Enhanced Core Stateless Fair Queuing with Multiple Queue Priority Scheduler

Nandhini Sivasubramaniam[1] and Palaniammal Senniappan[2]
[1]Department of Computer Science, Garden City College of Science and Management Studies, India
[2]Department of Science and Humanities, VLB Janakiammal College of Engineering and Technology, India

Abstract: *The Core Stateless Fair Queuing (CSFQ) is a distributed approach of Fair Queuing (FQ). The limitations include its inability to estimate fairness during large traffic flows, which are short and bursty (VoIP or video), and also it utilizes the single FIFO queue at the core router. For improving the fairness and efficiency, we propose an Enhanced Core Stateless Fair Queuing (ECSFQ) with multiple queue priority scheduler. Initially priority scheduler is applied to the flows entering the ingress edge router. If it is real time flow i.e., VoIP or video flow, then the packets are given higher priority else lower priority. In core router, for higher priority flows the Multiple Queue Fair Queuing (MQFQ) is applied that allows a flow to utilize multiple queues to transmit the packets. In case of lower priority, the normal max-min fairness criterion of CSFQ is applied to perform probabilistic packet dropping. By simulation results, we show that this technique improves the throughput of real time flows by reducing the packet loss and delay.*

Keywords: *CSFQ, quality of service, MQFQ, priority scheduler.*

Received November 21, 2011; accepted July 29, 2012; published online January 29, 2013

1. Introduction

1.1. Queuing Algorithms in Networking

Queueing Theory is a branch of applied probability theory. Its applications are in different fields, e.g., communication networks, computer systems, machine plants and etc., [22].

Queues and queuing algorithms [2, 3, 11] are important elements of traffic handling in a network to provide Quality of Service (QoS). Queuing happens only when the interface is busy. As long as the interface is idle, packets will be transmitted without special treatment. Regular queues invariably employ the FIFO principle: The packet that has been waiting the longest is transmitted first. When the queue is full and additional packets come in, tail drop happens. More sophisticated queuing mechanisms usually employ several queues for service in which the packets are classified by user-configurable means and then they are placed in the appropriate queue. When the interface is ready to transmit, a queue from which the next packet will be transmitted is selected as per the queuing algorithm. When a queue becomes inefficient, congestion occurs which results in dropping of packets. The traffic overflow can be managed by using appropriate queuing algorithm that can sort the traffic and find a prioritizing technique for forwarding the packets to output link [4]. A G/G/1 queuing model algorithm has been developed [18], which reduces response time variance in existence of bursty traffic.

1.2. Fair Queuing Techniques

Queuing in networking includes techniques like priority queuing [7, 11, 13], Fair Queuing (FQ) [2, 10, 19] and Round Robin Queuing [12, 15, 16].

FQ is intended to make sure that each flow has fair access to network resources and avoid bursty flows from consuming more than its fair share of output port bandwidth. This system classifies the packets into flows and it is assigned to queue which exclusively dedicated to that flow. The queue is serviced in round robin fashion. FQ is also termed as per-flow or flow based queuing [17]. The FQ mechanism includes Weighted Fair Queuing (WFQ) [1] and Core Stateless Fair Queuing (CSFQ).

1.3. Limitations of CSFQ and Proposed Enhancements

CSFQ is a distributed technique of FQ. The core component includes processing overhead on core network links. The concept of core component is that only certain routers have necessity to take special queuing mechanism whereas other routers remain as such. Thus, only edge routers take on congestion control mechanism allowing normal operation for core network links without any reduction of existing routing speeds. By using the 'flow id' field in packet headers, CSFQ approximates fair bandwidth allocation at its edge nodes. This data is stored in the packets and passed along as they travel [9]. The limitations of CSFQ include its inability to estimate fairness in

situations where large traffic flows are present and where such traffic is of short and bursty (VoIP or video). It uses the single FIFO queue at the core router. For improving the fairness and efficiency in the technique, we propose a solution that combines priority queuing and max-min fairness technique with CSFQ techniques. We consider multimedia flows that include VoIP flows and video flows. When the packets enter into the ingress edge router, first the priority scheduler is applied to the flows. In case of VoIP and video flows, the packets are treated as higher priority whereas for the best effort traffic the packets are treated as lower priority. These priority values are marked along with flow arrival rate and transmitted to core router.

In core router, for higher priority flows the Multiple Queue Fair Queuing (MQFQ) [5] is applied that allows a flow to utilize multiple queues to transmit the packets. In case of lower priority, the normal max-min fairness criterion of CSFQ [20] is applied to perform probabilistic packet dropping. This technique of applying individual queue techniques to every flow improves fairness in the transmission and avoids congestion.

2. Related Works

Hwang *et al.* [6] proposed an Adaptive Weighted Fair Queuing with Priority (AWFQP) scheduler for Diffserv networks based on Traffic THresholds (TH). The proposed AWFQP scheduler can guarantee the QoS for the Expedited Forwarding (EF) and Assured Forwarding (AF) traffics and avoid the starvation for Best Effort (BE) traffic when the traffic load is high. Overall, the AWFQP can improve the resource utilization while providing end-to-end QoS guarantees to all DiffServ classes and improves system throughput and reduces jitter and delay time.

Georg *et al.* [5] proposed MQFQ which employs a fixed number of FIFO and multiple hash functions. By applying the hash functions to headers of incoming packets, MQFQ strives to associate each flow with the subset of the FIFO queues. When a packet arrives from a flow, the router places the packet into the shortest queue associated with the flow. The authors have concluded that when the number of misbehaving flows is large, MQFQ balance all individual flow throughputs much more fairly than Stochastic Fair Queuing (SFQ) or Stochastic Fair Blue (SFB).

Jin *et al.* [8] have proposed a distributive flow control algorithm for networks with multiple paths among source-destination pairs. They employed a utility max-min fair resource allocation algorithm among competing users which is more appropriate for practical networks. The proposed approach removes typical oscillation performance in multipath networks by combining first order Lagrangian method and filtering mechanism. The factors such as delay and dynamic network behaviors such as stability are not considered in modeling the utility functions.

Elshaikh *et al.* [3] proposed a new scheduler that can be used effectively in a DiffServ networks and investigated the effects of using different scheduling mechanisms on a traffic stream entering a DiffServ network. It has been shown that for loss sensitive applications WFQ is the most appropriate since it has the smallest number of dropped packets in edrop and overall dropped although WIRR performs better in terms of Edrop however, in both cases FWFQ performs better up to 50% network provision level. In general WFQ has a better performance overall among the all compared algorithms making it more suitable for those sensitive to loss applications. For delay sensitive applications, FWFQ is better; it gives a better performance in terms of delay and delay jitter.

Tsao *et al.* [21] proposed a Minimum-Service First Request Scheduling (MSF-RS) scheme. MSF-RS always selects the next request from the class receiving the minimum service, to provide user-based weighted fairness, which ensures more bandwidth for high-class users. Next, MSF-RS uses a window-based rate control on releasing requests to maintain full link utilization and to reduce the user-perceived latency. Finally, the authors have concluded that to reduce the overhead, implementing MSF-RS in the kernel space may be considered.

3. Enhanced Core Stateless Fair Queuing with Multiple Queue Priority Scheduler (ECSFQ-MQPS)

3.1. Overview

For improving the fairness and efficiency in core stateless FQ technique, we propose a solution that combines priority queuing and max-min fairness technique. When the packets enter into the ingress edge router, first the priority scheduler is applied to the flows. In case of VoIP and video flows, the packets are treated as higher priority whereas for the best effort traffic the packets are treated as lower priority. These priority values are marked along with flow arrival rate and transmitted to core router. In core router, for higher priority flows the MQFQ is applied that allows a flow to utilize multiple queues to transmit the packets. In case of lower priority, the normal max-min fairness criterion of CSFQ is applied to perform probabilistic packet dropping.

3.2. Core Stateless Fair Queuing

We take active queue management system called core stateless queuing into consideration. CSFQ architecture has two key aspects:

1. In CSFQ, only edge routers maintain per flow rate, while core routers do not maintain per flow state

instead uses the per flow information carried through a label in each packet's header. This label contains an estimate of the flow's arrival rate. However, it is initialized by the edge router based on per-flow information, and then updated at each router along the path based only on aggregate information at that router.

2. FQ requires FIFO queuing to be used with probabilistic dropping on input to overcome per flow buffering and scheduling. The probability of dropping a packet is a function of the rate estimate carried in the label and of the fair share rate at that router, which is estimated based on measurements of the aggregate traffic. This technique provides a remedy that offers to avoid both the need to maintain per-flow state and the need to use complicated packet scheduling and buffering algorithms at core routers [20].

3.3. Flow Classifier

The flow classifier identifies the ingress traffic flow as best effort or real-time based on the estimated delay and loss.

3.3.1. Delay and Loss Estimation

At all ingress routers the real time flows are sampled. The path of a real time flow has been probed by the header of the sampled packet. As the user does not get altered frequently inside a network domain, the probe and user traffic travel in same path with the high probability. Thus, a rough estimation of delay value experienced by the sampled flows in the network domain is evaluated.

In case of probing the delay, the ingress routers encode the current timestamp T_c into the payload and header is marked with a new protocol value. Those packets are recognized by egress router and removed from the network. Before that, the egress router computes edge-to-edge link delay for a packet. The link delay is the resultant of difference between the own time of packets and T_c. The egress classifies the probe packets as belonging to flow i, and updates the average packet delay, PD_{avi} for delay sample $D_i(t)$ at time t using an Exponential Weighted Moving Average (EWMA):

$$PD_{avi}(t) = \mu * PD_{avi}(t-1) + (1-\mu) * D_i(t) \quad (1)$$

Where μ is a small fraction $0 \leq \mu \leq 1$ to emphasize recent history rather than the current sample alone. The edge-to-edge probing investigates excessive packet loss and causes due to loss within a network domain. The back to back probe packets for a small sample interval of T seconds are utilized to deduce link loss. This is done by computing the correlation of a packet loss within a set of probe packets at different destination. In this technique, source forwards a series of probe packets along a path P_1 to the destination, with no delay during the transmissions of successive packets. The loss ratio L_i at a node N_i along the path P_1 at the interval T can be calculated as:

$$L_i^T = P_{Lo} / R_a \quad (2)$$

Where P_{Lo} is the number of packets lost and R_a is the estimated arrival rate of the packet. Then, the total loss ratio at destination can be calculated as:

$$L^T = \sum L_i^T \quad (3)$$

Now the actual traffic flows are transmitted for the same sample interval of T seconds through the ingress router which marks the flow arrival rate as label according to CSFQ. The actual loss ratio (L_{act}) at each node along P_1 at the interval T can be estimated similarly as equation 1. Then the total actual loss ratio L_{act} at destination can be calculated as:

$$L_{act}^T = \sum L_{act}^T \quad (4)$$

At egress router, the difference in loss ratios can be then estimated as:

$$D = L_{act}^T - L^T \quad (5)$$

3.3.2. Flow Classification by Ingress Nodes

The links possessing high losses and egress router through which the flows are exiting are found. The flows that consume high bandwidth are isolated. These rates are forwarded to ingress routers through which the flows enter into the domain. The rate at which the flow is entering and exiting the network domain is compared by ingress router.

The real time flows can be reported either in per flow or aggregate fashion. If the flow value is greater than the threshold, then the feedback is done by aggregate manner for each ingress router. The aggregation is performed based on the traffic class. The real time flows with high bandwidth are reported to the egress router. From the CSFQ labels, the identity of the ingress router is obtained. This identification code is used to relate a flow and its entry point else the egress does not know through which ingress routers the flow is entering into the domain. The flow arrival rates and the corresponding source ids are collected from the labels of the packets which are marked by ingress node.

If the value of D as equation 5 exceeds to a threshold T_1 and if the delay as equation 1 exceeds a threshold T_2, then the flows are marked as real time flows by the egress node, otherwise they are considered as best effort traffic. Then the flow arrival rate and the flow id are sent to the source by the egress router.

3.4. Priority Scheduler

We apply priority scheduler to the above identified flow categories as per following condition:

1. If flow is VoIP or video, then
Flow is marked as higher priority in flow label
Else
2. If the flow is best effort traffic, then
Flow is marked as lower priority in flow label
End if

These priority values are marked along each flow and passed to the core router. The core router checks the priority values. For higher priority flows, multiple fair queuing technique is applied and for lower priority flows, max-min fairness criteria of CSFQ is applied.

3.5. Multiple Queue Fair Queuing Technique

MQFQ technique allows a flow to use multiple queues. It utilizes multiple hash functions to determine a set of FIFO queues for a flow and serves all queues in the round robin order. The steps involved in multiple fair queuing techniques are:

1. MQFQ uses multiple hash functions to determine a set of FIFO queues for a flow. When a packet arrives, all hash functions are applied to the packet header by MQFQ for computation of effective queues.
2. MQFQ puts the packet into the queue with the soonest service.
3. If one queue associated with a flow grows large, then the flow uses another of its queues and there by passes the congestion.
4. As a flow can flood multiple queues, there exists a trade-off between the degree of extra capacity surrendered to a misbehaving flow and a number of flows starved by the misbehaving flow.

Though MQFQ uses different queues for placing the packets, packet reordering does not occur when packets are of similar size. In case the packet sizes are dissimilar, reordering is possible but restricted within one round of queue traversal. This can be avoided by buffering packets for one round for restoring their order before forwarding them into the link. On the other hand, if router fragments incoming packets into similar sized cells, MQFQ is applied to the cells and packets are reassembled before forwarding to the link and then reordering does affect cells or reassembled packets.

3.6. Max-Min Fairness Criterion

The lower priority flows are applied with max-min fairness criterion to perform probabilistic packet dropping. The following notations are assumed for the computations of fairness.

- Flow arrival rate R_f.
- Fair share rate at time t $\lambda(t)$.
- Output link speed of a router model S_l.
- Total arrival rate $TAR(t)$.

3.6.1. Flow Arrival Rate R_f

R_{fi} is estimated at the edge routers and these are inserted into the packet labels. At each edge router, the exponential averaging is used to estimate the rate of a flow [20]. Let A_{fi}^n and L_i^n be the arrival time and length of the n^{th} packet of flow i. The estimated rate of flow i, R_{fi} is updated every time a new packet is received:

$$R_{fi}^{new} = \left(1 - e^{-T_i^n / X}\right) \frac{L_i^n}{T_i^n} + e^{-T_i^n / X} Rfi, old \quad (6)$$

Where $T_i^n = t_i^n - t_i^{n-1}$, is the inter-arrival time between the current and previous packet and X is a constant.

3.6.2. Link Fair Rate Estimation

The rate with which the algorithm accepts a packet is a function of the current estimate of the fair share rate, which is denoted as $\lambda(t)$. Considering $F(\lambda(t))$ to be the acceptance rate, we have:

$$F(\lambda(t)) = \sum_{i=1}^{n} min(R_{fi}(t), \lambda(t)) = S_l \quad (7)$$

3.6.3. Total Arrival Rate

For TAR, we use exponential averaging with a parameter e^{-T/K_a} where T is the inter-arrival time between the current and the previous packet:

$$TAR_{new} = (1 - e^{-T/K_a}) \frac{L}{T} + e^{-T/K_a} TAR_{old} \quad (8)$$

Where, TAR_{old} is the value of TAR before updating. From the above computation, the following conditions are obtained:

- If the link is congested $(TAR(t)>S_l)$, we choose $\lambda(t)$ to be unique solution to $F(x)=S_l$.
- If the link is not congested $TAR(t)<S_l$, we take $\lambda(t)$ to be the largest rate among the flows that traverse the link, i.e., $\lambda(t)=max_{1\le i\le n}(R_{fi}(t))$.

From equation 7, if we knew the arrival rate $(R_{fi}(t))$, we could compute $\lambda(t)$ directly. To avoid having to keep such per-flow state, we seek instead to implicitly compute $\lambda(t)$ by using only aggregate measurements of F and TAR.

3.6.4. Fairness Computation

Max-min fair bandwidth allocations are characterized by the fact that all flows that are bottlenecked by the router have a same output rate. This rate is termed as fair share rate of the link. If max-min bandwidth

allocation is achieved, each flow '*i*' receives a service at a rate given by min ($R_{fi}(t)$, $\lambda(t)$). $R_{fi}(t)$, and $\lambda(t)$ is computed from equations 6 and 7, respectively:

$$\therefore TAR(t) = \sum_{i=1}^{n} R_{fi}(t) \quad (9)$$

If $TAR(t) > S_1$, then the fair share $\lambda(t)$ is the:

$$S_1 = \sum_{i=1}^{n} min(R_{fi}(t), \lambda(t)) \quad (10)$$

If $TAR(t) \leq S_1$, then no bits are dropped and thus, $\lambda(t) = max_i\ R_{fi}(t)$. If $R_{fi}(t) \leq \lambda(t)$, i.e., flow '*i*' sends no more than the link's fair share rate, all of its traffic will be forwarded. If $R_{fi}(t) > \lambda(t)$, then a fraction $\frac{R_{fi}(t) - \lambda(t)}{R_{fi}(t)}$ of its bits. Will be dropped so '*i*' will have an output rate of exactly $\lambda(t)$. This suggests a very simple probabilistic forwarding algorithm that achieves fair allocation of bandwidth: Each incoming bit of flow '*i*' is dropped with the probability:

$$Max\left(0, 1 - \frac{\lambda(t)}{R_{fi}(t)}\right) \quad (11)$$

When these dropping probabilities are used, the arrival rate of flow '*i*' at the next hop is given by min ($R_{fi}(t)$, $\lambda(t)$). Figure 1 gives the steps involved in the design of Enhanced CSFQ technique.

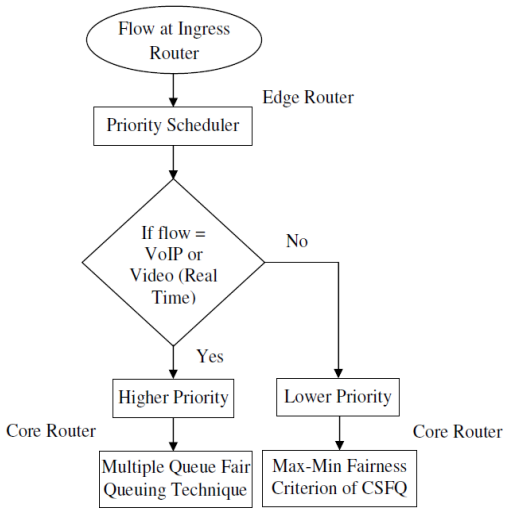

Figure 1. Flow chart of the proposed technique.

4. Simulation Results

4.1. Simulation Model and Parameters

In this section, we examine the performance of our Enhanced Core Stateless Fair Queuing with Multiple Queue Priority Scheduler (ECSFQ-MQPS) with an extensive simulation study based upon the ns-2 network simulator [14]. We compare our results with the traditional CSFQ. The topology used in the simulation is depicted in Figure 2. We use a mixture of video, VoIP and TCP traffic flows. The packet size is 512 bytes and there are totally 10 flows. The link bandwidth and link delay is set as 10 Mb and 10 ms respectively. The bottleneck bandwidth for the links (0, 1), (0, 2) and (0, 3) is set as 5Mb initially.

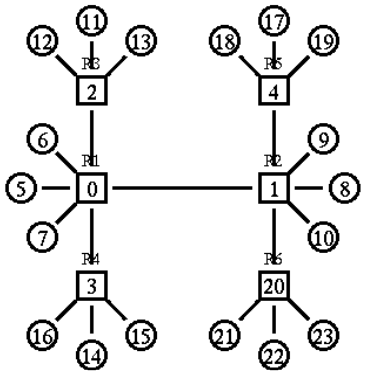

Figure 2. Simulation topology.

4.2. Performance Metrics

In the simulation experiments, we vary the bottleneck bandwidth and traffic rate. We measure the following metrics:

1. Throughput.
2. Delay.
3. Packet loss.

The results are described in the next section.

4.3. Results

4.3.1. Effect of Varying Bottleneck Bandwidth

In our first experiment, we vary the bottleneck bandwidth for the links (0, 1), (0, 2) and (0, 3) as 2Mb, 4 Mb, …, 8 Mb in order to calculate the packet loss, delay and throughput. In our experiment, we use TCP for responsive traffic and for unresponsive traffic we use VoIP and video:

- *Case 1. VoIP:* In this case, a set of VoIP and TCP flows are used.

Figure 3 gives the TCP throughput occurred for varying the bottleneck bandwidth. It shows that the TCP throughput is more in the case of ECSFQ-MQPS when compared with CSFQ.

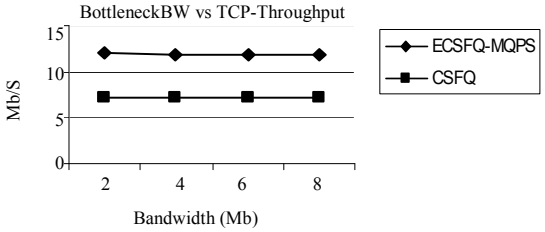

Figure 3. Bottleneck BW vs TCP-throughput.

Figure 4 shows the delay variation. It shows that the out proposed ECSFQ-MQPS has less delay than the CSFQ. When varying the bottleneck.

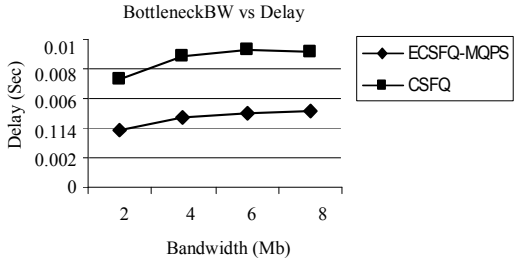

Figure 4. Bottleneck BW vs delay.

Figure 5 shows that the packet loss is high in CSFQ when compared with ECSFQ-MQPS by varying the bottleneck bandwidth.

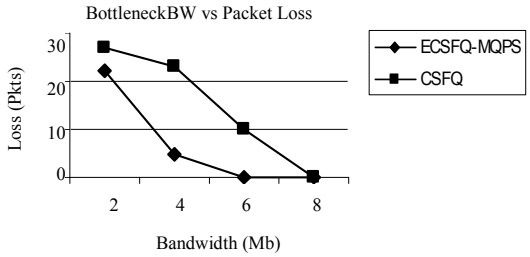

Figure 5. Bottleneck BW vs loss.

Figure 6 gives the VoIP throughput occurred for varying the bottleneck bandwidth. As we can see from the Figure, the VoIP throughput is more in the case of ECSFQ-MQPS when compared with CSFQ.

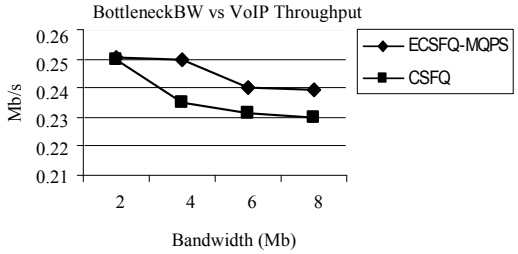

Figure 6. Bottleneck BW vs VoIP-throughput.

- *Case 2. VIDEO:* In this case, a set of video and TCP flows are used.

  Figure 7 gives the TCP throughput video occurred for varying the bottleneck bandwidth. It shows that the TCP throughput video is more in the case of ECSFQ-MQPS when compared with CSFQ

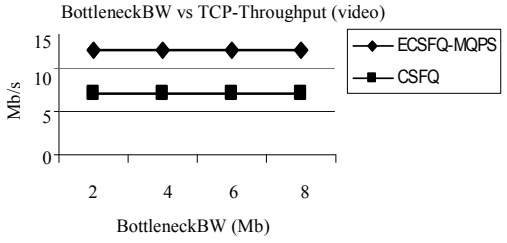

Figure 7. Bottleneck BW vs TCP-throughput (video).

Figure 8 shows the delay variation. It shows that out proposed ECSFQ-MQPS has less delay than the CSFQ. When varying the bottleneck.

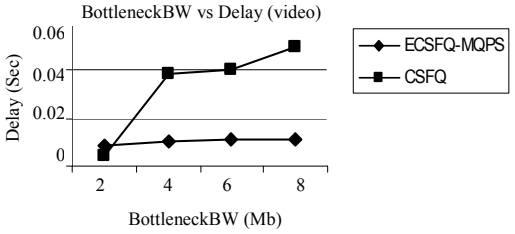

Figure 8. Bottleneck BW vs delay (video).

Figure 9 shows that the packet loss is high in CSFQ when compared with ECSFQ-MQPS by varying the bottleneck bandwidth.

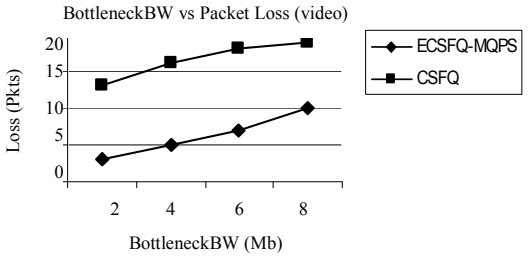

Figure 9. Bottleneck BW vs packet loss.

Figure 10 gives the video throughput occurred for varying the bottleneck bandwidth. As we can see from the Figure, the video throughput is more in the case of ECSFQ-MQPS when compared with CSFQ.

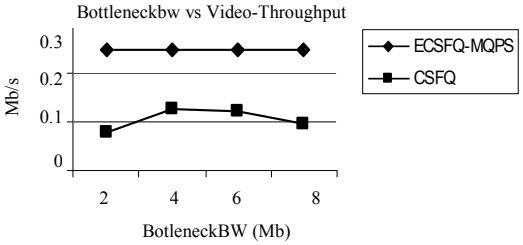

Figure 10. Bottleneck BW vs video-throughput.

4.3.2. Effect of Varying Rates

In our second experiment, we vary the rate as 250, 500, …, 1000 Kb in order to calculate the packet loss, delay and throughput. We use TCP for responsive traffic and for unresponsive traffic we use VoIP and video:

- *Case 1. VoIP:* In this case, a set of VoIP and TCP flows are used.

  Figure 11 gives the TCP throughput occurred for varying the Rate. It shows that the TCP throughput is more in the case of ECSFQ-MQPS when compared with CSFQ.

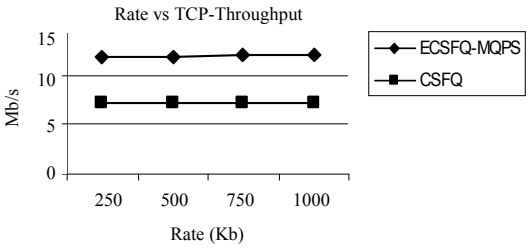

Figure 11. Rate vs TCP-throughput.

Figure 12 shows the delay variation. It shows that out proposed ECSFQ-MQPS has less delay than the CSFQ when varying the rate.

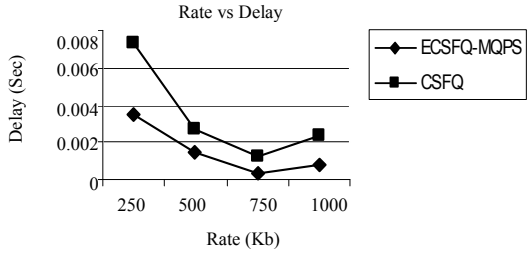

Figure 12. Rate vs delay.

Figure 13 shows that the packet loss is high in CSFQ when compared with ECSFQ by varying the rate.

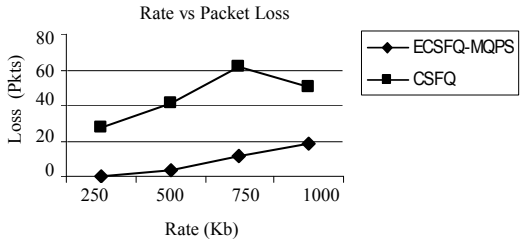

Figure 13. Rate vs loss.

Figure 14 gives the VoIP throughput occurred for varying the rate. As we can see from the figure, the VoIP throughput is more in the case of ECSFQ when compared with CSFQ.

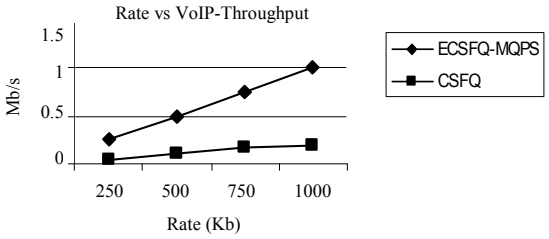

Figure 14. Rate vs VoIP-throughput.

- *Case 2. Video:* In this case, a set of video and TCP flows are used.

  Figure 15 gives the TCP throughput video occurred for varying the Rate. It shows that the TCP throughput video is more in the case of ECSFQ-MQPS when compared with CSFQ.

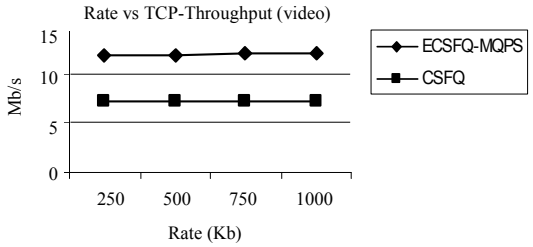

Figure 15. Rate vs TCP throughput (video).

Figure 16 shows the delay variation. It shows that out proposed ECSFQ-MQPS has less delay than the CSFQ when varying the rate.

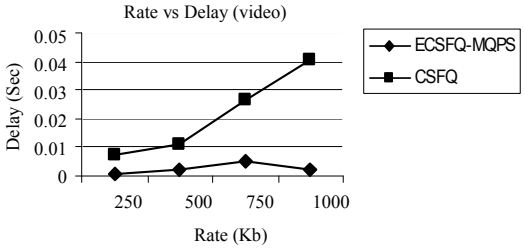

Figure 16. Rate vs delay (video).

Figure 17 shows that the packet loss is high in CSFQ when compared with ECSFQ-MQPS by varying the rate.

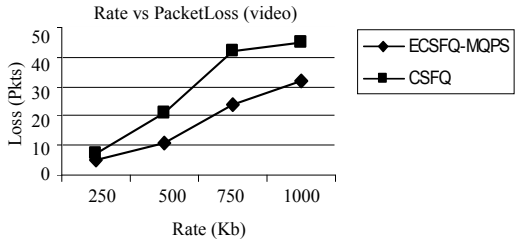

Figure 17. Rate vs packet loss (video).

Figure 18 gives the video throughput occurred for varying the rate. As we can see from the figure, the video throughput is more in the case of ECSFQ-MQPS when compared with CSFQ.

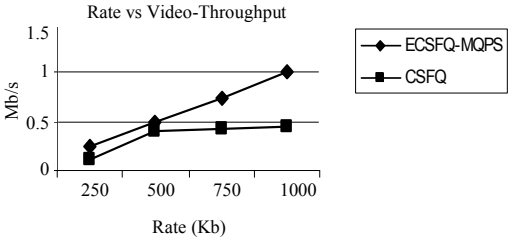

Figure 18. Rate vs video-throughput.

5. Conclusions

In this paper, we have proposed ECSFQ-MQPS. Here the flows are differentiated as lower priority flows and high priority flows by applying the priority scheduler for the incoming flows in the ingress edge router. The best-effort flows are marked as lower priority, the video or VoIP flows are marked as high priority and the marked packets along with the flow arrival rate are transmitted to the core router. MQFQ is applied for high priority queues in core router and for low priority; the normal max-min principle of CSFQ is applied. The simulation result shows improvement in performance with our ECSFQ-MQPS. In this paper, we have proposed a model considering congestion has happened. Further to this work, future work can be focused on congestion detection and then applying our model ECSFQ-MQPS.

References

[1] Banchs A. and Perez X., "Distributed Weighted Fair Queuing in 802.11 Wireless LAN," *in Proceedings of IEEE International Conference on Communications*, Germany, vol. 5, pp. 3121-3127, 2002.

[2] Elshafei A. and Baroudi U., "Wireless Fair Queuing Algorithm for Window-Based Link Level Retransmission," *in Proceedings of IEEE/ACS International Conference on Computer Systems and Applications*, Doha, Qatar, pp. 386-391, 2008.

[3] Elshaikh M., Othman M., Shamala S., and Desa J., "A New Fair Weighted Fair Queuing Scheduling Algorithm in Differentiated Services Network," *the International Journal of Computer Science and Network Security*, vol. 6, no. 11, pp. 267-271, 2006.

[4] Fengnet., available at : http://fengnet.com/book/ Optimizing-Applications /ch05lev1sec7.html, last visited 2011.

[5] Georg M., Jechlitschek C., and Gorinsky S., "Improving Individual Flow Performance with Multiple Queue Fair Queuing," *in Proceedings of the 15th IEEE International Workshop on Quality of Service*, Evanston, USA, pp. 141-144, 2007.

[6] Hwang I., Hwang B., and Ding C., "Adaptive Weighted Fair Queueing with Pritority (AWFQP) Scheduler for Diffserv Networks," *Journal of Informatics & Electronics*, vol. 2, no. 2, pp. 15-19, 2008.

[7] Ioannou A. and Katevenis M., "Pipelined Heap (Priority Queue) Management for Advanced Scheduling in High-Speed Networks," *IEEE/ACM Transactions on Networking*, vol. 15, no. 2, pp. 450-461, 2007.

[8] Jin J., Wang W., and Palaniswami M., "Utility Max-Min Fair Resource Allocation for Communication Networks with Multipath Routing," *Computer Communications*, vol. 32, no. 17, pp. 1802-1809, 2009.

[9] Josevski N. and Chilamkurti N., "A Stabilized Core-Stateless Fair Queuing Mechanism for a Layered Multicast Network," *in Proceedings of International Conference on Networking and Services*, California, USA, pp. 16-21, 2006.

[10] Kaur J. and Vin H., "End-to-End Fairness Analysis of Fair Queuing Networks," *in Proceedings of the 23rd IEEE Real-Time Systems Symposium*, Texas, USA, pp. 49-58, *2002*.

[11] Kong C., Guo Z., Ping L., and Peng X., "PSRED: A Queue Management Algorithm with Priority Self-adaptive Random Early Detection for Ad-hoc Network," *in Proceedings of the International Workshop on Information Security and Application*, China, pp. 557-560, 2009.

[12] Matsufuru N. and Aibara R., "Efficient Fair Queueing for ATM Networks Using Uniform Round Robin," *in Proceedings of the 18th Annual Joint Conference of the IEEE Computer and Communications Societies*, New York, USA, vol. 1, pp. 389-397, 1999.

[13] Mishkoy G., Giordano S., Andronati N., and Bejan A., "Priority Queueing Systems with Switchover Times: Generalized Models for QoS and CoS Network Technologies and Analysis," *Technical Report*, Free International University of Moldova, Moldova, 2006.

[14] Network Simulator, available at: http://www.isi.edu/nsnam/ns, last visited 2011.

[15] Rai I. and Alanyali M., "Uniform Weighted Round Robin Scheduling Algorithms for Input Queued Switches," *in Proceedings of IEEE International Conference on Communications*, vol. 7, pp. 2028-2032, 2001.

[16] Saha D., Mukherjee S., and Tripathi S., "Carry-Over Round Robin: A Simple Cell Scheduling Mechanism for ATM Networks," *IEEE/ACM Transactions on Networking*, vol. 6, no. 6, pp. 779-796, 1998.

[17] Semeria C., *Supporting Differentiated Service Classes: Queue Scheduling Disciplines*, Juniper Networks, USA, 2001.

[18] Singh L. and Srivastava R., "Design and Implementation of G/G/1 Queuing Model Algorithm for its Applicability in Internet Gateway Server," *the International Arab Journal of Information Technology*, vol. 5, no. 4, pp. 111-118, 2008.

[19] Stamoulis A., Sidiropoulos N., and Giannakis G., "Time-Varying Fair Queuing Scheduling for Multicode CDMA Based on Dynamic Programming," *IEEE Transactions on Wireless Communications*, vol. 3, no. 2, pp. 512-523, 2004.

[20] Stoica I., Shenker S., and Zhang H., "Core-Stateless Fair Queuing: A Scalable Architecture to Approximate Fair Bandwidth Allocations in High Speed Networks," *IEEE/ACM Transaction on Networking*, vol. 11, no. 1, pp. 33-46, 2003.

[21] Tsao S., Lai Y., Tsao L., and Lin Y., "On Applying Fair Queuing Discipline to Schedule Requests at Access Gateway for Downlink Differential Qos," *the International Journal of Computer and Telecommunications Networking*, vol. 52, no. 18, pp. 3392-3404, 2008.

[22] Willig A., A Short Introduction to Queueing Theory, avalible at: http://www.cs.ucf.edu/ ~lboloni/Teaching/EEL6785_Fall2010/slides/QueueingTheory.pdf, last visited 1999.

Nandhini Sivasubramaniam has obtained her M.Sc in mathematics and Master of Philosophy in mathematics. She is currently working as a senior lecturer in the Department of Computer Science at Garden City College of Science and Management Studies, India. She has presented papers in national conferences. Her area of interest includes queueing theory, computer networks, graph theory and operations research.

Palaniammal Senniappan received her PhD degree in applied mathematics from PSG college of Technology, Coimbatore in 2006. She is currently working as a professor and head in the Department of Science and Humanities at VLB Janakiammal College of Engineering and Technology, India. She has published 57 papers in national/ international journals/ conferences. Her areas of interest are queueing theory, data mining and networking. Under her guidance 9 scholars are pursuing their PhD and 2 M.Phil scholars has been completed. She has also authored 9 books in mathematics for B.E/B.Tech students and 2 of her books "Probability and Queueing Theory", "Probability and Random Processes" are published in Prentice Hall of India.