EFFICIENT SCHEDULING OF VARIABLE-LENGTH IP PACKETS ON HIGH-SPEED SWITCHES*

Ge Nong, Mounir Hamdi and Khaled Ben Letaief
Department of Computer Science
The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
Email: nong@ieee.org, hamdi@cs.ust.hk and eekhaled@ee.ust.hk

Abstract

ATM switches have been proposed as the switching fabric cores of many high-performance IP switches. In this paper, we present an efficient algorithm called IP-PIM for scheduling variable-length IP packets on these switches and compare its performance with the original *parallel iterative matching* (PIM) ATM cell scheduling algorithm. The mean IP packet delays using both the PIM and the IP-PIM scheduling algorithms are analyzed using queueing analysis and extensive simulations. Our results demonstrate that the packet level scheduling such as the IP-PIM could be a potential way to significantly improve the performance of PIM switches, especially in the context of IP networks.

1 Introduction

Recently, there is an urging need for the design and analysis of high-speed IP switches that can handle the undergoing traffic explosion in the Internet. Several methods for high-speed IP switching have been proposed. In particular, the IP-over-ATM solution is receiving the most attention [3, 7]. However, ATM switches have been designed to carry ATM cells. As a result, one of the critical issues that need to be addressed when used as the core of high-performance IP switches is how to efficiently schedule *variable-length* IP packets onto these core ATM switches.

In this paper, we specifically address and present a solution to this problem on ATM switches with multiple input queues [2]. In particular, we present efficient scheduling algorithms for variable-length IP packets on these switches and compare their performance with the original parallel iterative matching (PIM) ATM cell scheduling algorithm [2].

*This research work was supported in part by the Hong Kong Research Grant Council under the Grant RGC/HKUST 100/92E.

The outline of this paper is as follows. In Section 2 we first briefly describe the PIM switch architecture and then propose our IP Parallel Iterative Matching algorithm called IP-PIM. The mean IP packet delays for both the PIM and the IP-PIM scheduling algorithms are analyzed using an analytical queueing model in Section 4 and extensive simulations in Section 5, respectively. Some conclusions are drawn in Section 6.

2 The Switch Model and PIM Scheduling.

In this section, we describe the PIM switch architecture and present our IP-PIM scheduling algorithm.

2.1 The Switch Model

The 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 maintains a seperate queue for cells destined for each output. The first cell in each queue can be selected for transmission across the switch in each time slot, provided that the selected cells satisfies that (i) at most one cell can be transmitted from an input; and (ii) at most one cell can be received by an output.

To schedule the transmission of the head-of-line cell in each queue to their destined output, Anderson *et al.* [2] proposed an algorithm, called *parallel iterative matching* (PIM)[1] to accomplish this task. A number of high-speed switches have been built using the PIM or PIM-like scheduling algorithms [2, 5]. As a result, we selected the PIM algorithm as a concrete example for carrying out our discussion. However, while our focus in this paper is on PIM scheduling, our method

[1]Here we don't give the detailed description for the PIM algorithm, interested readers may refer to [2].

and results can be easily applied to other scheduling algorithms as well.

3 The IP Parallel Iterative Matching (IP-PIM)

We have to note that the PIM scheduling algorithm has been designed for the transmission of ATM cells, not IP packets. The traffic in IP networks are usually characterized by the concept of *flows* [7]. When transmitting IP packets onto these switches, an IP packet is first segmented into multiple ATM cells. Then, these ATM cells are transmitted by the core ATM switch. Consequently, the ATM cells belonging to the same IP packet are destined to the same output port.

Motivated by the above observation, and especially the fact that traffic in IP switches are bursty in nature, we propose a variation of the above PIM algorithm and denote it as the IP-PIM scheduling algorithm—which performs scheduling at the IP packet level rather than at the cell level. Using the IP-PIM scheduling algorithm, the matching of inputs with outputs are no longer in units of cells but rather in IP packets. As mentioned previously, each IP packet is segmented into a sequence of variable number of ATM cells. We denote this sequence of variable number of ATM cells as a **burst of cells**. In particular, each burst of cells participates during the scheduling only when the burst's head cell is at the HOL position. The burst that the HOL cell belongs to is referred to as the HOL burst in this paper. Once a matching between an input and an output is set up to transmit the head cell of a HOL burst, that matching will be *valid* until all cells of this HOL burst have been transmitted. With these denotations, our burst-based IP-PIM scheduling algorithm is formally given below.

1. Keep all the matchings in the last time slot which are still being *valid* (e.g., the last cell of the burst did not get transmitted yet) at the current time slot unchanged.
2. Each unmatched input sends a request with a given priority to every output for which it has a buffered *head cell of the HOL burst* .
3. If an unmatched output receives any requests, it chooses the *highest priority* one to grant.
4. If an input receives any grants, it chooses the *highest priority* one to accept and notifies that output.
5. Iterate steps 2-4 until a maximal matching has been found or until a fixed number of iterations have been performed.

Step 1 of the IP-PIM guarantees that all cells of a burst (IP packet) will be transmitted across the switch in a (continuous) sequence of time slots without any interruption. Moreover, the probability that a large number of the HOL bursts complete their transmissions at the same time slot is very low especially when the traffic load is high, i.e., the expected number of matchings in the last time slot to be *invalid* at the current time slot are small. As a consequence steps 2-5 are performed only on a subset of the inputs and outputs which is one major advantage of burst (IP packet) level scheduling over cell level scheduling. Hence, less iterations would be required to find a *maximal* matching among the smaller set of unmatched inputs/outputs. The priority in steps 2-4 can be defined elaborately. However in this paper, for comparison purposes, we also use a *random select* policy as which used in PIM algorithm.

To investigate the effects of iterations on the performances of PIM and IP-PIM scheduling algorithms, we proceed to build a queueing model for analyzing the mean delays of IP packets scheduled by both the PIM and the IP-PIM algorithms with only 1 iteration. One of the main reasons for comparing the performances of PIM and IP-PIM scheduling algorithms is to illustrate the fact that when using ATM switches as the core of an IP switch, we have to rethink the design of scheduling traffic onto these switches.

4 Queueing Model

The bursts of ATM cells corresponding to IP packets that arrive at each input port are modelled by a 2-states Markov Modulated Bernoulli Process (2-MMBP) [1] shown in Fig. 1. The traffic source modelled by a 2-MMBP alters between state 0 and 1 with probabilities r_i and s_i and thus the mean dwell times in the two states are $1/r_i$ and $1/s_i$, respectively. Furthermore, the traffic source generates a cell with probability $q_i(p_i, resp.)$ if it is currently in state 0(1, resp.). In our case, we let $q_i = 0$ and $p_i = 1$ to model the traffic arrivals at each input by an ON-OFF process. Specifically, a *burst* (IP packet) consists of the cells generated by the consecutive active states. All the cells of a burst are destined to the same output and the output is denoted as the burst's destination.

We focus our attention on finding the *mean burst delay*, that is, the mean IP packet delay. The *burst delay* is defined as the time interval between the appearance of the first cell of the burst at the switch's input link and the arrival of the last cell of the burst at the switch's

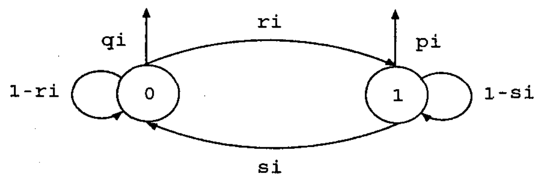


Figure 1: The diagram for the 2-MMBP traffic model.

output link.

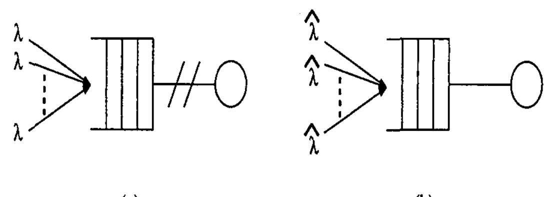
The switch size and the buffer of input queues are assumed to be infinite. In addition, the traffic arriving at each input are *i.i.d* and distributed over all outputs uniformly, namely, the traffic is homogeneous. Moreover, we assume that at any time slot the requests granted by the free outputs address the free inputs uniformly once the system attains steady state. With these assumptions, it can be proven[2] that at each time slot the number of granted requests to any free input is governed by a Poisson distribution with the mean arrival rate given by $\lambda_{ip} = -\ln\left[\frac{\tau(\lambda-1)}{\lambda\tau-\tau-\lambda}\right]$, where λ is the mean traffic load of the system. Consequently, the distribution of grants to a free input is given by

$$\Pr\{A_j = k|IP-PIM\} = e^{\lambda_{ip}}\frac{\lambda_{ip}^k}{k!},$$

where $A_j = k$ denotes the event that the number of grants to the input j is equal to k. For the 1 iteration PIM scheduling algorithm, $\Pr\{A_j = k|PIM\}$ is essentially a Poisson distribution with a mean arrival rate of $\lambda_p = -\ln(1-\lambda)$, i.e.,

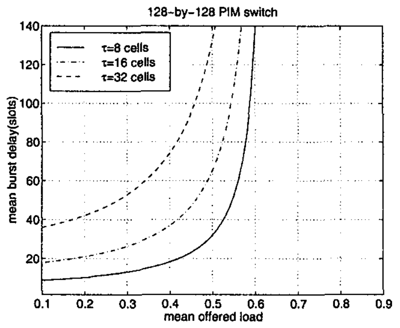
$$\Pr\{A_j = k|PIM\} = e^{-\lambda_p}\frac{\lambda_p^k}{k!}.$$

Due to the homogeneous property of the system studied here, the mean burst delay of the system will be the same as the ones at each input queues. This enforces us to study the queueing delay of the system by exploring the queueing behavior of the burst at any queues. From the view points of all queues for an output, a free output can be regarded as a server which maybe on vacation as illustrated by Fig. 2. The queueing systems with the servers that may be interrupted have been formerly studied such as in [4] and the results established there provide potential exact solutions to our queueing model. However, instead of applying the involved analysis procedures to obtain the exact mean burst delay, we approach this objective by approximating the original non-work-conserving system by a 2-MMBPs/D/1/∞ queue[3].

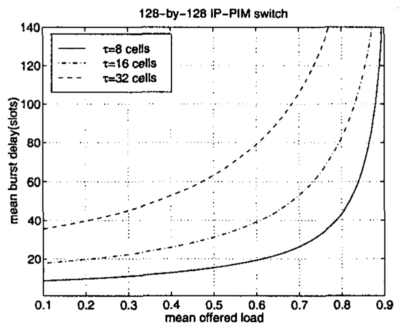


(a) (b)

Figure 2: The illustration of (a) the 2-MMBPs/G/1/∞ and (b) the 2-MMBPs/D/1/∞ queueing models.



(a) PIM scheduling



(b) IP-PIM scheduling

Figure 3: The mean burst delays for the 1-iteration PIM and IP-PIM scheduling.

Having transformed the original non-work-conserving

[2]We omit the proof here for the sake of the limited space.

[3]The detail of how to construct the 2-MMBPs/D/1/∞ queue is omitted due to the limited space again.

system into a 2-MMBPs/D/1/∞ queue which is essentially a multiplexer, a lot of available results on the multiplexers can be used immediately. The dynamic flow analysis was applied in [6] to get a closed form computing formulas for individual sojourn delay of an ATM switch which is modeled as an H-MMBPs/D/1/∞ queueing system. Let L denote the mean queue length, the main results established there [6] show that

$$L = \frac{2\hat{\lambda} - \hat{\lambda}^2 - \sum_{i=1}^{m} \hat{\lambda}_i^{\,2} + 2\sum_{i=1}^{m} h_i}{2(1-\hat{\lambda})},$$

where,

$$\begin{aligned} h_i &= \Big(\frac{1}{\hat{r}_i + \hat{s}_i}\Big)\Bigg[\frac{\hat{r}_i\hat{p}_i^{\,2} + \hat{s}_i\hat{q}_i^{\,2} - \hat{r}_i\hat{s}_i(\hat{p}_i - \hat{q}_i)^2}{\hat{r}_i + \hat{s}_i} \\ &\quad -\hat{\lambda}_i(1 + \hat{\lambda}_i - \hat{\lambda}) + (1-\hat{\lambda}) \\ &\quad \cdot \frac{\hat{r}_i\hat{p}_i(1-\hat{p}_i) + \hat{s}_i\hat{q}_i(1-\hat{q}_i) + \hat{r}_i\hat{s}_i(\hat{p}_i - \hat{q}_i)^2}{\hat{r}_i(1-\hat{p}_i) + \hat{s}_i(1-\hat{q}_i)}\Bigg], \end{aligned}$$

$$\hat{\lambda}_i = \frac{\hat{r}_i\hat{p}_i + \hat{s}_i\hat{q}_i}{\hat{r}_i + \hat{s}_i}.$$

All the parameters in hats are corresponding to those in the 2-MMBPs/D/1/∞ queueing system, which are derived from the parameters of the original 2-MMBPs/G/1/∞ queueing system.

The mean cell delay of the original system is thus given by (Little's law)

$$D_{cell} = \frac{L}{\lambda}.$$

The mean burst delay is derived from the mean cell delay as follows. Let $W(\ell, x)$ be the probability that the waiting delay[4] of a burst with size of ℓ cells, we have

$$D_{cell} = \sum_{\ell=0}^{\infty}\sum_{x=0}^{\infty} W(\ell,x)(\ell x + \ell)/\ell = 1 + \sum_{\ell=0}^{\infty}\sum_{x=0}^{\infty} W(\ell,x)x,$$

$$D_{burst} = \sum_{\ell=0}^{\infty}\sum_{x=0}^{\infty} W(\ell,x)(x+\ell) = \tau + \sum_{\ell=0}^{\infty}\sum_{x=0}^{\infty} W(\ell,x)x.$$

Elementary algebra operations on the above two equations leads to $D_{burst} = D_{cell} + \tau - 1$.

Fig. 4.(a) and (b) show the mean burst delay using the above queueing model for the 1 iteration PIM and IP-PIM as function of the mean traffic load by varying the mean burst size τ from 8 to 32 cells/burst. The switch size is set to 128. It is indicated by this figure that the improvement on maximum throughput increases as the mean burst size increases (from 0.63 to $\geq$0.88). Compared to the maximum throughput of around 0.88 achieved by the 2-iteration PIM [8], the performance improvement of IP-PIM over PIM is significant. These results illustrate the importance of designing scheduling algorithms specifically tailored for the transmission of IP packets rather than relying on the ATM cells scheduling algorithms. In next section, we continue to investigate the effects of the number of iterations on the performance of IP-PIM by simulations.

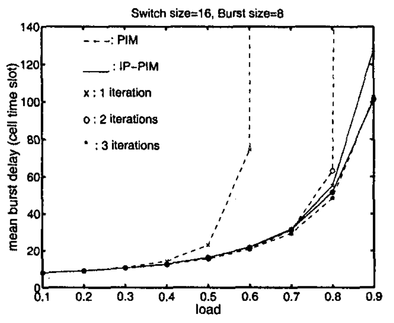
[4]The waiting delay of a burst is the time interval between the echoes of the burst's first cell arrived and the service begins.

5 Simulation Experiments

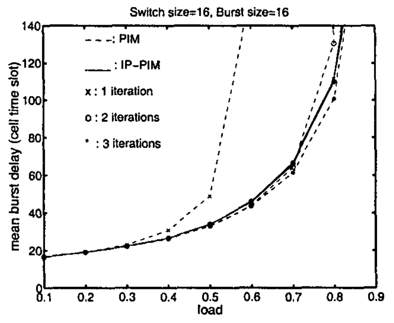
To investigate how much the original PIM algorithm can take advantage from the burst-based scheduling concept, a series of simulations were carried out with the traffic being generated by the 2-MMBP model at each input port. The bursts' destinations are uniformly distributed over all outputs.

Fig. 4(a)-(b) show the mean burst delays as a function of offered traffic load with a mean burst length of 8 or 16 cells, for a 16×16 switch. In these figures, the line types distinguish the scheduling strategies; the markers, i.e., x-mark, circle and star indicate the maximum iteration number of the scheduling strategies. For example, the solid lines with x-mark show the mean burst delays for traffic scheduled by the IP-PIM.

An interesting observation drawn from these figures is that the performance of the IP-PIM scheduling algorithm is insensitive to the iteration number, especially when the mean burst length is large. This is one major advantage over the PIM scheduling algorithm as it does not require expensive high-performance microprocessors to be executed on time. We can see that the various curves for the IP-PIM scheduling algorithm are almost clustered together when the burst length is 16 cells, which indicates that one iteration is enough for the IP-PIM scheduling algorithm to find a (nearly) *maximal* matching under any traffic load. As we can see from the figures, the curves of 1-iteration of the IP-PIM scheduling algorithm lie between the curves of 2-iteration and 3-iteration PIM scheduling algorithms while being very close to the one of the 3-iteration PIM scheduling algorithm. More significant is that the 1-iteration IP-PIM scheduling algorithm has almost the same throughput as the 3-iteration PIM scheduling algorithm, which can be seen from the figures where the curves of 1-iteration



(a) 16 − *by* − 16 switch, τ=8.



(b) 16 − *by* − 16 switch, τ=16.

Figure 4: The mean burst delays as functions of the mean traffic loads for the 1-iteration PIM and IP-PIM scheduling algorithms.

IP-PIM scheduling and 3-iteration PIM scheduling increase dramatically when the traffic load reaches 0.8. As a result, the 1-iteration IP-PIM scheduling has almost the same maximum throughput as the 3-iteration PIM scheduling. Actually, as we observed from the simulation results, the maximum throughput of 1 iteration IP-PIM is approximately the maximum traffic load for the given mean burst size, i.e., $\tau/(\tau+1)$. As a consequence, the packet level scheduling concept can greatly improve the performance of PIM scheduling algorithm under bursty IP traffic. This advantage of packet level scheduling can be explored to design large PIM switches operating at extremely high speeds where the time slot is too short to execute the PIM scheduling with multiple iterations.

6 Conclusion

Both the analysis and the simulation results indicate that our proposed IP-PIM scheduling outperforms the original PIM scheduling significantly in terms of the mean IP packet delay especially for very bursty traffic. Moreover, the suitability of our proposed algorithm is more significant when the link speeds become higher and the traffic become burstier which what we expect from future IP networks.

The queueing model developed in this paper for analyzing the mean burst delays of both the PIM and the IP-PIM scheduling algorithms provide a novel way to solve the difficult problem of performance analysis on PIM algorithms [2, 8].

References

[1] A. Adas. Traffic models in broadband networks. *IEEE Communications Magazine*, 35(7):82–89, July 1997.

[2] T. E. Anderson, S. S. Owicki, J. B. Saxe, and A. P. Thacker. High-speed switch scheduling for local-area networks. *ACM Transactions on Computer Systems*, 11(4):319–352, Nov 1993.

[3] S. Keshav and R. Sharma. Issues and trends in router design. *IEEE communications magazine*, 36(5):144–51, May 1998.

[4] K. Laevens and H. Bruneel. Delay analysis for discrete-time queueing systems with multiple randomly interrupted servers. *European Journal of Operational Research*, 85(1):161–177, Aug. 1995.

[5] N. McKeown, M. Izzard, A. Mekkittikul, W. Ellersick, and M. Horowitz. Tiny tera: a packet switch core. *IEEE Micro*, 17(1):26–33, Jan.-Feb. 1997.

[6] W. C. Miao and J. F. Chang. Individual sojourn delay analysis of an ATM switch receiving heterogeneous Markov-modulated Bernoulli processes under FIFO and priority service disciplines. *IEICE Transactions on Communications*, E80-B(5):712–25, May 1997.

[7] P. Newman, G. Minshall, and T. Lyon. IP switching-ATM under IP. *IEEE/ACM Trans. on Netw.*, 6(2):117–29, Apr. 1998.

[8] G. Nong, J. K. Muppala, and M. Hamdi. Analysis of non-blocking ATM switches with multiple input queues. *IEEE/ACM transactions on networking*, 7(1):60–74, Feb. 1999.