Delay Analysis of Multicast Switches with Multiple Input Queues

Min Song, Weiying Zhu
Department of Electrical and Computer Engineering
Old Dominion University
Norfolk, VA 23529, USA
{msong,wzhu}@odu.edu

Abstract – This paper examines the delay performance of multicast switches with multiple input queues per input buffer. Under the assumptions of a Poisson uniform traffic pattern, random packet assigning policy, and random packet scheduling policy, we derive the packet delay and service time under different fanouts. To verify this analysis, extensive simulations are conducted with various fanouts, the numbers of queues, and packet arrival rates. It is shown that the theoretical results agree with the simulation results well. The analysis proves that it is possible to predict how much the packet delay could be decreased through introducing more input queues per input buffer.

Keywords – multicast switches, delay analysis, input queuing, queuing theory

I. INTRODUCTION

As the Internet has evolved from a convenience to a mission-critical platform for conducting research, education, and business, the multicast applications such as distributed interactive simulations, distance learning, and video-on-demand are exponentially increasing. As a result, there is an increasing demand for high-performance packet switches that efficiently support multicast traffic. The distinguished feature of a multicast switch is its ability to deliver a multicast packet to multiple output ports. The number of destinations of a packet is termed as fanout. Thus in a multicast packet switch, a packet copying or replication process is needed; thus all destinations could receive a copy of the multicast packet. Due to the connectionless property of IP (Internet Protocol) networks, packets from different flows belonging to different service classes interact with each other when they are multiplexed at the same link of a switch. When multiple packets need to access the same resource at the same time, only one can succeed; other packets must be buffered. The way to organize these buffers is referred to queuing. The process of how to schedule the buffered packets is called packet scheduling. In principle, queuing, copying, and switching are the three major functions performed by a multicast switch. Due to its intrinsic multicasting support [3], [7], a crossbar switch fabric is assumed in this paper. The speed ratio of the switch fabric to the external links is referred to speedup. A speedup of one is assumed in this paper.

While virtual output queuing is a widely accepted queuing scheme for unicast traffic, it is not practical for multicast switches [5], [10]. One queuing scheme for multicast switches is to allocate a single FIFO (first-in first-out) queue at each input port for all multicast traffic [9]. The performance of such multicast switches was studied by Hayes et al. [4] and Hui [5]. While the former focuses on small size switches and the latter focuses on large size switches, both assume one FIFO queue at each input port and the random packet selection policy with fanout splitting [9]. The performance of multicast switches with one input queue per input buffer with no fanout splitting was studied in [1] and [8]. Although the 1-queue scheme can be easily implemented due to its simplicity, the performance of such multicast switches is limited due to the head-of-line (HOL) blocking. To overcome the HOL blocking, window-based queuing was introduced so that packets behind the HOL packet can be transmitted prior to the HOL packet during the current time slot, such as MSCS [2]. The increased scheduling space results in an approximate 10% increase in throughput on average. The price paid for this is that a reordering circuit is needed at each output port and that the input queues must have the random access capability. Another approach to alleviate HOL blocking is to allocate a number of queues at each input port. Thus, the packet scheduler sees more HOL packets. Simulation study for such multicast switches is performed by Gupta and Aziz [3] and Song et al. [10]. In [3], a number of queues are allocated at each input port based on the destination masks. Thus, multiple HOL packets at each input port are eligible for transmission at the same time slot. However, it is not clear how to partition the output ports and how to decide the number of queues at each input port. In [10], it is shown that per-vector queuing on-demand improves the performance. However, the price paid is the complex queue management and poor of scalability. We believe that a small number of input queues are enough to obtain a high switch performance.

The contributions of this paper are twofold. First, we derive the packet delay and service time under different fanouts and number of input queues. As far as we know, this is the first paper that deals with the delay analysis for multicast switches with multiple input queues per input buffer. Secondly, we show that it is possible to calculate how much the packet delay could be decreased through introducing more input queues. This result could be used to choose the optimal number of queues per input buffer according to the delay requirement. Section 2 presents the multicast switch architecture used in this paper. Section 3 introduces two analytical models and derives the packet delay and service time under different fanouts. The theoretical and numerical results are provided in Section 4. Finally, conclusions are given in Section 5.
II. MULTICAST SWITCH ARCHITECTURE

In this paper, the switch under consideration is an \( N \times N \) multicast switch as shown in Figure 1. The switch size \( N \) is assumed to be a very big number. At each input port, there are \( M \) FIFO queues each being referred as the \( k \)-th queue, \( 1 \leq k \leq M \). The number of input queues (\( M \)) at each input port is assumed to be much less than \( N \). The switch fabric is the crossbar with a speedup of one, i.e., at each time slot, no more than one packet can be transmitted from an input and no more than one packet can be delivered to an output. However, it is possible that multiple copies of one HOL packet are forwarded to different output ports during the same time slot. In essence, the service discipline is fanout-splitting. Random policy is adopted to solve the output contention. When more than one HOL packet contends for the same output, one of them will be selected randomly with equal possibilities. The saturation throughput for the switch system described above has been investigated by Song et al. in [11]. In this paper, we mainly address the delay performance analysis. Here, the delay means the input queue delay, which is defined as the number of time slots between the time it enters the input queue and the time when a packet leaves the input queue.

![Figure 1. The multicast switch architecture](image)

III. DELAY ANALYSIS

A. Analysis Models

Some assumptions need to be made before the further discussion. The input multicast traffic arriving at each input link is assumed to follow the Poisson process with the uniform average arrival rate. All the \( k \)-th queues across all the input ports are logically organized as a group \( Q^k \), for example, \( Q^1 \) contains all the first queues at each input port. An incoming packet is randomly assigned to one of the \( M \) queues with equal probability. This is called the \textit{randomly assigning policy}. An arriving multicast packet is destined for a random number of output ports. The average fanout is assumed to be much less than \( N \). In this paper we assume each input multicast packet has a constant fanout of \( f \). (It is straightforward to extend the result in this paper to the case of multicast packets with variable fanouts.) The \( f \) destinations are assumed distinct and uniformly distributed among the \( N \) output ports. Thus, the Poisson process and the uniform distribution properties still follow for each queue. All packets are assumed equal size.

It is likely that there are more than one HOL packets at a single input port. Due to the assumption of speedup of one, only one of them could be selected for transmission in one time slot. Therefore, the input contention may occur. This is solved through performing the matching between input ports and output ports in \( M \) rounds during each time slot. In the \( m \)-th \( (1 \leq m \leq M) \) round, one of the remaining un-served groups is randomly chosen with equal probability. The queues belonging to the selected group are considered as being served during this round even if there is no HOL packet in some of them. The HOL packets of the served queues at the available inputs send requests to all its residual destinations. Once an available output receives the requests, it randomly grants one of the requests and this results in the input received the grant and the output initiated the grant getting matched. This is called the \textit{randomly serving policy}. An input/output port is considered available if it has not yet matched during the previous rounds within the same time slot. A HOL packet will be removed from the input queue when its copies are transmitted to all its destined output ports. The number of time slots spent by a HOL packet transmitting to all its destined output ports is called the service time of the HOL packet.

In the initial model aforementioned, we assume that there is an observer who observes the \( N \times M \) queues at the beginning of each time slot. Therefore, the observer sees \( N \times M/M/1 \) queues with identical statistics property. Because all the input queues served in the \( m \)-th round are symmetric, the number of packets in any one of the queues counted by the observer has the same statistical property as others. We define this number as the queue length of the \( m \)-th round and represent it using \( N_m \). Similarly, all the queues in \( Q^k \) are also symmetrical and have the identical statistics property. The number of packets in one queue of \( Q^k \) counted by the observer is defined as the queue length of the \( k \)-th group and represented as \( L_k \). In the initial model, there is no fixed relationship between the queues served in the \( m \)-th round and the queues in a group \( Q^k \). Because of the adoption of \textit{randomly assigning policy} and \textit{randomly serving policy}, all groups have an identical statistics property. Therefore, for any \( i, j \in [1, 2, \ldots, N] \), we have

\[
E[L_i] = E[L_j] = \mathcal{L},
\]

and

\[
E[\mathcal{L}_i] = E[\mathcal{L}_j] = \mathcal{L},
\] (1)
\[ \text{Pr}\{\text{group } Q \text{ is served in the } m\text{-th round}\} = \frac{1}{M}. \]  

(2)

where \( \mathcal{L} \) is a symbol used to denote the identical average queue length.

Consequently, we have

\[ E[\mathcal{N}_m^+] = \sum_{k=1}^{M} (\text{Pr}\{\text{group } Q \text{ is served in the } m\text{-th round}\} \times E[\mathcal{L}^+]]) = \mathcal{L} \]

(3)

Thus, the average queue lengths during any two rounds equal to each other.

To facilitate the analysis in the rest of this paper, we need to modify the initial model by transforming it into a logically equivalent model. In the modified model, we fix the relationship between the queues served in the \( m \)-th round and the queues in the group \( Q \), i.e., queues in \( Q \) are always served in the \( m \)-th round. Correspondingly, the arrival multicast packets virtually should be assigned into the \( M \) queues randomly according to a dedicated arrival packet assignment ratio for a given offered load. Different arrival packet assignment ratios result in different average system delay. In the modified model, the Poisson process is still followed by the traffic for each queue. At the beginning of each time slot, we still assume that there is an observer to observe the system. Because of the equivalence of the initial model and the modified model, the queue length in the \( m \)-th round counted by the observer should be the same. Therefore, the average queue lengths in any two rounds equal to each other in the modified model. This character determines the unique arrival packet assignment ratio that makes the modified model equivalent to the initial model. In the modified model, during each round, the switch could still be modeled as \( N \) symmetrical and independent \( M/G/1 \) queues. In the remaining of this paper, we are going to derive the average system delay and the delay for each queue using the modified model.

**B. Delay Analysis**

To continue our analysis for the modified model, we define the following notations:

- \( \lambda_{\text{input}} \) : The offered load of an input link.
- \( \lambda_{m,\text{-input}} \) : The packet arrival rate for a queue in \( Q^m \).
- \( \lambda_m \) : The throughput in the \( m \)-th round.
- \( q_m \) : The probability that one destination of the HOL packet in the queue of \( Q^m \) gets service in each time slot.
- \( X_m \) : The service time of the HOL packet in the queue of \( Q^m \).
- \( T_m \) : The delay of the packet in the queue of \( Q^m \).
- \( T \) : The delay of the packet in any queue.
- \( A_m \) : The queue length of the queue of \( Q^m \).

When the overall packet arrival rate is less than the saturation packet arrival rate, i.e., before the system delay becomes infinite, the queuing system is stable and we have

\[ \lambda_{\text{input}} = \sum_{m=1}^{M} \lambda_{m,\text{-input}}, \]  

(4)

and

\[ \lambda_{m,\text{-input}} = \frac{\lambda_m}{f}. \]  

(5)

According to [11], we have

\[ E[X_m] = \sum_{k=1}^{f} \left( \frac{f}{k} \right) (-1)^{k-1}, \]  

(6)

where

\[ q_m = \begin{cases} \frac{2(1-\lambda_m)}{(2-\lambda_m)} & m = 1 \\ \frac{2(1-\sum_{k=1}^{M} \lambda_k)(1-\sum_{k=1}^{M} \lambda_k)}{(2-\lambda_m)} & m \geq 2 \end{cases}. \]  

(7)

According to [5] and [11], we have

\[ E[X_m^2] = \sum_{k=1}^{f} \left( \frac{f}{k} \right) (-1)^{k-1} \frac{2(1-q_m)^{k-1}}{(1-(1-q_m)^f)^2} + E[X_m], \]  

(8)

where \( E[X_m] \) and \( q_m \) are given by (6) and (7), respectively. Based on the \( \text{P-K} \) formula [6], we have

\[ E[T_m] = E[X_m] + \frac{\lambda_{m,\text{-input}} \times E[X_m^2]}{2(1-\lambda_{m,\text{-input}} \times E[X_m])}. \]  

(9)

According to Little’s theory,

\[ E[A_m] = \lambda_{m,\text{-input}} \times E[T_m]. \]  

(10)

As stated early, the average queue lengths of any two queues in the modified model equal to each other, i.e.,

\[ E[A_m] = E[A_{m'}] = \ldots = E[A_{M'}]. \]  

(11)

Therefore, combining (4) through (11), for a given offered load, \( \lambda_{m,\text{-input}} \) and \( E[T_m] \) could be calculated. Consequently, the average system delay could be gained by
An example of the calculation is as follows. For a multicast switch with three queues ($M = 3$) per input buffer, when all the multicast packets have a constant fanout of $4$ ($f = 4$), we substitute different values of $\lambda_{\text{input}}$ within $(0, 0.25)$ into (4) through (11). The corresponding $\lambda_{\text{input}}$ (Y-axis) vs. the offered load (X-axis) is plotted in Fig. 2; the corresponding $E[T_n]$ ($m = 1, 2, 3$) and $E[T]$ are plotted in Fig. 3.

**VI. NUMERICAL RESULTS**

Extensive simulations were performed for different fanouts to verify the analysis results and to infer further conclusions. All simulation runs have been fixed at one million time slots and statistics are collected during the last half million time slots. Infinite queue size is assumed to avoid packet loss. The simulation results are gained through the simulation of a $256 \times 256$ switch with random assigning policy and random selection policy as described in the initial model. The simulation and analysis average delay as the function of the offered load for the switch with $M$ ($M = 2, 3, 4$) queues under different fanouts $f$ ($f = 2, 4, 8$) are plotted in Figs. 4, 5, and 6, respectively. All the simulation results agree with the theoretical results well.
Figure 6.  Average delay vs. offered load for switches with $M$ ($M = 2$, 3, and 4) queues and a constant fanout of 8.

Based on the results shown in Figs. 4, 5, and 6, we can observe the following characters.  When the offered load is much less than the saturation arrival rate, which indicates the switch capacity, there is no noticeable difference among the average delays for multicast switches with two input queues, three input queues, and four input queues.  However, as the offered load increases and approaches to the saturation arrival rate, the average delay difference between multicast switches with different number of input queues becomes more and more obvious.  In each of the three figures, curves shift to right as the number of input queues increases, which indicates that the switch capacity increases accordingly.

In addition, using the equations provided in Section 3, it is straightforward to calculate the number of input queues in order to meet some specific delay requirement for a multicast switch.  Therefore, the optimal number of queues could be derived for a given delay requirement.  When the number of input queues is bigger than a certain value, the delay performance gain by adding more queues is not significant.

V. CONCLUSION

Using the $M/G/1$ model and the queuing theory, the delay performance of multicast switches with multiple input queues is analyzed.  The equation for the average service time is also provided.  Extensive simulations were performed to verify the analysis.  Based on the theoretical and simulation results, we conclude that 1) the delay of a multicast switch decreases as the number of input queues increases; and 2) for a big size multicast switch, when the number of input queues is bigger than a certain value, the delay performance gain by adding more queues is not significant.  Results from this paper prove that it is possible to design advanced packet queuing and scheduling algorithms for a multicast switch with multiple input queues to achieve delay guarantee for differentiated traffic classes by assigning them to different input queues.

ACKNOWLEDGMENT

The authors would like to thank the support from the Department of Electrical and Computer Engineering at Old Dominion University.

REFERENCES