High performance scheduler for multicast switches

N. Narayanan Prasanth*

School of Computer Engineering,
VIT University Vellore,
Tamilnadu, India
Email: narayana.prasanth@gmail.com
*Corresponding author

R. Chithra Devi

Department of IT,
Dr. Sivanthi Aditanar College of Engineering,
Tiruchendur 628205, Tamilnadu, India
Email: chitra_rajan2001@yahoo.co.in

S.P. Raja

Department of CSE,
Kalasalingam University,
Srivilliputur, Tamilnadu, India
Email: avemariaraja@gmail.com

Abstract: Day-by-day, the desideratum for high speed internet connectivity increases and consequently, high performance switches are imminent. These switches require efficient schedulers to reach its maximum potential. In this paper, a high performance scheduler is proposed for buffered crossbar switches to work under multicast traffic. The scheduler is able to reduce the starvation effect and thereby achieves maximum throughput with minimum delay than existing algorithms.

Keywords: crossbar switch; scheduler; starvation; throughput; delay.

Reference to this paper should be made as follows: Prasanth, N.N., Devi, R.C. and Raja, S.P. (2018) 'High performance scheduler for multicast switches', *Int. J. Information Systems and Change Management*, Vol. 10, No. 1, pp.3–15.

Biographical notes: N. Narayanan Prasanth is working as an Associate Professor at the Department of Computer Science and Engineering, VIT Vellore, Tamilnadu, India. He completed his PhD at the Manonmaniam Sundaranar University Tirunelveli, India in 2017. His research interest includes network switch scheduling and routing process. He has research papers published in various national and international journals.

R. Chithra Devi is working as an Assistant Professor at the Department of Information Technology, Dr. Sivanthi Aditanar College of Engineering Tiruchendur, Tamilnadu, India. She is pursuing her PhD at the Anna University Chennai since 2016. Her research interest includes network switch scheduling and its related issues. She has had research papers published in various national and international journals.

S.P. Raja completed his BTech in Information Technology in 2007 from the Dr. Sivanthi Aditanar College of Engineering, Tiruchendur India. He completed his ME in Computer Science and Engineering in 2010 from the Manonmaniam Sundaranar University, Tirunelveli. He completed his PhD in 2016 from the Manonmaniam Sundaranar University, Tirunelveli. His area of interest is image processing and cryptography. He has published more than 40 papers in various conferences and journals.

1 Introduction

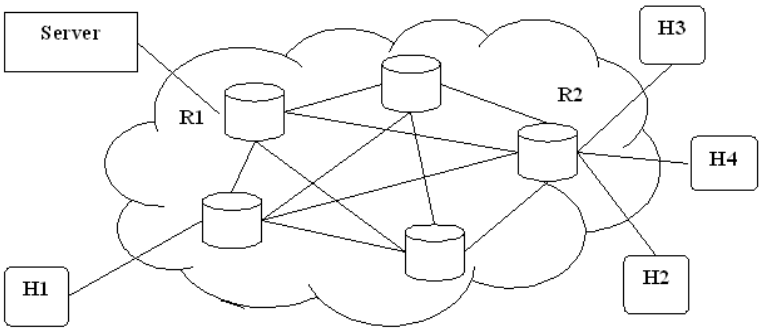
In recent years, advanced high speed communications have become vital to the economic growth of a country. Most of the online applications such as video games and conferencing, internet protocol TV, distance learning, etc. works with multicast traffic (Guo and Chang, 1998). These applications demand high speed internet support in order to attain its features. Applications with streaming video and audio are generated, processed and handled from a single source and are sent to many receivers. For example, a single video feed is transmitted from a video server to thousands of destined receivers. As a result, the entire community raises concerns about bandwidth usage and CPU overheads involved in the same (Sahasrabuddhe and Mukherjee, 2000). In general, if the video server needs to send a video feed to each individual receiver then it has to generate thousands of video streams. This result in so many streams which would most likely saturate the link connected to the server and render the server useless due to the high CPU load incurred. The concept of multicasting allows a one-to-many transmission that consists of a single data stream that is propagated to any host that wants to receive the data stream, without unnecessarily sending that data stream to hosts that do not want to receive the data stream (Eriksson, 1994; Giaccone et al., 2001).

Packet transmission takes place through three different strategies such as unicast, multicast and broadcast (Dai et al., 2008). A unicast packet is sent from a single source to a single destination and is also known as one-to-one communications. A broadcast packet is sent from a single source to all the destinations on the given environment and is also known as one-to-all communication. Multicast traffic is designed to send a packet to the fixed number of destinations in the given environment. It is also known as one-to-many communications. Multicasting ensures only a single copy of a one-to-many transmission is sent over the paths necessary to reach the appropriate destination that need to receive it.

Figure 1 shows the multicast traffic in core routers. R1 and R2 are the routers which receive the message from the server and transfers to the host or next intermediate router. H1, H2 and H3 are the hosts waiting for the information from the server. Any number of hosts can be connected to a router. In Figure 1, H1, H2 and H3 are connected to R2. If all the three host request for the same message from the server, it can send same message three times to all the hosts. Otherwise, it can use the routers to split the message into

three copies and sent to each host. The second method is the better one in terms of performance and cost which is termed as multicasting. Only requirement is the routers must be designed to support multicast traffic (Salama, 1996; Navaz and Balasubramanian, 2016).

Figure 1 Multicast traffic



Multicasting is implemented through layer 2 and layer 3 switches. When multicasting with layer 2 switches, all the attached network devices receive the packets irrespective to their requirements. On the other case, when multicasting with layer 3 switches, the packets are sent only to the exact receiver who needs them. Multicasting allows for a reduction in bandwidth usage, increasing network efficiency and performance. Any application that needs to send large amount of information to fixed number of multiple destination then Multicast is the best option (Navaz and Balasubramanian, 2015). Example application includes:

- Multimedia applications that consume high bandwidth, such as streaming video and TV servers.
- Remote conferencing system, i.e., voice.
- Software distribution applications.
- Routing protocols such as open shortest path first (OSPF), enhanced interior gateway routing protocol (EIGRP) and routing information protocol (RIP) v2.

In this paper, a high performance scheduling scheme is proposed and its performance is analysed under various uniform and non-uniform traffic patterns in multicast environment. Section 2 discusses about the BCS support to multicast scheduling. Section 3 provides the proposed scheduler and Section 4 analyse the throughput and delay performance of the proposed scheduler. Section 5 concludes the paper.

2 Multicasting in BCS

To support multicasting, BCS is considered as the suitable architecture due to its scalability, flexibility, low cost and intrinsic multicast capabilities (Marsan et al., 2003; Prasanth and Balasubramanian, 2015). BCS holds buffer in the switch fabric rather than in the line cards which means the switch and buffer implemented in a single chip, thereby

reducing the implementation cost (Chen et al., 2016). At each timeslot, BCS requires two schedulers to switch a cell namely, arrival and departure scheduler. Arrival scheduler selects a cell from the HOL of a queue and places it in an empty crosspoint buffer. Departure schedule selects a cell from a non-empty buffer and transferred it to output through destination port. At each timeslot, based on the employed scheduling algorithm, a cell is scheduled from VOQ to one or more crosspoint buffer and from buffer to output port. Amount of cells stored in the crosspoint buffer is based on its size and it is practically viable to implement multi-sized buffer in the crosspoint of a switch [Pan and Yang, (2005), p.23].

Multicasting is a process of transferring a cell from a single source (port) to multiple destinations at minimum cost and time. Number of destination ports that a cell needs to be transferred in a switch is said to be fanout set (Liu et al., 2016; Mhamdi, 2009). In BCS/VOQ architecture, copies of the cell to be multicast are placed in the alternate queues of VOQ such that multiple copies can be delivered at the same timeslot. This is possible subject to the availability of empty buffers and employed scheduling algorithm. These situations are dealt with two procedures namely *no fanout splitting* and *fanout splitting* (Lee and Un, 1997; Prabhakar et al., 1997).

No fanout splitting will switch the cell only if all the buffers corresponding to destination ports are empty, i.e., cell will be switched only once. Fanout splitting will switch the cell irrespective to the availability of buffers at the crosspoint and the process continues until all the port receives the cell. No fanout splitting is easy to implement but offers low throughput because of work conserving. Fanout splitting is complex to implement since cells are transferred partially to the destinations ports (Hui and Renner, 1994).

Several scheduling algorithms are proposed for the multicast traffic under BCS (Sun et al., 2005). At each timeslot, a HOL cell is switched to empty crosspoint buffer results in minimum residue left. Another scheduling algorithm opts for a cell which has maximum possible destinations. A new algorithm works with respect to maximum service ratio (MSR), which is the number of reachable destination outputs divided by the fanout number of a cell. Mhamdi et al. (2007) proposed multicast crosspoint round robin scheduler (MXRR) scheduler and its performance are compared as best to other schedulers such as TATRA and iSLIP (Mhamdi and Hamdi, 2004). In Divanovic et al. (2012), author proposed longest queue first and round robin scheduler for multicast support and their performances are analysed. From all the above work, it is understood that most of the algorithms try to cover certain issues but fails with other requirements. None of the algorithms proved efficient in terms of throughput and delay performance. Also, these algorithms have taken a very less effort to reduce the starvation effect.

The prolonged waiting period of a cell in a virtual output queue is called as starvation. Starvation is one of the primary factors which disturb the performance of a scheduling algorithm. TPQRS and D-PQRS are the commonly used unicast algorithms which made an impressive attempt to reduce the impact of starvation. PQRS uses priority queue scheduler as input schedule and round-robin algorithm as output schedule. It reduces the difference between the average waiting of the queues in a VOQ and thereby reduces starvation. Simulation ensures that the performance of the switch gets increased by the reducing the starvation. PQRS is updated as D-PQRS which uses delay-based priority queue scheduler as input schedule and round-robin algorithm as output schedule. It further reduces the starvation effect by stabilising the average waiting time. Simulation

result shows that D-PQRS achieves high throughput with minimum delay by reducing the starvation effect in the input queues.

3 DPQRS-M scheduling scheme

D-PQRS scheduler for multicast support is called as D-PQRS-M or DPQRS-M. Proposed algorithm has taken at-most care to reduce the starvation effect and is given.

3.1 Arrival schedule

1 Incoming cells are stored in the VOQ's of the switch.

2 Multicasting cells are stored in alternate queues of the VOQ.

3 At each timeslot, a cell is transferred to multiple destination ports based on fanout set subject to the availability of buffer.

4 Selection of cell to be switched is based on the priority value (PV) of a queue.

5 Initially, PV is assumed to be the size of the queue and for each timeslot, PV is updated with respect to bonus value (BV), i.e., PV = PV + BV.

6 BV is the waiting time of HOL cell in a queue which is represented in terms of timeslots.

7 During every schedule, PV of the currently used queue will be reduced by one.

8 If two queues have same PV, then queue which is not recently selected will be allowed to switch their cell or queue which is not selected at least once will be selected or FIFO principle will be used.

9 If high prioritised queue is empty, then switch the cell from next high priority queue.

3.2 Departure schedule

Round robin scheduling (RRS) is a proven output queued scheduler and is used to switch the cell from buffer to destination ports unless the buffer is empty.

3.3 An example

The proposed scheduling algorithm made an attempt to overcome the starvation problem by prioritising the waiting cells with BV. BV for a cell is computed based on its waiting time in the queue. For example, consider a 4 × 4 switch works under uniform switch which assumes all the input cell arrived at the same time. For each schedule, BVs are computed for the queue cells of each VOQ and are shown in Table 1.

At timeslot T_0, numbers of cells entered in the queues are (5, 7, 8, 6) and are considered as queue PVs. As per the proposed scheduler, Q_2 is selected for the first schedule because it has the highest PV and the priority table at T_1 is updated as (6, 8, 7, 7). At T_1, Q_1 is selected for the schedule and Table 1 is updated as (7, 7, 8, 8). At the third timeslot, Q_3 is selected for schedule even though Q_2 and Q_3 have same PV. This

is because Q_2 is already switched once but not Q_3. The process continues until all the incoming cells are scheduled or the scheduling period gets over. From the above example, it is clear that the proposed approach made a significant effort to serve the cells equally from all the queues and thereby reduces the starvation effect.

Table 1 BV computation

Timeslots ↑ / *Queues* ↓	T_0	T_1	T_2	T_3
Q_0	5	$6^{(+1)}$	$7^{(+1)}$	$8^{(+1)}$
Q_1	7	$8^{(+1)}$	$7^{(-1)}$	$8^{(+1)}$
Q_2	8	$7^{(-1)}$	$8^{(+1)}$	$9^{(+1)}$
Q_3	6	$7^{(+1)}$	$8^{(+1)}$	$7(^{-1)}$

4 Performance analysis

A 4 × 4 buffered crossbar switch is used to multicast the packets with buffer size 1. Each packet is segmented into equal sized cells in the input side and is reassembled in the output side before they move out of the switch. JNetworkSim developed by Nick Mckeown Group of Stanford University is used to simulate the switching process with a total amount of 10,000 cells. At each timeslot, multicasting a cell is implemented through both the *fanout splitting* and *no fanout splitting* principle to analyse its outcomes. Proposed algorithm is tested and compared with MXRR algorithm (Dai et al., 2008) under non-uniform unbalanced and non-uniform bursty traffic patterns.

4.1 Average waiting time analysis

Waiting time of a cell is a measure of total time spent by a cell in the queue of a VOQ. The primary objective of every scheduler is to reduce the waiting time of a cell so that the overall performance will increase. Waiting time of a cell is measured in terms of milliseconds/timeslots. Figure 2 shows the averating waiting time of a VOQ under no fanout splitting scheme. For DPQRS-M, queue 1 of the VOQ has the minimum waiting time of 3.7 ms and queue 3 has the maximum waiting time of 5.1 ms with a difference between them is 1.4 ms. For MXRR, queue 3 of the VOQ has the minimum waiting time of 4.3 ms and queue 2 has the maximum waiting time of 6.7 ms with a difference between them is 2.4 ms. In Figure 3, under no fanout splitting scheme, the difference between the minimum and maximum average waiting time is 1.2 ms for DPQRS and 1.9 ms for MXRR. Therefore in both the cases, the average waiting between the queues has been considerably reduced in DPQRS-M compared to MXRR, which means starvation is also reduced.

Figure 2 Averating waiting time of a VOQ under no fanout splitting scheme (see online version for colours)

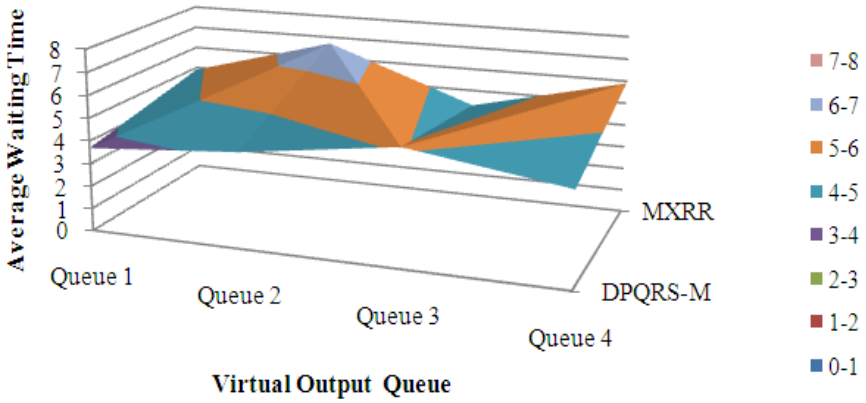
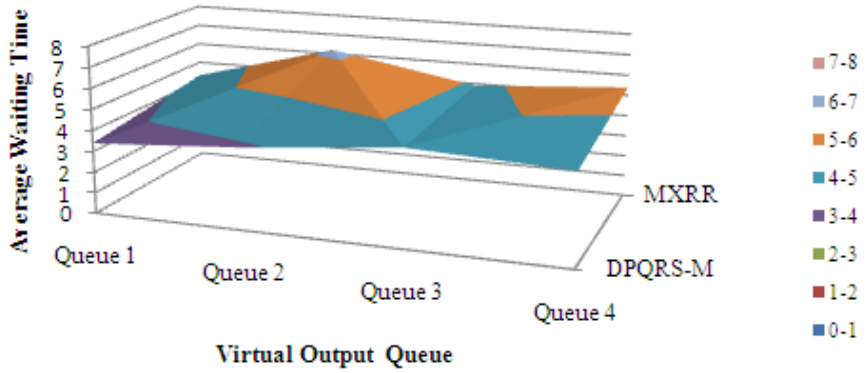


Figure 3 Averating waiting time of a VOQ under fanout splitting scheme (see online version for colours)



4.2 Throughput analysis

Throughput performance of DPQRS-M and MXRR algorithms are analysed under non-uniform bursty traffic and is depicted in Figure 4. It adopts no fanout principle. DPQRS-M throughput performance is greater than 70% when minimum load (20%) is supplied and thereafter throughput drops up to 50% when maximum load is supplied. Comparing with MXRR, less than 5% of throughput is separating both the algorithms in favour of DPQRS-M. Figure 5 shows the throughput performance of DPQRS-M and MXRR algorithms under non-uniform bursty traffic with a burst size of 100. Both the algorithms offer similar performance slightly lesser than 70% when minimum load is offered and greater than 40% for the maximum load offered. To improve the performance further, a speedup of two can be introduced in the switch against line speed. But considering the implementation cost, speedup is not used in our work. It is understood that the throughput performance is average because of the no fanout splitting.

Figure 4 Throughput performance of DPQRS-M and MXRR under non-uniform unbalanced traffic using no fanout splitting (see online version for colours)

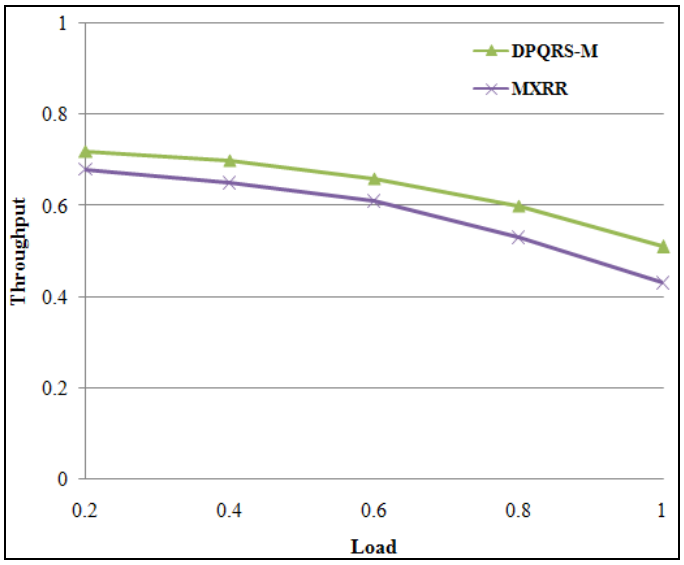


Figure 5 Throughput performance of DPQRS-M and MXRR under non-uniform bursty traffic using no fanout splitting (see online version for colours)

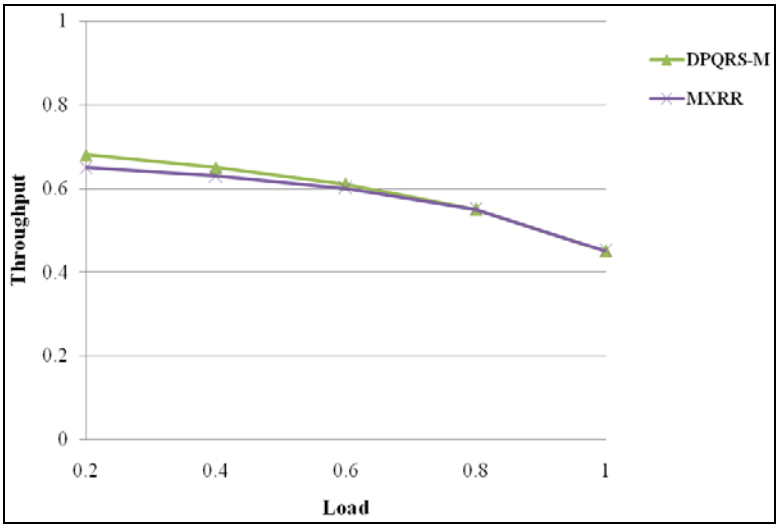


Figure 6 shows the throughput performance of DPQRS-M and MXRR algorithms under non-uniform unbalanced traffic which uses fanout splitting procedure. Overall throughput is increased by 20% for DPQRS-M which is slightly greater than 90% for minimum load and lesser than 80% when maximum load is offered. For any load, DPQRS-M throughput performance is 5% better than MXRR. Figure 7 shows the throughput performance of DPQRS-M and MXRR algorithms under non-uniform bursty traffic which uses fanout splitting procedure. Again, a 20% increase is noted as similar to unbalanced traffic.

Throughput is slightly less than 90% and greater than 70% for minimum and maximum load, respectively. DPQRS-M and MXRR schedulers' throughput are much improved under fanout splitting and difference between them is around 5%. From the simulation results, it is understood that proposed scheduler offers better throughput performance than MXRR. Fanout splitting principle should be used to achieve maximum throughput performance.

Figure 6 Throughput performance of DPQRS-M and MXRR under non-uniform unbalanced traffic using fanout splitting (see online version for colours)

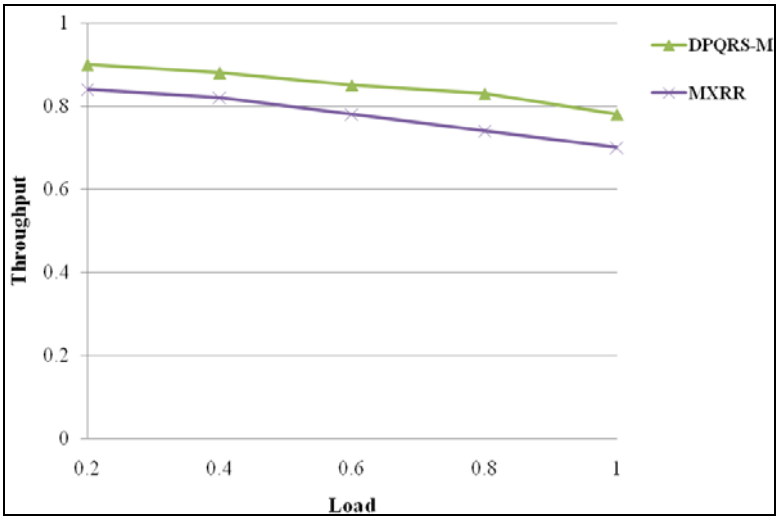
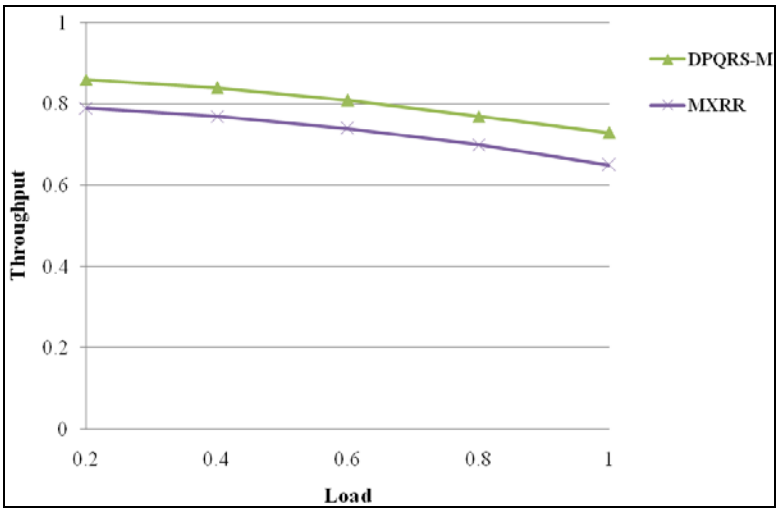


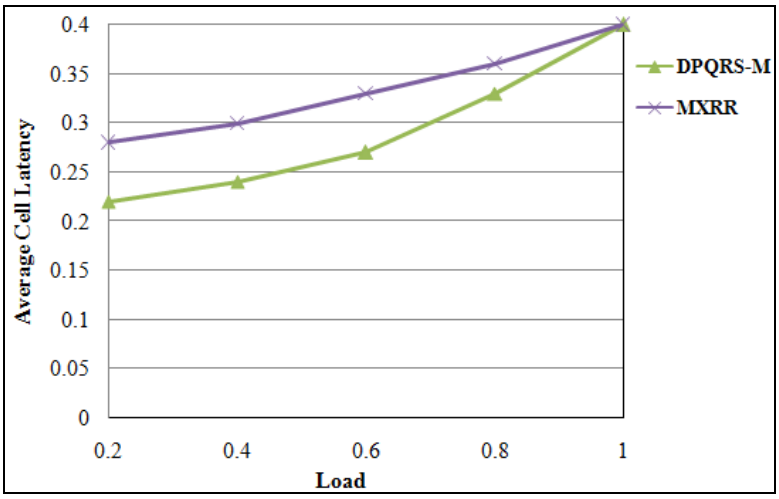
Figure 7 Throughput performance of DPQRS-M and MXRR under non-uniform bursty traffic using fanout splitting (see online version for colours)



4.3 Delay analysis

Average cell latency of D-PQRS-M and MXRR algorithms are analysed under non-uniform unbalanced traffic using no fanout splitting. Figure 8 depicts the average delay performance of DPQRS-M compared to the MXRR running on 4 × 4 buffered crossbar switch. For DPQRS-M, ACL is 22% for minimum load and 40% for maximum load. Comparing with MXRR, DPQRS-M ACL is better by 5% when minimum load is offered and when load exceeds 70% (p = 0.7), both the algorithms deliver similar delay performance. Figure 9 shows the average cell latency of DPQRS-M and MXRR under non-uniform bursty traffic using no fanout splitting. For bursty traffic, both the algorithm offers similar delay performance (30%) when minimum load is offered. When maximum load is supplied, delay increases up to 55% for MXRR which is 7% greater than DPQRS-M. Similar to throughput, switch delay performance also suffered under no fanout splitting.

Figure 8 Average cell latency of DPQRS-M and MXRR under non-uniform unbalanced traffic using no fanout splitting (see online version for colours)



Average cell latency of D-PQRS-M and MXRR algorithms are analysed under non-uniform unbalanced traffic using fanout splitting. Figure 10 depicts the average delay performance of D-PQRS-M under non-uniform unbalanced traffic running on 4 × 4 buffered crossbar switch. Delay performance of D-PQRS-M is much improved to 5% for minimum load and 20% for maximum load. For any load structure, D-PQRS-M delay performance is better than MXRR by 10%. Figure 11 shows the average cell latency of DPQRS-M and MXRR under non-uniform bursty traffic using fanout splitting. Delay performance of DPQRS-M is under 10% for minimum load and slightly greater than 20% for maximum load. A delay of 10% difference is found between DPQRS-M and MXRR schedulers in favour of DPQRS-M.

Figure 9 Average cell latency of DPQRS-M and MXRR under non-uniform bursty traffic using no fanout splitting (see online version for colours)

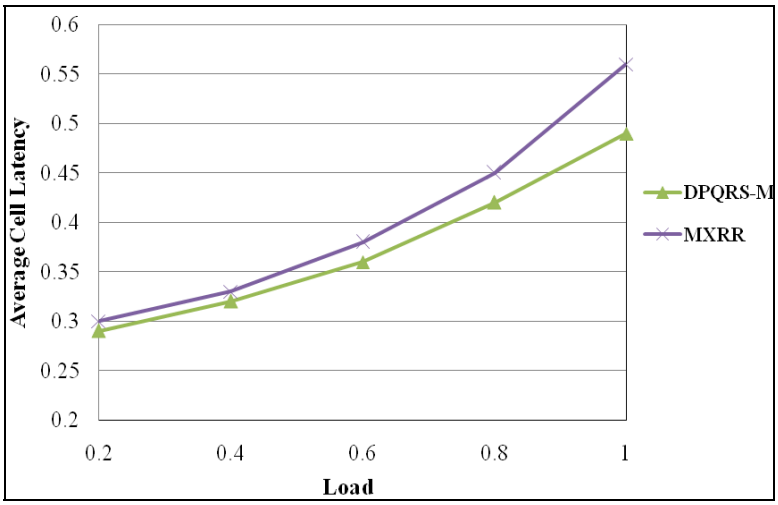
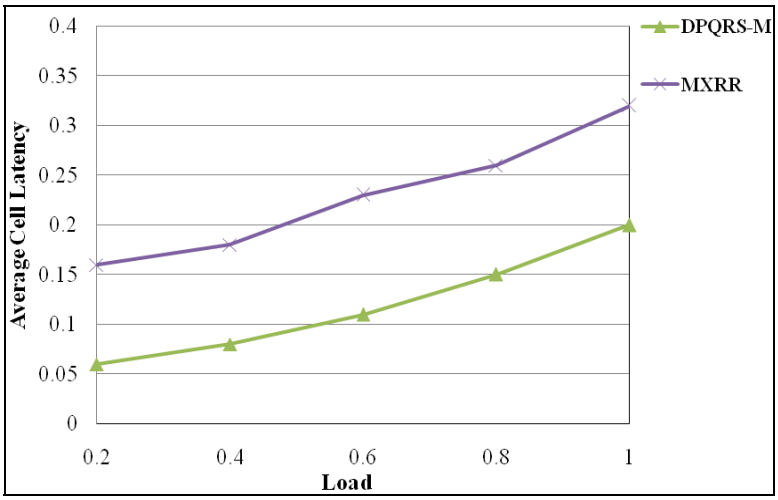
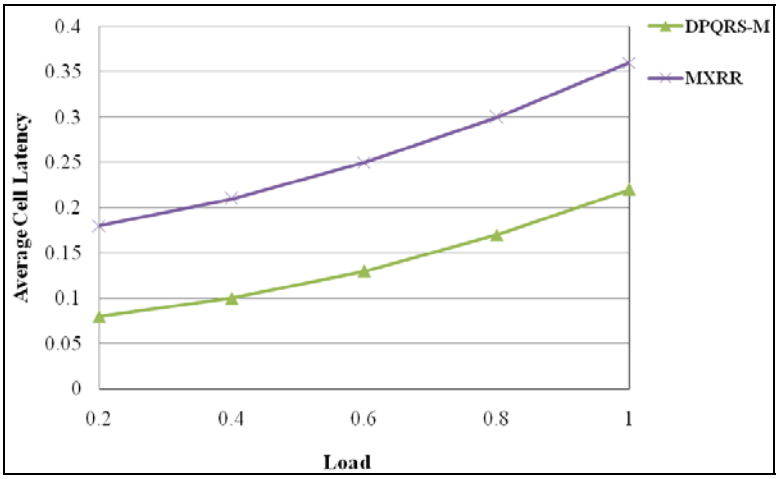


Figure 10 Average cell latency of DPQRS-M and MXRR under non-uniform unbalanced traffic using fanout splitting (see online version for colours)



From the simulation results, it is proved that DPQRS-M provides better delay performance than MXRR for any non-uniform load structures. As discussed, scheduler deployed fanout splitting procedure delivers better performance than no fanout splitting.

Figure 11 Average cell latency of DPQRS-M and MXRR under non-uniform bursty traffic using fanout splitting (see online version for colours)



5 Conclusions

A DPQRS-M algorithm is proposed with the intent of reducing the starvation effect in a switch scheduling process and thereby increasing the switching performance. DPQRS-M for multicast support is analysed with respect to throughput and delay performance and the results are compared with MXRR algorithm. Proposed scheduler is implemented in a BCS under unbalanced and bursty traffic patterns. Simulation is carried out under no fanout splitting as well as fanout splitting strategies to transfer the cells in and out of the switch. Simulation result shows D-PQRS-M offers better throughput and delay performance than MXRR for any non-uniform traffic patterns. D-PQRS-M performance under fanout splitting procedure is 20% greater than non-fanout splitting. In the future, the performance of DPQRS-M can be analysed with different crossbar buffer sizes and also with multistage switching architectures.

References

Chen, G., Zhao, Y. Pei, D. and Sun, Y. (2016) 'Analyzing the impact of buffer capacity on crosspoint-queued switch performance', *Journal of Communications and Networks*, Vol. 18, No. 3, pp.523–530.

Dai, Y., Su, J.S. and Zhang, Y. (2008) 'A coordination scheduling mechanism to guarantee packet ordering in parallel packet switch', *International Journal of Electronic Security and Digital Forensics*, Vol. 1, No. 4, pp.362–373.

Divanovic, S., Kovacevic, V., Radonjic, M., Yoshigoe, K. and Radusinovic, I. (2012) 'Crosspoint queued switch performance analysis under multicast traffic', *20th Telecommunications Forum TELFOR 2012*, Belgrade, pp.226–229.

Eriksson, H. (1994) 'MBONE: the multicast backbone', *Communications of the ACM*, Vol. 37, No. 8, pp.54–60.

Giaccone, P., Prabhakar, B. and Shah, D. (2001) 'Towards simple, high-performance schedulers for high-aggregate bandwidth switches', *Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, Vol. 3, pp.1160–1169.

Guo, M.H. and Chang, R.S. (1998) 'Multicast ATM switches: survey and performance evaluation', *ACM SIGCOMM Computer Communication Review*, Vol. 28, No. 2, pp.98–131.

Hui, J.Y. and Renner, T. (1994) 'Queuing analysis for multicast packet switching', *IEEE Transactions on Communications*, Vol. 42, No. 2, pp.723–731.

Lee, J.Y. and Un, C.K. (1997) 'Performance analysis of two fanout splitting schemes for a multicast packet switch with capacity m', *Elsevier – Performance Evaluation*, Vol. 29, No. 1, pp.1–14.

Liu, K., Yan, J., Lu, J. and Chen, X. (2016) 'Predictive unicast and multicast scheduling in onboard buffered crossbar switches', *IEEE Communications Letters*, Vol. 20, No. 3, pp.498–501.

Marsan, M.A., Bianco, A., Giaccone, P., Leonardi, E. and Neri, F. (2003) 'Multicast traffic in input-queued switches: optimal scheduling and maximum throughput', *IEEE/ACM Transactions on Networking*, Vol. 3, No. 11, pp.465–477.

Mhamdi, L. (2009) 'On the integration of unicast and multicast cell scheduling in buffered crossbar switches', *IEEE Transactions on Parallel and Distributed Systems*, Vol. 20, No. 6, pp.818–830.

Mhamdi, L. and Hamdi, M. (2004) 'Scheduling multicast traffic in internally buffered crossbar switches', *IEEE International Conference on Communications*, Vol. 2, pp.1103–1107.

Mhamdi, L., Gaydadjiev, G. and Vassiliadis, S. (2007) 'Efficient multicast support in high speed packet switches', *Journal of Networks*, Vol. 2, No. 3, pp.28–35.

Navaz, K. and Balasubramanian, K. (2015) 'OQSMS: optimal queue selection based multicast scheduling algorithm for input-queued switches', *Australian Journal of Basic and Applied Sciences*, Vol. 9, No. 27, pp.373–378.

Navaz, K. and Balasubramanian, K. (2016) 'Multicast due date round robin scheduling algorithm for input queued switches', *International Journal of Computer Network and Information Security*, Vol. 8, No. 2, pp.56–63.

Pan, D. and Yang, Y. (2005) 'FIFO-based multicast scheduling algorithm for virtual output queued packet switches', *IEEE Transaction on Computers*, Vol. 54, No. 10, pp.1283–1297.

Prabhakar, B., McKeown, N. and Ahuja, R. (1997) 'Multicast scheduling for input-queued switches', *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 5, pp.885–866.

Prasanth, N. and Balasubramanian, K. (2015) 'Performance analysis of buffered crossbar switch scheduling algorithms', *International Journal of Information and Computer Security*, Vol. 7, No. 1, pp.49–63.

Sahasrabuddhe, L.H. and Mukherjee, B. (2000) 'Multicast routing algorithms and protocols: a tutorial', *IEEE Network*, Vol. 14, No. 1, pp.90–102.

Salama, H.F. (1996) *Multicast Routing for Real-time Communication of High-speed Networks*, Doctoral dissertation, North Carolina State University.

Sun, S., He, S., Zheng, Y. and Gao, W. (2005) 'Multicast scheduling in buffered crossbar switches with multiple input queues', *IEEE Workshop on High Performance Switching and Routing*, pp.73–77.

Websites

http://www.cisco.com/c/en/us/td/docs/ios/solutions_docs/ip_multicast/White_papers/mcst_ovr.html

http://yuba.stanford.edu/JNetworkSim/