4V－6

A Packet Scheduling Mechanism for Supporting Real-Time Applications on High Speed Networks

Onur ALTINTAS and Yukio ATSUMI

UNCL
Ultra-high Speed Network and Computer Technology Laboratories

Abstract Packet scheduling is one of the key mechanisms that will be employed in the network nodes for supporting real-time applications. In this paper we propose a new frame-based packet scheduling algorithm which calculates and keeps an index for each flow in order to keep track of instantaneous bursts. With this approach, flows which might be in need of momentary service can be detected. After a brief introduction of packet scheduling issues, we describe the operation of our algorithm. We then give some simulation results showing the delay performance of the proposed algorithm.

1 Background

Guaranteeing performance by employing packet scheduling at the intermediate nodes has been recently drawing attention. Conventionally, networks have used FIFO discipline at the network nodes which can not be effective for QoS provision. In a packet switching network supporting various classes of applications, since packets from different flows interact with each other at each queueing point, the network should schedule its resources. However, the choice of such a packet service discipline is a major design challenge. Recently, a number of scheduling disciplines have been proposed (a comprehensive overview of most of these algorithms is given in [5]). These are categorized into *timestamping-based* algorithms and *frame-based* algorithms according to their methods of handling packets/flows.

WFQ (Weighted Fair Queueing) [3] has been the most well-known timestamping algorithm. It is a packet-by-packet emulation of a hypothetical system called the Generalized Processor Sharing (GPS). WFQ timestamps each packet with an index indicating their virtual finish times. Despite its analytically interesting characteristics about delay and fairness, the main opposition to WFQ is its complexity which makes it impractical to implement on high-speed networks. On the other hand, algorithms in the frame-based category divide the time axis into frames of service allotments. WRR (Weighted Round Robin) [2] is one of the most well-known algorithms of this category. Compared to the timestamping schedulers, frame-based ones have much smaller computation times. However, algorithms in this category usually have loose delay bounds especially under bursty traffic. In this paper, we propose a new frame-based scheduling discipline which we call *Urgency-based Round Robin (URR)*.

2 Urgency-based Round Robin Scheduling

In this section we describe the operation of the algorithm we propose. The basic contribution of this work is to employ an index table which is an adaptive approach taking into consideration the instantaneous needs of flows incoming to a queueing point. Incoming flows are isolated and put into different buffers which operate as FIFO queues. Fixed size packets (or cells) from various flows arrive into the system and get served according to their reserved bandwidth shares (weights). Normally, in a frame-based scheduling discipline such as WRR, the time axis is divided into equal size *frames*. During each time frame, the server allocates w_i time quanta (slots) to flow i in order to satisfy a guaranteed rate, where w_i is the weight reserved for flow i. The server then serves the non-empty queues in an orderly fashion sending one packet at a time from each flow until satisfying the assigned weights. Actually, WRR is an efficient and easy to implement scheme for utilizing network resources, however its delay performance becomes poor under bursty traffic conditions. Next, we introduce the *urgency index*: At time t, for a flow with queue size of $q_i(t)$ and service share (weight) of w_i, $u_i(t) = q_i(t)/w_i$ is called the urgency index of that flow indicating the total duration it should get service to empty its queue.

URR algorithm calculates and updates a table of urgency indexes, right after a round completes. The calculated index is kept only for each queue (flow or session) rather than timestamping each packet. Basically, the scheduling engine starts each new round from the flow with the largest urgency index. In other words, the next packet to be forwarded out first in the next round is selected based on the instantaneous needs of flows (Figure 1). Note that the system is not static as in a simple priority-based scheme nor its cycle order is fixed as in WRR. Also, note that our scheme is work-conserving. The underlying argument of our approach is the following: The assignment of service rates to sessions is static, however, it is possible that within a small amount of time, a particular queue temporarily exhibit a higher arrival rate than its assigned weight. In other words, even if the source in question is shaped and well-behaving at the edge of the network, it is known that it is still possible that the packets belonging to that flow gets bursty inside the network due to the interaction with packets from other flows.

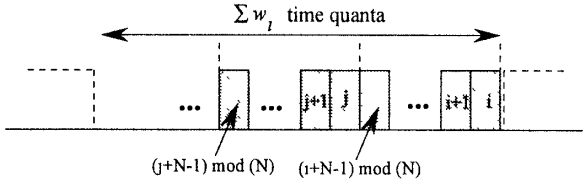

Figure 1.Output stream of the URR scheduler for N flows.

3 Simulation Results

In the model used in the following simulations, all sources produce traffic for the same destination node. All traffic flows through a single scheduler which is connected to the destination. Our primary goal is first to investigate how the proposed scheduler performs under several

patterns of traffic load within a single node. Each source is connected to the scheduler node via an infinitely fast link. Two kinds of traffic source models are used in the simulations: ON-OFF model, and the Poisson model. In the following, we will be interested in the delay characteristics of the packets produced by the ON-OFF sources in the presence of other type of traffic sources sharing the same link. ON-OFF source is, basically, a two-state Markov modulated process. In the source we use, the packet stream consists of arrivals with T ms intervals when the model is in ON state. The ON and OFF state durations are distributed exponentially with means α and β, respectively. We choose T = 16 ms and the packet length L = 512 bits. Also, we set α to 352 ms, and β to 650 ms. Setting β to 650 ms gives a standard packetized voice source model; and setting it to values less than 352 ms corresponds to a model of continuous packet rate source which is typical in real-time remote sensor devices [4]. ON-OFF sources reserve 32 Kb/s. Poisson sources have exponentially distributed interarrival times with mean $1/\lambda$, λ depending on the experiment. Note that Poisson sources reserve a rate of 512λ Kb/s.

Next we present the results of two sets of simulations. In the first set of simulations, there are 10 sources of which 9 behave as Poisson sources, and one as an ON-OFF source. All of the sources reserve equal amount of shares. The average delays experienced by the Poisson sources do not differ under two disciplines, therefore we omit results related to these sources. Since our underlying argument is that burstiness might occur inside the network even if the sources are well-behaving and shaped at the edges, in the first set of simulations, we assume that the scheduler is one of the intermediate nodes on a several hop path, and intentionally squeeze T. We obtain packet arrivals during the ON state ranging from values corresponding to the reserved bandwidth of the source, to values roughly corresponding to two times the reserved bandwidth. Specifically, T is varied from 15 ms to 8 ms, yielding what we call a degree of burstiness of 1.07 to 2. Note that, the utilization for this source is still well below the reserved rate (0.702 for the "busiest" setting). Actually, this scenario is quite possible for a flow passing through several nodes and getting more concentrated at each node. Also, note that, it is not our aim to prioritize a malicious user, but to detect and relieve a bursty duration for any flow. In Figure 2 we show mean delays and 99th percentile delays for the first set of simulations. The gap between URR and WRR performances widens as the burstiness of the ON-OFF source gets more intense. This situation is explained as follows. In WRR, during any frame, the ON-OFF source normally waits for its slot corresponding to its share for 9 slots of time. However with URR, if it has the eligibility to enter from the first slot (i.e. if its u_i is larger than others), it takes service immediately. However, this does not mean that other flows are sacrificed. In a single round, none of the flows are skipped, rather their orders of service are changed.

In the second set of simulations, all of the sources are ON-OFF sources with varying mean OFF durations corresponding to models ranging from standard voice to real-time remote sensors. We show results of this set in Table 1 where β/α is the ratio of mean OFF duration to mean ON duration. In all cases URR has better delay performance compared to WRR. However, as expected, URR performs much better for those flows with small mean OFF durations.

4 Conclusions and Future Work

We have proposed and evaluated a new packet scheduling discipline called URR which can be considered as a version of WRR with improved delay characteristics. We have shown through simulations that the delay performance of the proposed algorithm under various traffic loads is better compared to WRR. Future work will include extending the urgency index table to a sorted one which will constitute the basis of the output pattern (i.e. in descending order).

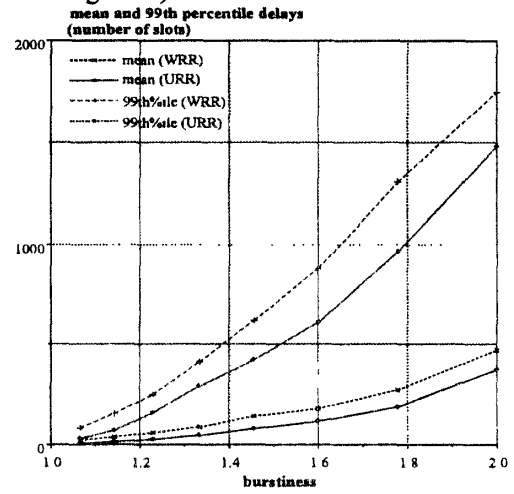

Figure 2. Mean and 99th percentile delays of the ON-OFF source in the first set of simulations.

node	β/α	Mean (WRR)	Mean (URR)	99th %ile (WRR)	99th%ile (URR)
1	0.018	1.698	0.722	6.869	4.944
2	0.053	1.606	0.779	7.306	6.344
3	0.111	1.623	0.831	7.612	6.125
4	0.250	1.680	0.954	7.963	6.694
5	0.429	1.540	0.949	7.963	6.125
6	0.818	1.571	1.037	4.988	4.988
7	1.847	1.378	1.041	5.993	5.556

Table 1. Mean and 99th percentile delays for all flows in the second set of simulation (values are in number of slots).

References

[1] O. Altintas, Y. Atsumi and T. Yoshida, "On a packet scheduling mechanism for supporting delay sensitive applications on high speed networks," *Proc. ICICS'97*, Sept. 1997.

[2] M. Katevenis, S. Sidiropoulos, and C. Courcoubetis, "Weighted round-robin cell multiplexing in a general-purpose ATM switch chip," *IEEE JSAC*, pp. 1265-1279, Oct. 1991.

[3] A.K. Parekh and R.G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The single node case," *IEEE/ACM Trans. on Networking*, pp. 344-357, June 1993.

[4] D. Yates, J. Kurose, D. Towsley, and M.G. Hluchyj, "On per-session end-to-end delay and the call admission problem for real-time applications with QoS requirements," *J. of High Speed Networks*, pp. 429-458, Dec. 1994.

[5] H. Zhang, "Service disciplines for guaranteed performance service in packet-switching networks," *Proceedings of the IEEE*, pp. 1374-1396, Oct. 1995.