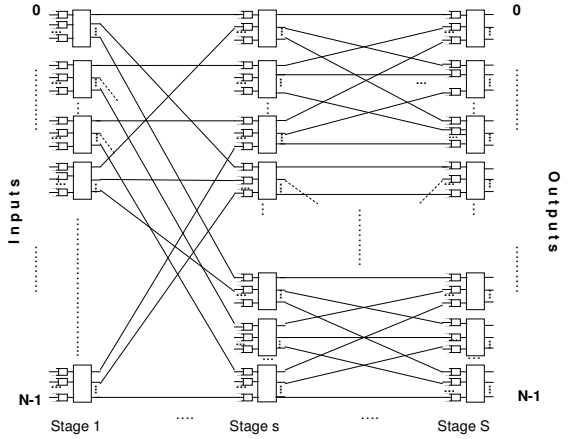

Fig.1 An NxN typical single-layer S-stage MIN constructed by 2 × 2 SEs

A processor that is connected to an input port and can generate and send requests to other processors is called a sender while a processor that is connected to an output port and receives requests is called a receiver. A circuit connection between a sender and a receiver is called a path.

For instance the MIN illustrated in "Fig. 1" is a Modified Data Manipulator type and is established by numbering the SEs at each stage (S). Numbers are coded to the base of c (here c for our study is equal to 2), starting with 0. This means that each SE is numbered by an (s–1)-digit number $v_{n-1}, v_{v-2,} \ldots v_2, v_1$, where $0 \le v_i \le c-1$ and $1 \le i \le S-1$. Then, SE $v_{n-1}, v_{v-2,} \ldots v_2, v_1$ at stage s is connected to c SEs at stage $s+1$ that are numbered $\ldots \ldots \ldots \ldots \lozenge\, v_{n-k-2} \ldots v_1$, where $\lozenge$ equals all values from 0 to $c-1$.

A 2 × 2 SE is shown in "Fig. 1". A request from either one of the inputs can be connected to either one of the outputs if the output is not occupied by some other request. If the switching element connects an input to an output as requested, we say that the request is passed. A path between an input and an output in an SE is called a switching connection.

When a request has arrived at the input of an SE in arbitrary stage (s) and if it has not passed the switching element, we say that the request is at stage (s). In general, there are three possible states for a request at an SE. In the first state, there is no

previous request from the inputs and the probability of a new request passing is equal to 1. In the second state, a request has already passed, and the probability of new request passing is equal to 1/2.

Finally, in the third state the SE has two requests arriving in the same cycle. If both requests ask for the same output, one will be passed while the other will not be serviced.
The choice between the requests in the conflicting case depends on the priority set in the switching element. The probability of a request passing in the third state is equal to 0.75 given that the intended output for each request is equally distributed between both outputs.
When a sender generates a request, it is delivered through a link to an input of an SE in stage (s). The request will pass stage 1 with a probability of 0.5 or 0.75 depending on the state of the other input of the switching element. Requests that pass stage 1 will progress to stage 2 and so on in the same way.

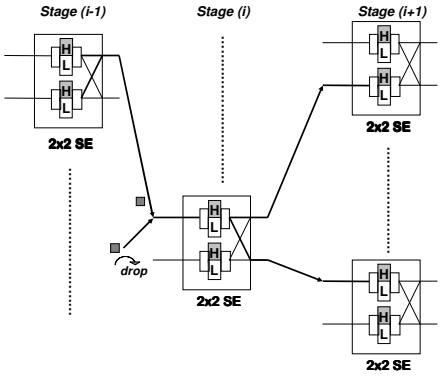

Fig. 2. Internal view of multistage interconnection network with double priority service

When a request passes the last stage s ($S = \log_2 N$, where N is the number of senders or receivers) a path is established between the sender and the receiver. A request can pass one stage during a time cycle and therefore at least (S) cycles are required to establish a path between a sender and a receiver.
In the packet drop strategy, for example a relaxing blocking mechanism, the switching element that has decided to block a request sends a release signal back to the sender along the partially established path. This path is cleared at the end of the cycle. The blocked request, now dropped, goes under the same process starting from stage 1 until a path is established.
In the packet hold strategy or backpressure blocking mechanism, the blocked request keeps the partial path and continues the connection effort in the same stage. The idea behind applying the relaxing blocking mechanism is to reduce the requests traffic in the network, which increases the probability of passes in the stages near to the receivers. But if a request makes a path achieves nearness to its destination, and is then dropped, there is much waste of effort. On the other hand, the idea behind the packet hold strategy is to prevent that kind of waste, but it causes more effort to be required in the switching fabric.

2.2. Basic assumptions

In our paper, we consider in the first part of the study a single layer MIN with the banyan property which operates under the following assumptions:
The traffic feeding the first stage of the switch fabric follows Bernoulli distributions, that is, the probability that a packet will arrive within a clock cycle is constant and the arrivals are independent of each other. We will denote this total probability of arrivals as λ, and this includes high and low priority packets.
All the packets that reach this fabric have identical and constant size.
A packet arriving at the first stage ($s = 1$) is discarded if the buffer of the corresponding SE is full.
The packets are uniformly distributed across all the inputs and their destinations and each queue use a FIFO policy for all output ports.
The service time of the output queues at each switch is assumed to be fixed and equal to the network cycle time.
The network clock cycle consists of two phases. In the first phase, flow control information passes through the network from the last stage to the first one. Flow control information generally includes data regarding the size of the queues in the subsequent stages as well as congestion control signals' flags. In the second phase, packets flow from one stage to the next in accordance with the flow control information. SEs operates in a slotted time model and routing is performed in a pipeline manner, meaning that the routing process occurs in parallel in every stage.
A packet is blocked at an arbitrary stage if the next destination buffer at the next stage is full.
When two packets at a stage contend for a buffer at the next stage and there is not adequate free space for both of them to be stored (i.e. only one buffer position is available at the next stage), there is a conflict. In the single priority (or no-priority) scheme MINs, one packet will be accepted at random and the other will be dropped. In this proposed internal-priority scheme, if a conflict occurs it is resolved in the same way; for example one of the packets will be serviced while the other

packet will be dropped, and the transmitting SE will realize a signal to the sender during the next network cycle.
Finally, all packets in input ports contain both the data to be transferred and the routing tag. In order to achieve synchronously operating SEs, the MIN is internally clocked. As soon as packets reach a destination port they are removed from the MIN, so packets cannot be blocked at the last stage.
Our analysis introduces a novel model which considers not only the current state of the associated buffer, but also the previous one (one clock history consideration).

2.3 Basic definitions

Number of MIN stages S is the number of stages of which an $(N \times N)$ MIN consists. In our simulation experiments S is assumed to be N = 4, 6, 8, and 10. For the case of $(2x2)$ SEs, is $S = \log_2 N$ always.
Buffer size (b) is the maximum number of packets that an input buffer of an SE can hold. In this work we consider a finite-buffered b = 1, 2, 4, 8, and 10.
Probability of packet arrivals (λ) is the steady-state fixed probability of packets arriving (high and low priority) at each queue at a MIN's inputs. In our simulation λ is assumed to be 0.1, 0.2, ..., 0.9, and 1.
Probabilities of high and low packet arrivals are (λ_1), (λ_2). These probabilities are the steady-state fixed probabilities high and low priority packets, respectively, arriving at each input queue, $\lambda = \lambda_1 + \lambda_2$ always. The total offered load at every input is parceled out in two fractions among two separate queues. The packet arrivals follow a Bernoulli type distribution.
$p_{i,j}^{(s)}(n)$ is the probability that (i) number of high class priority packets and (j) number of low class priority packets arrive at the pair of buffers of an SE at an arbitrary stage (s) in a random (n) timeslot.
$x_{i,j}^{(s)}(n)$, represents the joint probability occupied by (i) number of high class priority packets and (j) number of low class priority packets in a pair of adjoining queues of an arbitrary stage (s) in a random time cycle (n).
$y_1^{(s)}(n)$, $y_2^{(s)}(n)$ are the probabilities that packets of high and low class priority packets respectively existing at the head of corresponding queue demanded to be transmitted to the next stage. All those number of packets are under the queues server control and make tendency the system.

2.4 Basic performance metrics

In order to evaluate the performance of an $(N \times N)$ MIN with $(\log_c N)$ intermediate stages of $(c \times c)$ SEs, the following metrics are used. Let T be a relatively large time period divided into ω discrete time intervals $(\tau_1, \tau_2, \tau_3, \ldots, \tau_\omega)$.
Average throughput $Th_{(avg)}$ is the average number of packets accepted by all destinations per network cycle. This metric is also referred to as bandwidth. If $\nu(i)$ denotes the number of packets that reach their destinations during an arbitrary i^{th} time cycle then for a number (ω) of consecutive time slots this number reaches $\sum_{i=1}^{\omega} \nu(i)$. Hence, $Th_{(avg)}$ can be defined formally by the following expression:

$$Th_{(avg)} = \lim_{\omega \to \infty} \frac{\sum_{i=1}^{\omega} \nu(i)}{\omega} \qquad (2.1)$$

The throughputs for packages of high and low priority packets $(Th_{High(avg)}, Th_{Low(avg)})$, respectively, are defined similarly.
Normalized throughput $Th_{(Norm)}$ is the ratio of the average throughput $Th_{(avg)}$ to network size (N). Formally, $Th_{(Norm)}$ can be defined by: $Th_{(Norm)} = \frac{Th_{(avg)}}{N}$.
Normalized throughput is a good metric for assessing the MIN's cost effectiveness. Also, as normalized throughput of high priority packets $(Th_{High(Norm)})$ is defined as: $Th_{High(Norm)} = \frac{Th_{High(avg)}}{N}$, and the complement metric of normalized throughput of low priority packets $Th_{Low(Norm)}$ is defined similarly, it is obvious that we always have:

$$Th_{Norm} = Th_{High(Norm)} + Th_{Low(Norm)}$$

Average packet delay (D_{avg}) in an NxN MIN is defined as the average time slots that are needed by packets to traverse the MIN from inputs to outputs. The average delay of packets (D_{avg}) in traversing an NxN MIN is defined as:

$$D_{avg} = \lim_{\omega \to \infty} \frac{\sum_{i=1}^{\omega} t(i)}{\omega} \qquad (2.2)$$

where (ω) denotes the total number of packets accepted by destinations in (ω) consecutive time intervals and $t(i)$ represents the total number of time slots that the i^{th} packet takes to arrive at the MIN's output. This total number of time slots $t(i)$ includes the total number of time slots taken for packet transmissions through the MIN and the total number of time slots taken by the packet being in a waiting state in all the intermediate buffers that are used by it.

Similarly, the *average packet delay* ($D_{High(avg)}$, $D_{Low(avg)}$) for high and low class priority packets in a MIN is defined. These average delays are denoted as the average time slots that are needed by high and low class priority packets respectively to traverse the MIN from inputs to outputs.

Normalized packets delay (D_{Norm}) in a MIN with S number of stages is defined as a ratio of average delay (D_{avg}) in time slots to the minimum packet delay in time slots. Minimum packet delay can be considered to be the total number of time slots needed by a packet for transmission to its destination without any blocking. Thus, for a MIN with (S) stages the minimum packet delay time is equal to $S \cdot (time\,slot)$. Formally, D_{avg} can be defined as:

$$D_{Norm} = \frac{D_{avg}}{S} \qquad (2.3)$$

Similarly, *normalized packet delay* ($D_{High(Norm)}$, $D_{Low(Norm)}$) for high and low priority packets in a MIN is defined as the average time slots that are needed by high and low priority packets respectively to traverse the MIN from inputs to outputs.

Normalized packet loss (p_{loss}). This probability shows the packet loss from all the MIN's stages. Here, for the MINs that apply the packet drop strategy in conflicting cases, packets may be lost in any stage of the MIN except the last stage, where packet conflict does not appear. *The probability of packet loss* can be given as:

$$p_{loss(Norm)} = p_{loss(Norm)}(0) + \sum_{s=1}^{S-1} p_{loss(Norm)}(s) \qquad (2.4)$$

Where $p_{loss(Norm)}(0)$ represents the packet loss probability that happens at the MIN's inputs before they enter the fabric and $p_{loss(Norm)}(s)$ represents the probability of packet loss in a queue of an s^{th} intermediate stage of a MIN with S total number of stages. It should be remembered that the last stage does not have packet conflict and hence does not have dropped packets. In the same way, $p_{High\_loss(Norm)}$ and $p_{Low\_loss(Norm)}$ for high and low priority packets are defined.

3 Analysis – performance evaluation methodology

Lets $p_{i,j}^{(s)}(n)$ be the probability that (i) number of high priority packets and (j) number of low priority packets arrive at a pair of adjoined parallel buffers of an SE at an arbitrary stage (s) in a random (n) timeslot. Then $p_{i,j}^{(s)}(n)$ can be calculated by the expression:

$$p_{i,j}^{(s)}(n) = \binom{i+j}{i} \cdot (\lambda_1^{(s)}(n))^i \cdot (\lambda_2^{(s)}(n))^j \cdot \binom{N}{i+j} \cdot (1 - \frac{\lambda^{(s)}(n)}{N})^{N-i-j} \cdot \left(\frac{1}{N}\right)^{i+j} \qquad (3.1)$$

Where s = 1, 2, ..., S stages of the MIN and (n) is the random time slot, $i = 0,...b_1$ is the packet population of high priority packets, and $j = 0,...b_2$ is the packet population of low priority packets. Also, the sum of i, j represents the total number of packets. Time slot n indicates the time interval $[n, n-1]$. In addition, the corresponding packet arrivals on the inputs per time slot ($\lambda_1^{(s)}(n)$, $\lambda_2^{(s)}(n)$) can be written as follows:

$$\lambda_1^{(s)}(n) = \sum_{i=1}^{b_1} \sum_{j=0}^{b_2} x_{1,j}^{s-1}(n-1) \cdot y_1^{(s-1)}(n-1) \qquad (3.2)$$

where $s = 2,3,....,L$ while for s=1: $\lambda_1^{(1)}(n) = p_1(n)$

$$\lambda_2^{(s)}(n) = \sum_{j=1}^{b_2} x_{0,j}^{(s-1)}(n-1) \cdot y_2^{(s-1)}(n-1) \qquad (3.3)$$

Where $s = 2,3,....,L$ while for s=1: $\lambda_2^{(1)}(n) = p_2(n)$

It should be remembered that $x_{i,j}^{(s)}(n)$ represents the joint probability that there are i high priority packets and j low priority packets in the adjoining queues in a random stage (s) and in an arbitrary time cycle (n). (2.4)

Also, $y_1^{(s)}(n)$, $y_2^{(s)}(n)$ represents the probabilities that packets of high and low priority classes respectively

exist at the heads of corresponding adjoining queues ready to be transmitted.

The probabilities $y_1^{(s)}(n)$, $y_2^{(s)}(n)$ can be calculated as follows:

For the stage $s = 1,2,....,\ S-1$:

$$y_1^{(s)}(n) = p_{H\_pass}^{(s+1)}(n) \cdot \sum_{i=1}^{N-1} \sum_{j=0}^{N-1-1} \frac{p_{i,j}^{(s)}(n)}{i+1} \quad (3.4)$$

where $p_{H\_pass}^{(s+1)}$ denotes the probability that the next queue is ready to accept a high priority packet from the current stage (s), that is the relevant queue of the next stage is not in a blocking stage.

In the same way, the probability $y_2^{(s)}(n)$ is calculated by:

$$y_2^{(s)}(n) = p_{L\_pass}^{(s+1)}(n) \cdot \sum_{t=0}^{b_2} p_{0,t}^{(s)} \cdot \sum_{i=1}^{N-1} \sum_{j=0}^{N-1-1} \frac{p_{i,j}^{(s)}(n)}{j+1} \quad (3.5)$$

where $p_{L\_pass}^{(s+1)}$ denotes the probability that the next queue is ready to accept low priority packets from the current stage (s). Hence, the relevant queue of the next stage is not in a blocking stage.

Also, because it is considered that the last stage ($s = S$) is never blocked, we can write: $p_{H\_pass}^{(s=S)} = 1$ and $p_{L\_pass}^{(s=S)} = 1$.

Where for the remaining stages $s = 1,2,....,S-1$, the probability of the next stage being ready to accept high priority packets happens when the next stage is not full and the packet will certainly be transmitted from the current stage to the next. This concept is expressed by the formula:

$$p_{H\_pass}^{(s)} = 1 - P_{full\_high}^{s+1}(n) \cdot (1 - y_1^{s+1}(n)) \quad (3.6)$$

where $P_{full\_high}^{(s+1)}(n) = \sum_{j=0}^{b_2} p_{b_1,j}^{(s+1)}(n)$

Similarly, the concept of the next stage not being full and the low priority packet certainly being transmitted is depicted by the expression:

$$p_{L\_pass}^{(s)} = 1 - P_{full\_low}^{s+1}(n) \cdot (1 - y_2^{s+1}(n)) \quad (3.7)$$

where $P_{full\_low}^{(s+1)}(n) = \sum_{i=0}^{b_1} p_{i,b_2}^{(s+1)}(n)$

Last stage ($s = S$):

For the last stage, the following can be written: $y_1^{(s=S)}(n) = 1$ and $y_2^{(s=S)}(n) = 1$, because the last stage is considered as a non-blocking stage.

In order to evaluate the performance metrics in this scenario, we make the approximate assumption of interstage independence (which seems to be more accurate, as buffer size is getting smaller). Normally the blocking has a stronger dependence among stages but this dependency becomes smaller and also a drop resolution mechanism is applied here.

3.1 A-phase of calculations

Let us use for simplicity $x_{i,j}^{(s)}$, $y_1^{(s)}$, $y_2^{(s)}$, and $p_{i,j}^{(s)}$ instead of $x_{i,j}^{(s)}(n)$, $y_1^{(s)}(n)$, $y_2^{(s)}(n)$, and $p_{i,j}^{(s)}(n)$ in a random stage (s). This simplification in writing is adopted in order to form the equations more easily without any loss-making to the generality. Hence, based on the above analysis the probabilities of the queue's packet population can be calculated as follows:

The probability of both adjoining queues (high and low) being empty can be calculated as:

$$x_{0,0}^{(s+1)} = \left(x_{0,0}^{(s)} + x_{1,0}^{(s)} + x_{0,1}^{(s)} \right) \cdot p_{0,0}^{(s+1)}$$

Working in the same way and in general for the intermediate case of holding i number of high priority packets and j number of low priority packets in adjoining parallel buffers, $x_{i,j}^{(s+1)}$ can be written:

$$\begin{aligned} x_{i,j}^{(s+1)} = {} & x_{0,0}^{(s)} \cdot p_{i,j}^{(s+1)} + \sum_{t=1}^{i} x_{t,0}^{(s)} \left[y_1^{(s)} \cdot x_{i-t+1,j}^{(s+1)} + (1 - y_1^{(s)}) \cdot p_{i-t,j}^{(s+1)} \right] + \\ & x_{i+1,0}^{(s)} \cdot y_1^{(s)} \cdot p_{0,j}^{(s+1)} + \sum_{t=1}^{j} x_{0,t}^{(s)} \cdot \left[y_2^{(s)} \cdot p_{i,j-t+1}^{(s+1)} + (1 - y_2^{(s)}) \cdot p_{i,j-t}^{(s+1)} \right] + \\ & x_{0,j+1}^{(s)} \cdot y_2^{(s)} \cdot p_{i,0}^{(s+1)} + \sum_{t=1}^{i} \sum_{\rho=1}^{\xi} x_{t,\rho}^{(s)} \cdot \left[y_1^{(s)} \cdot p_{i-t+1,j-m}^{(s+1)} + (1 - y_1^{(s)}) \cdot p_{i-1,j-m}^{(s+1)} \right] + \\ & \sum_{t=1}^{j} x_{i+1}^{(s)} \cdot y_1^{(s)} \cdot p_{0,j-t}^{(s+1)} \end{aligned} \quad (3.8)$$

where $i = 1,2,..,(b_1 - 1)$ and $j = 1,2,..,(b_2 - 1)$.

Moreover, in Appendix A' a set of equations is demonstrated which shows the joint probability of high and low priority packets in adjoining buffers in some marginal cases in an arbitrary stage ($s+1$) and in a random time cycle.

Formulas (3.8) as well as the remaining formulas (A1)-(A6) in Appendix A, in conjunction with formulas (3.1), (3.2), and (3.3), give us the joint probability $x_{i,j}^{(s+1)}$ for $i = 1,2,..,(b_1 - 1)$ and $j = 1,2,..,(b_2 - 1)$.

Calculate marginal distribution of full high and low packet populations

For $i = b_1$ and $j = b_2$, the joint probability $x_{b_1,b_2}^{(s+1)}$, which represents the existence of (b_1) number of high priority packets and (b_2) number of low priority packets in a pair of two corresponding priority queues of an arbitrary stage (s), is calculated by the expression:

$$x_{b_1,b_2}^{(s)} = 1 - \sum_{i=0}^{b_1-1} \sum_{j=0}^{b_2-1} x_{i,j}^{(s)} \qquad (3.9)$$

Subsequent to the above analysis, in the next subsection we use a convergent algorithm in order to estimate the steady state of a buffer packet population for every stage.

3.2 Apply a convergence algorithm

However, in the general case, in order to get the parameters of the arrival process and service time of a queue at an arbitrary stage (s), it is necessary to know the solutions to the next stages. This special requirement will be taken into account to some extent by adopting a convergence procedure. Thus, our approximation scheme will be entirely a convergence procedure which is repeated until the queue probability of buffer occupation by high and low priority packets $x_{i,j}^{(s)}$ does not change any more.

The convergence algorithm which is applied here is presented in the following:

The Algorithm

Set S stages, b_1, b_2: maximum *buffers size*, N stages, and λ, λ_1 *offered lo*ad.

Initialize: $x_{i,j}^{(s)}, y_1^{(s)}, y_2^{(s)}, p_{H\_pass}^{(s)}, p_{L\_pass}^{(s)}$

Begin

Calculate based on Eqs. (3.1), (3.2), and (3.3): $p_{i,j}^{(s+1)}$, $\lambda_1^{(s+1)}$, and $\lambda_2^{(s+1)}$

Do

Begin

Calculate based on Eq. (3.8) and Eqs. (A1)–(A6) in the Appendix: $x_{i,j}^{(s+1)}$

End

Until $\left| x_{i,j}^{(s+1)} - x_{i,j}^{(s)} \right| \le \varepsilon$

End

Output $x_{i,j}^{(s)}$

By applying this algorithm a fixed-point iteration ($\varepsilon<10^{-5}$) over the state of joint probability $x_{i,j}^{(s+1)}$ is taken, and a steady state is reached from which the performance metrics of interest are determined.

3.3. B-phase of calculations

Intermediate results

After buffer occupancy estimation, we estimate the marginal distribution of high and low packet populations:

$$x_{1,i}(s) = \sum_{j=0}^{b_2} x_{i,j}^{(s)}, \quad i = 1,2,..,(b_1 - 1) \text{ and}$$

$$x_{2,j}(s) = \sum_{j=0}^{b_1} x_{i,j}^{(s)}, \quad j = 1,2,..,(b_2 - 1)$$

Based on the above results we can calculate the marginal distribution expected for high and low priority buffer occupancies. If $X_{High}(s)$ and $X_{Low}(s)$ denote the numbers of high and low priority packets that are utilized by two adjoining buffers in a stage s, they are estimated by the following expressions:

$$Average\ X_{High}(s) = \sum_{j=0}^{b_1} i \cdot x_{1,j}^{(s)} \quad \text{for } j = 1,2,..,(b_2 - 1)$$

and

$$Average\ X_{Low}(s) = \sum_{i=0}^{b_1} j \cdot x_{2,j}^{(s)} \quad \text{for } i = 1,2,..,(b_1 - 1)$$

Final results

A. Calculate throughput.

- The *high priority throughput* of the MIN is: $Th_{High(Norm)} = 1 - x_{1,0}^{(S)}$
- The *low priority throughput* of the MIN is: $Th_{Low(Norm)} = (1 - x_{2,0}^{(S)}) \cdot x_{1,0}^{(S)}$
- The total MIN's *throughput* is: $Th_{Norm} = Th_{High(Norm)} + Th_{Low(Norm)}$

B. Calculate packet loss probabilities.

Hence the packet *loss probabilities* of the system are calculated as follows:

The *loss probability* of high priority packets is estimated as:

$$p_{High\_loss(Norm)} = \frac{\lambda_1 - Th_{High(Norm)}}{\lambda_1}$$

The *loss probability* of low priority packets is estimated as:

$$p_{Low\_loss(Norm)} = \frac{\lambda_2 - Th_{Low(Norm)}}{\lambda_2}$$

The *total loss probability* of the MIN is:

$$p_{loss(Norm)} = p_{High\_loss(Norm)} + p_{Low\_loss(Norm)}$$

C. Calculate packet latency.

The *packet latencies* of high and *low priority packets* respectively are calculated by:

$$D_{High(Norm)} = \frac{\sum_{s=1}^{S} X_{High(s)}}{X_1(S)}, \text{ and}$$

$$D_{Low(Norm)} = \frac{\sum_{s=1}^{S} X_{Low}(s)}{X_2(S)} \quad \text{respectively}$$

4 Simulation and performance results

4.1 Simulation

The performance of MINs is usually determined by modelling, either by simulation [19], [20] or by analytical approaches [21], [22]. In this paper the network performance is also estimated using an analytical method and simulations. A simulator for MINs in a packet communication environment is developed with C++ programming language. The simulator can work for various switch types, inter-stage interconnection patterns, and variable load conditions, and implements a drop and backpressure mechanism with single or double schema of priorities.

We focused on an (N × N) MIN that consists of (2 × 2) SEs, using double queuing architecture. Each (2 × 2) SE in all stages of the MIN was modelled by two non-shared buffer queues. Buffer operation was based on the FCFS principle. In the case of non-priority scheme MINs, when there was a contention between two packets, it was solved randomly.

The performance of non-priority (or single priority) MINs was compared against the performance of two priority MINs with relaxing blocking, where contentions were resolved by favoring the packet transmitted from the SE with the longest transmission queue. The simulation was performed at packet level, assuming fixed-length packets transmitted in equal-length time slots, (although the above analysis is applied for queues of unequal lengths) where the slot was the time required to forward a packet from one stage to the next.

Results for the packet traffic model were varied across simulation experiments to generate various total offered loads and traffic patterns. Basic performance statistics such as packet throughput and packet delays were collected at the output ports. Extensive simulations to validate our results were performed. All performance results which were collected by simulation were obtained by a simulator running for 10^5 clock cycles. The number of simulation runs was adjusted to ensure a steady-state operating condition for the MIN.

4.2 Performance Results

"Fig. 3" shows the *normalized loss packets* of a 10-stage MIN versus the high priority offered load on inputs for MINs with various *buffer sizes* b (where b is equal to 2, 4, 6, or 8). The solid and dotted curves depict the *probabilities* of *packet loss* estimated by the analytical model and simulation respectively.

From "Fig. 3" it is obvious that when the *buffer size* is increased the *probability of packet loss* decreases. For example, for full high priority load on inputs of a 10-stage MIN with $b = 2$, the *loss probability* is equal to 0.42, whereas when the same fabric is constructed with buffer size $b = 8$ the loss probability drops to 0.2.

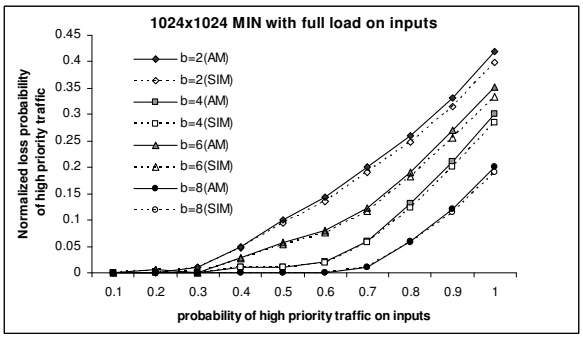

Fig. 3. *Normalized loss probability* of high priority load versus the high priority *offered load*

The near coincidence of results obtained from the analytical model and simulation is a first indication that our results are accurate.

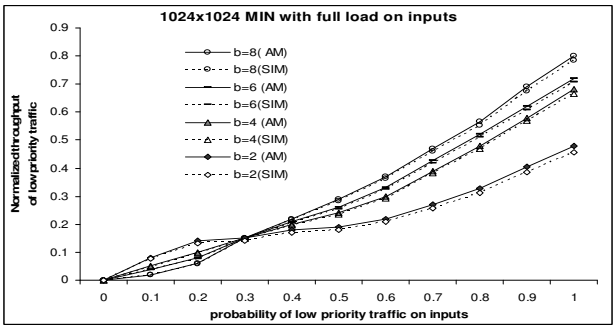

Fig. 4. *Normalized throughput* of low priority traffic versus *low priority traffic* on inputs

"Fig. 4" illustrates the *normalized throughput* of low priority traffic of a 10-stage MIN with various buffer sizes b versus the low priority offered load on MIN inputs. The solid and dotted curves depict the *throughputs* estimated by the analytical model and simulation respectively.

From "Fig. 4" it can be observed that there is a crucial point ($\lambda_2 = 0.3$); when the low priority traffic overcomes this point the *throughput* rises sharply as well as the buffer size is increased. On the other hand when the low priority load remains below this point the situation of the low priority *throughput* is reversed, thus in the second case the increment of buffer size is an disadvantage.

"Fig. 5" presents the corresponding increments and the simultaneous reductions in *normalized throughput* of high and low priority packets respectively versus the probability of high priority load arriving on inputs. The *throughput* results are given for various 10-stage MINs constructed with different buffer sizes (b = 2, 4, 6, 8, and 10) when the *offered load* on inputs is full.

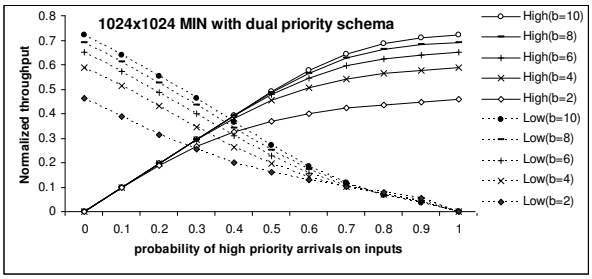

Fig. 5. *Normalized throughput* of the MIN's high and low traffic respectively versus *probability of high priority traffic* on inputs

From the curves it is obvious that there is a gain in *throughput* when the buffer size is large because it enables more packets to be accumulated.

Also, in "Fig. 6", with the same MIN configurations as in "Fig. 5", the high values of packet delay that appear when the buffer size increases and the total offered load is full are obvious.

From both "Fig. 5" and "Fig. 6" it is evident that when a fabric is implemented with a high value of buffer size (e.g. b = 10) then high values of *throughput* and high values of *packet delay* appear. Nevertheless, those performance metrics values oppose each other and therefore the choice depends entirely on the requirements of the applications that we have to service.

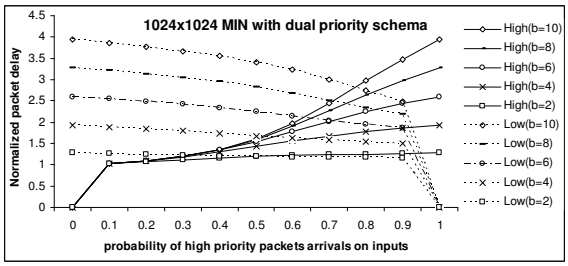

Fig. 6. *Normalized packet delay* of *high* and *low* priority packets versus *probability of high priority packet arrivals* on inputs.

For example, applications which demand low values of *packet delay* and *jitter* will be accompanied by MINs that have a small value of buffer size (e.g. $b = 1$).

"Fig. 7" depicts the *normalized throughput* of a 10-stage MIN with full load for different values of buffer size where b is equal to 4, 6, 8, and 10.

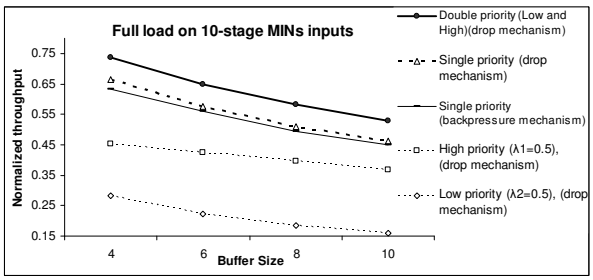

Fig.7. *Normalized throughput* of various 10-stage MINs with *buffer sizes* of 4, 6, 8, and 10 respectively.

The two solid curves depict the total *normalized throughputs* for a single priority 10-stage MIN with a backpressure blocking mechanism and a corresponding double priority fabric with drop mechanism respectively. The second one (with drop mechanism) always achieves better *throughput* as well as alleviating the MIN's tendency which appears in all MIN constructions with the backpressure mechanism. So, for a given buffer size the *throughput* of a single priority multistage fabric tends to be higher when it employs a double priority schema with a drop mechanism in contrast to the corresponding single priority MINs (with drop or backpressure blocking mechanism).

But in all cases the fabrics which are constructed by buffers with small sizes achieve entirely better values of *throughputs*. That and the corresponding costs are the reasons why construction companies prefer to build MINs with small buffer sizes.

Thus, all our experiments indicate that both major performance metrics (in terms of *throughput* and *latency*) are improved because the packets exploit more room (double parallel buffers, traffic dichotomy in high and low traffic) and because on the other hand the drop technique alleviates the MIN's tendency.

5 Model verification and semi-layer MIN's performance

5.1 Model Verification

"Fig. 8" illustrates the *normalized throughput* which is estimated by our analytical model (AM – solid curves) for three-stage MINs with various values of buffer sizes (b = 1, 3, and 9) versus the *offered load* on inputs. In "Fig. 8", we show the results of our model in marginal cases in which our model works with all the traffic in the high class priority ($\lambda = \lambda_1 = 1$) or the opposite, where all the traffic is in the low class priority $\lambda = \lambda_2 = 1$.

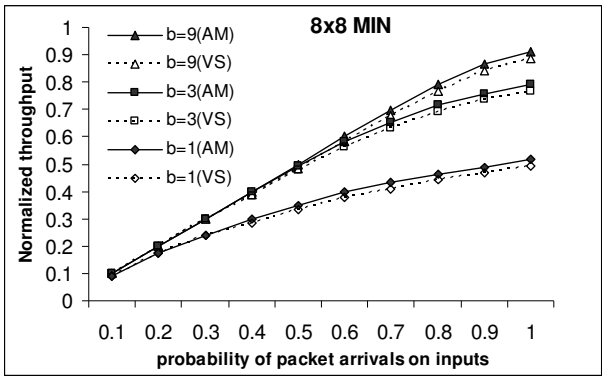

Fig. 8. *Normalized throughput* of various three-stage MINs with *buffer sizes* of 4, 6, 8, and 10

"Fig. 8" shows that our results are in close agreement with the corresponding results of simulation reported in paper [16] by Vasiliadis and his colleagues (SV – dotted curves). In paper [16], "Fig. 3", the curves $BL\_\beta = 0$, $BL\_\beta = 2$, and $BL\_\beta = 8$ show the results for an 8 × 8 MIN with buffer sizes of 1, 3, and 9 respectively. That is because the authors consider a single buffered MIN (b = 1 here) as an unbuffered MIN and they denote it as a MIN with $\beta = 0$. They handle the subsequent cases of buffer sizes in the same way. Those authors and some others prefer, not to count the buffer position which is handled by the queue server in the number of the buffer size. This is simply a matter of denotation. In [16] the authors studied a three-stage MIN which works with single priority, and their results are extracted by simulations.

The near coincidence of our results with the corresponding results reported in "Fig. 3" of [16] is an additional indication that convinces us that this analytical approach provides accuracy and correct results.

5.2 Semi-layer fabrics with packet drop technique

Multilayer MINs were introduced recently by Tusch and Brenner [20]. Single-layer MINs (SiLMINs), replicated MINs, and semi-layer MINs (SeLMINs) can be considered special cases of multilayer multistage interconnection network (MLMINs) architectures.

In papers [17], [24], and [25], Garofalakis et al. have given the definition of semi-layer MINs and have also given a definition of three basic factors (start replication factor G_s, growth factor G_F, and layer limit factor G_L) which help in the description of plenitude multilayer constructions.

According to [20], [25], if the structure of a MIN replicates the number of layers in a stage, and this replication is equal to the number of inputs per switch (whereby the growth of the number of layers in a stage – the growth factor G_F – is c, where c is the number of inputs per switch), then this structure ensures that no internal blocking occurs in SEs, even if all SE inputs broadcast their packets to all SE outputs [20].

In this work semi-layer MINs whose first segment (single layer part) applies a drop strategy in handling conflicting packets while the second is a multilayer one (with two layers' growth) are also studied. Moreover it should be remembered that the second segment operates in a non-blocking fashion. When the packets claim the same resource, then one of them can earn this resource while the other has the ability to escape via the alternative route which exists and is led to a higher level. In addition, it is considered that when the packets reach the end of the fabric the multiplexer beyond the fabric or data sink has adequate capacity to accept all the packets that are outputted from all levels with high or low priority.

5.2.1 Throughput of Semi-layer MINs

Let us denotes as $Th_{SL(avg)}$ the *average throughput* that appears at the end of the first segment of a semi-layer MIN; then, because the second segment (the full *fan-out*) is free of blocking, the *average throughput* at the end of the MIN has the same value:

$Th_{Semi-layerMIN(avg)} = Th_{SL(avg)}$.

5.2.2 Packet delay of Semi-layer MINs

- Average packet latency of a semi-layer MIN $(D_{Semi-layerMIN(avg)})$

Average packet latency ($D_{Semi-layerMIN(avg)}$) is the average time a packet spends passing through the semi-layer MIN. Thus, the *average packet latency* in a SeLMIN is calculated as the sum of packet delays in both segments – the first (with drop mechanism) segment and second (non-blocked) segment of a semi-layer fabric. Hence, this *average packet latency* is given by the following formula:

$$D_{Semi-layerMIN(avg)} = D_{SL(avg)} + D_{ML(avg)} \quad (5.1)$$

Where $D_{SL(avg)}$ represents the *average packet delay* in the single layer (the first segment of the semi-layer MIN), which in the case of a single buffered segment can be calculated based on this analytical method. The $D_{ML(avg)}$ represents the *average packet delay* in the multilayer, second and unblocked segment of the semi-layer MIN. The *average packet latency* $D_{ML(avg)}$ of the second segment of the SeLMIN (the fan-out) is calculated as: $D_{ML(avg)} = S_{ML}$ where S_{ML} depicts the number of stages of the second segment (the fan-out) of a MIN. Because there is no blocking in the second segment of the MIN, the *packet delay* is equal to the number of stages.

- *Normalized packet latency of a semi-layer MIN* $(D_{Semi-layerMIN(Norm)})$

The *normalized latency* $D_{Semi-layerMIN(Norm)}$ can be defined by the ratio of the packet's average latency $D_{Semi-layerMIN(avg)}$ to the minimum delay that a packet needs to traverse the SeLMIN without any blocking. This minimum *packet delay* depends on the number of stages that a semi-layer MIN has. The *normalized packet latency* of a semi-layer MIN is expressed by:

$$D_{Semi-layer(Norm)} = \frac{D_{Semi-layer(avg)}}{S} \quad (5.2)$$

After the above description of semi-layer MINs, the results of an experimental testbed using a 1024 × 1024 semi-layer MIN are described in the following subsection.

5.3 Performance results of semi-layer MINs

In "Fig. 9", a sample of a semi-layer MIN performance evaluation is illustrated, in particular, the normalized packet delay of a 10-stage semi-layer MIN which starts layer replication at stages 6 $(G_S = 6)$ and 8 $(G_S = 8)$ respectively with growth factor $G_F = 2$ and corresponding numbers of total layers 4 and 16 when they accept a full load with double priority on inputs. In the experiment which is described in "Fig. 9", the high priority traffic is considered constant and equal to 30% of the total offered load. Moreover, all the curves that are pictured in "Fig. 9" have been estimated by applying our analytical model.

In "Fig. 9" the solid curves (HP) depict the normalized packet delay of high priority packets. It can easily be seen that the high priority packets benefit from low values of delay while on the other hand the low priority packets experience higher values of packets delay.

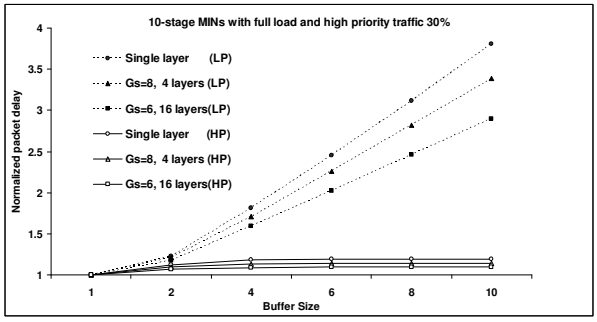

Fig.9. *Normalized packet delays* of various 10-stage MINs with *buffer sizes* of 1, 2, 4, 6, 8, and 10 respectively with full load on inputs with constant (30%) high priority traffic.

In addition the corresponding 10-stage MIN with a single layer presents the highest values of *delay* for both high and low priority packets respectively. Moreover it is obvious that the earlier the layers replication starts, the lower the values of *packets delay* that are achieved in both packet streams (high and low priority).

Nowadays, there are many applications which demand such low level values of *latency* and *jitter* (e.g. media streams, online TV, etc.). So, the

quantity of interest is the available *packet latency*. Nevertheless, an earlier starting layer replication leads to a larger size construction and hence to higher values of construction cost. Thus, a balance between performance and cost should be sought.

6 Conclusions

We studied the service quality of MINs working with a relaxing blocking technique. We built an analytical model in the single layer configuration. With this new model, we can better predict the actual performance level of the network. Also, we found that the drop strategy is better than the hold strategy in a single network except for short data transfer times. In addition, the performance level of the network with simulation is measured. Moreover, the quality of service of semi-layer MINs implementing the packet drop mechanism in their first segment is also investigated.

This work can be extended in many directions as follows. The analysis can be extended to more than two priority levels. The service time can be assumed to be general. The server can serve the packets with a scalable service according to priority class. Another research scenario is considered for the arrival and service rates which can be taken as a state dependent event. In all these variations, the present analytical approach can be considered as a guide. While this analytical approach has promise, evaluating it in wider settings remains a goal for future work.

Appendix A. Marginal calculations

Here some packets' probabilities of occupying adjoining parallel buffer sets are demonstrated. The following probabilities give the packet populations of adjoining queues in some marginal cases of queues. More specifically:

The probability of having packets if the high priority queue has (i) number of packets and the low priority one is empty is given by:

$$x_{i,0}^{(s+1)} = \left(x_{0,0}^{(s)} + x_{0,1}^{(s)} \cdot y_2^{(s)} \right) \cdot p_{i,0}^{(s+1)} + \sum_{t=1}^{i} x_{1,0}^{(s)} \cdot [y_1^{(s)} \cdot p_{i-t,1,0_{i-1,1,0}}^{(s+1)} + (1 - y_1^{(s)}) \cdot p_{i-t,0}^{(k+1)}] + x_{i+1}^{(s)} \cdot y_1^{(s)} \cdot p_{0,0}^{(s+1)} \quad \text{(A.1)}$$

where $i = 1,2,..,(b_1 - 1)$.

– Also, the marginal case of the high priority queue being full and the low priority queue being empty is:

$$x_{b_1,0}^{(s+1)} = \left(x_{0,0}^{(s)} + x_{0,1}^{(s)} \cdot y_2^{(s)} \right) \cdot \sum_{t=b_1}^{N} p_{t,0}^{(s+1)} + \sum_{t=1}^{b_1} p_t^{(s)} \left[y_1^{(s)} \cdot \sum_{\rho=b_1-t+1}^{N} p_{\rho,0}^{s+1} + (1 - y_1^{(s)}) \cdot \sum_{\rho=b_1-t}^{N} p_{\rho,0}^{s+1} \right] \quad \text{(A.2)}$$

– Working in the same way, the probability $x_{0,j}^{(s+1)}$ can be written as follows:

$$x_{0,j}^{(s+1)} = \left[x_{0,0}^{(s)} + x_{0,1}^{(s+1)} \cdot y_2^{(s)} \right] \cdot p_{0,j}^{(s+1)} + \sum_{t=1}^{j} x_{0,t}^{(s)} \cdot \left[y_2^{(s)} \cdot p_{0,j-t+1}^{(s+1)} + (1 - y_2^{(s)}) \cdot p_{0,j-t}^{(s+1)} \right] + \sum_{t=1}^{j} x_{1,t}^{(s)} \cdot y_1^{(s)} \cdot p_{0,j-1}^{(s+1)} + x_{0,j+1}^{(s)} \cdot y_2^{(s)} \cdot p_{0,0}^{(s+1)} \quad \text{(A.3)}$$

where $j = 1,2,..,(b_2 - 1)$.

– In addition, the probability in the case of a full low priority queue and empty adjoining high priority queue $x_{0,b_2}^{(s+1)}$ is calculated by:

$$x_{0,b_2}^{(s+1)} = \left[x_{0,0}^{(s)} + x_{1,0}^{(s)} \cdot y_1^{(s)} \right] \cdot \sum_{t=b_2}^{N} p_{0,t}^{(s+1)} + \sum_{t=1}^{B_2} x_{0,t}^{(s)} \cdot \left[y_2^{(s)} \cdot \sum_{\rho=b_2-t+1}^{N} p_{0,\rho}^{(s+1)} + (1 - y_2^{(s)}) \cdot \sum_{\rho=b_2-t}^{N} p_{0,\rho}^{(s+1)} \right] + \sum_{t=1}^{b_2} x_{1,t}^{(s)} \cdot y_1^{(s)} \cdot \sum_{r=b_2-1}^{N} p_{0,\rho}^{(s+1)} \quad \text{(A.4)}$$

– Moreover we continue in the same way for the symmetrical cases. Hence, for the case with a full high priority queue and (j) number of packets in low priority queues $x_{b_1,j}^{(s+1)}$ is given by:

$$x_{b_1,j}^{(s+1)} = x_{0,0}^{(s)} \cdot \sum_{t=b_1}^{N} p_{t,j}^{(s+1)} + \sum_{t=1}^{b_1} x_{t,0}^{(s+1)} \cdot \left[y_1^{(s)} \cdot \sum_{\rho=b_1-t+1}^{N} p_{\rho,j}^{(s+1)} + (1 - y_1^{(s)}) \cdot \sum_{\rho=b_1-t}^{N} p_{\rho,j}^{(s+1)} \right] + \sum_{t=1}^{j} x_{0,t}^{(s)} \cdot \left[y_2^{(s)} \cdot \sum_{\rho=b_1-t+1}^{N} p_{\rho,j}^{(s+1)} + (1 - y_2^{(s)}) \cdot \sum_{\rho=b_1}^{N} p_{\rho,j-t}^{(s+1)} \right] + x_{0,j+1}^{(s)} \cdot y_2^{(s)} \cdot \sum_{t=b_1}^{N} p_{t,0}^{(s)} + \sum_{t=1}^{b_1} \sum_{\rho=1}^{\xi} x_{1,\rho}^{(s)} \left[x_1^{(s)} \cdot \sum_{v=b_1-t+1}^{N} p_{v,j-\rho}^{(s+1)} + (1 - y_1^{(s)}) \cdot \sum_{v=b_1-t}^{N} p_{v,j-\rho}^{(s+1)} \right] \quad \text{(A.5)}$$

And finally, for the case of a full low priority queue, $x_{i,b_2}^{(s+1)}$ can be calculated as:

$$
\begin{aligned}
x_{i,b_2}^{(s+1)} &= x_{0,0}^{(s)} \cdot \sum_{t=b_2}^{N} p_{i,t}^{(s+1)} + \\
&\sum_{t=1}^{i} x_{t,0}^{(s)} \cdot \left[y_1^{(s)} \cdot \sum_{\rho=b_2}^{N} p_{i-t+1,\rho}^{(s+1)} + (1-y_1^{(s)}) \cdot \sum_{\rho=b_2}^{N} p_{i-t,\rho}^{(s+1)} \right] + \\
&x_{i+1,0}^{(s)} \cdot y_1^{(s)} \cdot \sum_{t=b_2}^{N} p_{0,t}^{(s+1)} + \\
&\sum_{t=1}^{b_2} x_{0,t}^{(s)} \cdot \left[y_2^{(s)} \cdot \sum_{\rho=b_2-t+1}^{N} p_{i,\rho}^{(s+1)} + (1-y_2^{(s)}) \cdot \sum_{\rho=b_2-t}^{N} p_{i,\rho}^{(s+1)} \right] + \\
&\sum_{t=1}^{i} \sum_{\rho=1}^{b_2} x_{t,\rho}^{(s)} \cdot \left[x_1^{(s)} \cdot \sum_{v=b_2-\rho}^{N} p_{i-t+1,v}^{(s+1)} + (1-y_1^{(s)}) \cdot \sum_{v=b_2-m}^{N} p_{i-t,v}^{(s+1)} \right] + \\
&\sum_{t=1}^{b_2} x_{i+1}^{(s)} \cdot y_1^{(s)} \cdot \sum_{\rho=b_2-t}^{N} p_{0,\rho}^{(s+1)}
\end{aligned}
\qquad \text{(A.6)}
$$

where i and j can vary as follows:

$i = 1,2,..,(b_1 - 1)$ and $j = 1,2,..,(b_2 - 1)$.

7 References:

[1] T. Feng, "A survey of Interconnection Networks," Computer, Dec. 1981, pp. 12-27.

[2] L.N. Bhuyan and C.W. Lee, "An Interference Analysis of Interconnection Networks", Proc. of the International Conference on Parallel Processing, pp. 2-9 Aug. 1983.

[3] D.M. Dias and J.R. Jump, "Analysis and Simulation of Buffered Delta Network", IEEE Transactions on Computers, Vol. C-30, August 1981, pp. 273-282.

[4] C. Wu and T. Feng, "On a Class of Multistage Interconnection Networks", IEEE Transactions Computers, Vol. C-29, Sep. 1980, pp. 694-702.

[5] J.H. Patel, "Performance of Processor – Memory Interconnection for Multiprocessors", IEEE Transactions Computer, Vol. C-30, no. 10, , pp.771-780, Oct. 1981.

[6] S. Thanawastien and V.P. Nelson "Interference Analysis of Shuffle/Exchange Network", IEEE Transactions Computer, Vol. C-30, April 1981, pp. 545-556.

[7] L. N. Bhuyan and C.W. Lee, "An Interference Analysis of Interconnection Networks", Proc. Inter. Conference on Parallel Processing, pp. 2-9, 1983

[8] D.M. Dias and J.R. Jump, "Analysis and Simulation of Buffered Delta Network", IEEE Trans. on Computers, vol. C-30, pp.273-282, April 1981

[9] J. Turner, R. Melen. Multirate Clos Networks, IEEE Communications Magazine, 41, no. 10 pp. 38-44, 2003.

[10] R. German, Performance Analysis of Communication Systems, John Wiley and Sons (2000).

[11] P. J. Haas. Stohastic Petri Nets. Springer Verlag (2002).

[12] C. Lindermann, Performance Modelling with Deterministic and Stohastic Petri Nets, John Wiley and Sons (1998).

[13] S.H. Hsiao and R. Y. Chen, Performance Analysis of Single-Buffered Multistage Interconnection Networks, 3rd IEEE Symposium on Parallel and Distributed Processing (1991) pp. 864-867.

[14] T.H. Theimer, E. P. Rathgeb and M.N. Huber, Performance Analysis of Buffered Banyan Networks, IEEE Transactions on Communications, vol. 39, no. 2 (1991) pp. 269-277.

[15] M. Jurczyk, Performance Comparison of Wormhole-Routing Priority Switch Architectures. Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications 2001 (PDPTA '01); Las Vegas (2001) pp. 1834-1840.

[16] D. C. Vasiliadis , G. E. Rizos , S. V. Margariti , L. E. Tsiantis, Comparative study of blocking mechanisms for packet switched Omega networks, Proceedings of the 6th WSEAS International Conference on Electronics, Hardware, Wireless and Optical Communications, p.18-22, February 16-19, 2007, Corfu Island, Greece.

[17] John Garofalakis, Eleftherios Stergiou, «Mechanisms and analysis for supporting multicast traffic by using Multilayer Multistage Interconnection Networks», International Journal of Network Management, 15 SEP 2010.

[18] J.H. Patel, Processor-memory interconnections for mutliprocessors, Proceedings of 6th Annual Symposium on Computer Architecture New York (1979) pp. 168-177

[19] D.C. Vasiliadis, G.E. Rizos, C. Vassilakis, "Performance Analysis of blocking Banyan Switches", Proceedings of the IEEE sponsored International Joint Conference on Telecommunications and Networking CISSE 06 (2006).

[20] D. Tutsch, M. Brenner, "MIN Simulate. A Multistage Interconnection Network Simulator", 17th European Simulation Multi-conference: Foundations for Successful Modelling & Simulation (ESM'03); Nottingham, SCS (2003) pp. 211-216.

[21] D. Tutsch, G. Hommel, "Generating Systems of Equations for Performance Evaluation of Buffered Multistage Interconnection Networks", Journal of Parallel and Distributed Computing, 62, no. 2, pp. 228-240, 2002.
[22] J. Garofalakis and E. Stergiou "An analytical performance model for multistage interconnection networks with blocking", Proc. of CNSR 2008, May (2008).
[23] Tutsch, D., Hommel, G. "Multilayer Multistage Interconnection Networks". Proceedings of 2003 Design, Analysis, and Simulation of Distributed Systems (DASD'03). Orlando, USA, 2003, pp. 155-162.
[24] John Garofalakis, Eleftherios Stergiou, "Performance Evaluation for Single and Semi Layer Multistage Interconnection Networks servicing Multicast traffic by Full multicast operation", International Journal of Communication Systems, Wiley, 2010.
[25] John Garofalakis, Eleftherios Stergiou, "Analytical Model for Performance Evaluation of Multilayer Multistage Interconnection Networks servicing Unicast and Multicast Traffic by Partial multicast operation", Elsevier Performance Evaluation, Elsevier, Volume 67, Issue 10, Pages 959-976, October 2010. ..