I. J. Computer Network and Information Security, **2016, 2, 56-63**
Published Online February 2016 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2016.02.07

Multicast Due Date Round-Robin Scheduling Algorithm for Input-Queued Switches

Navaz K
Manonmaniam Sundaranar University, Department of CSE, Tirunelveli, 627 012, Tamilnadu, India
E-mail: navazit@gmail.com

Dr.Kannan Balasubramanian
Mepco Schlenk Engineering College, Department of CSE, Sivakasi, 626 005, Tamilnadu, India
E-mail: kannanbala@mepcoeng.ac.in

***Abstract*—**In recent years, the Internet has incremented the several incipient applications that rely on multicast transmission. This paper discusses the challenges of scheduling algorithms for multicast in high-speed switches that reduces the overhead of adaptation by selecting a HOL (Head of Line Cell) using Round Robin pointer. The objective of this paper is to design a scheduling algorithm called MDDR (Multicast Due Date Round-Robin) scheduling to achieve maximum throughput and low delay that has two phases request and grant. In request phase, MDDR assigns a Due Date (Request Time Slot) for HOL cells of each queue in the input port. Round Robin Pointer is utilized in the grant phase to select a request if HOL occurs. MDDR achieves more preponderant performance than MDRR (Multicast Dual Round-Robin), since the request shall be made when the Due Date is reached. MDDR mainly minimizes many requests made for output ports and time complexity. The simulation results show that the proposed algorithm has good switching performance in throughput and average time delay under Bernoulli and bursty traffic conditions.

***Index Terms*—**Due Date, Grant, Multicast, Request, Scheduling.

I. Introduction

The number of emerging applications of multicast are growing day by day and there is a requisite for the design of high-speed switches/routers. Multicasting is useful for many types of one-to-many applications, such as multimedia, video conferencing, collaborative computing and Data casting(file Distribution) or database synchronization etc. Mobile computer support such as remote address book updating and distribution of organizational publications. Multicast IP conserves bandwidth by forcing the network to do packet replication only when necessary, and offers an attractive alternative to unicast transmission for the delivery of network ticker tapes, live stock quotes, multiparty video-conferencing, and shared whiteboard applications (among others). This is important because of the growing proportion of multicast traffic on the Internet (audio, video, IPTV, etc.). If we consider the example in Fig. 1, and assume that the 2 hosts connected to router D are receiving the same media content from the server. If the server sends the same message to hosts, H3 and H4, it either sends the same message two times (one per destination) or it can send the message only once over routers B and D. Once reaching D, the message gets split into two copies, one copy per destination host. Obviously, the latter case is a better choice as it optimizes the network resources and the time taken for the hosts to receive the data. In order to achieve this, routers B and D must be designed to support multicast traffic.

In addition, people are more interested in sharing knowledge and information for various purposes. Innovations in information sharing are continuously accelerated to cater user needs in such environments. These motives have encouraged the construction of sophisticated environments for effective communication to deliver information.

Internet users are demanding faster and higher-quality services. To cater such requirements, Content Delivery Networks (CDNs) were introduced. CDNs are implemented to achieve low-latency content delivery such as; data streaming, on-line gaming and e-commerce web accesses by placing content servers near the customer. In [16] author proposed a new method of CDN Request Redirection (RR) (SoR-based RR), which is designed to redirect packets based on the content of packets and the status of content servers using an SoR as an edge router of a CDN. In order to minimize the delay in the network, multicast supporting edge router is essential.

It is important to note that the applications for IP Multicast are not solely limited to the Internet. Multicast IP can also play an important role in large distributed commercial networks. The demand for network bandwidth is very essential and many of the networking applications require high speed switching for multicast traffic at the switch/router level to preserve network bandwidth. It causes an incrementing interest in the input-queued switches.

A switch consists of three components: 1) input queues for cells arriving at the input links 2) output queues for cells exit on output links 3) A switch fabric for transferring cells from the inputs to the desired outputs. LAN and Asynchronous Transfer Mode (ATM) switches are considered as a high performing internetworking protocol and uses a crossbar switch based on switched backplane. Further, these systems use the input queues for holding packets which are waiting to traverse through the switch fabric. Thus, it is known that the first in first out (FIFO) input queues can be used to maintain packets.

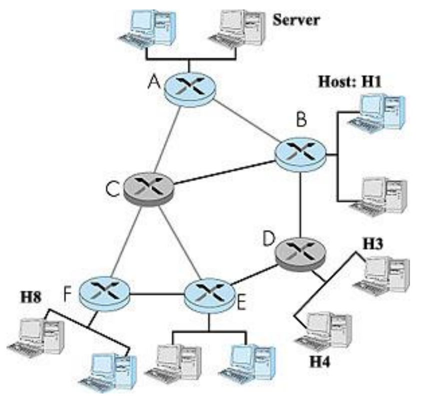

Fig.1. Multicast Traffic Support in Core Routers

A scheduling algorithm is utilized to configure the crossbar switch, to decide the order in which packets will be accommodated. Many integrated scheduling algorithms have been proposed earlier. They have been mainly proposed for input queued (IQ) crossbar-fabric-predicated switching architecture but multicast scheduling, mainly concerns how to transmit as many cells as possible from input to output. In the unicast traffic, Head-of-Line (HOL) blocking quandary occurs that is induced by first-in-first-out (FIFO) queue which gets avoided by utilizing virtual output queuing(VOQ) technique. Here, in this type of technique every single input maintains a separate queue for each output[7]. Numerous unicast scheduling algorithms have been proposed so far. iSLIP is the fast and efficient algorithm which has achieved 100% throughput in a single iteration for uniform traffic. In [11], MRR (Modified Round Robin Algorithm) proposes that it can show a performance equivalent to iSLIP, yet require less number of processing steps. Using multicast traffic [2], [5] we can avoid HOL blocking by utilizing 2^N -1 queues for each input port in $N \times N$ switch. This type of queue architecture is called Multicast Virtual Output Queuing (MC-VOQ). However, in the medium/sizably voluminous switches, because of its low Scalability, it is virtually not tackled. One such practical queuing scheme utilized for multicast switches is to assign a single FIFO queue at each input for all multicast traffic, however, the HOL blocking quandary limits the throughput. Whereas the other algorithms [1], [3] considered a circumscribed number of FIFO queues is maintained at each input to reduce the HOL blocking problem. Thus queuing architecture is denominated as k-MC-VOQ and performance of these multicast switches are analyzed theoretically [12], [15]. As the link speed grows dramatically, high speed switches will have less time to perform scheduling process. As a result, iterative schemes and high matching overhead would cause delay, Matching overhead scales up very expeditiously, the link speed and the switch size increases, the desideratum for simple and high performance switches becomes very essential. Hence this paper, proposes an incipient scheduling algorithm called Multicast Due Date Round-Robin (MDDR).

The rest of the paper is organized as follows. In section II- the works related to designing multicast scheduling algorithm is discussed. In Section III- assumed switching architecture and proposed algorithm design is summarized. In section IV- the performance evaluation and analysis of the result is presented. Finally, we conclude the paper in Section V.

II. Related Works

The majority of subsisting multicast switches [9]-[10] require in-switch packet replication, and a sophisticated central scheduler to maximize performance of the switch. TATRA [10] is a multicast algorithm predicated on single FIFO queue, where each input port has a single prevalent queue for both unicast and multicast traffic. The central scheduler maintains the N virtual queues and each is destined for one output port. In each time slot, the head-of-line(HOL) packet of each input queue is scheduled to join different virtual queues according to its destination output ports. Fan-out splitting [4], which sanctions a multicast packet to be sent to a subset of its outputs, is adopted to increment the switch throughput. However, TATRA suffers from serious HOL blocking because of its single queue nature. TATRA evades starvation but is additionally perplexed to implement a hardware due to heavy computations.

To minimize the HOL blocking, multiple dedicated multicast queues have been utilized in [3] and [14]. In [14], each input port has a set of multicast queues. When a multicast packet arrives, it culls one of the multicast queues to join according to its load balancing policy. In each time slot, the scheduling priority is given to either a unicast packet or a multicast packet. According to the accommodation ratio of the two types of traffic. An iterative scheduling algorithm is adopted to maximize the switch throughput.

To reduce the HOL blocking further, a multicast packet split scheme is proposed in [3]. In [3], the set of output ports is divided into m non–overlapped subsets, and each input port maintains m unicast / multicast shared queues and each is dedicated to a subset of outputs. When a multicast packet arrives, if its fan-out set wHOLly fit in a queue, it will join the queue; otherwise, the multicast packet is divided into smaller ones (each with a modified fan-out set) to join multiple queues. Again, an iterative scheduler is adopted to maximize throughput.

ESLIP [6] adopts the VOQ structure to buffer unicast packets and puts all the multicast packets in a special single queue at each input port. It utilizes a variant of the iSLIP algorithm to schedule mixed unicast and multicast traffic. As can be expected, ESLIP eliminates the HOL blocking for unicast traffic, but not for multicast traffic.

In an extreme situation, where all the incoming packets are multicast packets, ESLIP cannot benefit from the VOQ structure and it authentically works on the single input queued switch.

In [9], an efficient multicast scheduling algorithm called FIFOMS is proposed to avoid HOL blocking. The rudimentary conception is to discretely store unicast/multicast packets and memory addresses. FIFOMS uses prevalent unicast VOQs as pointer queues. More concretely, when a multicast packet with a fan-out of f (f = 1 for unicast packet) arrives, it is time stamped and stored in shared memory, and its memory address / pointer joins f different VOQ queues according to the fan-out set . In each time slot, the scheduling priority is given to pointers which are the unicast copies of a multicast packet with the most diminutive timestamp. Indeed, In Scheduling all multicast packets are “converted" into a unicast. At this end, the HOL blocking is consummately eliminated. But in order to maximize switch throughput, in-switch packet replication is still utilized for sending multiple replicas of a multicast packet in the same slot. This is achieved by an iterative scheduling algorithm, which incurs considerable amount of communication overhead.

In [13], Multicast Dual Round Robin Scheduling Algorithm called MDRR is proposed to achieve maximum throughput with low-matching overhead. Here input schedulers are distributed to each input, and a global pointer ‘g’ is collectively maintained by all output schedulers. Each input scheduler has two priority pointers that guarantee high throughput: a primary pointer and a secondary pointer. MDRR needs more message transfer between the input and output ports in the request phase. It does not ensure a minimum delay compared with MaxService[3]. When the number of queues and the fan-out size (ef) increase MDRR could not obtain a maximum throughput than MaxService scheme done. Dual pointer utilization in the input ports are overhead here which takes longer execution time.

We present a scheduling scheme called MDDR which primarily minimizes the request overhead at the output ports and eliminates the dual pointer utilization in input ports. It shows that MDDR more preponderant than the MDRR algorithm.

III. Switch Architecture And Algorithm Design

In this section, we give in detail the multicasting problem, switch architecture and the Multicast Due Date Round-Robin Scheduling algorithm for input queued crossbar switch which works in two phases request and grant. The pseudo code and steps involved in this algorithm are also detailed here.

A. *The Multicasting Problem*

The number of destination output ports of a multicast cell is known as its fan-out set. If we consider an NxM router with multicast capabilities, a multicast cell arriving at any of the N input ports can have any set of destinations between 2 and M. In order to avoid the HOL problem, the router must maintain up to 2^M-1 separate FIFO queues per input to cover all possible fan-out set configurations. This architecture is known as the multicast VOQ (MC-VOQ)[12]. Because of the huge number of queues maintained at each input the extensive order to schedule the traffic, this architecture is the amount of information which exchanges and is considered to be impractical. Instead, researchers have used just one FIFO queue per input. This approach is very practical, however it has poor performance due to the HOL problem. Another solution is to maintain a small number, k, of queues per input for multicast traffic. This was a good compromise to achieve high performance while maintaining affordable hardware requirements. Cells with different fan-out sets will have to be placed in the same input queue because k is much smaller than 2^M-1. This mapping is known as the multicast cell placement policy.

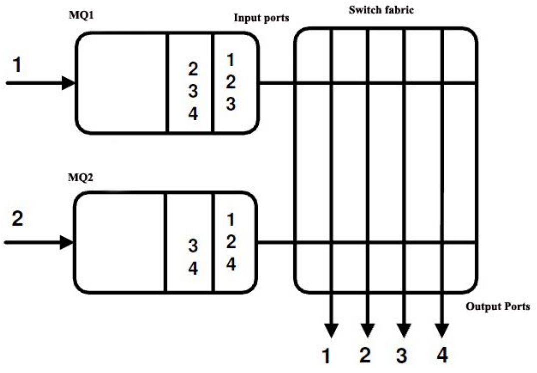

Fig.2. A 2x4 Multicast Crossbar Switch

If we consider that router D (in Fig.1) uses just one FIFO queue per input for multicast traffic, its architecture can be described as depicted in Fig.2. By considering that the crossbar fabric operates at the same rate as the external lines, at each time slot every input can send at most one cell and every output can receive at most one cell. Because of the intrinsic multicast capabilities of the crossbar fabric, a cell (multiple copies) can be sent to all its destinations at the cost of one by simply closing those cross points corresponding to its output ports provided they are available.

Subject to output availability and the scheduling algorithm used a cell may not reach all its destinations, indicated by its fan-out set, during one time slot. There are two known service disciplines used to deal with such situation. The first is known as no fan-out splitting. Time slot is defined as the time between two cell consecutive arrivals/departures to/from an input/output port of the router and the latter is known as fan-out splitting. When no fan-out splitting discipline is used, a cell must traverse the crossbar fabric only once. Meaning that a cell gets switched to its output destination ports if and only if all its destination outputs are available at the same time. If one, or more, of the output destinations is/are busy, the cell loses contention and all of its copies remain in the input port. If we consider no fan-out splitting discipline in Fig. 2, then either of the two HOL cells of queues MQ1 and MQ2 will be switched out but not both. The reason is because both cells have output ports 1 and 2 in their fan-out sets and knowing that an output port can receive at most one cell and the no fan-out splitting discipline does not allow partial cell switching resulting in only one cell of the two being eligible for transfer. The no fan-out splitting discipline is easy to implement, however it results in low throughput because it is not work conserving. This can be seen from the example above as either output 3 or output 4 will receive a cell but not both depending on which MQ has been selected.

However, fan-out splitting discipline is used, a cell can be partially sent to its destination output ports over many time slots. Copies of the cell that are not switched, due to output contention, during one time slot continue competing for transfer during the following time slot(s). The flexibility of allowing partial cell transfer comes at a little increase in implementation complexity, however it provides higher throughput because it is work conserving. In this paper, we consider fan-out splitting. Consider the example of Fig. 2 again and assuming a fan-out splitting discipline is used, then both the HOL cells of MQ1 and MQ2 can send copies to a subset of their output ports. Output 3 and 4 are receiving one cell each and therefore both copies destined to them, in the input queues, are transferred with no contention. Additionally, both HOL cells of MQ1 and MQ2 have cells destined to outputs 1 and 2. However, we know that each output can receive at most one cell at a time. Therefore, at the end of the time slot, we will have remaining cells for output ports 1 and 2. These remaining cells are referred to as the residue.

Depending on the policy used, the residue can either be concentrated on the input ports or it can be distributed over the input ports. As defined in, the residue is the number of cells left at the HOL of the input queues after losing contention for the output ports at the end of each time slot. In the example of Fig. 2, the residue is {1, 2}. A concentrating policy is one that leaves the residue on the minimum number of input ports. If we consider a concentrating policy in Fig. 2, the residue with be left (concentrated) on either MQ1 or on MQ2 but not on both. On the other hand, a distributing policy is one that leaves the residue on the maximum number of input ports. Using a distributing policy in Fig. 2 would result in the residue being distributed over MQ1 and MQ2 but not on one queue only.

B. Packet Switch Architecture

The proposed scheduling algorithm is fully made ready for synchronous input-queued (IQ) switches. The fixed-size packet which is transmitted by the switch fabric is called cell. But only the fan-out splitting discipline is consider because the cells may deliver outputs over several cell times. Any multicast cell is characterized by its fan-out set, i.e., by using the set of outputs to which the cell is directed. We define the fan-out size ‘f’ as the number of destinations of a multicast cell. The NxN switch architecture is shown in Fig. 3. Let us assume NxN switch having N input ports and N output ports, and the fabric is connecting input ports and output ports for any time slot. A small number k of FIFO queues dedicated to multicast traffic is maintained at each input port. Q_{ij} is the jth queue in the ith input port. Arriving multicast cells are partitioned into the k queues according to the fan-out size. Each queue contains the multicast cells with fan-out sets. A scheduling algorithm does the arbitration between the N input ports and N output ports, obtained by solving the bipartite graph-matching problem. This matching is a collection of edges, from the set of non-empty input queues to the set of output ports. Such that each input is connected to at most N outputs and each output is connected to at most one input. In each time slot, Input ‘i’ is connected with set of output destinations. If the fan-out of the cell is completely served, a cell is removed from the corresponding queue to output destinations by properly configuring the non-blocking multicast switch fabric otherwise a cell is retained until all its destinations are served.

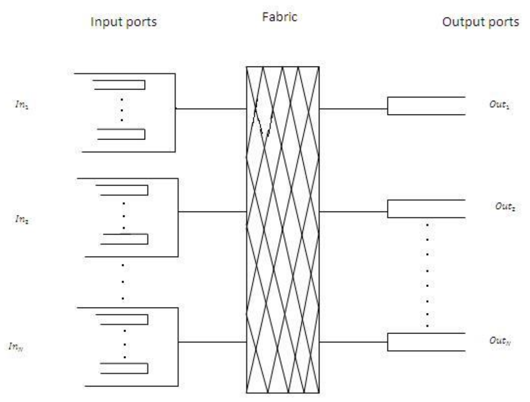

Fig.3. NxN Input Queued Switch Architecture.

C. Multicast Due Date Round-Robin(MDDR)

In MDDR, Input schedulers are distributed at each input and a global pointer *g* is collectively maintained by all the output schedulers. Each input maintains a Due Date to be sent. This due date is generated based on the priority of cells contained in the fan-out. The highest fan-out size port gets the first priority and next fan-out size has the second priority and so on. By keeping this order the throughput will be increased. This algorithm works in the following phases.

Request: The input sends request to all the destined output ports corresponding to the first nonempty queue. At request phase, fan-out size of the current non-empty queue is measured in each input port and prioritize the input ports based on fan-out size. Next step is to assign the due dates to the cells within the fan- out. This Due dates are assigned in a priority input port which will assign the first Due Date (Due Date = 1) to the cells. On the second priority port, elements already presented in first priority are assigned to second Due Date (Due Date = 2) and remaining cells are assigned to the first Due Date (Due Date = 1) and so on. On the completion of these Due Dates, the requests will be made to output ports.

Grant: In the Grant phase, if more than one request is made for the same port, the global pointer pointing one is granted and the others are rejected then the global pointer is incremented to the next position.

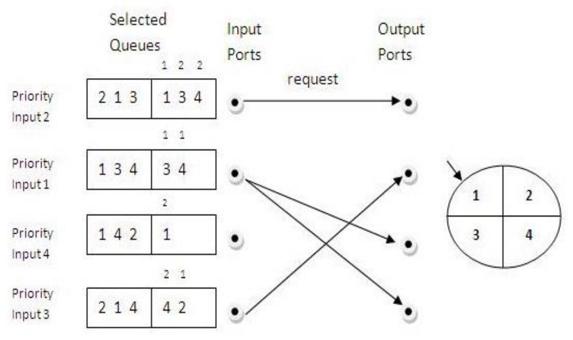

Fig.4. An Example of MDDR for 4x4 Switch With k=2 in time slot 1

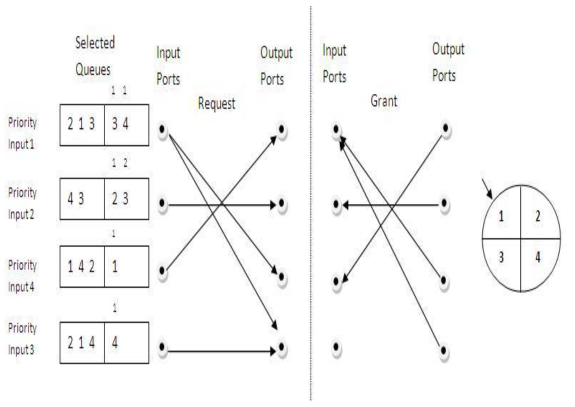

Fig.5. Request and Grant Phase of MDDR in Time Slot 2

Fig. 4 shows an example of the MDDR algorithm for a 4x4 switch where each input port maintains two multicast queues. Input 2 is the priority input 1 which is based on the highest fan-out size. All the inputs and outputs perform the arbitration in parallels to find its match. In a request phase, Input ports are prioritized as priority input 1, 2 and so on. All the input ports choose the highest fan-out size queue as selected queues as shown in Fig.4 and Fig. 5. In the Priority order all inputs make request to the output ports at the time slot. In the initial time slot, no output port receives more than one request because of after fan-out splitting only 4 fan-outs of cells are assigned due date 1 so all the other requests are granted in the grant phase. In the next time slot, all the due date 2 assigned cells of the previous time slot are reassigned due date 1 which leads to output ports to receive more than one request. In this case, output round robin pointer currently pointing one is granted and others are rejected and then the pointer is moved to the next position. This scenario is shown in Fig. 5.

When considering alternative multicast switch schedules, we can evaluate how they affect cost structure and whether they make operation easier to manage. With this aspect, MDDR got the following characteristics.

- Supports multicast and multi queues at input ports.
- Utilization of output ports in well since the throughput is increased.
- MDDR is easy to implement in the hardware.
- MDDR will manage maximum load offers.
- Have the ability to flex up to meet queue demand when multicasting.
- Execution of Due Date assignment is fast.

D. *Pseudocode of MDDR*

Input **:** Time slot (t_n), Input ports In_i,

```
loop each (time slot)-k
Fan-outsizes={(i,Size),(..,..),.......}
Fan-outsizes=Decending order of   Fan-outsizes
loop each cells m having duedate 2
assign Duedate(m)=1
end loop
loop each i (input ports) in Fan-outsizes-i
myfan-out->fan-out(i, Queue, k)
for each cells in fan-out
myfan-outnode->current fan-out cell
 if{cells having no due date}
     {
 if{no cell having current due date}
     {
Duedate(myfan-outnode)=1
Send Req i
     }
else
     {
Duedate (myfan-outnode)=2
     }
     }
if {i =global_pointer }
grant (i)
 end loop (fan out)
 if { global_pointer=count(inp) } {
global_pointer->0
 }
else
{
global_pointer [global_pointer+1]
 }
 end loop (input ports)
end loop (time slot)
```

E. *Steps involved in MDDR Algorithm*

- Prioritize input ports based on high fan out size
- Iterative through Priority order of input ports
- Set Due Date for the fan out cells in the current queue of current input port. If two or more queues in the input port have same fan-out size, first non-empty queue is selected. For each Due Date unassigned cells set Due Date 1 to priori non exist cell, and 2 to priori exist cell.
- Make request for the 1st due date cells to the output ports.
- If output port received more than one request, choose one from the global pointer points and grant it.
- If output port receives only one request, it directly grants it. Repeat the iteration with the non-granted cell requests and remaining fan outs further.

IV. Performance Evaluation And Result Analysis

We implemented the simulation in NS2 that models the input queued crossbar switch of size $N \times N$. In general for all the experiments, we used a 8×8 and 16×16 VOQ switch. The VOQ's are supplied with Bernoulli uncorrelated and Bursty correlated multicast traffic.

NS2 is an open-source event-driven simulator designed specifically for research in computer communication networks. Since its inception in 1989, NS2 has continuously gained tremendous interest from industry, academia, and government. Having been under constant investigation and enhancement for years, NS2 now contains modules for numerous network components such as routing, transport layer protocol, application, etc. To investigate network performance, researchers can simply use an easy-to-use scripting language to configure a network, and observe results generated by NS2. Undoubtedly, NS2 has become the most widely used open source network simulator, and one of the most widely used network simulators.

We consider Bernoulli uncorrelated and Bursty correlated traffic conditions and compare the algorithms to evaluate the performance of the proposed scheduling algorithm. A traffic generation model is a stochastic model of the traffic flows or data sources in a communication network, for example a cellular network or a computer network.

Bernoulli processes are the discrete time analog of Poisson processes. In a Bernoulli process the probability of an arrival in any time slot is p, independent of any other time slot. The time between arrivals corresponds to a Geometric distribution.

Bursty traffic model is found in Jain and Routhier's Packet Trains model. This model is principally designed to recognize that the addresses locality applies to routing decisions; that is, packets that arrive near each other in time are frequently goes to the same destination. In generating a traffic model that allows for easier analysis of locality, the authors created the notion of packet trains, a sequence of packets from the same source, travelling to the same destination (with replies in the opposite direction). These two traffic as two different arrival processes, fix the average burst size E[B] to be 16 cells for all experiments.

Let λ be the average arrival rates, equal to the input load, and μ be the output load, then $\mu = \lambda ef$.

This research work concentrates on two Performance metrics which are Delay and Throughput. The graphs drawn in Fig. 6 to 9 shows that the overall performance of Delay and Throughput comparison of MDRR and MDDR algorithms.

Delay: A multicast cell is stored in the queue until all the destinations in its fan-out set are reached. The multicast delay of a cell is calculated as the cell times that the cell stays in the queue until it is removed. Delay increases when they become unstable as the offered load increases.

Throughput: Throughput is the another performance measurement used in this investigation which is defined as the ratio between the total number of cells forwarded to output interfaces, and the total number of cells arrived at input interfaces. It is essentially a measure of the cell loss probability at input queues.

Our proposed work is compared with the existing approach MDRR and can judge the proposed work achieves high throughput than the existing work. The delay-throughput performance of MDDR and MDRR schemes has been well demonstrated under two traffic conditions.

We first apply the Bernoulli traffic to the switch for the 8x8 switch. Fig. 6 shows the load-delay performance of the MDDR and MDRR algorithms under Bernoulli traffic pattern. We varied the input load from 0 to 1.0. For the load upto 0.6, both algorithms maintain almost equal delay. At higher offered load MDDR achieves minimum delay compared with MDRR since Due Dates are assigned to the cells which minimizes the number of request to the output ports. Fig. 7 shows the simulation results when Bursty traffic is applied. Fig. 6 and 7 compares the average multicast delays under various traffic loads. At lower offered load MDDR and MDRR achieves closest delay under both traffic conditions but the request made by input ports are minimized in MDDR.

Now fix the switch size as stable of 16x16, and we have taken the throughput by increasing the number of queues in each input port. We examine the average number of queues be 2, 4, 6,8,14 and 20. In Fig. 8, it is shown, and we observe that, Throughput is increasing for a particular range only. When we increase the number of queues, MDDR achieving more throughput than MDRR because of higher fan-out size queue is selected each time for a particular port. Fan-out size is an another factor affecting the performance of the switch, if fan-out size increased, delay may increase and if fan-out size decreased throughput may decrease. From this point of view we examine a switch of 16x6 with k=4. From this observation we can conclude when fan-out size increases MDDR having higher throughput than MDRR. It is shown on the Fig. 9.

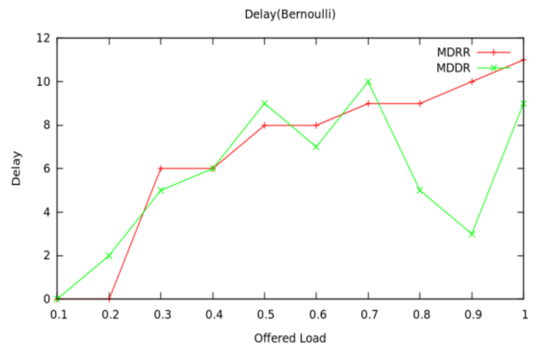

Fig.6. Offered Load Vs Delay (Bernoulli)

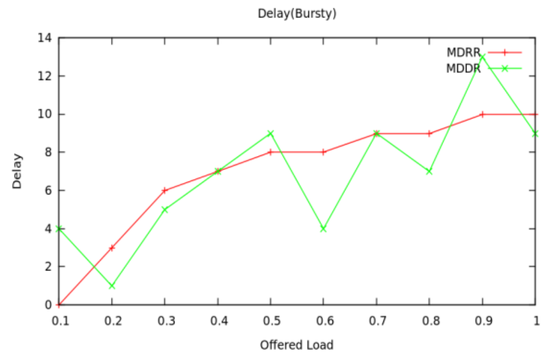

Fig.7. Offered Load Vs Delay (Bursty)

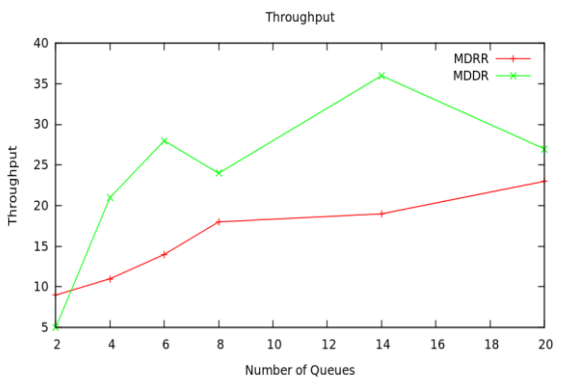

Fig.8. No.of. Queues Vs Throughput

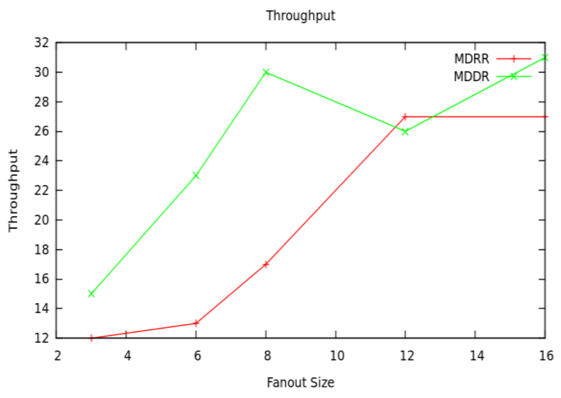

Fig.9. Fan-out Size Vs Throughput

Our future work includes, testing the MDDR algorithm with Enhanced Core Stateless Fair Queuing (ECSFQ) discipline which has been proposed in [8] and to analyse the switching performance for varying switch sizes.

V. Conclusion

This study proposes a flexible, simple, and fair multicast scheduling algorithm which reduces the complexity of scheduling greatly with very little lose of switching performance, and offers a reasonable choice for high-speed input queued switches/routers. In this paper we have implemented and simulated the scheduling algorithms MDDR and MDRR and we have achieved better performance in MDDR comparing with MDRR. The single pointer algorithms are having better performance than more than one pointers. So our current work is enhanced to reduce the pointer overhead and achieve the maximum throughput in the input-queued multicast switches.

References

[1] Bianco A, Giaccone P, Leonardi E, Neri F, and Piglione C., “On the number of input queues to efficiently support multicast traffic in input queued switches,” *In Proceedings of Workshop on High Performance Switching and Routing*, pp. 111–116, 2003.

[2] Bianco A, Scicchitano A., “Multicast support in multi-chip centralized schedulers in input queued switches,” *Computer Networks*, vol. 53, no. 7, pp. 1040–1049, 2009.

[3] Gupta S, and Aziz A., “Multicast scheduling for switches with multiple input-queues,” *In Proceedings of High Performance Interconnects Symposium*, pp. 28–33, 2002.

[4] Marsan M.A, Bianco A, Giaccone P, Leonardi E, and Neri F, “Multicast traffic in input-queued switches: optimal scheduling and maximum throughput,” *IEEE/ACM Transactions on Networking*, vol. 11, no. 3, pp. 465–477, 2003.

[5] McKeown N, and Prabhakar B., “Scheduling multicast cells in an input queued switch,” *In Proceedings of IEEE INFOCOM*, vol. 1, pp. 271–278, 1996.

[6] McKeown N., “A Fast Switched Backplane for a Gigabit Switched Router,” *Business Communication Review*, vol. 27, no. 12, 1997.

[7] McKeown N, “The iSLIP scheduling algorithm for input-queued switches,” *IEEE/ACM Transactions on Networking*, vol. 7, no. 2, pp. 188–201, 1999.

[8] Nandhini Sivasubramanian, Palaniammal Senniappan., “Enhanced Core Stateless Fair Queuing with Multiple Queue Priority Scheduler,” *In Proceedings of The International Arab Journal of Information Technology*, Vol. 11, No. 2, 2014.

[9] Pan D. and Yang Y., “FIFO-based multicast scheduling algorithm forvirtual output queued packet switches,” *IEEE Transactions on Computers*, vol. 54, no. 10, pp. 1283–1297, 2005.

[10] Prabhakar B, McKeown N, and Ahuja R., “Multicast scheduling for input-queued switches,” *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 5, pp. 855–866, 1997.

[11] Shanmugam Arumugam, Shanthi Govindaswamy., “Performance of the Modified Round Robin Scheduling Algorithm for Input-Queued Switches Under Self-Similar Traffic,” *In Proceedings of The International Arab Journal of Information Technology*, vol.3, no.2, 1996.

[12] Song M, and Zhu W., “Throughput analysis for multicast switches with multiple input queues,” *IEEE Communications Letters*, vol. 8, no. 7, pp. 479–481, 2004.

[13] Yongbo Jiang, Zhiliang Qiu, Ya Gao, and Jun Li, “Multicast Support in Input Queued Switches with Low Matching Overhead”, *IEEE Communications Letters*, vol. 16, no. 12, 2012.

[14] Zhu W, and Song M., “Integration of unicast and multicast scheduling in input-queued packet switches,” *Computer Networks*, vol. 50, pp. 667– 687, 2006.

[15] Zhu W, and Song M., “Performance analysis of large multicast packet switches with multiple input queues and gathered traffic,” *Computer Communications*, vol. 33, no. 7, pp. 803–815, 2010.

[16] Janaka L. Wijekoon, Erwin H. Harahap, Shinichi Ishida, Rajitha L. Tennekoon and Hiroaki Nishi., “Router-based Content-aware Data Redirection for Future CDN Systems” I.J. Computer Network and Information Security, 2014, 7, 1-10.

Authors’ Profiles

Navaz K received the B.Tech degree in Information Technology in 2006, and the M.E degree in Computer Science and Engineering in 2009, all from Anna University, Tamilnadu, India. He is working towards his Ph.D in the department of Computer Science and Engineering, Manonmaniam Sundaranar University, Tirunelveli, Tamilnadu, India.

His areas of interest in research are Computer Networks, Network Design and Simulations, Switch Architecture and Scheduling Algorithms for High Performance Switches.

Kannan Balasubramanian received the Ph.D degree in Computer Sceince from UCLA, and the M.Tech degree in Computer Sceince and Engineering from IIT Bombay, India and his Msc(Tech) degree in Computer Sceince from BITS., Pilani, India. He is a Professor Mepco Schlenk Engineering College, Sivakasi, India. His research interest includes Network Architecture, Protocols, Security and Performance.

How to cite this paper: Navaz K, Kannan Balasubramanian,"Multicast Due Date Round-Robin Scheduling Algorithm for Input-Queued Switches", International Journal of Computer Network and Information Security(IJCNIS), Vol.8, No.2, pp.56-63, 2016.DOI: 10.5815/ijcnis.2016.02.07