Performance Evaluation of Parallel Packet Switches using the Proportional Bounded Copy Multicast Scheme

Jyun-Ming Chen, Guan-Hong Jhou, and Woei Lin
Department of Computer Science, National Chung-Hsing University
{s9256018,s9456009,wlin} @cs.nchu.edu.tw

Abstract

A parallel packet switch (PPS) has been a promisingly practical scheme to build high-capacity packet switches. But most of the feasible researches presented so far assume the input traffic pattern is unicast-only. It is known that, for multicast traffic, an N-port PPS can mimic a multicast FCFS output-queued switch with theoretic speedup requirement $2\sqrt{N}+1$. It would face a scalability problem as the number of ports increases. To make multicast PPS practical, in this paper, we develop a method of multicasting named the proportional bounded copy multicast scheme (PBC).

Our proposal uses a bound to limit the multicast replication in the input where the value of bound is adjusted periodically by a proportional scheme. Since it not only induces efficient memory usage but also provides fully distributed approach. The proposed algorithm can achieve improved performance than other traditional multicast schemes in terms of the cell loss probability under overall traffic load.

Index Terms – PPS; multicast; proportional bounded copy multicast.

1. Introduction

The speed of memory in the traditional packet switches at least runs equal to external line rate. Driven by increasing Internet traffic, external line speeds, increasing from OC192 (40 Gbps), OC768 (160 Gbps), or even OC3072 (160 Gbps), make it become difficult to buffer packets as fast as they come using conventional packet switches. In attempt to overcome this problem, the parallel packet switch (PPS) was proposed by Iyer et al. [1], [2]. Figure 1 shows an example of an $N \times N$ PPS architecture with a center stage which consists of K output-queued (OQ) switches. In each demultiplexer, packets arriving at rate R are load-balanced over K identical lower speed switches, operating at rate $r = S\times(R/K)$ for an internal speedup of S, then recombined at a multiplexer before departing from the system.

Recently, most of the theoretical results of PPS presented so far assume the input traffic pattern is unicast-only [1], [2], [3] and [4]. In order to support continuous growth in the demand for multicast-dependent services, *the bounded copy strategy* has been proposed in [5].

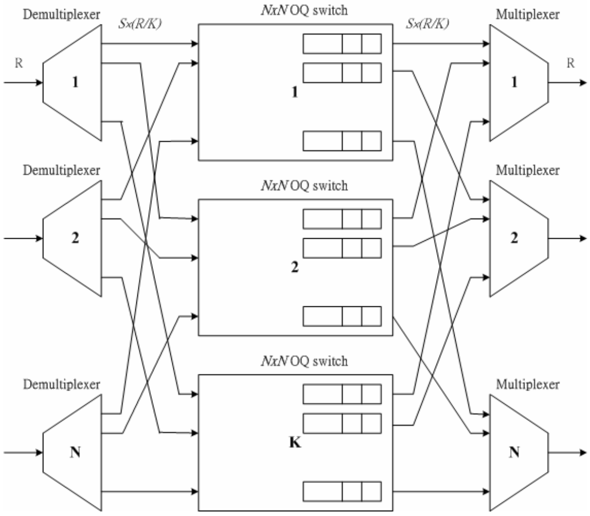

Fig. 1: Architecture of an $N \times N$ PPS.

According to the strategy, the speedup requirement of an N-port *bufferless* PPS (i.e., with no coordination buffers in either the demultiplexers or the multiplexers as in [1]) increases to $S > 2\sqrt{N}+1$. Obviously, for a large N of switch size, the speedup requirement may become impracticably large. This induces scalability problems due to the resulting increase of speedup and the need for a centralized scheduling algorithm, which leads to large communication complexity and inefficient utilization of memory in center-stage switches.

Another approach, named *SDME PPS*, has been proposed in [6] to support multicast traffic in PPS. In an *SDME PPS*, each central switch-layer is appended with a $1 \times N$ multicast module where each multicast cell is duplicated and transmitted to the set of its multicast destinations. Although an *SDME PPS* can support multicast traffic with a relative queueing delay bound, it requires double size of coordination buffer at each multiplexer. Furthermore, if the multicast load is low, many of the multicast modules will be idle and wasted.

A PPS with each demultiplexer and multiplexer containing a fixed size $N \times K$ coordination buffer, named as *buffered* PPS, can allow a fully distributed implementation and no speedup requirement under unicast traffic [4]. For such reason, we focus our work on multicast switching in a *buffered* PPS architecture. We investigate the traditional multicast PPS schemes and propose a novel algorithm the proportional bounded copy scheme (PBC for short) operated in demultiplexers to support multicasting in PPS.

The rest of this paper is organized as follows. Section 2 introduces terminologies and traditional multicast schemes. Section 3 describes our proposed multicast scheduling algorithm. Section 4 contains the traffic models used for simulations and results evaluation. The conclusion follows in Section 5.

2. BACKGROUND

Buffered PPS: A PPS contains a fixed size $N \times K$ of coordination buffer in each demultiplexer and multiplexer where a coordination buffer is configured as *K* equal size of FIFOs [4].

Fanout: The number of outputs that a multicast cell is destined to is called the fanout and is denoted by *fanout*.

Copy Multicast: A multicast cell can be copied to create multiple unicast cells at the input of a packet switch and each unicast cell can be switched individually.

Fanout Multicast: The input of the switch delivers a multicast cell into the switch fabric just once. The central switch layers take care of delivering the cell to each output.

The copy multicast and the fanout multicast here are considered as the traditional multicast schemes which have been proposed in the past. The survey and issues involved in their design can be found in [5] and [7].

3. THE PROPORTIONAL-BOUNDED COPY MULTICAST SCHEME

In this section, we start to elaborate on our algorithm. Except the problem of scheduling multicast cells from demultiplexers to center-stage switches, the rest of the method for scheduling unicast cells is the same as in [4]. The main goal of our proposed algorithm is to utilize both buffering capabilities of demultiplexers and of central switch layers to handle the traffic expansion from multicast replication. The PBC scheme comprises two parts which are both distributed approaches. The subsection 3.1 shows the one part of algorithm which is used to maintain a parameter *B* which is the maximum number of copies made from a given multicast cell. The subsection 3.2 depicts the other part of mechanism which describes how to dispatch incoming cells into coordination buffers of a demultiplexer.

3.1. MAINTAINING BOUNDED COPY VALUE

Like the main idea of the bounded copy strategy, our algorithm attempts to limit the number of copies made from a multicast cell by a bound. However, to attain a certain level of performance, the size of bound may not be the same in the various architectures. For such reason, the value of bound here is measured and determined over time intervals. We define *B* as the bounded copy value which is the maximum number of copies that can be made from an incoming multicast cell. And a parameter *H* is defined as a time interval, in units of internal time slot, for which the schedule is generated. When the schedule is generated, each demultiplexer determines *B* respectively by following equation

$$B = \lfloor N \cdot (1 - Q^{\alpha}) \rfloor .$$

Here *N* is the switch size, *Q* is the buffer occupation factor, and *α* is a parameter. The buffer occupation factor *Q* is calculated by

$$Q = \frac{L_{\max}}{N},$$

where L_{max} is the length of a queue which currently has the maximum length in a demultiplexer. Because the size of each FIFO in a demultiplexer is *N*, *Q* can represent the degree of buffer occupation in the demultiplexer. To find L_{max}, each FIFO was assigned a counter to record its queue length. The counter increases as cells are buffered into, but it decreases as cells are sent to central switches in every internal time slot.

3.2. CELL DISPATCHING METHOD

The PBC distributes incoming cells or packets to central switch layers using round robin by flow at demultiplexers. Flows can be identified as a unique source and destination address pair in IP packet or ATM cell. For the reason to spread traffic uniformly across the center-stage switches, each demultiplexer will maintain a set of separate round robin pointers [4]. Figure 2 depicts the definition of parameters and cell dispatching mechanism.

The separate round robin pointers denoted as srr[ ] in figure 2 is dedicated to each distinct output port, respectively. A cell destined for an output index is directed into its corresponding FIFO by the corresponding round robin pointer. For example, a cell to output port *i* will be assigned to FIFO[*k*] if srr[*i*] = *k*. After that srr[*i*] increases to *k*+1 in the round-robin manner.

Now we give a detailed description of our proposed cell dispatching mechanism. Within each demultiplexer, cells are received and classified. If an incoming cell is distinguished as unicasting, it will be queued into an FIFO by flow which is identified as *i* in figure 2.

However, when an incoming cell is multicasting, the *distribute* function will firstly distribute all fanout of this multicast cell as evenly as possible between *B* copies. For instance, if the fanout is 10 and current *B* is 3, there will be 3 copies (denoted as *m* in figure 2) made from the multicast cell, and the fanout of these copies is 4, 3, and 3 respectively. After that the 3 copies will be dispatched to FIFOs according to the separate round robin scheme. Thus, we can derive that each copies (denoted as $C[j]$, $j \in \{1,...,m\}$) will contain at most $\lceil fanout / B \rceil$ destinations where $m = B$ if $fanout \geq B$, or $m = fanout$ if $fanout < B$.

We adopt the bounded copy value *B* to bound the maximum copies from a multicast cell, which makes it possible to mitigate the traffic expansion from multicast replication and also provides more efficient buffer utilization in the demultiplexer. Furthermore, the *distribute* function can achieve load-balancing to center-stage switches by distributing the fanout of a multicast cell evenly over each copies.

//Define the following parameters:	
// *i*	Destination port index.
// srr[ ]	Separate round robin pointers.
// *K*	Number of center-stage switches.
// *B*	Bounded copy value.
// C[ ]	Copies made from a multicast cell.
// FIFO[ ]	Several equal-size FIFOs which are configured from coordination buffers in a demultiplexer.
// *m*	A value returned by the function: distribute( ). It is the number of copies actually made from a multicast cell.

//Define the following functions:	
//receive( )	Receives a cell.
//classify(cell)	Returns destination port index. If the cell is multicasting, returns the smallest output index in its fanout.
//queue(cell,buffer[ ])	Queues a cell into an FIFO. It will drop the cell if the FIFO is full.
//distribute(cell, *B*)	Distributes all fanout of a given multicast cell as evenly as possible over *B* copies, denoted as C[ ], which are indexed from 1 to *m*.

```
//Pseudo-code for cell dispatching mechanism:
For each demultiplexer:
While ( true ) {
   cell = receive( ); //a cell arrives.
   if cell is unicasting {
      i = classify( cell );
      queue( cell, buffer[ srr[i] ] ); //buffer the cell.
      srr[i] = ( srr[i]+1 ) mod K;    //update the pointer.
   }
   if cell is multicasting {
      m = distribute(cell, B);
      for j = 1 to m {
         i = classify( C[j] );
         queue( C[j], buffer[ srr[i] ] );
         srr[i] = ( srr[i]+1 ) mod K;
      }
   }
}
```

Fig. 2: The cell dispatching mechanism.

4. COMPARATIVE PERFORMANCE EVALUATION OF MULTICAST SCHEMES

In this section, performance of multicast schemes as applied to PPS is evaluated under bursty traffic which consists of a mix of unicast and multicast cells. We assume the proposed model of *N* input ports and *N* output ports, and each incoming packet is segmented into several fixed-sized cells. All cells arrival at every input port are equally likely sent to every output port at the probability of $\frac{1}{N}$. The link arrival rate is denoted as λ.

4.1 Traffic Models

We use the bursty arrival to simulate the burst nature of realistic networks. Thus, at each input link, cells arrive between consecutive time slots for a duration of busy burst period and a duration of idle period. As a random variable, the length of a busy burst, b, follows geometrical distribution with a mean of $E[b]$, which is also called mean burst length (denoted as *mbl*); the length of an idle burst follows geometric distribution with a mean of $(1/\lambda - 1)\times E[b]$.

The arrival rates of unicast and multicast traffic are denoted as λ_u and λ_m, respectively. Hence, we have $\lambda_u = P_u \times \lambda$ and $\lambda_m = (1-P_u)\times\lambda$, where P_u is the probability that an arrival cell is unicast. The considered distribution of fanout sets are the following:

uniform fanout: the fanout set is chosen uniformly at random among all the possible fanout sets. Let $E[F]$ be the mean fanout size, then $E[F] = N/2$.

binomial fanout: the fanout set is chosen according to a non-uniform binomial distribution with mean fanout size $E[F]$. Hence, its p.m.f. can be expressed as

$P_f = C_f^N \cdot P^f \cdot (1-P)^{N-f}$, where

$P = E[F]/N$.

We can calculate the effective load λ_{eff} by following equation:

$\lambda_{eff} = \lambda \times P_u + \lambda \times (1-P_u)\times E[F]$.

And we only consider admissible traffic, i.e., $0<\lambda_{eff}<1$ and $0<P_u<1$.

4.2 Simulation Results

We have conducted simulations to compare the performance of the PBC with two other traditional multicast schemes, i.e., copy multicast and fanout multicast. We evaluate the cell loss probability by computer simulations which are developed in *Java* and *C*. The parameters of simulation are listed in table 1.

Table 1: Simulation parameters.

Symbol	Description
N	Switch size.
K	Number of center-stage switches.
S	Speedup factor of PPS.
L	Queue size in a center-stage switches.
Pu	Probability that an arrival cell is unicast.
mbl	Mean burst length.
α	A parameter.
H	Time interval in units of internal time slot.

We built 16×16 and 32×32 buffered PPSs, respectively. The speedup factor S of PPS is set to 1, thus the number of center-stage switches only requires at least $N+1$ to achieve the goal of PPS where the internal link speed is lower than external line rate (in our simulations, K is set to $N+1$). Each demultiplexer contains $N\times K$ coordination buffers which are arranged as K size of N FIFOs. To examine capability of these multicast schemes under high ratio of multicast traffic, we assume P_u is 0.2 that the multicast cells occupy a large part in all arriving cells.

4.2.1 Effect of Multicast Schemes under Uniform Fanout Distribution

Figure 3 and 4 show the cell loss probability versus effective load in uniform fanout distribution. As can be seen in figures, our proposal PBC outperforms copy multicast and fanout multicast under overall effective loads. The trend is clear even as burst degree goes heavier. Also, we notice that copy multicast performs the worst performance obviously. The reason is that copy multicast only uses the coordination buffers in demultiplexers to handle multicast replication which may induce large number of cells lost in demultiplexers due to buffers overflowing. Since the curves coincide with each other with slight difference at heavy loads, we will show the improving rate of PBC to fanout multicast in table 2 and 3 for observing. The improving rate of PBC to fanout multicast can be calculated by

$$rate_{PBC/fanout} = \frac{Throughput_{PBC} - Throughput_{fanout}}{Throughput_{fanout}}.$$

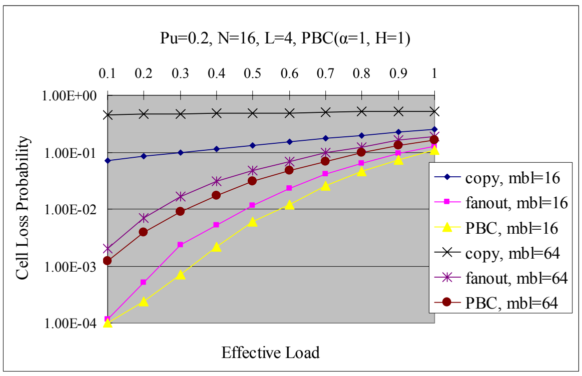

Fig. 3: Switch 16×16 with uniform fanout under different burst degrees.

Table 2: Improving rate of PBC to fanout multicast in figure 3.

16×16 Switch			
	λ_{eff} =0.8	λ_{eff} =0.9	λ_{eff} =0.99
PBC, *mbl*=16	28.71%	21.16%	14.60%
PBC, *mbl*=64	19.98%	18.88%	13.36%

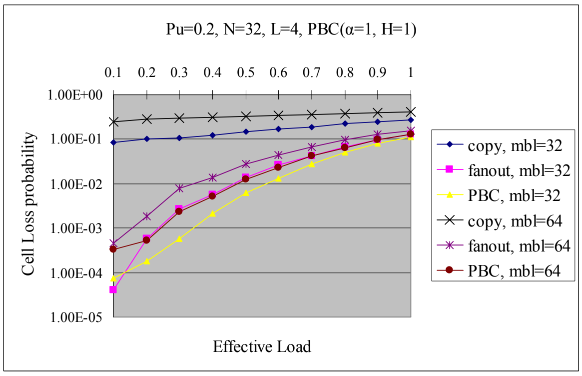

Fig. 4: Switch 32×32 with uniform fanout under different burst degrees.

Table 3: Improving rate of PBC to fanout multicast in figure 4.

32×32 Switch			
	λ_{eff} =0.8	λ_{eff} =0.9	λ_{eff} =0.99
PBC, *mbl*=32	25.20%	17.08%	15.01%
PBC, *mbl*=64	31.60%	23.71%	20.37%

4.2.2 Effect of Multicast Schemes under Binomial Fanout Distribution

Figure 5 and 6 show the cell loss probability under binomial fanout distribution. The improving rate of PBC to fanout multicast at high loads is represented in table 4 and 5. We note that the effect of PBC does not get beyond fanout multicast as much as in uniform fanout distribution when switch size increases to 32. The reason for this result can be derived from following discussion. We make the experiment with a higher mean fanout size (0.75N) in figure 6, which will lead to a large fanout dominated multicast traffic. When offered load is high, the buffer occupation factor Q increases and reaches the buffering saturation rapidly. From the reasons above, the heavy multicast traffic expansion may cause the bounded copy value B to decline to a small value quickly in a burst. In other words, the result makes PBC behave as fanout multicast.

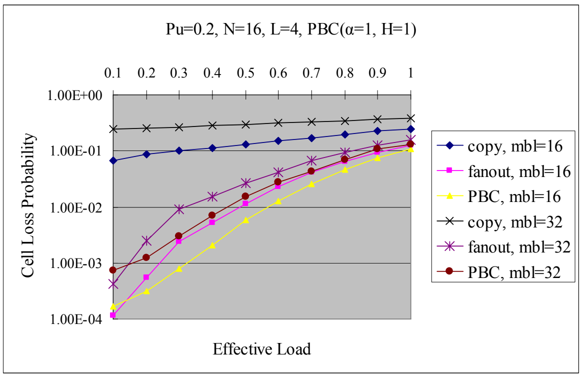

Fig. 5: Switch 16×16 with binomial fanout of mean fanout set (0.5N) under.

Table 4: Improving rate of PBC to fanout multicast in figure 5.

16×16 Switch			
	λ_{eff} =0.8	λ_{eff} =0.9	λ_{eff} =0.99
PBC, *mbl*=16	29.68%	20.29%	14.49%
PBC, *mbl*=32	24.79%	13.39%	19.29%

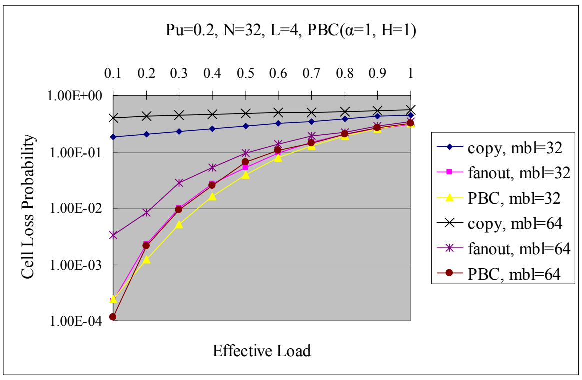

Fig. 6: Switch 32×32 with binomial fanout of mean fanout set (0.75N) under.

Table 5: Improving rate of PBC to fanout multicast in figure 6.

32×32 Switch			
	λ_{eff} =0.8	λ_{eff} =0.9	λ_{eff} =0.99
PBC, *mbl*=32	7.99%	2.24%	1.75%
PBC, *mbl*=64	6.12%	7.49%	6.75%

Here we make some conclusions before subsequent experiments. By the above simulation results, it is worth noting that fanout multicast does not suffer serious cell loss rate when comparing with copy multicast. Since fanout multicast does not replicate multicast cells in demultiplexers, which may lead to distribute multicast traffic unevenly among the center-stage switches. In such scenario, although multicast cells can experience no cell loss in demultiplexers, it is still possible for buffers in a center-stage switch to overflow even though buffers in other switches are not full.

On the other hand, the PBC can reduce possible buffer overflow in demultiplexers and center-stage switches by efficiently utilizing the copy capabilities of both layers. The size of B maintained over time intervals can also provide effect on flow estimation. Moreover, the *distribute* function ensuring even distribution can help to balance multicast traffic load over center-stage switches, making it also possible to reduce cell loss in central switch layers.

4.2.3 Design Alternatives for PBC

In order to attain a certain level of performance, we experiment the parameter H. Figure 7 shows the performance with different scheduling horizons (the time duration for which the schedule is generated). Intuitively, the optimal value H should be as small as possible for fine-grained tuning. Nevertheless, as can be seen from the figure, there is an optimal H, where H is equal to one unit of internal time slot, beyond which there is nearly no measurable throughput benefit. Hence, an optimal H depends on burst degree and the duration of internal time slot. This result also provides the practicality of implementation for maintaining B periodically in the lower speed switch system.

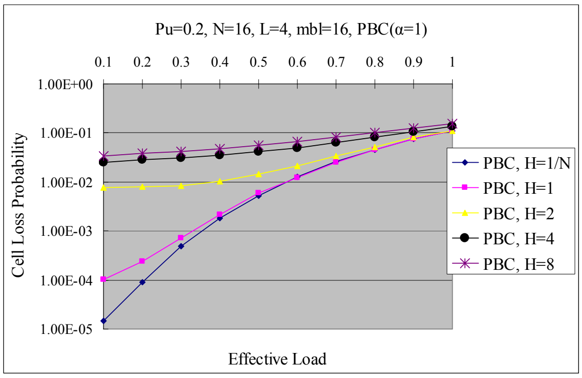

Fig. 7: Switch 16×16 with uniform fanout under various scheduling horizons (H).

Next, we examine the parameter α from 1 to 4 under uniform fanout distribution and binomial fanout distribution. Because the difference of performance between adjacent α is not significant, we only show the comparison between α=1 and α=4. Figure 8 and 9 illustrate the cell loss probability versus effective load with different values α. We can see that a higher α leads to a worse performance than a lower one at low offered load. This is because that a higher α indicates smaller changes on B, which makes it adjust B to a proper value more insensitively. When offered load is low, the variation of buffer occupation may change tremendously and frequently, which cause PBC with a high α can not modify B sharply in such scenario. On the contrary, it can be observed that curves of a higher α is a little better than a lower one at high offered load. The reason can be derived from that as offered load is high, a fined-grained tuning is much fitter when the variation of buffer occupation is slight.

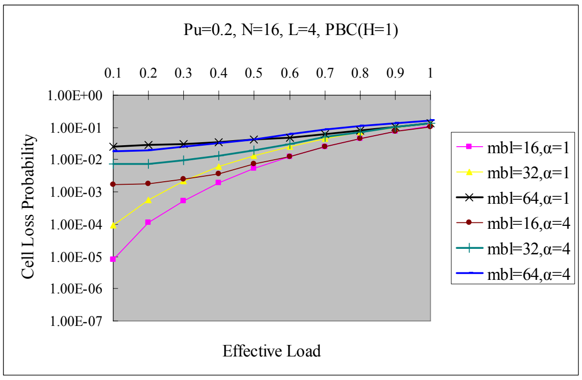

Fig. 8: Switch 16×16 with uniform fanout under various parameters (α).

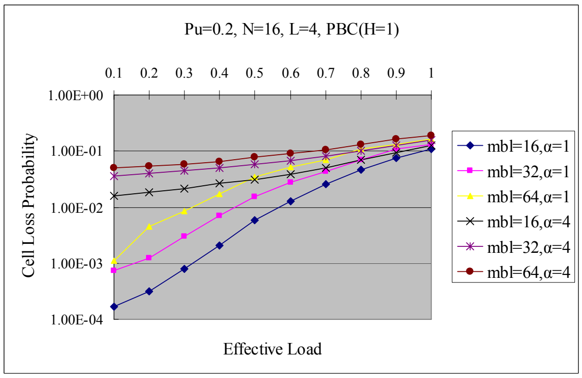

Fig. 9: Switch 16×16 with binomial fanout of mean fanout set (0.5N) under various parameters (α).

5. Conclusion

From the above discussions, the contribution of this paper was to introduce a novel multicast algorithm on the PPS. Our proposal, the proportional bounded copy scheme (PBC), uses a bounded copy value B to limit the multicast replication in demultiplexers where B is adjusted periodically by a proportional scheme. Furthermore, a cell dispatching mechanism for multicast and unicast traffic is also proposed. In our mechanism, the fanout of a multicast cell will be distributed evenly over copies which are bounded by the bounded copy value B.

We evaluate the cell loss probability by computer simulations. By utilizing the copy capability of each layer and balancing the traffic expansion of multicast cells to center-stage switch layers, our proposal enables the PPS to handle multicast traffic more efficiently. As shown by simulation evaluations, the PBC outperforms traditional multicast schemes in terms of the cell loss probability. Besides, it also has advantages in efficient memory usage, which can make it possible to reduce the buffer requirement in central switch layers.

REFERENCES

[1] S. Iyer, A. Awadallah and N. McKeown, "Analysis of a packet switch with memories running slower than the line rate," in *Proc. IEEE INFOCOM 2000,* vol. 2, pp. 529-537, March 2000.

[2] S. Iyer and N. McKeown, "Making parallel packet switches practical," in *Proc. IEEE INFOCOM 2001,* vol. 3, pp. 1680-1687, April 2001.

[3] D. Khotimsky and S. Krishnan, "Stability analysis of a parallel packet switch with bufferless input demultiplexers," in *Proc. IEEE ICC 2001*, vol. 1, pp. 100-106, June 2001.

[4] S. Iyer and N. McKeown, "Analysis of the parallel packet switch architecture," *IEEE/ACM Trans. On Networking*, vol. 11, no. 2, pp. 314-324, April 2003.

[5] S. Iyer and N. McKeown, "On the speedup required for a multicast parallel packet switch," *IEEE Commun Lett.*, vol. 5, pp. 269-271, June 2001.

[6] Ximing Hu, Binqiang Wang and Jing Yu, "Making multicast parallel packet switch practical," in *Proc. IEEE WCNM 2005*, vol. 2, pp. 1393-1397, September 2005.

[7] M.-H.Gua and R.-S. Chang, "Multicast ATM switches: Survey and performance evaluation," *omputer Commun. Rev.*, vol.28, no.2, pp. 98-131, April. 1998.

中文摘要

平行封包交換器（Parallel Packet Switch）已成為網路交換器領域裡熱門的討論主題之一，主要原因是其利用平行分散處理的概念提供內部交換器能以低於外部線速的速度來處理網路封包。近年來已經有許多平行封包交換器的相關研究被提出，然而大部分的研究都假設封包的型態為單一傳送(unicast)，隨著未來網路逐漸走向多重傳送(multicast)為主的應用，研究平行封包交換器支援多重傳送的封包型態變的極為迫切。本篇論文的主要目的是在平行封包交換器上提出一個新的多重傳送演算法，藉由程式模擬來和傳統多重傳送演算法做比較，驗證我們所提出的演算法的可行性，且可提供平行封包交換器在多重傳送下有更佳的處理能力。