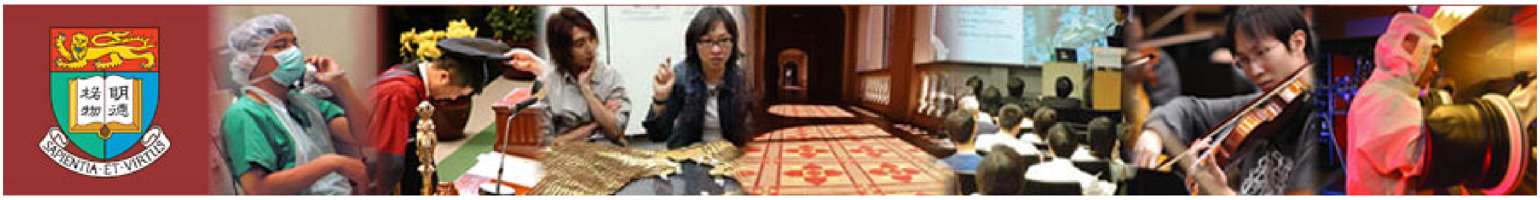
View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by HKU Scholars Hub

The HKU Scholars Hub The University of Hong Kong 香港大學學術庫



Title	On the scalability of feedback-based two-stage switch
Author(s)	Hu, B; He, C; Yeung, LK
Citation	The 2012 IEEE International Conference on Communications (ICC 2012), Ottawa, Canada, 10-15 June 2012. In IEEE International Conference on Communications, 2012, p. 2956-2960
Issued Date	2012
URL	http://hdl.handle.net/10722/165313
Rights	IEEE International Conference on Communications. Copyright © IEEE.

On the Scalability of Feedback-based Two-stage Switch

Bing Hu[1], Chunzhi He[2] and Kwan L. Yeung[2]

[1]Department of Information Science & Electronic Engineering
Zhejiang University
Hangzhou, PRC
E-mail: binghu@zju.edu.cn

[2]Department of Electrical & Electronic Engineering
The University of Hong Kong
Hong Kong, PRC
E-mail: {czhe, kyeung}@eee.hku.hk

***Abstract*—The feedback-based two-stage switch does not require a central scheduler and can provide close to 100% throughput [3]. But the number of crosspoints required for the two stages of switch fabric is $2N^2$, and the average packet delay performance (even under light traffic load) is on the order of $O(N)$ slots, where N is the switch size. To improve the performance of feedback-based two-stage switch when N is large, we adopt the Clos network for constructing a large switch from a set of smaller feedback-based switch modules. We call it a Clos-feedback switch. The potential problem of packet mis-sequencing is solved by using application-flow based load balancing. With recursive decomposition, a Clos network can degenerate into a Benes network. We show that for a Clos-feedback switch, the number of crosspoints required is reduced to $4N(2\log_2 N-1)$ and the average packet delay is cut down to $O(\log_2 N)$ slots.**

Keywords- Feedback-based two-stage switch; Clos-feedback switch; packet mis-sequencing

I. INTRODUCTION

With the continuous growth of bandwidth in fiber links, the need for building high speed switches/routers is urgent in order to keep pace with the increased transmission rate. Load-balanced switches [1] have received a great deal of attention recently because they are simple and can provide close to 100% throughput. A load-balanced switch consists of two stages of switch fabric, as shown in Fig. 1. The first switch fabric converts the non-uniform traffic into uniform and the second fabric delivers packets to their correct outputs. Each switch fabric is configured by a predetermined, periodic sequence of N configurations, where N is the switch size. The basic requirement of the sequence is that each input is connected to each output exactly once in the sequence. Accordingly, a central scheduler for determining the best switch configuration in each time slot in real time is not needed. This makes load-balanced switch suitable for high-speed implementation.

From Fig. 1, we can see that the outputs of the first switch fabric collocate with the inputs of the second switch fabric. Unless otherwise specified, we call them middle-stage ports. We call the outputs of the second switch fabric as outputs of the load-balanced switch, or simply outputs. The basic operation of a load-balanced switch is as follows. When a packet arrives at an input (and assume there is no input buffer), it will be immediately delivered to a middle-stage port based on the current switch configuration used in the first switch fabric. Due to the periodic sequence of configurations used, a burst of packets arrived at an input will be evenly spread out to different middle-stage ports. Ideally, the (non-uniform) input traffic will be converted into uniform before entering the second switch fabric. Packets arrived at middle-stage ports join the corresponding VOQs (virtual output queues) based on their outputs. When a middle-stage port is connected to an output (according to the periodic switch sequence used), a packet (if any) from the corresponding middle-stage port VOQ will be sent.

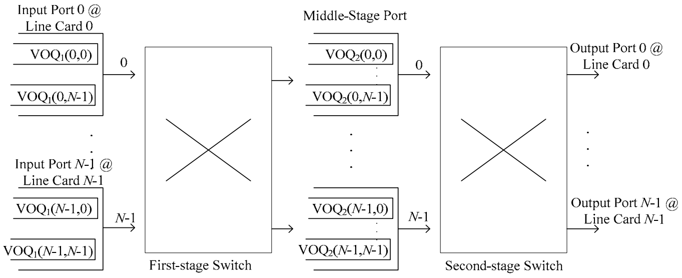


Figure 1. A feedback-based two-stage switch.

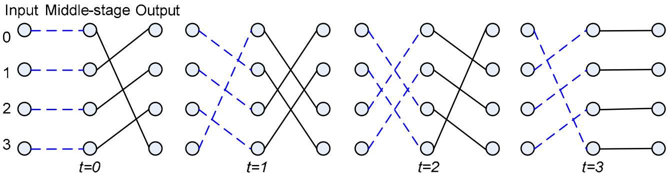


Figure 2. A joint sequence for a 4×4 feedback-based two-stage switch.

It can be easily shown that if the traffic entering the second switch fabric is uniform, 100% throughput can be guaranteed. The issue is if the load balancing performance rendered by the first switch is good enough. In [1], it is proved that if the input traffic is stationary and weakly mixing [2], the first switch fabric can convert any non-uniform traffic into uniform. From the basic operation of a load-balanced switch above, we can see that packets of the same flow (i.e. arriving at the same input and destined for the same output) will arrive at their output via different middle-stage ports, due to the load-balancing mechanism at the first switch fabric. Besides, packets may experience different delays at different middle-stage ports. As a result, when packets of the same flow arrive at the output, their order cannot be guaranteed.

Many efforts [3-10] are then made to address this notorious packet mis-sequencing problem. Among them, the feedback-based (two-stage) switch [3] provides an elegant solution. The key idea is to ensure that packets of the same flow, no matter which middle-stage port they traversed, always experience the

This work was supported in part by Small Project Funding 201007176210 The University of Hong Kong, Zhejiang Provincial Natural Science Foundation of China (No. Y1100388), Zhejiang Provincial Public Technology Research of China (No. 2010C31071), Fundamental Research Funds for the Central Universities (No. 2010QNA5032).

978-1-4577-2053-6/12/$31.00 ©2012 IEEE

same amount of middle-stage delay. It is further shown that the feedback-based switch provides the best delay-throughput performance. (For a detailed review of the feedback-based switch, please refer to Section II.) Nevertheless, the delay performance under even very light traffic loading is on the order of $O(N)$ slots. If the switch size N is large, this delay can be significant. Assume crossbar switch fabric is used. A load-balanced switch consists of two crossbar switch fabrics and a total of $2N^2$ crosspoints is required. Again, when N is large, there is a need to cut down the switch complexity.

In this paper, we focus on improving the delay performance of the feedback-based switch and its implementation complexity. In particular, we propose to construct a large feedback-based switch based on the three-stage Clos network [11], where each switch module in the Clos network is a feedback-based switch. We call it a Clos-feedback switch. Although packet order within each switch module of the Clos-feedback switch is ensured, out of order packet delivery can occur if packets of the same flow traverse through different switch modules. To address this problem, an application-flow based load balancing mechanism is designed. With recursive decomposition, a Clos network can degenerate into a Benes network. Then the total number of crosspoints required for constructing an $N×N$ Clos-feedback switch can be cut down from $2N^2$ (of the original feedback-based switch) to $4N(2\log_2 N－1)$, and the average packet delay can be reduced from $O(N)$ to $O(\log_2 N)$ slots.

The rest of this paper is organized as follows. In Section II, the original feedback-based two-stage switch is reviewed. In Section III, we present our Clos-feedback switch. Its delay and throughput performance is studied in Section IV. Simulation results are presented in Section V and we conclude the paper in Section VI.

II. Feedback-based Two-stage Switch

Fig. 1 shows the feedback-based two-stage switch architecture [3], where $\text{VOQ}_1(i,k)$ represents the VOQ (Virtual Output Queue) at input i with packets destined for output k, and $\text{VOQ}_2(j,k)$ denotes the VOQ at middle-stage port j with packets destined for output k. In Fig. 1, each middle-stage $\text{VOQ}_2(j,k)$ only needs a single packet buffer, and the two stages of the switch fabric are configured using a tailor-made sequence of switch configurations. An example sequence is shown in Fig. 2. Specifically, at time slot t, the connection patterns between input i, middle-stage port j and output k are given by:

$$j = (\, i + t\,) \bmod N, \qquad k = (\, j\ -1-t\,) \bmod N. \tag{1}$$

There is an interesting property of the joint sequence in (1). From Fig. 2, we can see that if middle-stage port j connects to output k in current time slot, then in next slot, input k will connect to middle port j. Since each $\text{VOQ}_2(j,k)$ only has a single packet buffer, an N-bit vector is enough to denote the occupancy of all N $\text{VOQ}_2(j,k)$s (k=0, 1, …, N-1) at middle-stage port j. This vector is piggybacked onto the data packet sent to output k (from middle port j), and is then immediately made available to input k (because both input k and output k reside on the same switch linecard). Based on the received occupancy vector, input k selects the best packet for sending to its currently connected middle port j. Specifically, among the set of queues with the corresponding middle-stage $\text{VOQ}_2(j,k)$ empty, a packet from the longest $\text{VOQ}_1(i,k)$ (k=0, 1, …, N-1) is selected for sending.

With the above mechanism, it is shown [3] that packets of the same flow always experience the same middle-stage port delay (bounded by [1, N] slots), no matter which middle-stage port it passes through, and/or the actual traffic loading. Under uniform traffic, the average packet delay at middle-stage ports can be easily derived as $(1+N)/2$ slots. In general, the overall packet delay is on the order of at least $O(N)$ slots.

III. Clos-feedback Switch Design

A. Clos Network Construction

We propose to construct a large $N×N$ switch based on the Clos network [11] architecture, where each switch module is a feedback-based switch. The resulting Clos-feedback switch is shown in Fig. 3. Without loss of generality, we assume $N = p·q$. Based on the Clos network construction, there are q $p×p$, p $q×q$ and q $p×p$ switch modules in the first, second and third stages respectively. Switch modules in the first and second stages are connected by a perfect shuffle exchange, where for i=0, 1,. . ., p-1, j=0, 1,. . ., q-1, the i-th output from the j-th switch module in the first stage is connected to the j-th input of the i-th switch module in the second stage. The same applies to the connections between the second and third stages of switch modules. Note that the feedback mechanism only executes inside a switch module and there is no feedback between different modules.

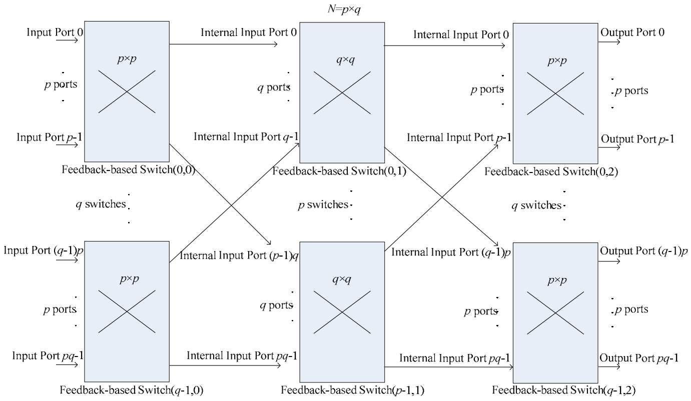


Figure 3. Clos-feedback switch based on Clos network.

Our proposed Clos-feedback switch operates as follows:

1) In the first stage, there are p VOQs at each input port of each switch module. When a packet arrives, it is randomly placed to join a VOQ with probability $1/q$. In doing so, the same flow packets will be uniformly distributed to p VOQs, and thus the p switch modules at the second/middle stage.

2) Each *internal* input port of a second stage switch module maintains q VOQs. If internal input i of the second stage receives a packet destined for output j, the packet joins VOQ(i,m), where $m·p \le j < m·p+p$.

3) There are p VOQs at an internal input port of the third stage. When a packet with destination output j arrives at the internal input i of the third stage, it joins VOQ(i,m), where $m = j \bmod p$.

From the above operation, we can see that the first stage switch modules are responsible for converting non-uniform traffic to uniform. Packets of the same flow are then "re-assembled" in the second and third stages. Compared with a single $N\times N$ feedback-based switch, the number of crosspoints required by the Clos network construction is reduced from $2N^2$ to $4p^2q+2pq^2$. Since $N = p\cdot q$, the number of crosspoints required can be minimized to $2(2N)^{1.5}$ by setting $q = \sqrt{2N}$.

B. Benes Network Construction

Without loss of generality, assume that N is a power of 2. Then we can recursively decompose the Clos network until each switch module becomes a 2×2 feedback-based switch, as shown in Fig. 4. In this case, the Clos network degenerates into a Benes network [12]. For an $N\times N$ Benes switch, there are $2\log_2 N$-1 stages and each stage has $N/2$ 2×2 switches. The number of crosspoints required becomes $4N(2\log_2 N - 1)$.

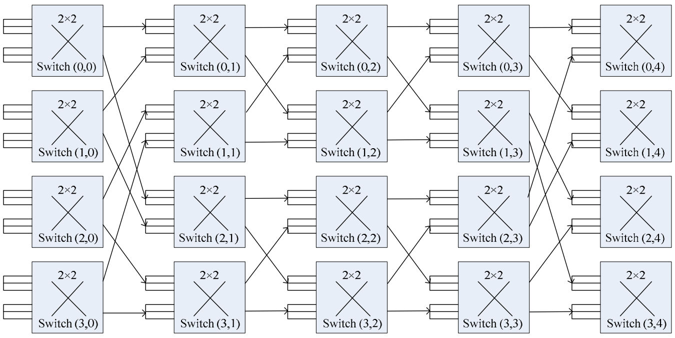


Figure 4. Clos-feedback switch based on Benes network.

Assume a packet destined for output j arrives at the (internal) input i located in switch(u,v) (see Fig. 4). If $v < \log_2 N$, it is placed to one of the 2 VOQs with equal probability. Otherwise, it is stored at VOQ(i,m), where m is given by

$$\frac{(m-1)N^2}{2^{v+2}} \le j \bmod \left(\frac{N^2}{2^{v+1}}\right) < \frac{mN^2}{2^{v+2}}$$

In the above Benes construction, the first $\log_2 N$-1 stages perform load balancing where the same flow packets are uniformly spread over different switch modules. The same flow packets are then "re-assembled" in the next $\log_2 N$ stages.

C. Application-flow Based Load Balancing

Since each switch module in the Clos-feedback switch is a feedback-based switch, the packet sequence within a module can be ensured. But packets of the same flow will go through different switch modules, and thus experience different amount of transit delays. When they finally reach output ports, packet mis-sequencing problem will occur.

To address this problem, we first differentiate between a switch-flow and an application-flow. We define that packets arriving at the same input i and going to the same output j of a switch belong to the same switch-flow. Similarly, packets coming from the same source host and going to the same destination host belong to the same application-flow. We know that a switch-flow consists of many application-flows. If we can ensure packets of the same switch-flow are delivered in-order, application-flow order is also guaranteed. To ensure in-order packet delivery, we can route each switch-flow to always use the same set of switch modules in our Clos-feedback switch. But this cannot balance the traffic load among different switch modules. This defeats the original purpose of load balancing, and 100% throughput is impossible.

Our approach is to route the packets of the same application-flow to go through the same internal switch path, whereas different application-flows, though belonging to the same switch-flow, can go through different paths for load balancing. This imposes two immediate questions: a) how can we identify an application-flow, and b) is the load balancing performance based on application-flows good enough?

To answer the first question, we use the pair of source and destination IP addresses as an application-flow identifier. Given the huge number of application-flows that a backbone router/switch needs to handle, it is reasonable to make the following assumption:

Assumption 1: In a backbone router/switch, the IP address pair associated with each application-flow is uniformly distributed over $[0, 2^{64}\text{-}1]$.

Consider the Clos network in Fig. 3. When a packet arrives at input port i of the first stage, it is assigned to join VOQ(i,j) if its (64-bit *address pair*) *mod* $p = j$. Packets stored in VOQ(i,j) will be delivered to the second stage switch module j. Since the address pair is uniformly distributed over $[0, 2^{64}\text{-}1]$, j will also be uniformly distributed over $[0, p\text{-}1]$. Then all VOQs of the input port will be balanced. The same argument applies to the Benes construction. As such, we solve the packet mis-sequencing problem by application-flow based load balancing.

IV. Analytical Model

A. Throughput

Statement 1: If the incoming traffic is admissible, then under Assumption 1, traffic enters each feedback-based switch module of the Clos-feedback switch is admissible.

Proof: To prove Statement 1, we only need to show that all (internal) input and output ports are not overloaded. Without loss of generality, we consider the Clos network construction shown in Fig. 3. Since the incoming traffic is admissible, input ports of the first stage and output ports of the third stage cannot be overloaded. Note that the traffic entering an internal input of the second and third stages is provided by an internal output of the first and second stages respectively. Since links connecting switch modules are of same line rate, it is impossible for the traffic coming from one port to overload another. Therefore, the internal inputs of the second and third stages will not be overloaded.

With the proposed application-flow based load balancing mechanism and Assumption 1, a first stage switch module will equally divides one switch-flow into p groups of application-flows, each with an arrival rate no larger than $1/p$. In any first stage switch module, each input sends one group of application-flows to an internal output. On average, every internal output will handle p groups of application-flows, whose total traffic rate is still no larger than 1. We can conclude that the internal outputs of the first stage modules are not overloaded.

Let us have a closer look at the second stage switch modules in Fig. 3. All packets entering a switch module possess the same value of (64-bit *address pair*) *mod* $p = j$. For example, any packet going to the first switch module in the second stage must have its $j = 0$. Because of Assumption 1, these packets are uniformly distributed among the last stage output ports. On the other hand, an internal input in the second stage stores packets in q VOQs based on their final destination outputs. Then each VOQ has an incoming traffic rate no larger than $1/q$. For an internal output in the second stage, the traffic comes from q VOQs (one for each internal second stage input and with traffic rate no larger than $1/q$). Therefore, an internal second stage output will not be overloaded.

In summary, all (internal) ports are not overloaded if the incoming traffic to the switch is admissible, and thus the traffic entering each feedback-based switch module in the Clos-feedback switch is admissible. #

Theorem 1*: (Sufficiency)* Under Assumption 1, the Clos-feedback switch can achieve 100% throughput with a speedup of 2 for any admissible traffic pattern.

Proof: From [3], the feedback-based two-stage switch can achieve 100% throughput with a speedup of 2. Due to Statement 1, packets can pass through every feedback-based switch module in Fig. 3 with a bounded delay. Thus the total delay for traversing the whole Clos-feedback switch is also bounded (under a speedup of 2). Then we finished the proof. #

Note that a switch with a speedup of M can remove up to M packets from each input and deliver up to M packets to each output in a time slot. In our Clos-feedback switch, the speedup of two is only required in theory. In practice, simulation results show that it can deliver close to 100% throughput without any speedup. (Please see Section V.)

B. Delay

Recall that in a feedback-based switch, the delay experienced by a packet consists of input port queuing delay and middle-stage port queuing delay. Under uniform traffic, the average packet delay experienced at middle-stage ports [3] is $(1+N)/2$ slots (for an $N\times N$ switch). In our Clos construction in Fig. 3, we cannot cut down the input queuing delay but we can reduce the middle-stage port delay. Note that a packet passes through three feedback-based switch modules, one at each stage. The total middle-stage port delay of the three switch modules is $0.5(3+q+2p)$. Since $N = p\cdot q$, this delay can be minimized to become $1.5+\sqrt{2N}$ by setting $q = \sqrt{2N}$. It is interesting to point out that the Clos construction simultaneously minimizes the delay and the number of crosspoints by setting $q = \sqrt{2N}$.

Similarly in Benes network construction, the total middle-stage port delay in the $2\log_2N$-1 feedback-based switch modules is $3\log_2N$-1.5. We can see that the average packet delay is cut down from $O(N)$ to $O(\log_2N)$ slots.

V. Performance Evaluation

In this section, we study the delay performance of our proposed Clos-feedback switch by simulations. For comparison, the original feedback-based two-stage switch [3] is implemented. We also implement the recently proposed quasi-output-buffered (QOB) switch [13]. Notably, the QOB switch adopts the same Clos and Benes network constructions. To address the problem of packet mis-sequencing, the notion of "frame" is adopted. The QOB switch can cut down the number of crosspoints [13]. But its delay performance is still on the order of $O(N)$ slots, while that for our design is $O(\log_2N)$. Last but not the least, iSLIP algorithm [14] (with a single iteration) and output-queued switch are implemented, which serve as a benchmark for single-stage input-queued switch and optimal delay performance, respectively.

Although our work in this paper is targeted at large switch size, the long simulation time is formidable. To this end, we only simulate a 32×32 switch (without speedup). We believe the simulation results below provide sufficient evidence/insight to justify our proposed Clos-feedback switch architecture. To be fair, we use the same set of parameters for our Clos-feedback and QOB switches, i.e. $p = 4$ and $q = 8$.

A. Uniform Traffic

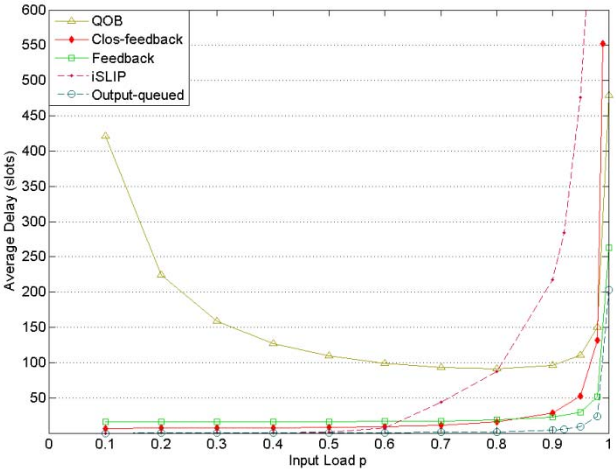


Figure 5. Delay vs throughput, under uniform traffic.

Uniform traffic is generated as follows. At every time slot for each input, a packet arrives with probability p (input load p) and destines to each output with same probability. From Fig. 5, we can see that Clos-feedback can obtain up to 100% throughput and the best delay performance among all load-balanced switches. Compared with QOB, Clos-feedback gives significantly smaller delay. When $p = 0.8$, QOB requires 90.9 time slots, and Clos-feedback only 15.8, cutting down the delay by more than 4 times. Note that when $p < 0.9$, although not obviously in Fig. 5, Clos-feedback beats the original feedback-based switch. For example, when $p = 0.6$, the delays using Clos-feedback and the original feedback-based switch are 8.4 and 16.2 time slots respectively.

B. Uniform Bursty Traffic

Bursty arrivals are modeled by the ON/OFF traffic model, which is a special instance of the two-state Markov-modulated Bernoulli process [15]. In the ON state, a packet arrival is generated in every time slot. In the OFF state, there are no packet arrivals. Packets of the same burst have the same output and the output for each burst is uniformly distributed. Given the average input load of p and average burst size s, the state transition probabilities from OFF to ON is $p/[s(1-p)]$ and from ON to OFF is $1/s$. Without loss of generality, we set burst size $s = 30$ packets. From Fig. 6, we can see that delay builds up quickly with input load. This is because for bursty traffic, the input port queuing delay dominates the total delay performance. In this case, the middle-stage ports queuing delay that Clos-feedback cuts down is less than the increase in the input port queuing delay due to Clos-feedback. As such, the original feedback-based switch yields better delay performance. But it should be noted that the feedback-based switch requires $O(N^2)$ crosspoints, while that for Clos-feedback is $O(N^{1.5})$. From Fig. 6, we can also see that Clos-feedback is better than QOB when $p < 0.7$. For example at $p = 0.6$, with QOB packets experience a delay of 189.4 time slots, whereas for Clos-feedback is just 153.4.

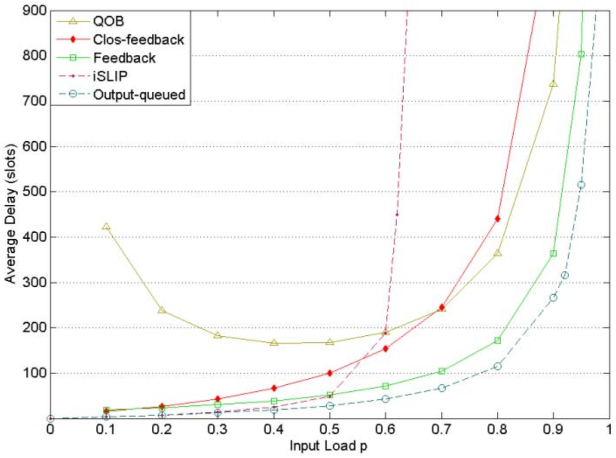


Figure 6. Delay vs throughput, under uniform bursty traffic.

C. *Hot-spot Traffic*

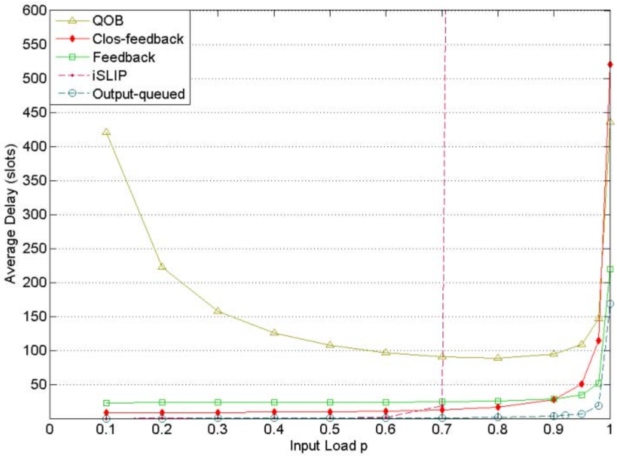


Figure 7. Delay vs input load, under hot-spot traffic.

We assume packets arriving at each input port in each time slot follow the same independent Bernoulli process with probability p. Hot-spots are generated as follows. For input port i, packet goes to output $i+N/2 \bmod N$ with probability 0.5, and goes to other outputs with the same probability $1/[2(N-2)]$. From Fig. 7, again we can see Clos-feedback consistently outperforms QOB and the original feedback-based switches.

In summary, Clos-feedback yields the best delay performance under uniform and hot-spot traffic. Under bursty traffic, the original feedback-based switch performs the best. But it should be noted that the Clos-feedback renders a much less hardware complexity than feedback-based switch.

VI. Conclusions

Aiming at improving the performance of the original feedback-based switch when switch size N is large, we proposed a Clos-feedback switch. Clos-feedback switch is constructed based on the Clos network and with (smaller) feedback-based switches as switch modules. The packet mis-sequencing problem was solved by using application-flow based load balancing. With recursive decomposition, a Clos network can degenerate into a Benes network. As compared with the original feedback-based switch, we showed that the Benes construction of our Clos-feedback switch can cut down the number of crosspoints from $2N^2$ to $4N(2\log_2 N-1)$, and the average packet delay from $O(N)$ to $O(\log_2 N)$ slots.

References

[1] C. S. Chang, D. S. Lee and Y. S. Jou, “Load balanced Birkhoff-von Neumann switches, part I: one-stage buffering,” *Computer Communications*, Vol. 25, pp. 611 – 622, 2002.

[2] M.G. Nadkarni, “Basic ergodic theory,” *Birkhäuser Basel*, 1998.

[3] B. Hu and K. L. Yeung, “Feedback-based scheduling for load-balanced two-stage switches,” *IEEE/ACM Transactions on Networking,* Vol. 18, Issue. 4, pp. 1077-1090, Aug. 2010.

[4] C. S. Chang, D. S. Lee and C. M. Lein, “Load balanced Birkhoff-von Neumann switches, part II: multi-stage buffering,” *Computer Communications*, Vol. 25, pp. 623 – 634, 2002.

[5] C. S. Chang, D. S. Lee and Y. J. Shih, “Mailbox switch: a scalable two-stage switch architecture for conflict resolution of ordered packets,” *INFOCOM 2004*, March 2004, Hong Kong.

[6] Y. Shen, S. Jiang, S. S. Panwar and H. J. Chao, “Byte-Focal: a practical load-balanced switch,” *IEEE Workshop on High Performance Switching and Routing*, May 2005, Hong Kong.

[7] I. Keslassy, S. T. Chuang, K. Yu, D. Miller, M. Horowitz, O. Solgaard and N. McKeown, “Scaling the Internet Routers using Optics,” *ACM SIGCOMM*, Aug. 2003, Karlsruhe, Germany

[8] C. Y. Tu, C. S. Chang, D. S. Lee and C. T. Chiu, “Design a simple and high performance switch using a two-stage architecture,” *GLOBECOM*, Nov. 2005, St. Louis, USA,.

[9] I. Keslassy and N. McKeown, “Maintaining packet order in two-stage switches,” *INFOCOM 2002*, June 2002, New York, USA.

[10] H. I. Lee, B. C. Lee and S. W. Seo, “A load balancing scheme for two-stage switches maintaining packet sequence,” *IEEE ICC 2006*, June 2006, Istanbul, Turkey.

[11] C. Clos, “A study of nonblocking switching networks,” *BSTJ*, Vol. 32, pp. 406-424, 1953.

[12] V. E. Benes, “Mathematical theory of connecting networks and telephone traffic,” *New York: Academic Press*, 1965.

[13] C. S. Chang, J. Cheng, D. S. Lee and C. F. Wu, “Quasi-output-buffered switches,” *IEEE Transactions on Parallel and Distributed Systems*, Vol. 22, Issue. 5, pp. 833-846, May 2011.

[14] N. McKeown, “Scheduling algorithms for input-queued cell switches,” *PhD. Thesis*, University of California at Berkeley, 1995.

[15] B. Hu, K. L. Yeung, “Load-balanced optical switch for high-speed router design,” *IEEE/OSA Journal of Lightwave Technology,* Vol. 28 , Issue. 13, pp. 1969-1977, July 2010.