Performance of the Modified Round Robin Scheduling Algorithm for Input-Queued Switches Under Self-Similar Traffic

Shanmugam Arumugam[1] and Shanthi Govindaswamy[2]
[1]Bannari Amman Institute of Technology and Science, Coimbatore, India
[2]ECE Department, PSG College of Technology, Coimbatore, India

Abstract: *The iSLIP algorithm has been proved to be a very efficient, high throughput scheme for scheduling in input-queued switches. In this paper, we discuss a Modified Round Robin (MRR) scheme, which involves less number of arbitration steps in each iteration when compared to iSLIP. We prove through simulation studies, that the MRR algorithm shows a performance equivalent to iSLIP, but requires less number of processing steps. We compare the performance of the two algorithms under different types of traffic, including Bernoulli independent identically distributed (i. i. d) and bursty traffic. We study the load-delay performance for different switch sizes and for traffic of different burst lengths. We also consider several traffic generation models for generating self-similar traffic and study the performance of both the algorithms under this type of traffic.*

Keywords: *High-speed packet switches, self-similar traffic, input-queued switches, packet schedulers, scheduling algorithms.*

Received December 4, 2004; accepted June 292005

1. Introduction

The replacement of copper with fibre has made significant amounts of bandwidth available, making it necessary for packet switches to improve their speed and efficiency of operation. Different architectures are commonly used to build packet switches. The fastest switches and routers usually transfer packets across the switching fabric in fixed size units that we shall refer to as "cells." Variable length packets are segmented into cells upon arrival, transferred across the switch fabric and then reassembled again before they depart. At the beginning of each cell time, a (usually centralized) *scheduler* selects a configuration for the switching fabric and then transfers cells from inputs to outputs. Using fixed sized cells simplifies the switch design, and makes it easier for the scheduler to configure the switch fabric for high throughput.

One well known architecture is the Output-Queued (OQ) switch, which has the following property: When a packet arrives, it is immediately placed in a queue that is dedicated to its outgoing port, where it will wait for its turn to depart. OQ switches have a number of appealing characteristics. They are work-conserving, they provide 100% throughput for all types of traffic, minimize queuing delays and can provide delay guarantees and various qualities of service [11]. However, OQ switches are not scalable due to their memory speed requirements and are not preferred normally. Hence, Input- Queued (IQ) switches, where the switch fabric runs at the line rate are commonly used. For such switches, the Virtual Output Queued (VOQ) architecture is commonly used [16], where each input port has N queues – one for each output port. However, such architecture requires a scheduler to decide the switch configuration during each time slot.

Scheduling algorithms are broadly classified into Maximum Size Matching (MSM) and Maximum Weight Matching algorithms (MWM). Maximum size matching algorithms provide high instantaneous throughput and delay performance but poor overall throughput. Whereas the maximum weight matching algorithms tend to give 100% throughput for independent and identically distributed uniform arrivals, but do not guarantee Quality of Service (QoS) and fairness. A scheduling algorithm is said to be fair if it provides equal chance for all the inputs, for being serviced in a given time slot. Of the many algorithms discussed in the literature, the iSLIP algorithm proposed by Nick Mckeown [9] achieves 100% throughput for uniform traffic but shows a low throughput for non-uniform traffic [11]. A variation of the same algorithm has been implemented in the Cisco12000 series routers which operate at a maximum speed of 360 Gbps [2].

This paper has been organized as follows. In section 2, we present a generic switch model. In section 3 we discuss several self-similar traffic models for generating traffic using the algorithms which have been tested. In section 4, we present the iSLIP and the

Modified Round Robin algorithms and discuss their relative merits. The simulation results are given in section 5.

2. Switch Model

We consider an NxN switch i. e., a switch, which has 'N' input ports and 'N' output ports, as shown in Figure 1. Input and output links to the switch are assumed to be of the same speed. Incoming cells are assumed to be of fixed size, and the channel-time is slotted, with the slot-size being equal to the cell transmission time. The switch slot may be exactly equal to the link-slot or may be higher, depending on the switch architecture and speed-up. The switch fabric is assumed to be non-blocking. The average amount of traffic at each input is called the input load and is measured in cells per time slot. We normalize input load to line rates i. e., a fully utilized input line (one cell per time slot) corresponds to a load of 1. The traffic at the input of a switch is said to be admissible, if no input load is larger than 1. An input traffic is said to be sustainable, if it can be transferred through switch.

The stationary and ergodic arrival process A_{ij} *(n)* at the input '*i*' for output '*j*', *1<= i <= N, 1<= j <= N,* at rate λ_{ij} is in general a discrete-time process of fixed size packets or cells. The set of all arrival processes is defined as *A (n) = [Aij (n), 1<= i <= N, 1<= j <= N]*. At the beginning of each time-slot, either zero or one cell arrives at each input, each containing an identifier that indicates which output '*j*', *1 <= j <= N* the cell is destined for. Each cell is placed in a FIFO queue (Virtual Output Queue) $Q_{ij.}$

A scheduling algorithm selects a matching 'M' between inputs and outputs, obtained by solving a bipartite graph-matching problem. This matching is a collection of edges, from the set of non-empty input queues to the set of outputs, such that each input is connected to at most one output and each output is connected to at most one input. The set of all departure process $[D_{ij}$ *(n)],* at rate μ_{ij} is also a discrete-time process, which is also stationary and ergodic.

Problem definition: We assume that the set of all arrival processes *A (n) = [A* $_{ij}$ *(n)]* satisfy the strong law of large numbers i. e,

$$\lim_{n \to \infty} A_{ij}(n)/n = \lambda_{ij}, \forall i, j = 1,2,3.....N \quad (1)$$

The average rate of the arrival process A_{ij} *(n)* at input '*i*' for output '*j*' is denoted by λ_{ij} and the cell arrival rate matrix is denoted by $\Lambda = [\lambda_{ij}]$. The admissibility criterion at each input port is defined as follows:

$$\sum_{i=1}^{N} \lambda_{ij} \le 1, \forall j = 1,2,3.....N \quad (2)$$

and at each output port by:

$$\sum_{j=1}^{N} \lambda_{ij} \le 1, \forall i = 1,2,3.....N \quad (3)$$

A switch operating under a scheduling algorithm is said to be rate stable if:

$$\lim_{n \to \infty} D_{ij}(n)/n = \lambda_{ij} \forall i, j = 1,2,3.....N \quad (4)$$

for any departure process *D (n) = [*D_{ij} *(n))]* with rate $\mu_{ij.}$

The scheduling algorithm should select a set of input output pairs with no conflicts such that each input is connected with at most one output and each output is connected with at most one input. In each slot, if input '*i*' is connected with output '*j*', a cell is removed from Q_{ij} and transferred to output '*j*' by properly configuring the non-blocking switch fabric.

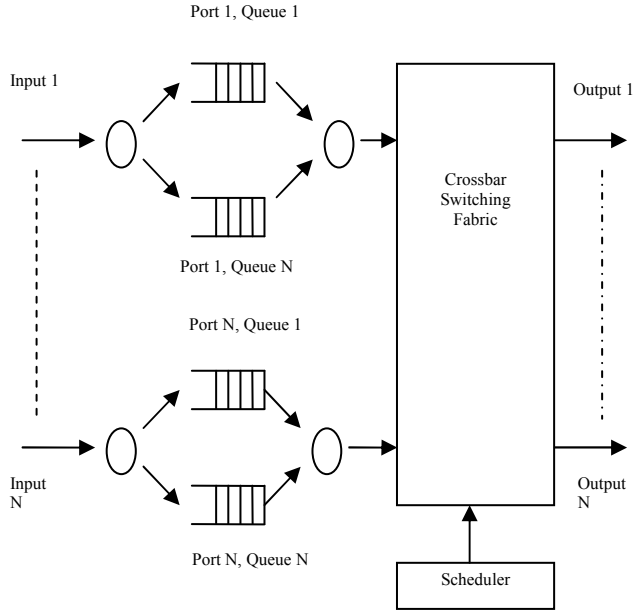

Figure 1. Generic model of the input-queued switch with virtual output queues.

3. iSLIP and Modified Round Robin Algorithms

3.1. iSLIP Algorithm

A fast and efficient algorithm, which overcomes the complexity problem of the Maximum Size Matching algorithms, is the iSLIP algorithm, proposed by Nick McKeown [9, 10]. This algorithm is based on the Parallel Iterative Matching (PIM) algorithm [1]. The main difference is that it uses a round robin schedule instead of a random schedule for figuring out which input or output is matched next. This is achieved by maintaining different pointers, one for each input port and output port. The iSLIP algorithm carries out the following steps in each iteration:

- *Request:* Each unmatched input sends a request to every output for which it has a queued cell.
- *Grant:* If an output receives any requests, it chooses the one that appears next in a fixed round robin schedule starting from the highest priority element. The output notifies each input whether or not its request was granted.
- *Accept:* If an input receives a grant, it accepts the one that appears next in a fixed round robin schedule starting from its own output pointer. The output pointer of the round robin schedule is incremented by one location after the accepted output.

The iSLIP algorithm has some advantages over the PIM approach. First of all, it performs better than PIM even under heavy load and is relatively easy to implement. It can achieve 100 % throughput in a single iteration for uniform traffic [9]. PIM has the problem that random selection is difficult to achieve. Another issue is that the iSLIP algorithm avoids the starvation problem by virtue of its round robin schedule. Performance results for the iSLIP algorithm are discussed in section 5.

3.2. Modified Round Robin Algorithm

This algorithm selects the first non-empty VOQ at each input in a round robin fashion and sends a request to the corresponding output for which that particular VOQ needs to send the cell. Each output may receive one or more multiple requests and services them in a round robin fashion. It selects one input port from the contenders and sends a grant to that input. Following this matching procedure the switch transfers the packet from the corresponding VOQ at that input. The algorithm involves two steps as follows:

- *Step 1*: Each input checks first for a non-empty VOQ among N FIFO queues maintained for each output, in a round robin order. The selection of the non-empty queue is based on the current position of the round robin pointer i. e., it chooses the input which occurs next to the one served in the previous cell time. After the selected input sends a request to the output for which that FIFO needs to send the packet, the round robin pointer is updated only after the request is granted in *Step2*.
- *Step 2*: Since each output may receive one or more requests, it serves them in the round robin order. Again the round robin pointer provides the grant to that input which occurs next in the round robin order after the input that was serviced in the previous cell time. The pointer will not be updated if there are no requests for that output.

In the modified round robin algorithm, only two stages of arbitration occur i. e., (1) A total of N requests are sent to the outputs, (2) The same number of grants are sent to each input. However, the iSLIP algorithm takes three steps to complete the same arbitration:

1. The inputs send a maximum of N^2 requests to outputs.
2. Outputs send grants to inputs.
3. Inputs send accept signals to outputs [4].

Hence, the modified algorithm takes only two operations per cycle to complete the arbitration and hence the data to be swapped for these exchanges is also reduced. For example, for a 256 x 256 switch this results in a saving of $256^2 - 256$ memory accesses to update the accept matrix when compared to the iSLIP algorithm. For switches running at Terabit speeds, this is a significant saving in terms of processor cycles and hence the switching speed will significantly increase. With 0.25μm CMOS technology, the arbitration time can be as small as 12ns. This allows a 256x256 switch with an incoming line bandwidth of 5 Gbps and an internal speedup of 2 to have an aggregate switch bandwidth of (256 x 5Gbps x 2) = 2.56 Tbps.

In addition, both algorithms do not suffer from the starvation problem, since the pointers at both the input and output ports move forward only after providing grants. All the inputs and outputs are given a fair chance for contention. At the worst case any input may have to wait for at the most (N-1) cycle times to get its turn for service and there is little chance for mutual deadlock. The next sections show the performance of the modified round robin algorithm under uniform, bursty and self similar traffic models.

4. Self-Similar Traffic Model

Traditionally, for the sake of mathematical tractability and simplicity, Poisson models which have exponential inter-arrival times have been widely used for studies in wide-area networks. However, it has been shown that the Poisson model is incapable of capturing the bursty nature of wide-area and local-area network traffic. The papers by Leland *et al.* [7] and Paxson [14] established through experimental results that most network traffic exhibits a fractal-like behaviour where we have long-term spikes, ripples and swells. Also, significant correlations are exhibited across arbitrarily large time scales, resulting in Long Range Dependence (LRD). In general, self similar traffic traces have the following main properties [3, 8, 12, 13, 17]:

1. *Slowly decaying variances*: Usually for most other processes, the variances of the sample mean decrease along with the sample size. However, in the case of self-similar process, the variance does not decrease as fast as sample size. Specifically, the variance exhibits hyperbolic decay instead of exponential decay.
2. *Long-range dependence:* This property is manifested as a non-summable autocorrelation

function. The summation of autocorrelations for various time scales will be infinity.

The self-similar nature of internet traffic has great impact on the following parameters of traffic flow:

1. *Queue Length Delay*: A heavy-tailed service distribution as in self-similar traffic results in packets spending more time in queues, resulting in more queue lengths and queuing delays.
2. *Packet Loss*: Significant loss in throughput is experienced because of large queue lengths and burstiness.
3. *Throughput*: Large queue lengths and packet losses result in reduced throughput.

There are several strategies for self similar traffic modelling [5, 6, 18]. For the present work, we generated self-similar traffic using the following methods:

1. The Fast Fourier Transform Fractional Gaussian Noise (FFT_FGN) method [5, 7].
2. The fast approximation to FFT_FGN method [5].
3. The FGN_Inverse DWT model [5].

We analyzed the traffic traces generated by the above methods using the following parameters and selected the five traces indicated in Table 1, which fitted the specifications given by us.

Table 1. Self-similar traffic traces.

	Mean	Variance	Hurst Parameter (from VT plot)	Hurst Parameter (from R/S plot)	Running Time (s)
1	9.98	18.69	0.68	0.7	160.
2	9.98	18.69	0.68	0.7	160.
3	9.92	18.58	0.54	0.61	9 E (-3)
4	9.92	18.58	0.54	0.61	9 E (-3)
5	9.92	18.68	0.78	0.76	9 E (-4)

In Table 1, trace 1 and trace 2 have a sample size of 20,502 samples and were generated by the FFT_FGN method. Trace 3 with a sample size of 25,785 samples was generated by the fast approximation to the FFT_FGN method. Trace 4 and Trace 5, each with a sample size of 15,769 samples was generated by the FGN_Inverse DWT method. We generated traffic by various methods and found that the performance of our algorithm remained the same irrespective of the method of traffic generation.

Figure 2 shows the autocorrelation plots for trace 1 with 20,502 samples generated by the FFT_FGN method and for trace 2 with 25,785 samples generated by the fast approximation to the FFT_FGN method. These two plots validate the accuracy of the methods used.

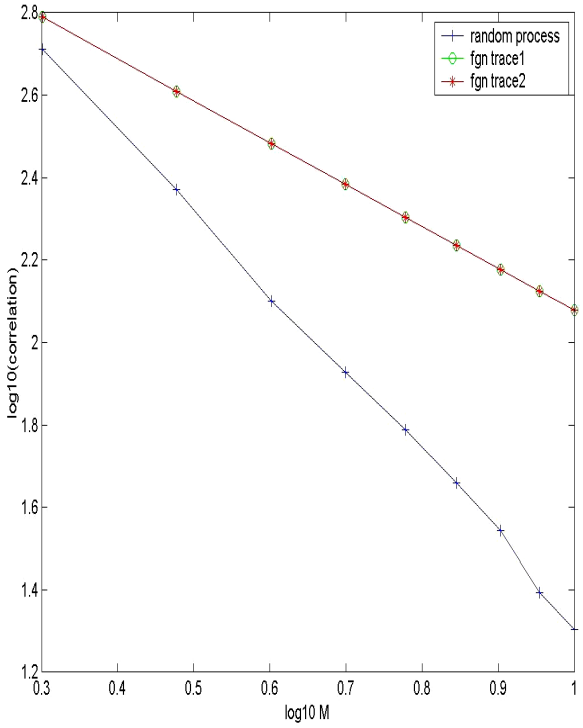

Figure 2. Autocorrelation plot for self similar series for 20,502 samples (FFT_FGN-trace 1 and trace 2).

5. Simulation Results

For the simulation study of the switch scheduling algorithms, we used a slotted-time simulator, written in ANSI-C and running under the Linux operating system. The parameters for simulation were chosen in a run-time configuration file. We varied the input load from 0 to 1.0 in steps of 0.1. For low values of the load we chose the simulation length as 1,00,000 time slots. As the switch approached saturation, we increased the length to as high as 1,000,000 time slots in order to get more accurate results. In the simulator, the user may select any combination of the elements-traffic: *input action*, *fabric*, *scheduling algorithm and output action* in a configuration file. The self-similar traffic generator module, which we coded in C, was interfaced to the TRAFFIC module. The results obtained were plotted using MATLAB.

Figure 3 shows the load-delay performance of the original iSLIP algorithm under various traffic models. The figure shows that the delay under Bernoulli traffic is less than the delay under Bursty model. The delay under self-similar traffic is the largest. For a load of 0.6, the Bernoulli traffic and bursty model have negligible delay, while the self-similar traffic model indicates a delay of around 380 time slots.

Figure 4 shows the behaviour of the MRR algorithm under the same traffic models. Again, considering a load of 0.6, the delay under the Bernoulli model is negligible while the delay under the bursty model is around 40 time slots, and under the self–similar model it is around 450 time slots. This shows that the performance under DRR algorithm is slightly less than

that under the iSLIP algorithm. However, the considerable savings in hardware because of less processor cycles has to be taken into account.

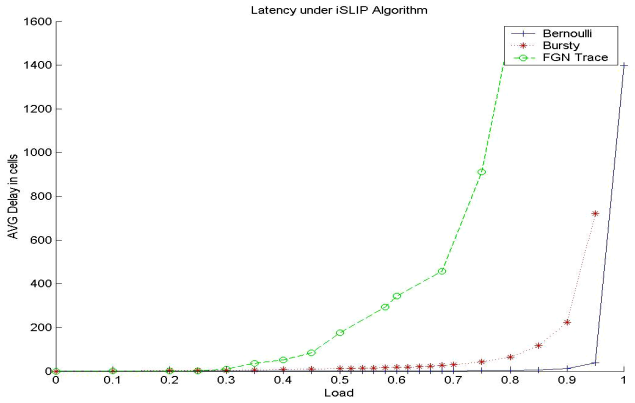

Figure 3. Load-delay performance of the iSLIP algorithm under various traffic models.

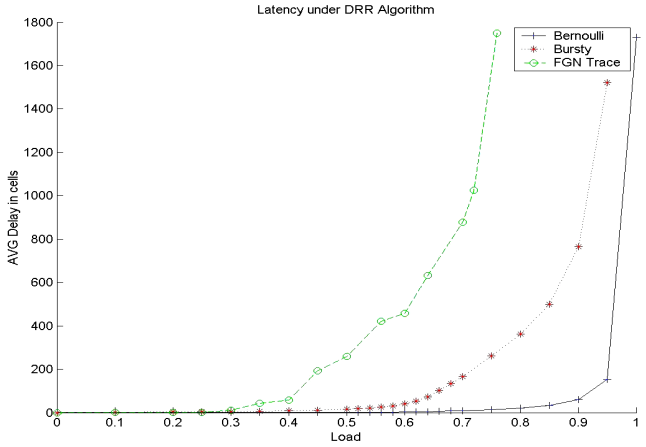

Figure 4. Load-delay performance of the MRR algorithm under various traffic models.

Figure 5 shows the effect of increasing burst length, for the bursty model. As the burst length increases, the average delay is shown to increase and hence, the throughput is found to decrease. With burst lengths of above 40, the performance is found to be the similar to that under the self-similar traffic model. The performance of the modified round robin algorithm for switches of different sizes is indicated in Figure 6. The load delay graph indicates that switches of size 8x8, 16x16, and 32x32 show an average delay of 400 time slots at loads of 0.58, 0.5 and 0.4, respectively. This leads to the conclusion that smaller switches show better performance.

Round robin arbitration of both the input and output ports tends to synchronize the positions of input and output pointers at consecutive cell times [9]. Hence, as an alternative we propose the following variations:

1. The round robin arbiter at each input and output port may be offset by 1, that is, each consecutive pointer points to a different VOQ. This will lead to de-synchronization of pointers and hence may give better performance results.
2. The order of round robin pointers may be reversed at the input and output sides, i. e., for an 8x8 switch, if first input pointer is at position 1, the pointer for the first output may start at position 8, and so on.

Our future work includes, testing the above modifications and VLSI implementation of the above algorithms on FPGA boards using fast programmable priority encoders for arbitration.

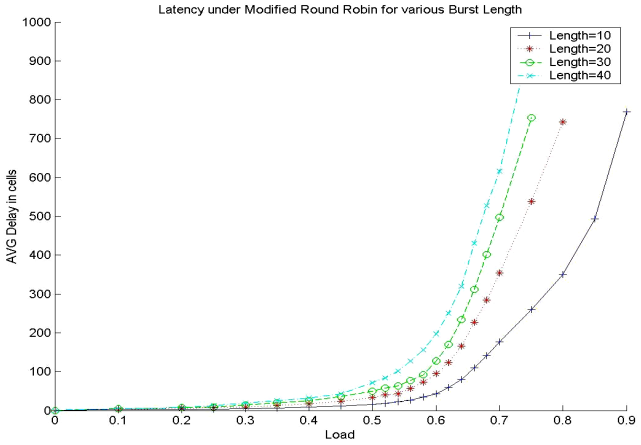

Figure 5. Load-delay performance of MRR algorithm under different burst lengths.

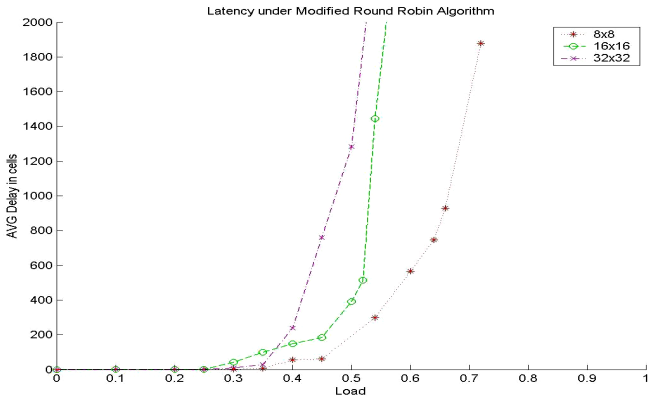

Figure 6. Load-delay plot under modified round robin algorithm for different switch sizes.

6. Conclusion

Research is going on at a rapid pace for designing scheduling algorithms for high speed packet switches. In this paper we have discussed the performance of the modified round robin algorithm which performs scheduling in two steps for each iteration and compared it with the well-known iSLIP algorithm. The load-delay performances of the two algorithms have been studied for different traffic models with particular emphasis on the self-similar traffic model. The simulation results for switches of different sizes are presented. The behaviour of the algorithms under traffic of different levels of burstiness has also been discussed. Further, it has been well established that the

performance of a high speed packet switch can be improved by introducing speed-up in the switch fabric [15]. Hence, we believe that the modified round robin algorithm with integral speed-up factors may well approach the performance of an ideal Output-Queued switch.

References

[1] Anderson T., Owicki S., Saxe J., and Thacker C., "High Speed Switch Scheduling for Local Area Networks," *ACM Transactions on Computer Systems*, vol. 11, no. 4, pp. 319-352, November 1993.

[2] Cisco, "Cisco 12000 Gigabit Switch Router," *Product Overview*, www.cisco.com, April 2000.

[3] Cox D. R., "Long-Range Dependence: A Review," *in Statistics: An Appraisal*, in David H. A. and David H. T. (Eds), The Iowa State University Press, Ames, Iowa, pp. 55-74, 1984.

[4] Gupta P. and Mckeown N., "Designing and Implementing a Fast Crossbar Scheduler," *IEEE Microelectronics*, vol. 19, no. 1, pp. 20-29, 1999.

[5] Jeong H. D. J., McNickle D., and Pawlikowski K., "Fast Self-Similar Tele-Traffic Generation Based on FGN and Wavelets," *in Proceedings of IEEE International Conference on Networks (ICON'99)*, Brisbane, Australia, pp. 75-82, 1999.

[6] Lau W. C., Erramilli A., Wang J. L., and Willinger W., "Self-Similar Traffic Generation: The Random Midpoint Displacement Algorithm and its Properties," *in Proceedings of IEEE ICC'95*, Seattle, USA, pp. 466-472, 1995.

[7] Leland W., Taqqu M., Willinger W., and Wilson D., "On the Self-Similar Nature of Ethernet Traffic (Extended Version)," *IEEE/ACM Transactions on Networking*, vol. 2, no. 1, pp. 1-15, February 1994.

[8] Mandelbrot B. B. and Van Ness J. W., "Fractional Brownian Motions, Fractional Noises and Applications," *SIAM Review 10*, pp. 422-437, 1968.

[9] Mckeown N., "Scheduling Algorithms for Input-Queued Cell Switches," *PhD Thesis*, University of California, Berkeley, USA, 1995.

[10] Mckeown N., Anantharam, V., and Walrand J., "Achieving 100% Throughput in an Input-Queued Switch," *in Proceedings of INFOCOM*, pp. 296- 302, 1996.

[11] Mekkittikul A., "Scheduling Non-Uniform Traffic in High Speed Packet Switches and Routers," *PhD Thesis*, Stanford University, 1998.

[12] Park K., Kim G., and Crovella M., "On the Effect of Traffic Self-Similarity on Network Performance," *in Proceedings of the SPIE International Conference on Performance and Control of Network System*, pp. 296-310, 1997.

[13] Park K. and Willinger W., *Self Similar Network Traffic: An Overview Self-Similar Network Traffic and Performance Evaluation*, John Wiley & Sons, New York, 2002.

[14] Paxson V. and Floyd S., "Wide-Area Traffic: The Failure of Poisson Modeling," *IEEE/ACM Transactions on Networking*, vol. 3, no. 3, pp. 226-244, June 1995.

[15] Prabhakar B. and Mckeown N., "On the Speedup Required for Combined Input and Output Queued Switching," *Technical Report*, CSL-TR-97-738, Computer Lab, Stanford University, 1997.

[16] Tamir Y. and Frazier. G. "High Performance Multi-Queue Buffers for VLSI Communication Switches," *in Proceedings of the 15th Annual Symposium on Computer Architecture (ISCA)*, pp. 343-354, June 1988

[17] Taqqu M. S., "A Bibliographical Guide to Self-Similar Processes and Long-Range Dependence," *Dependence in Probability and Statistics*, Eberlein E. and Taqqu M. S. (Eds), Birkhauser, Basel, pp. 137-165, 1985.

[18] Taralp T., Devetsikiotis M., and Lambadaris I., "Efficient Fractional Gaussian Noise Generation Using the Spatial Renewal Process," *in Proceedings of the IEEE International Communications Conference (ICC'98)*, Atlanta, 1998.

Shanmugam Arumugam obtained his PhD degree from Bharathiar University, Coimbatore, India. He worked at PSG College of Technology, Coimbatore from 1979 to 2004. During his career, he has undertaken several consultancy and sponsored research activities. Currently, he is a principal, Bannari Amman Institute of Technology and Science, Coimbatore, India. His major sponsored research activities include modelling, characterization and synthesis of ASICs for fuzzy based traffic polling in broadband networks using ATM mode, and modernization of fibre-optic laboratory, both from AICTE, New Delhi, India. He has more than 75 papers to his credit in national and international conferences, including 13 journal publications. He has so far supervised four PhD candidates, and he is now supervising six PhD candidates.

Shanthi Govindaswamy obtained her BE degree in electronics and communication from PSG College of Technology and her ME in applied electronics from Government College of Technology, Coimbatore, India in 1988 and 1992, respectively. She is a lecturer and research scholar in Electronics and Communication Department, PSG College of Technology, Coimbatore, India. She has 13 years of teaching experience and has been at PSG College of Technology since 2001. Currently, she is pursuing her research in the area high-speed packet switches. Her research interests include congestion control in Broadband networks and schedulers for high-speed packet switches.