A New Fair Weighted Fair Queuing Scheduling Algorithm in Differentiated Services Network

M. A. Elshaikh, M. Othman, S. Shamala and J. Desa
Department of Communication Technology and Network
Faculty of Computer Science and Information Technology
University Putra Malaysia
Selangor, D.E., Malaysia

Summary
Many new technologies has been proposed by the Internet Engineering Task Force (IETF) to cover the new real time applications which are becomes very important in today's Internet demands. One such technology is Differentiated Services (DiffServ). This has been introduced to provide better QoS where the routers provide PHBs to aggregate traffic for different levels of services and the scheduling algorithm used by the DiffServ routers is playing a critical role in implementing those PHBs. In this paper a new scheduler, Fair Weighted Fair Queuing (FWFQ), has been proposed that can be used effectively in a DiffServ networks. We evaluate the performance of our proposed FWFQ algorithm using extensive network simulation with a comparison to the current used algorithms WFQ and WIRR. The results from the simulation studies indicate that the scheduling algorithm we propose ensures both the required bandwidth fairness and end-to-end network delay bounds for QoS in DiffServ networks.

Key words:
Diffserv, FWFQ, Scheduling, QoS

1. Introduction

There is a need for mechanisms to support QoS in the Internet to provide appropriate services for delay and loss sensitive applications. The IETF has proposed two architectures for that, namely, IntServ and DiffServ.

The IP QoS architecture development began with the IntServ concept [1], which deals with individual flows and relies on signaling to reserve the network resources necessary to satisfy QoS requirements for each flow along the flow's path. The scalability problem led to the design and introduction of DiffServ architecture [2], in this architecture, aggregates of flows are allocated resources in accordance with a small number of standardized QoS specifications based on the PHB construct.

1.1 DiffServ

DiffServ model has been developed to provide an efficient platform for service providers to commit and fulfill contracts with customers [3]. DiffServ push the flow-based traffic classification and conditioning to the edge router of a network domain. The core of that domain is only having a responsibility of forwarding the packets according to the PHB associated with each traffic class; which is identified by the DiffServ Code Point (DSCP) field in the header of each packet. Currently, the IETF defines a set of PHBs which includes Expedited Forwarding (EF) PHB, Assured Forwarding (AF) PHB and Best Effort (BE) PHB [4, 5, 6]. The implementation of PHBs relies much on the marker, scheduling and queuing schemes used in switches and routers [7, 8].

In this paper, a new scheduler is proposed and implemented using NS-2. The results acquired shown that the proposed algorithm performs better than the current available algorithms for DiffServ.

1.2 Unfairness of Weighted Fair Queuing

Generalized Processor Sharing (GPS) is an ideal scheduling algorithm [9]. In this algorithm, packets from each flow are classified into different logical queues. GPS serves non-empty queues in turn and skips the empty queues. It sends an infinitesimally small amount of data from each queue, so that in any finite time interval it visits all the queues at least once [10]. There can be a service weigh associated with each queue. Queues receive service according to their associated weights. Because GPS posses the properties of ideal fairness and complete isolation, there are many research studies have been done on it. However, GPS is not implementable because serving an infinitesimal amount of data from each non-empty queue is not possible. Thus, various emulations of GPS have been proposed in the literature, one of those variations, which used in DiffServ networks is WFQ.

Packet by packet Generalized Processor Sharing (PGPS) and WFQ algorithms are both approximations of GPS. The difference between these algorithms and GPS is that unlike GPS they do not service an infinitesimal amount of data from each queue. Another improvement, which has

Manuscript received November 5, 2006.
Manuscript revised November 25, 2006.

been done to GPS in these algorithms, is that, in the case of flows' variable packet sizes, they need not to know the average packet size in advance [9, 11]. WFQ is essentially the same as PGPS, but they were independently developed. Thus, we only focus on explaining WFQ. WFQ was developed by Demers, Keshav, and Shenker in 1989. The idea behind the algorithm is that for each packet, WFQ computes the time at which service to the packet would be finished, deploying a GPS scheduler. Then the WFQ scheduler services the packets in the increasing order of their finish times. In other words, WFQ simulates GPS on one side and uses the results of this simulation to determine the packets' service order.

It has been prove that WFQ have an excellent use in creating firewalls between classes but at the same time it does punishing flows or classes [12], in the case of DiffServ, for using uncontended bandwidth which making it not suitable for DiffServ architecture.

There are some enhancements have been done for WFQ [12, 13, 14, 15] although all those algorithms are fair in the worst-case sense and tend to have low delay, they were not designed to provide service differentiation among classes in the context of DiffServ networks.

The rest of the paper is organized as follows. Section 2, describes the new proposed algorithm (FWFQ). The scenario of the experiment used in the paper is explained in section 3. Simulation results and analysis are discussed in section 4, while Section 5 concludes the paper.

2. FWFQ

FWFQ has been proposed in an attempt to correct the above-mentioned problem. It proposes the usage of two types of queue and uses the queue length as a parameter to in calculating the virtual time to ensure that the flows or aggregates are not punished for using uncounted bandwidth.

In FWFQ the virtual time will approach along with the real time same like the WFQ but will not be fixed so that it will be moving according to the queue length. Compare the current queue length with the weight when a packet arrive or depart.

We can summarize the algorithm as shown in Figure 1. The basic idea of FWFQ is dealing with current queue length in respects with the weight of that queue in the current class, which it can be summarized in the following equation

$$W_i = \mathrm{f}\,(\mathrm{Q\ length}_i^k) \tag{1}$$

Assume that W_i is not changing during the transmission for all i.

Then let X_i be the current number of packets for class C_i in the output queue and q_i is the current queue length, and W_i^0 is the basic weight for of class i.

$$X_i = \min\,(W_i^0 + Q_i\ ,\ W_i^{max}) \tag{2}$$

Analytically we can prove that the FWFQ resulting in a delay bound similarly to that of WFQ.

$$D_i^k < D_i^{k-1} + C_i \times \frac{1}{f(Q_i^{k-1})} \tag{3}$$

The delay of packet k in class i is $\mathrm{delay}_i^k = D_i^k - t_i^k$, similar to that of WFQ.

Packet arrival (P_i)
If (Scheduler not idle)
When a packet arrive at time t
  Update the system virtual time v'(t) using the last-vt-update v'(t) at time β
    $V(t) = v'(\beta) + (t - \beta)/Sum$; where $Sum = \sum_{i \in B(i)} r_i$
    Where $\beta <= t < \beta + 1$
    $V'(t) = t$
Else
    $V'(t) = t$;
    $V(t) = 0$;
Previous Queues finish time = last queue finish time / Gamma;
Compute the finish time for the packet (i.e. time stamp the packet)
  Fill the queues finish time to the previous finish time For $F_j = F_i$
  Get S^k_i for the new packet
    $S^k_i = \max\,(F_i^{k-1}, V(t))$
  Calculate
    $F_i = S^k_i + L_i^k/r_i$
Place the packet with time stamp to the related queue
  Get $Sum = \sum_{i \in B(i)} r_i$
  $V(t_i) = V(\beta) + (t_i - \beta) / Sum$
  Get the packet with a minimum $V(t_i)$ to be transmitted

Figure 1: FWFQ Algorithm

3. Experimental Model

In this section we outline the model of our proposed algorithm and the simulation model used to verify the performance of FWFQ compared to WFQ and WIRR. The result acquired displays the correlation between the scheduling mechanisms and the performance metrics.

3.1 Simulation Model

In [16] performance evaluation of Dynamic DiffServ Scheduling (DDS) and Priority Queuing Weighted Round Robin (PQWRR) was performed. However, in this paper WIRR, WFQ and FWFQ are considered because currently the most common used algorithms are WIRR and WFQ algorithms.

The network model shown in Figure 2 is utilized. The sources are connected with an edge router with link capacity of 10Mbps and the ingress connected to the core with a link capacity of 10Mbps and from the core to the ingress is connected with a link capacity 10Mbps, this core is connected to the egress edge router with a bottleneck link capacity 5Mbps and from the edge router (egress) to the core with a link capacity 5Mbps which is to congestion. Then, this egress router is connected to the destination nodes with a link capacity of 10Mbps. The delay time in all links is set to 5ms.

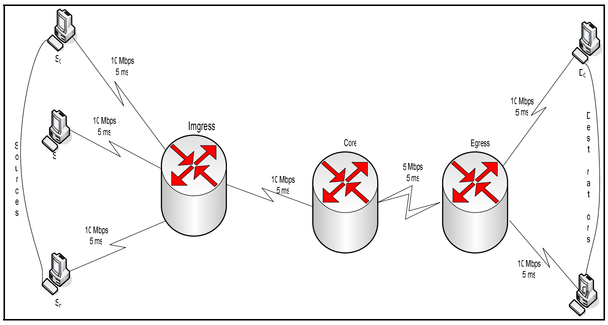

Figure 2: DiffServ Network configuration used in the simulation

Variable Bit Rate (VBR) traffic is generated from the sources to the destinations with a different traffic rate values. A DiffServ domain at its edge may control the amount of traffic that enters or exits the domain at various levels of drop precedence. These traffic conditioning may include traffic shaping, discarding packets and reassigning of packets to a different traffic class. In this paper the implementation of traffic conditioning is done via a token bucket shaper.

3.2 Model Scenario

The result will be discussed by usage of two kinds of scenarios. In the first one, we have used the model in order to evaluate our proposed algorithm with respects to delay losses and jitter. However, in the second scenario, we focused in the fairness of our proposed algorithm compared with other algorithm in the case of different classes are used in the DiffServ network.

4. Results and Discussions

The key aspects of our experiments in this paper is to evaluate FWFQ algorithm on its guarantee of bounds delay and jitter, as well as the minimum guaranteed bandwidth for the class that was given high priority, while equally observing its fair allocation of link bandwidth to other low priority service classes. These performance characteristics enable to determine whether the suggested algorithm is fair and efficient, thus it can support applications in DiffServ networks in order to achieve an acceptable performance.

Figure 3 show that WFQ has a better performance among all of the mechanisms used in terms of Idrop, which represents the packet drop due to an overflow. However, for the range of 10% up to 50% network provision FWFQ is performing better than the other algorithms.

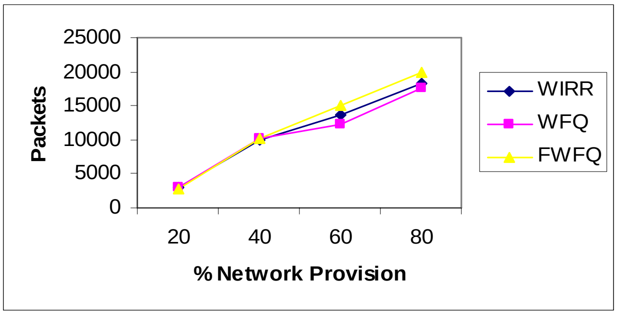

Figure 3: Idrop

In terms of the dropping due to a Random Early drop (RED) mechanism, which we denoted as Edrop, Figure 4; WIRR has a better performance as compared to WFQ and FWFQ. While up to 50% network provision FWFQ is performing better than WFQ algorithm.

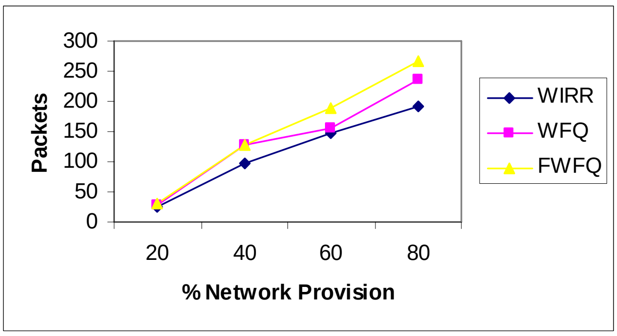

Figure 4: Edrop

In comparison of total packets dropped, it is observed that, as shown in Figure 5, WIRR has a better performance overall among the all compared algorithms making it more suitable for those sensitive to loss applications. However up to 50% network provision FWFQ performs better.

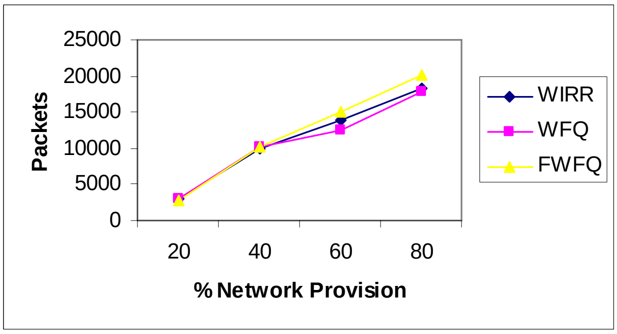

Figure 5: Total Drop

Table 1 show that, in contexts of delay FWFQ has almost a similar performance like WFQ, which is better than WIRR and in terms of jitter FWFQ is outperforms WFQ.

Scheduler Type	Delay (ms)	Jitter (ms)	Loss (% packets)
WIRR	215.185	42.1721	62.08
WFQ	194.095	47.4337	58.43
FWFQ	189.817	43.7446	58.65

Table 1: Average Delay, Loss and Jitter

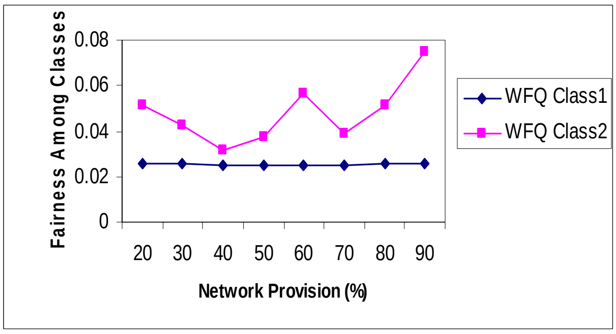

Figure 6: WFQ Fairness Index for Different Classes

Figure 6 shows the service fairness index for different DiffServ classes uses WFQ, it is clear that the algorithm is experiencing some sort of unfairness among the classes sharing the DiffServ domain.

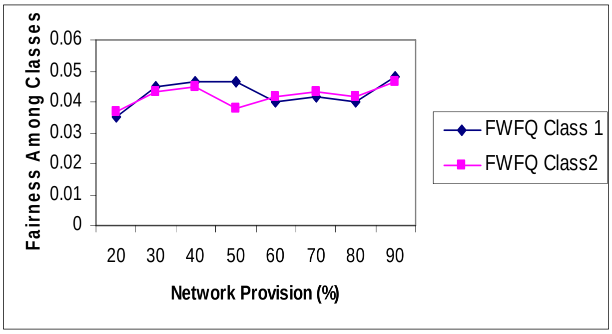

Figure 7: FWFQ Fairness Index for Different Classes

In Figure 7, it is clear that by using our proposed FWFQ the fairness of the classes sharing the DiffServ network becomes better making our algorithm suits the DiffServ architecture.

5. Conclusion

In this paper we have proposed a new scheduler that it 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 Idrop 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 we can see that WFQ has a better performance overall among the all compared algorithms making it more suitable for those sensitive to loss applications. However up to 50% network provision FWFQ performs better. For delay sensitive applications, FWFQ is better; it gives a better performance in terms of delay and delay jitter. We have also presents the evaluation results of a simulation based study on the fairness criteria of packet scheduling algorithms to support QoS in DiffServ networks and it shown that FWFQ is the most suitable one.

The simulation study evaluates the performance of our proposed FWFQ algorithm using extensive network simulation in comparison to WFQ and WIRR. For delay sensitive applications FWFQ is better; it gives a better performance in terms of delay and delay jitter and ensures the required bandwidth fairness among the classes sharing the DiffServ networks and also suites the loss sensitive applications in the rang of 10% to 50% network provision level.

References

[1] R. Braden, D. Clark, and S. Shenker, "Integrated services in the Internet: an overview", RFC1633, www.ietf.org/rfc/rfc1633.txt.

[2] S. Blake et al, "An architecture for differentiated services", RFC2474, www.ietf.org/rfc/rfc2475.txt

[3] J. A. Zubairi, M. A. Elshaikh and O. Mahmoud, "A study of VBR traffic treatment in DiffServ domain", International Journal of Modeling and Simulation, Vol. 23, No. 3, pp 187-193, 2003.

[4] B. Carpenter, and K. Nichols, "Differentiated services in the Internet" In proc. IEEE, vol. 90, No. 9, Sept. 2002, pp. 1479- 1494.

[5] M. A. Elshaikh and M. Othman, "Differentiated services: architecture overview study", Journal of Technology and Management, Vol. 1, No. 1, pp 7-11, 2003.

[6] D. Hang, H. Shao, W. Zhu, and Y.Q. Zhang, "TD2FQ: An integrated traffic scheduling and shaping scheme for DiffServ networks", in IEEE International Workshop on High Performance Switching and Routing, 2001.

[7] M. A. Elshaikh, M. Othman, S. Subramaniam and R. Johari, "An Enhancement of TSWTCM Algorithm to Improve Fairness in DiffServ Networks: Design and Implementation", Brunei International Conference of Engineering and Technology 2005, BICET 2005, Vol. 3, pp. 89 – 96, 2005.

[8] M. A. Elshaikh, M. Othman, S. Subramaniam and R. Johari, "Enhanced TSWTCM to Improve Fairness in DiffServ Networks", 13th IEEE International Conference on Networks, IEEE MICC-ICON Proceeding, Kuala Lumpur, Malaysia, pp. 302 – 307, 2005.

[9] Parekh, A., Gallager, R., " Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: the Single Node Case", IEEE/ACM Transactions on Networking, Vol. 1, No. 3, pp. 344-357, 1993.

[10] Keshav, S., "An Engineering Approach to Computer Networking", Addison Wesley, 1997.

[11] Parekh, A., Gallager, R., " Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: the Multiple Node Case", IEEE/ACM Transactions on Networking, Vol. 2, No. 2, pp. 137-150, 1994.

[12] CISCO Systems, "Documentation on QoS", http://www.cisco.com/warp/public/732/Tech/qos, 2002.

[13] Golestani, S., "A self-Clocked Fair Queuing Scheme for Broadband Applications", in Proc. Of IEEE INFOCOM'94, Toronto, Canada, Vol. 2, pp. 636-646, 1994.

[14] J. Heinanen , Telia Finland, R. Guerin, "A two rate tree color marker", RFC 2698, September 1999.

[15] Risso, F., "Quality of Service on Packet Switched Networks", PhD thesis, Faculty of Computer Science, Torino Polytechnic, Italy, 2000.

[16] Yang, M., Lu, E., Zheng, S., "Scheduling with Dynamic Bandwidth Allocation for DiffServ Classes", 12th International Conference **on Computer Communications and Networks (ICCCN 2003), Dallas, TX, 2003.**

M.A. Elshaikh received his BSc. in Computer Science from Sudan University of Science and Technology (SUST), Sudan and his MSc. in Computer Engineering from the International Islamic University Malaysia (IIUM), Malaysia; He worked as teaching and research assistant at SUST and IIUM. Currently he is a PhD student at University Putra Malaysia (UPM). His research interest includes network traffic engineering, parallel computation, distributed systems, mobile communication, network analysis, performance and design.

M. Othman completed PhD from University Kebangsaan Malaysia in 1999 (with best PhD thesis awarded by Sime Darby Malaysia and Malaysian Mathematical Science Society). Currently, he is Associate professor at the Department of Communication Technology and Networks, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia. His research interest includes parallel and distributed algorithms, high speed computer network, multiprocessor system on chip (MSoC), network management (network security and traffic monitoring) and scientific computing. He has published more than 80 articles in journals comprising of both local and international. He is associate researcher and the coordinator for the High Performance Computing at the Institute of Mathematical Research (INSPEM), University Putra Malaysia.