The CSI Journal on
Computer Science and Engineering
Vol. 2, No. 2&4 (b), Summer & Winter 2004
Pages 1-9
Regular Paper

Design and Analysis of a Fully-Distributed Parallel Packet Switch with Buffered Demultiplexers

Ali Asghar Khodaparast Siavash Khorsandi

Computer Engineering Department, Amirkabir University of Technology
Tehran, Iran

Abstract

A Parallel Packet Switch (PPS) is a multistage switch aimed at building a very high-speed switch using much slower devices. A PPS in general has three stages. Several packet switches are placed in the central stage, which operate slower than the external line's rate. Incoming packets are spread over the center-stage switches by demultiplexers at the input stage. Packets destined to each output port need to be collected and reordered if necessary at the output stage. The initial proposed architecture for the PPS was based on a centralized mechanism with high complexity to distribute incoming packets over the center-stage switches [1]. To reduce the complexity, a distributed algorithm has been proposed in [2] that performs the packet distribution at each demultiplexer independently. The algorithmic complexity of this scheme is in the order of K^2 that poses a scalability problem at high speeds as K grows, where K is the number of center-stage switches. In addition, each demultiplexer requires a high-speed buffer at the external line's rate. In this paper, we have proposed a fully distributed algorithm (at each input line level) with minimal complexity of O(1). Besides demultiplexer buffer in the proposed architecture operates at the low internal link rate. We show that the performance of our architecture is comparable to that of [2]. In particular, we prove that it is stable without any speedup, that is, a bounded delay is guaranteed. The resulting PPS architecture is more simple and implementable.

Keyword: system design, parallel packet switch, load balancing, synchronization, multistage switch

1. Introduction

Nowadays the data transmission speeds have reached the terabit range using optical fibers. Thus, the packet switches must be compatible with such speeds. Using optical components in the packet switches is one approach taken to overcome the switching speed limitations [3,4]. However, electronic components are still used to do the complicated tasks such as header processing and packet buffering. Thus the optical switches must be designed in such a way that their speed can't be limited by electronic components. Many packet switch architectures have been proposed for now which can't work rapidly enough because of their complicated scheduling algorithms [5,6,7].

In [1], a special multistage switch architecture has been proposed so called "Parallel Packet Switch" (PPS). The aim was to make a powerful switch that can operate at speeds equivalent to the optical links using parallelization of simple and slower packet switches. The general architecture of the PPS is shown in Figure 1. The PPS architecture is based on a 3-stage Clos Network [8]. The main difference is that a Clos network is a bufferless switch fabric, whereas the PPS contains buffered packet switches in its center stage. Each one of N input ports is connected to all of K center-stage switches, which operate independently and in parallel. Packets arriving at an input port are examined by the demultiplexer and then sent to one of the center-stage switches. Packets are processed individually; i.e. there is no guarantee that packets belonging to the same flow or to the same output port will pass through the same center-stage switch. In fact, the demultiplexer will ideally spread packets equally over all of the center-stage switches. Packets are stored in the output queues of the center-stage switches and are delivered to the multiplexer. Packets from each input operating at the line rate, R, are sent over links each operating at a rate of at least R/K. In general, the internal links in the center-stage switches operate at a rate $s.R/K$, where s is the speedup factor and K is the number of switches at the central stage.

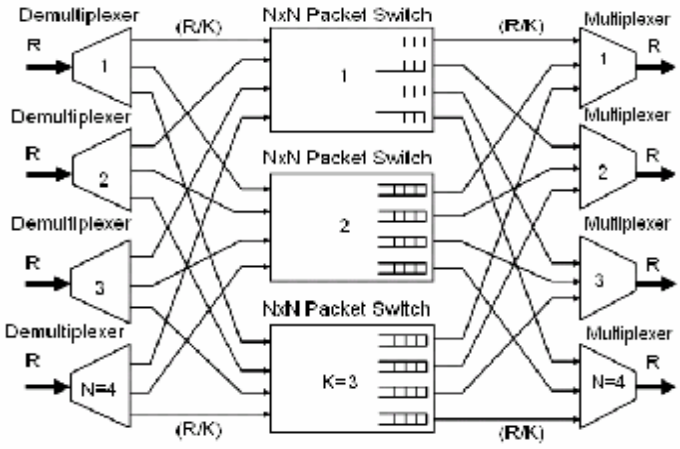


Figure 1. A parallel packet switch with 4 input ports, 4 output ports, and 3 center-stage switches

As discussed in Section II, the previous schemes of the PPS [1,2,9] suffer from complicated algorithm of the demultiplexer and high-speed buffer requirements. In this paper, we propose a new architecture and a fully distributed load balancing algorithm for the demultiplexer such that it is not only distributed at the demultiplexer level but also at each packet type within a demultiplexer. Hence, a centralized controller is not required in a demultiplexer. This facilitates its hardware implementation. The proposed load balancing algorithm has a time complexity of $O(1)$. We have proved this algorithm is stable, i.e. the delays of packets are bounded. Furthermore, all buffers work at the internal link rate.

The rest of this paper is organized as follows. Section 2 reviews and analyzes the previous schemes of the PPS. In Section 3, we describe the internal structure of the demultiplexer and propose a new fully distributed design. In Section 4, an analysis of traffic homogeneity in the center-stage switches is performed. In Section 5, we have analyzed the stability of the proposed algorithm and in Section 6 we present the experimental results. Finally, Section 7 concludes the paper.

2. Previous Work

Before begin this section, we need to have some definitions.

Definition 1: A packet of **type j** is defined to be a packet that is destined to the output port *j*, $j=1,\ldots,N$.

Definition 2: A packet of **flow (i,j)** is defined to be a packet that entered from the input port *i*, $i=1,\ldots,N$, and is destined to the output port *j*, $j=1,\ldots,N$.

Definition 3: An **internal time slot** or **time slot** in short is a time period used to transfer a maximal-size packet over an internal link. An internal time slot is equal to *K* consecutive time-slots over an external line.

The main issue in the design of a PPS is the demultiplexer architecture. A fast and efficient scheduling algorithm is required to distribute the load as evenly as possible among the center-stage switches on per output port basis. Contention among packets sent to the same center-stage switch requires buffering. To minimize the delay of packets in the PPS, a centralized optimal load balancing scheme in the demultiplexers, buffering and processing at the external line's rate, and high speedup factor is required. One of such designs has been proposed in [1] that suffers from high complexity of the distribution algorithm. An algorithm has been proposed in [2] that relieves speedup requirements and provides a near-optimal distributed algorithm for the demultiplexer. In this scheme, if a packet of type *j* enters a demultiplexer in an external time-slot, that demultiplexer must construct these two sets of center-stage switches:

1. The set of center-stage switches each one has idle link to that demultiplexer in the next idle internal time-slot in the future.
2. The set of center-stage switches each one has idle link to the multiplexer *j* in the next idle internal time-slot in the future.

Then, an intersection of these sets determines the destination center-stage switch for that packet.

Theorem 1: *the intersection process used in [2], has a time complexity of* $O(K^2)$.

Proof: Each of these two sets consists of at most *K* elements at each external time-slot. For the intersection process, we must first consider the first element of the first set and search it in the second set. If the first search was not successful, we must search the second set for the second element of the first set. These searches must be continued until either a shared element is found or all of the *K* elements of the first set are searched in the second set. Every search requires at most *K* comparisons.

If we want the search for only the first element of the first set not to be successful, the second set should not contain that element, i.e. the second set must contain at most *K-1* elements. In such a way, if we want the search for only the first *k* elements of the first set not to be successful, the second set should not contain those elements, i.e. the second set must contain at most *K-k* elements. Then, the number of required comparisons is equal to $k.(K-k)$. This is maximized if $k=K/2$. Then, the maximum number of required comparisons will be equal to $K^2/4$.

Construction of these two sets at each external time-slot has a time complexity of $O(1)$, since at most one packet will enter the demultiplexer at each external time-slot and then, at most one center-stage switch may be added to or removed from these sets. But the intersection process has a time complexity of $O(K^2)$ based on the theorem 1. A PPS with this complexity per external time-slot can't work at high speeds especially with optical lines. Furthermore, this limits the number of center-stage switches. In addition, this scheme requires buffer at external line's rate in each demultiplexer.

Another scheme has been proposed in [9] for a PPS with bufferless demultiplexers. In that scheme, a backpressure mechanism is used for each demultiplexer to have information about the lengths of buffers of the center-stage switches. Then for each incoming packet at each external time-slot, the demultiplexer chooses the destination center-stage switch one that has the minimum buffer length. This searching process has a time complexity of $O(K)$ and must be done on each external time-slot and limits the speed of the demultiplexer. Furthermore, all parts of the demultiplexer have to work at external line's rate.

3. A New Internal Design For The Demultiplexer

The PPS architecture includes *N* demutiplexers in the first stage, each one placed on one of the input ports. The general architecture of the demultiplexer is shown in Figure 2. It

includes buffers at the input and/or at the output. A distribution network is responsible for making a decision to send an incoming packet on which center-stage switch. The center-stage switches have *N* input and *N* output ports. We assume all of the center-stage switches have same input/output rates.

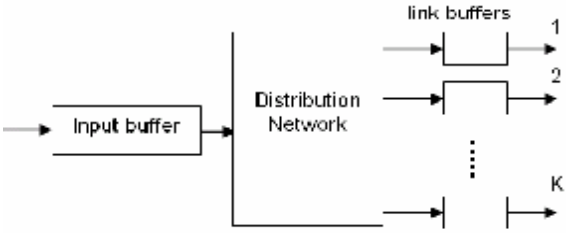


Figure 2. The general structure of a demultiplexer

In an ideal case, each demultiplexer must distribute the incoming packets destined to each of the output ports equally among the center-stage switches. That is, the state of the center-stage switches with regard to each packet type can differ only by one. Let us consider a distribution network based on a pure round-robin strategy. Suppose an incoming traffic scenario where the destination ports of packets periodically go from *1* to *K* (Figure 3). Using round robin operation, packets of type *j* will be sent to the *j*-th center-stage switch. If this scenario is repeated at all of the input ports, each center-stage switch like *k* will receive packets of type *k* at the external line's rate, *R*, while the rate of its link to the output *k* is *R/K*. This results in huge misbalance and loss of traffic in the case of the finite buffers.

In order to solve the fore-mentioned load imbalance problem, we envision applying the round-robin scheduling at each packet type level, rather than applying it at the line level, in a demultiplexer (Figure 4). When packets of type *j* enter the demultiplexer, they are distributed among center-stage switches. This results in perfectly uniform distribution of packets over center-stage switches.

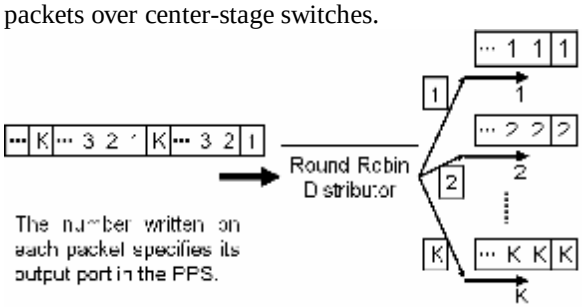


Figure 3. Operation of a pure round-robin scheduler in the demultiplexer

In Figure 4, the input buffers need to work at the line rate. Moreover, a collision can occur when two or more schedulers send packets to the same output buffer. This needs to use *K* multiplexers at the egress side of the demultiplexer.

Proposed Demultiplexer Architecture

We now propose a new design for the demultiplexer that is both efficient in per-output port load balancing at each center-stage switch and simple in its hardware implementation. The proposed architecture is shown in Figure 5. Its operation is described in the following.

Distributor(1): which is the first distributor in the demultiplexer, distributes incoming packets among allocators according to their destination output port.

Allocator(j): which is the *j*-th allocator in the demultiplexer, allocates the numbers *1,2,…,K* to its received packets in a round-robin manner. This number allocated to a packet is a center-stage switch's sequence number which that packet will be sent to it. There are *N* allocators in every demultiplexer, each one for one of the output ports of the PPS.

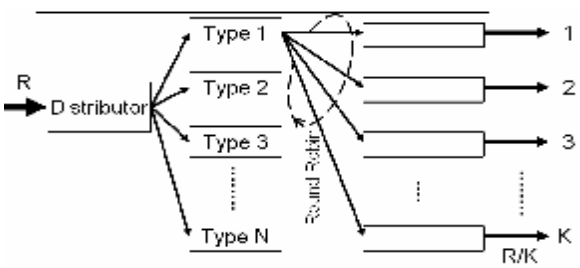
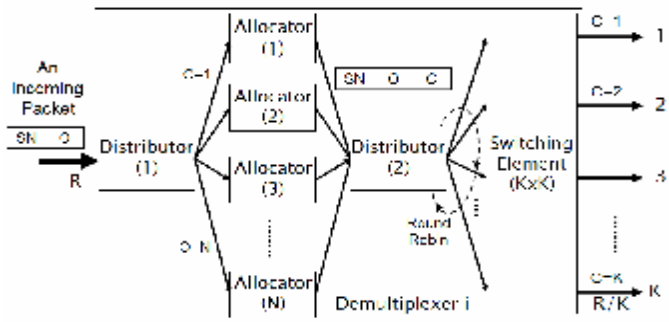


Figure 4. The architecture of a demultiplexer with round-robin scheduling at each packet type level

Distributor(2): which is the second distributor in the demultiplexer, gathers packets from the allocators and distributes them among the inputs of switching element in a round-robin manner.

Switching Element: which is a *K*×*K* packet switch, switches the incoming packets according to their destination center-stage switch.

Since the allocators operate at the external line's rate, there is no contention between the allocators at Distributor(2). The duty of Distributor(2) is to distribute packets uniformly among the inputs of the switching element. Thus, there is no need for this switch to operate at the external line's rate. Only distributors and allocators have to operate at the external line's rate, which are computationally very simple. In other words, the time complexity of this distribution scheme in the demultiplexer is of *O*(1).



SN: Sequence Number, O: Output port, C: Center-stage switch of the packet

Figure 5. The proposed internal structure of the demultiplexer

4. Traffic Homogeneity In The Center-Stage Switches

There is a special problem in the scheduling of packets in the PPS that leads to some other problems. That is the allocators in a demultiplexer may be synchronized on a number like *k*, and in this state, during *N* consecutive time-slots each allocator may receive only one packet. Then, *N* consecutive packets will be allocated to the *k*-th center-stage switch.

Probability of the synchronization of *N* allocators is very low and it usually happens on less number of allocators. However, the results of this event are:

1. With assumption that the switching element is output-queued, the queue size of the output *k* in that switch grows suddenly. This makes the sizes of queues in the various outputs of that switch different. Although this difference is bounded, it may lead to the results 3 and 4.
2. When a synchronization of any length is happening on a center-stage switch, no packet is sent to some of the center-stage switches on that time-slot and the throughput may be decreased.
3. The difference between the sizes of various output queues in the switching element may lead to the disordering of packets, because two consecutive packets of flow *(i,j)* will pass two different center-stage switches.
4. That difference may also lead to the loss of homogeneity of traffic in various center-stage switches.

In appendix I, it has been shown that the maximum difference of the number of packets of same type, between each two center-stage switches is of $O(N^2)$. In other words, when the number of packets of type *j* is equal to x in a center-stage switch, in another one, in the worst case, it can be at most $x + O(N^2)$.

Now, it is apparent that sequence of the incoming packets may be disordered in the PPS. Thus, we use a resequencing buffer in each multiplexer as used in [2]. In each multiplexer, when a packet comes from the input port *i*, while its previous packets from the same input have not been received, it will be saved into the buffer assigned to the input port *i* in that multiplexer. When all of the previous packets passed that multiplexer, that packet can continue. This operation needs each allocator in each demultiplexer to insert another number on its packets which specifies the incoming sequence of packets for destination multiplexer.

According to the appendix II, the maximum amount of disordering for two consecutive packets of flow *(i,j)* is $N.(N - \lceil N/K \rceil)$. This number is equal to the maximum number of packets of type *j* that have entered to the demultiplexer *i* after a specific packet, but have come to the multiplexer *j* before that packet. Thus, this number is equal to the size of buffer required in the multiplexer *j* for the resequencing of packets of flow *(i,j)*.

5. Stability Analysis of The Proposed PPS

Let's consider a pattern of traffic to the allocators of a demultiplexer in a time interval. With this incoming traffic, an allocator like allocator(*j*) counts as many as $r_{i,j}K + q_{i,j}$ numbers. In other words, it counts as often as $0 \le r_{i,j} < K$ from *1* to *K* and once as many as $0 \le q_{i,j} < K$ numbers among *1,2,…,K*. This consideration can be done for the other allocators in that time interval with various $r_{i,j}$ s and $q_{i,j}$ s. When the allocator(*j*) counts as often as $r_{i,j}$ from *1* to *K*, each number is counted as often as $r_{i,j}$ and this counting takes $r_{i,j}K$ time-slots. In this interval, an OQ switching element can process $r_{i,j}$ packets passed from the allocator(*j*) and going to the same center-stage switch. Thus after those $r_{i,j}K$ counts of each allocator, the size of each output queue in the switching element will remain unchanged.

For allocator(*j*), if a number like *k* is among those $q_{i,j}$ numbers, then that allocator will send packets destined to the center-stage switch *k*, to the switching element at a rate greater than the mean rate of each allocator. Also, if *k* is not among those $q_{i,j}$ numbers, then that allocator will send packets destined to the center-stage switch *k*, to the switching element at a rate less than the mean rate of each allocator. Thus, the maximum difference of the number of packets destined to the center-stage switch *k* from the mean number which an allocator sends to the switching element in any time interval is one.

Theorem 2: *if the switching element is output-queued, the maximum amount of the increment in queue size of the output k in that switch on any time-slot is* $N - \lceil N/K \rceil$.

Proof: if packets destined to the center-stage switch *k* in a time interval are sent to the switching element at a rate greater than the overall mean rate of the allocators, then queue size of the output *k* in that switch will be increased on that interval. Thus in any time interval, the maximum increment in the size of that queue will happen when all of the *N* allocators send packets destined to the center-stage switch *k* to the switching element at a rate greater than their mean rate. If that happens, since an allocator can send those packets at most one more than its mean number during that interval, then *N* packets destined to the center-stage switch *k* will be sent more than the overall mean number. During send of those packets, which takes at least *N* time-slots, at least $\lceil N/K \rceil$ packets destined to the center-stage switch *k* will be delivered to that center-stage switch. Thus, the maximum amount of that increment is $N - \lceil N/K \rceil$. The worst case of this number is obtained when $N \le K$ which is *N-1*. □

Theorem 3: *if size of the output queue k in the switching element is increased as many as* $N - \lceil N/K \rceil$ *in a time interval, then that queue has been empty just before that interval.*

Proof: Let's assume that PPS has started its work on the time-slot 0 and has been empty on that time-slot. We consider that time interval as $[t_1, t_2]$. Now, let's assume at least one packet has been in that queue just before the time-slot t_1. Then, we can conclude that from a time-slot like t_0 ($0 \le t_0 < t_1$) to just before the time-slot t_1, packets destined to the center-stage switch *k* have been sent to the switching element at a rate greater than the overall mean rate. Thus in the interval $[t_0, t_1)$, size of that queue has been increased at least by one packet.

According to the assumption of this theorem, the size of that queue has been increased as many as $N - \lceil N/K \rceil$ in the

interval $[t_1, t_2]$. Thus in the interval $[t_0, t_2]$, the size of that queue has been increased more than $N - \lceil N/K \rceil$. This result is against the theorem 2 and thus, that queue has been empty just before the interval $[t_1, t_2]$.

Theorem 4: *The maximum size of the queue k in an OQ switching element is* $N - \lceil N/K \rceil$.

Proof: Let's assume the size of that queue is one packet more than $N - \lceil N/K \rceil$ in a time-slot. Then, there is a time interval that size of the queue *k* had been at least one packet just before that interval and during it, the size of that queue has been increased as many as $N - \lceil N/K \rceil$ packets. This result is against the theorem 3 and the theorem 4 is proven.

Corollary 1: *The maximum difference between the sizes of various queues in an OQ switching element is* $N - \lceil N/K \rceil$.

6. Simulation Results

In the previous sections we have demonstrate that our PPS can impose a delay on packets which is at most $O(N^2)$ external time-slots greater than the packet's delay in an NxN output-queued switch. In this section, we compare the two switches by simulation.

Figure 6 shows average delay both in the PPS and in the OQ switch. Considering reality, we have also simulated the PPS with VOQ center-stage switches instead of OQ ones. The simulation is repeated for different traffic distributions. It is apparent from the figure that the PPS and the OQ switch have had approximately equal average delays under burst arrivals. Since the allocators have the most probability to be synchronized under a uniformly distributed arrival, the difference of the simulated switches is increased under the uniform traffic.

By increasing the uniformity of traffic, the delays have generally been decreased in all the switches, because fewer outputs were overloaded. A PPS with VOQ center-stage switch can not naturally operate as well as a PPS with OQ ones. This fact is also visible in the figure.

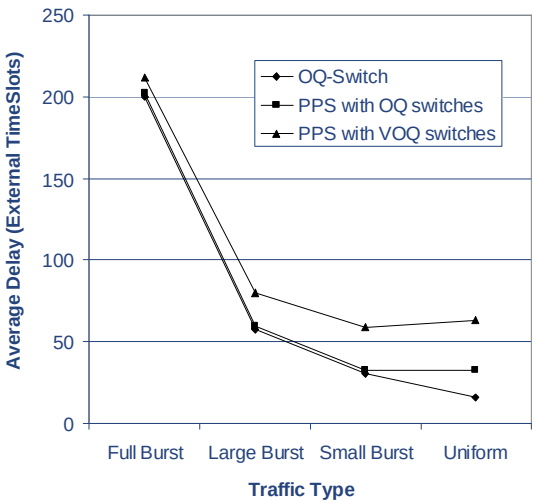


Figure 6. Comparing average delay between an NxN PPS and an NxN output-queued switch

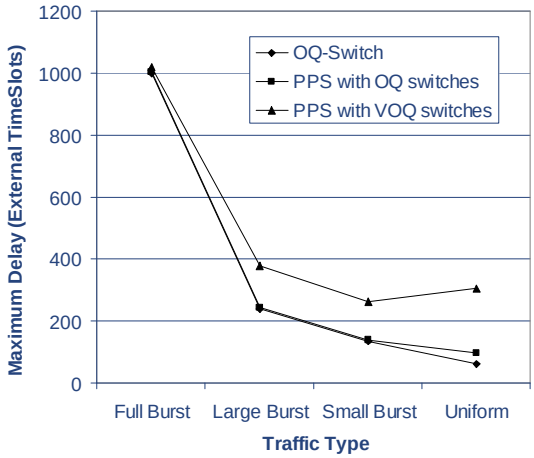


Figure 7. Comparing maximum delay between an NxN PPS and an NxN output-queued switch

Figure 7 shows maximum delay of packets both in the PPS and in the OQ switch. The observations are similar to those of Fig. 6. In general, we can conclude that the performances of the PPS and the OQ switch are comparable. That is, the $O(N^2)$ external time-slots delay penalty of PPS is too rare to happen.

7. Conclusions

We have tried to make the PPS proposed in [2] more simple and implementable in this paper. In our scheme, there are *N* allocators in the demultiplexer each one for one of the packet types. Each allocator allocates the packets of a particular type uniformly to the center-stage switches of the PPS. There is a switching element in each demultiplexer to switch these packets from the allocators to various center-stage switches. We have proved that the maximum amount of the disordering and delay of the packets in this scheme is of $O(N^2)$. While the delay in the previous schemes of the PPS [1,2,9] is bounded like our scheme, those schemes require high speed buffers and have a complicated load balancing algorithm in the demultiplexer and our load balancing algorithm has a complexity of $O(1)$. The simulations show that the performance of our PPS is comparable to that of an equivalent OQ switch. In other words, the $O(N^2)$ external time-slots delay penalty of our PPS is too rare to happen.

References

[1] S. Iyer, A. A. Awadallah and Nick McKeown, "Analysis of a Packet Switch with Memories Running Slower than the Line Rate," *IEEE Infocom, Tel-Aviv, Israel*, 2000.

[2] S. Iyer and N. McKeown, "Making Parallel Packet Switches Practical," *IEEE INFOCOM, Alaska,* USA, 2001.

[3] F. Tucker and C. Zhong, "Photonic Packet Switching: An Overview," *IEICE TRANC. COMMUN.,* vol. E82-B, no. 2, 1999.

[4] M. Renaud, "Key components for Optical Packet Switching," *Proc. of Photonics in Switching*, EFDI, Stockholm, Sweden, 1997.

[5] N. McKeown, A. Mekkittikul, V. Anantharam and J. Wairand, "Achieving 100% Throughput in an Input-Queued Switch", *IEEE Transactions on Communications*, vol. 47, no. 8, pp.1260-1267, 1999.

[6] P. Giaccone, B. Prabhakar, and D. Shah, "Towards Simple, High-Performance Schedulers for High-Aggregate Bandwidth Switches," *Proceedings of the IEEE Infocom*, 2002.

[7] T. Anderson, S. Owicki, J. Saxe and C. Thacker, "High-Speed Switch Scheduling for Local-Area Networks," *ACM Transactions on Computer Systems*, vol. 2, no. 4, pp. 319-352, 1993.

[8] C. Clos, "A Study of Non-blocking Switching Networks," *Bell Systems Technical Journal 32*, 1953.

[9] W. Wang, L. Dong and W. Wolf, "A Distributed Switch Architecture With Dynamic Load Balancing and Parallel Input-queued Crossbars for Terabit Switch Fabrics", *In Proc. IEEE INFOCOM 2002*, pp. 352-360 New York, 2002.

APPENDIX I: CALCULATING THE MAXIMUM DIFFERENCE OF LOAD BETWEEN THE CENTER-STAGE SWITCHES

To simplify the calculations, we assume that all of the center-stage switches and the switching elements are output-queued reaching 100% port utilization under heavy load. We therefore leave the impact of fabric scheduling algorithm out.

Definition 4: We define the PPS to be in the **stationary state** if any synchronization of the allocators in the demultiplexers has not happened for a long time. Hence, in the stationary state, (a) the sizes of all of the output queues in a switching element and (b) conditions in all of the center-stage switches are approximately the same.

To calculate the maximum difference of load based on packet types between the center-stage switches, let us consider the following.

1. If we don't have any synchronization in a demultiplexer, the sizes of various queues in the switching element of that demultiplexer will approximately be equal. If size of the queue *k* of the switching element is $0 \le L_{i,k} < N - \lceil N/K \rceil$, Then for any *k* and *k'* ($1 \le k, k' \le K$) we have $\left| L_{i,k} - L_{i,k'} \right| \le 1$.

2. If any synchronization happens on the queue *k* of the switching element, then the size of that queue will become as many as $0 < d \le N - \lceil N/K \rceil$ packets greater than the size of another queue like *k'* in that switch.

3. We assume the number of packets of type *j* is $V_k^{\,j}$ in the center-stage switch *k* and $V_{k'}^{\,j}$ in the center-stage switch *k'* just before the synchronization. Then, if the PPS is in the stationary state we have $\left| V_k^{\,j} - V_{k'}^{\,j} \right| \le 1$.

4. We assume the allocator(*j*) of the demultiplexer *i* sends $W_k^{\,i,j}$ packets (of type *j*) toward the center-stage switch *k* and $W_{k'}^{\,i,j}$ packets toward the center-stage switch *k'* after the synchronization. Then we have $\left| W_k^{\,i,j} - W_{k'}^{\,i,j} \right| \le 1$.

5. Only one packet among *d* packets synchronized on the queue *k* of the switching element, is of type *j*.

6. The difference in the number of packets of type *j* between the center-stage switches *k* and *k'* will be made because the demultiplexer *i* will send only one packet of type *j* to the center-stage switch *k* during transmission of those *d* packets synchronized on the queue *k* of the switching element, while it may send at most *d* packets of type *j* to another center-stage switch like *k'* in the same time interval. Thus, the maximum difference will be made when *d* has its maximum value. According to the theorem 2, the maximum value of *d* is $N - \lceil N/K \rceil$. Then according to the theorem 3, the queue *k* consists of only the synchronized packets just after that synchronization.

7. The demultiplexer *i*, in the worst case, can send $N - \lceil N/K \rceil$ packets of type *j* to the center-stage switch *k'* after *d* time-slots. If this event happens for all of the demultiplexers just like the demultiplexer *i*, then the number of packets of type *j* which are sent to the center-stage switch *k'* during those *d* time-slots is $N.\left(N - \lceil N/K \rceil\right)$.

8. We assume that synchronization happens on the time-slot t_0. Now we consider the time-slot t_1 which is the time-slot just after transmission of all of those *d* synchronized packets to the center-stage switch *k*. On the time-slot t_1, the number of packets of type *j* in the center-stage switch *k* is $V_k^{\,j} - Serv([t_0, t_1), k, j)$ and in the center-stage switch *k'* is

$$V_{k'}^{\,j} + N.\left(N - \left\lceil \frac{N}{K} \right\rceil \right) - Serv([t_0, t_1), k', j)$$

where $Serv(t, k', j)$ is the number of packets of type *j* that have been serviced in the center-stage switch *k'* during the interval *t*. Also, we have

$$t_1 = t_0 + \left(N - \lceil N/K \rceil\right).$$

9. Now, we consider the time-slot t_2 which is the time-slot just after transmission of all of those $W_{k'}^{\,i,j}$ packets. On the time-slot t_2, the number of packets of type *j* in the center-stage switch *k'* is

$$V_{k'}^{\,j} + \sum_{m=1}^{N} W_{k'}^{\,m,j} - Serv([t_0, t_2), k', j)$$

where $\sum_{m=1}^{N} W_{k'}^{\,m,j}$ is the number of packets of type *j* that have been transmitted to the center-stage switch *k'* by all of the demultiplexers during the interval $[t_0, t_2)$. On that time-slot, the number of packets of type *j* in the center-stage switch *k* is

$$V_k^{\,j} + \sum_{m=1}^{N} W_k^{\,m,j} - N.\left(N - \left\lceil \frac{N}{K} \right\rceil\right) - Serv([t_0, t_2), k, j)$$

.

10. Now we consider the time-slot t_3 which is the time-slot just after transmission of all of those $W_k^{\,i,j}$ packets. On this time-slot, the number of packets of type *j* in the center-stage switch *k* is

$$V_k^{\,j} + \sum_{m=1}^{N} W_k^{\,m,j} - Serv([t_0, t_3), k, j)$$

and in the center-stage switch *k'* is

$$V_{k'}^{\,j} + \sum_{m=1}^{N} W_{k'}^{\,m,j} - Serv([t_0, t_3), k', j).$$

11. We assume all of the center-stage switches are same and their throughput is 100%. If there have been some packets of type *j* in the center-stage switch *k* before that synchronization such that $V_k^{\,j} \geq (N - \lceil N/K \rceil)$, then according to the part 8, the maximum difference in the number of packets of type *j* between the center-stage switches *k* and *k'* on the time-slot t_1 will be made when $V_{k'}^{\,j} \geq V_k^{\,j}$. Then we have

$$Serv([t_0, t_1), k, j) = Serv([t_0, t_1), k', j) = N - \lceil N/K \rceil$$

.

Then, that difference between the center-stage switches *k* and *k'* on the time-slot t_1 will be

$$V_{k'}^{\,j} - V_k^{\,j} + N.\left(N - \left\lceil \frac{N}{K} \right\rceil\right).$$

12. If there has not been any packet of type *j* in the center-stage switches *k* and *k'* just before that synchronization (i.e. $V_k^{\,j} = V_{k'}^{\,j} = 0$), then we have $Serv([t_0, t_1), k, j) = 0$ and $Serv([t_0, t_1), k', j) = N - \lceil N/K \rceil$.

Then, that difference between the center-stage switches *k* and *k'* on the time-slot t_1 will be

$$(N-1).(N - \lceil N/K \rceil).$$

13. In the time-slot t_2 depending on $V_k^{\,j}$ and $V_{k'}^{\,j}$, that difference remains same as the time-slot t_1, because the arrivals of packets of type *j* to the center-stage switches *k* and *k'* are the same in the interval $[t_1, t_2)$.

14. For that synchronization, we have

$$Serv([t_0, t_3), k, j) = Serv([t_0, t_1), k, j) + Serv([t_1, t_2), k, j) + Serv([t_2, t_3), k, j)$$

$$Serv([t_0, t_3), k', j) = Serv([t_0, t_1), k', j) + Serv([t_1, t_2), k', j) + Serv([t_2, t_3), k', j)$$

If the numbers $\sum_{m=1}^{N} W_k^{\,m,j}$ and $\sum_{m=1}^{N} W_{k'}^{\,m,j}$ are large enough, then the last two terms of $Serv([t_0, t_3), k, j)$ and $Serv([t_0, t_3), k', j)$ will be equal. If the condition of part 11 is satisfied, then we have

$$Serv([t_0, t_1), k, j) = Serv([t_0, t_1), k', j) = N - \lceil N/K \rceil$$

.

Then, that difference between the center-stage switches *k* and *k'* on the time-slot t_3 will be $V_{k'}^{\,j} - V_k^{\,j}$.

15. If the condition of part 12 is satisfied (i.e. $V_k^{\,j} = V_{k'}^{\,j} = 0$), then we have $Serv([t_0, t_1), k, j) = 0$ and $Serv([t_0, t_1), k', j) = N - \lceil N/K \rceil$. Then, that difference on the time-slot t_3 will be $-(N - \lceil N/K \rceil)$.

16. Thus, if such synchronizations happen on the center-stage switch *k*, the number of packets of type *j* in another center-stage switch like *k'*, in the worst case which happens on the start of the synchronizations, will become at most $V_{k'}^{\,j} - V_k^{\,j} + N.(N - \lceil N/K \rceil)$ packets greater than their number in the center-stage switch *k*. This event happens if $V_{k'}^{\,j} \geq V_k^{\,j} \geq (N - \lceil N/K \rceil)$. Now we should calculate the maximum value of $V_{k'}^{\,j} - V_k^{\,j}$. The value of $V_{k'}^{\,j}$ will become greater than $V_k^{\,j}$ if

(a) Before those discussed synchronizations, the number of packets of type *j* in the center-stage switch *k'* becomes greater than the number of them in the center-stage switch *k* because of another synchronization on the center-stage switch *k*. Then this difference will be transient and according to the parts 14 and 15, the number of those packets in the center-stage switch *k'* will become equal or less than the number of them in the center-stage switch *k* on the end of that synchronization.

(b) Before those discussed synchronizations, the number of packets of type *j* in the center-stage switch *k'* becomes greater than the number of them in the center-stage switch *k* in the end of another synchronization on the center-stage switch *k'* according to the part 15.

Thus, the worst case will happens in the state (b) which is $V_{k'}^{\,j} - V_k^{\,j} = (N - \lceil N/K \rceil)$. Then, that difference will be equal to

$$\left(N - \left\lceil \frac{N}{K} \right\rceil\right) + N.\left(N - \left\lceil \frac{N}{K} \right\rceil\right) = (N+1).\left(N - \left\lceil \frac{N}{K} \right\rceil\right).$$

As can be seen, it is of $O(N^2)$.

APPENDIX II: CALCULATING THE MAXIMUM AMOUNT OF THE DISORDERING FOR TWO CONSECUTIVE PACKETS OF THE SAME FLOW

Here we again consider the assumptions taken in the beginning of the appendix I. Now we want to calculate amount of disordering, in the worst case, for two consecutive packets of flow *(i,j)* if one of them was added to end of the queue *k* and the other to end of the queue *k'* of the switching element in the demultiplexer *i*. For this purpose, let's consider the following. We assume the parts 1,2,3 are same as appendix I.

4. Only one packet among those *d* packets synchronized on the queue *k* of the switching element is of type *j*.

5. The disordering for two consecutive packets of flow *(i,j)* happens because the first packet may be added to the queue k of the switching element just behind those *d* synchronized

packets, while the second may be added to another queue of that switch like *k'*. Then, if the size of the queue *k'* is less than *d*, the second packet will be sent to the center-stage switch before the first one. Then, the second packet may be sent to the multiplexer before the first, too. Thus, the maximum amount of the disordering happens when *d* has its maximum value. According to the theorem 2, the maximum value of *d* is $N-\lceil N/K \rceil$. Then according to the theorem 3, the queue *k* consists of only the synchronized packets just after that synchronization.

6. The maximum amount of the disordering happens when those two packets are added to the queues *k* and *k'* just after that synchronization, since the queues *k* and *k'* have the maximum difference on that time.

7. Since the allocator(*j*) has participated in the synchronization, for those two packets to be added to the queues *k* and *k'* just after the synchronization, all of the packets entered to the multiplexer *i* must be of type *j* on the time-slot after that synchronization.

8. We assume that synchronization happens on the time-slot t_0. Now we consider the time-slot t_1 which is the time-slot just after transmission of all of packets that have composed the $L_{i,k'}$. Then, the number of time-slots required for the packet 2 to reach the multiplexer *j* is

$$L_{k'}{}^{i} + V_{k'}{}^{j} + recv([t_0,t_1),k',j) - Serv([t_0,t_1),k',j)$$

where $Serv(t,k',j)$ is the number of packets of type *j* which have been serviced in the center-stage switch *k'* during the interval *t* and $recv(t,k',j)$ is the number of packets of type *j* which have been sent to the center-stage switch *k'* during the interval *t* by all of the demultiplexers.

9. Now, we consider the time-slot t_2 which is the time-slot just after transmission of all of those packets synchronized on the center-stage switch *k* in the demultiplexer *i* such that $t_2 = t_0 + L_k{}^{i} + d$. The packet 1 should wait $L_k{}^{i} + d$ time-slots in the demultiplexer *i*. The number of time-slots required for this packet to reach the multiplexer *j* is

$$L_k{}^{i} + d + V_k{}^{j} + recv([t_0,t_2),k,j) - Serv([t_0,t_2),k,j)$$

10. Thus, the maximum amount of the disordering is equal to the maximum value of

$$(L_k{}^{i} + d - L_{k'}{}^{i}) +$$
$$(recv([t_0,t_2),k,j) - recv([t_0,t_1),k',j)) +$$
$$(Serv([t_0,t_1),k',j) - Serv([t_0,t_2),k,j)) + (V_k{}^{j} - V_{k'}{}^{j})$$

11. The maximum value of $(L_k{}^{i} + d - L_{k'}{}^{i})$ is equal to the maximum difference of the sizes of the queues *k* and *k'* in the demultiplexer *i*. Thus according to the corollary 1, we have

$$\max(L_k{}^{i} + d - L_{k'}{}^{i}) = N - \lceil N/K \rceil.$$

12. The maximum value of $(recv([t_0,t_2),k,j) - recv([t_0,t_1),k',j))$ is obtained if no packet of type *j* is sent to the center-stage switch *k'* during the interval $[t_0,t_1)$ while packets of type *j* are sent to the center-stage switch *k* (and probably the center-stage switch *k'*) with the maximum rate (one packet per time-slot) during the interval $[t_1,t_2)$. Then we have

$$\max(recv([t_0,t_2),k,j) - recv([t_0,t_1),k',j)) =$$
$$(N-1)\left(N - \left\lceil \frac{N}{K} \right\rceil\right)$$

13. Since $d = N - \lceil N/K \rceil$, $L_k{}^{i}$ and $L_{k'}{}^{i}$ are zero, and then $t_1 = t_0 + 1$.

14. We assume all of the center-stage switches are the same and their throughput is 100%. If there are enough packets of type *j* in the center-stage switch *k'* just before that synchronization, then according to the part 10, the maximum amount of the disordering for the packets 1 and 2 is obtained if $V_k{}^{j} \geq V_{k'}{}^{j}$. Then we have $Serv([t_0,t_1),k,j) = Serv([t_0,t_1),k',j)$. According to the assumption of part 12, we have $Serv([t_1,t_2),k,j) = N - \lceil N/K \rceil$. Then

$$\max(Serv([t_0,t_1),k',j) - Serv([t_0,t_2),k,j)) =$$
$$-(N - \lceil N/K \rceil)$$

Also, according to the part 16 of the appendix I we have

$$\max(V_k{}^{j} - V_{k'}{}^{j}) = N - \lceil N/K \rceil.$$

15. If there is no packet of type *j* in the center-stage switches *k* and *k'* just before that synchronization, then according to the assumption of the part 12, we have

$$Serv([t_0,t_2),k,j) = N - \lceil N/K \rceil$$

and $Serv([t_0,t_1),k',j) = 0$, then

$$\max(V_k{}^{j} - V_{k'}{}^{j}) = 0$$
$$\max(Serv([t_0,t_1),k',j) - Serv([t_0,t_2),k,j)) =$$
$$-(N - \lceil N/K \rceil)$$

16. Thus, the maximum amount of the disordering is obtained based on the part 14 which is equal to $N.(N - \lceil N/K \rceil)$.

Ali Asghar Khodaparast was born in 1979 in Iran. Mr. Khodaparast received his B.Sc. degree from Shiraz University in 2001 in Computer Engineering. He did his master of science under supervision of Dr. Khorsandi at Amirkabir University of Technology where he graduated in 2004. He was part of the simulation and analysis team working on Iran's national IP network from 2001 to 2003. Mr. Khodarast's research interests are Simulation and Analysis of Computer Networks, Design and Analysis of Network Algorithms and Design and Analysis of Packet Switches.

Siavash Khorsandi was born in 1965 in Iran. He received his BSc and MSc degrees in Electronics from Amirkabir University of Technology in 1987 and 1990 respectively. From 1991 to 1996 he was engaged in a doctoral program at the University of Toronto, Toronto, Canada where he successfully received his PHD in Electrical and Computer Engineering. From 1996 to 1998 he worked for Nortel Networks, Advanced Network Architecture Group. He has been an assistant professor with Amirkabir University of Technology since then. He has served as a member of board of directors of Computer Society of Iran from 1999-2003. His research areas of interest are high speed networking, system modeling and simulation, and network design and analysis.