DLB: A Novel Real-time QoS Control Mechanism for Multimedia Transmission

Jian Li, Ye-Qiong Song

DLB: A Novel Real-time QoS Control Mechanism for Multimedia Transmission

Jian Li and YeQiong Song

LORIA – INPL

Campus Scientifique - BP 239 - 54506 Vandoeuvre-lès-Nancy Cedex, France
e-mail: Jian.Li@loria.fr; Song@loria.fr

Abstract

This paper presents a new QoS guarantee scheme called R-(m,k)-firm (Relaxed-(m,k)-firm) which provides the guarantee on transmission delay of at least m out of any k consecutive packets (m≤k). It has several advantages: (1) during network congestion, packets are dropped according to the (m,k) model rather than uncontrollably as the case of TD and RED, avoiding thus undesirable long consecutive packet drops; (2) it allows to admit more real-time flows than the traditional over-provisioning approach. A new mechanism, called DLB (Double Leaks Bucket) is also proposed for dropping a proportion of packets of a flow or of aggregated-flows in case of network congestion while still guaranteeing the R-(m,k)-firm constraint. The sufficient condition for this guarantee is given for configuring the DLB parameters. It is easy to implement DLB in the actual IntServ and Diffserv architectures (by simply replacing the actual leaky bucket by DLB) for providing respectively per flow and per class (m,k) guarantee, or event per flow R-(m,k)-firm guarantee in Diffserv.

1. Introduction

Nowadays, Internet supports more and more real-time and business-critical applications. However, for providing them with transmission delay guarantee, neither Intserv nor Diffserv consists in an efficient solution. In fact, as a flow often generates bursty traffic (case of most VBR applications and aggregated Diffserv class of flows), for providing a bounded transmission delay to a flow (Intserv) or a class of flows (Diffserv), most of the existing solutions are based on the over-provisioning bandwidth reservation policy according to the peak rate of the flow. This results in a poor admission rate of real-time flows. Moreover, Diffserv does not provide any end-to-end real-time QoS guarantee [1]. Another problem we should deal with in providing real-time QoS is the

network congestion because of the router overload. This can occur when a path includes one or several routers which do not support Diffserv. As it may lead to packet drop, although occasional packet drops could be tolerated by many multimedia applications, long consecutive packet drops must be avoided since it can drastically decease the QoS for applications such as audio/video diffusion. Unfortunately, existing queue management schemes such as TD (Tail-Drop) do not address the consecutive packet drop problem. RED (Random Early Detection) [2] has been proposed to deal with the problem with random dropping. However, it does not give any guarantee on non-consecutiveness of the packet drops.

In this paper we propose a solution to deal with both *low admission control* caused by over-provisioning and *consecutive packet drop* problems. For reducing over-provisioning, the key idea we exploit is to take advantage of the "natural" packet loss tolerance of a large class of real-time applications to reduce the sufficient and necessary bandwidth reservation. In fact, if we consider applications such as video-on-demand (VoD), IP telephony, VoIP, Internet radio, teleconferencing, interactive games, distance learning, telemedicine, High Definition TV (HDTV), etc., many of them can tolerate packet losses to some extent. Graceful degradation of quality of service (QoS) is acceptable in case of *congestion* or when the peak rate reservation (i.e., over-provisioning) cannot be provided at *admission control*.

At first glance, the *(m,k)-firm* model [3] seems to be an interesting one for resolving both of the above problems. In *(m,k)-firm* model, the deadlines of at least *m* out of any consecutive *k* packets must be met. Notice that the term "any consecutive *k* packets" implies a sliding window guarantee for a flow. In other words, if the *(m,k)-firm* is deterministically guaranteed, there will never be more than *k-m* consecutive packet drops. On the other hand, specifying deadline miss tolerance of an application according to *(m,k)-firm* brings an interesting alternative for resource reservation comparing with the classic hard real-time model (which

can be considered as a *(k,k)-firm* one) and for more deterministic guarantee comparing with the soft real-time model.

Some work exists in applying *(m,k)-firm* to QoS management. [4] implemented *(m,k)-firm* model instead of RED for congestion control and the result showed its interest. However nothing is provided concerning the transmission delay guarantee. [5] proposed an integration of the existing *(m,k)-firm* scheduling algorithms into the *Diffserv* architecture, but only for providing average performance improvement. So deterministic *(m,k)-firm* delay guarantee is still not addressed although it is required by some critical real-time applications and by the property of non-consecutiveness of packet drops.

Intuitively, reserving resources according to *(m,k)-firm* requirement rather than *(k,k)-firm* should reduce the necessary resource reservation as only *m* instead of all the *k* packets should be processed in time. This is true, for example, for the case when flows are serviced by a WFQ server [6]. Unfortunately, it has been proven that in general, for achieving deterministic *(m,k)-firm* guarantee, one has to reserve resources according to (k,k)-firm since the worst case must be considered [7, 8, 9]. Moreover, the problem of non pre-emptive scheduling of *N* (m_i,k_i)-firm constrained flows (*i =1, …, N*) has been proved NP-hard in the strong sense, such that no optimal scheduling can be expected under such model.

To resolve this tedious technical problem, we propose in this paper *R-(m,k)-firm* to extend the concept of *(m,k)-firm* by relaxing packet deadline. Instead of considering the traditional guarantee of the individual packet deadline, we define a global deadline for any group of *k* consecutive packets. In an interval]*s, t*], the source has sent *k* packets to the network, then the destination should be assured to receive at least *m* among them before time *t+Δ*, where *Δ* is the maximum tolerable transmission delay caused by the network for any group of *k* consecutive packets. This QoS scheme allows a graceful QoS degradation between satisfying all packets' deadlines (i.e. *(k,k)-firm* guarantee) and *(m,k)-firm* guarantee.

In order to implement *R-(m,k)-firm*, we also designed a new mechanism, called DLB (Double Leaks Bucket) for dropping a proportion of packets of a flow or a class of flows in case of network congestion or admission control while still guaranteeing the *R-(m,k)-firm* constraint.

In the following of this paper, we first define the R-*(m,k)-firm* constraint in section 2; then, we describe the DLB mechanism and give the sufficient schedulability conditions for its configuration in section 3; section 4 shows a performance comparison among DLB, DT and RED; the applicability of DLB is discussed in section 4; finally, we conclude our contributions in section 5.

2. *R-(m,k)-firm* model

Let us consider a periodic or sporadic stream set described as $\Gamma =(\tau_1,… \tau_n)$. A stream (or flow) is denoted by $\tau_i = (c_i, p_i, d_i, m_i, k_i)$, where c_i stands for the transmission time of a packet, p_i stands for the period or minimum inter-arrival time of packets, d_i stands for the deadline before which the packet must be transmitted to the destination node, and m_i, k_i are the (m_i,k_i)-firm constraint. The utilization factor of such a stream set is given in terms of $\sum_{i=1}^{n} \frac{m_i}{k_i} \frac{c_i}{p_i}$.

2.1. Problem of (m,k)-firm constraint

Although the demonstration of the limit of the *(m,k)-firm* model is beyond the scope of this paper, in the sequel, we still briefly quote the main recent results on the schedulability analysis under *(m,k)-firm* constraint, to justify the proposition of *R-(m,k)-firm*.

The initial motivation of *(m,k)-firm* constraint is that the deadlines of at least *m* out of any *k* consecutive packets must be guaranteed. However, as shown, *(m,k)-firm* constraint just takes into account the quantity of the deadline met or miss, whereas a stream is defined with several other parameters, such as *transmission time, period and dropping rate*. This unilateralism of analysis results in bad utilisation of resource, and this poor utilization factor causes the considerable over-provisioning of the system.

Note that this over-provisioning causes the *(m,k)-firm* or its similar systems (DWCS) to loss their practical effect. These facts have been shown in [7, 8, 9]. The follows give a glance over the theorems of discussions on the pessimistic points for the provisioning schemes.

In [9], it has been proven that under an arbitrary low utilisation, there is always a stream set Γ, which causes the system violate the *(m,k)-firm* constraint. In fact, such theorem has also been given for *DWCS* constraint in [8].

More generally, in [7, 8], it has been proven that schedulablity of a stream set under *(m,k)-firm* or DWCS is a problem of NP-hard in strong sense.

Faced to this NP-hard schedulability problem, two ways are possible to get a gain. One is to specialize the stream set and the other is to propose new constraint models other than the *(m,k)-firm* one. In [10], the

stream model is specialized such that the packets of all the streams must have the same transmission time and the same period, then the utilisation factor is improved. In [11], the deadlines of packets are relaxed to reduce the resource requirement.

2.2. Definition of R-(m,k)-firm constraint

We are interested in an absolute guarantee for a system, and the system should accept general streams. Obviously, it is better to find a new real-time constraint suitable to universal stream set. This fact leads us to propose R-*(m,k)-firm*, as shown in Fig. 1.
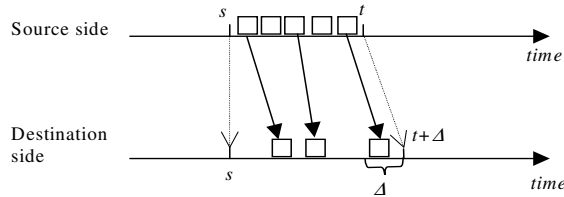

Fig. 1: R-(m,k)-firm constraint

Definition: In a time interval $]s, t]$, a source of flow sends k unit of workload via network, then it should be assured that the destination receives at least m among them before time $t+\Delta$, where Δ is the maximum tolerable transmission delay caused by the network.

Obviously, this QoS requirement is given facing on the integral flow instead of the per-deadline requirement of the conventional real-time constraint.
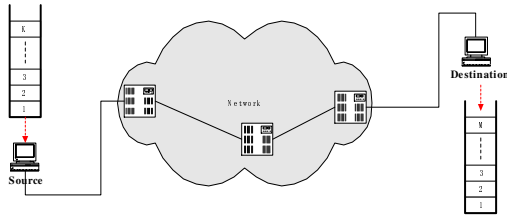

Fig. 2: Flow Integral QoS

This scheme works as shown in Fig. 2. The source has a virtual stack. It sends the packets from its stack head through the network. The destination has also a corresponding stack, and adds the received packets to the stack tail. Supposing that packets are indexed at the source side ($1…n$), and in the time interval $]s, t]$, k packets have been sent to the destination, the QoS provided by the network must assure the destination to receive at least m packets among the k for adding into its stack before $t+\Delta$, and the index of the added packets should be in increasing order. k-m packets could be discarded in any k consecutive ones.

This proposed real-time constraint takes into account the natural real-world degradation, which is acceptable for various applications, such as VoIP, VoD, etc. So far, it is also necessary to provide one schedule so as to guarantee this R-*(m,k)-firm* constraint. Double Leaks Bucket will be detailed in the next section, which can deterministically guarantee R-*(m,k)-firm* constraint.

3. DBL and sufficient condition

In the packet switching networks the information are encapsulated as packets, and the non-preemption is widely employed. According to the R-*(m,k)-firm* constraint given in the previous section, we develop one new mechanism from the traditional leaky bucket [12], which has two leaks named Serving Leak (SL) and Discarding Leak (DL), as shown in Fig. 3.
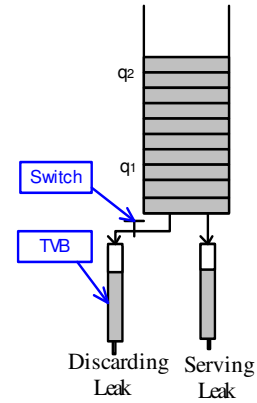
Fig. 3: Double-Leaks Bucket

The network should guarantee the packets that travel through SL, while the DL controlled by one switch gives the capacity to throw out packets from the bucket. With the service guarantee for SL, R-*(m,k)-firm* constraint could be satisfied. The packets going through the DL can be discarded, or treated with whatever method without jeopardizing the network QoS. The strategy to schedule the packets going through either SL or DL will be discussed in section 5, but right away we will introduce how this mechanism guarantees R-*(m,k)-firm* constraint.

As shown in Fig. 3, there are two new parts, named *temporary vessel buckets* (TVB) for DL and SL, respectively. They can take an entire packet from the DLB only after the service of the current packet in themselves. Additionally, TVB of the DL can only get the next during the switch is still in open state

Let C_1 and C_2 denote respectively the capacities of SL and DL. The switch of DL works according to the

workload in the bucket (denoted by q), controlled by *double threshold control function* (DTC function), as shown by Fig. 4, where 1 corresponds to the open state of the DL switch and 0 to the closed state.
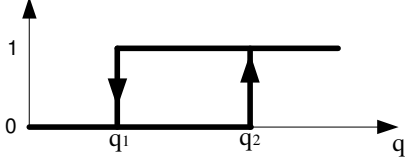
Fig. 4: Double Thresholds Control Function of the Switch

During the increment of the height, the switch remains closed until the height grows to q_2. Once it is opened at q_2, it remains in its open state until the height goes down to q_1. When the switch is opened (control function's value is 1), the Discarding Leak takes packet from the bucket.

The cumulative arrival curve is upper bounded by (r,b) [12], where r stands for the average arrival rate, and b stands for the burst.
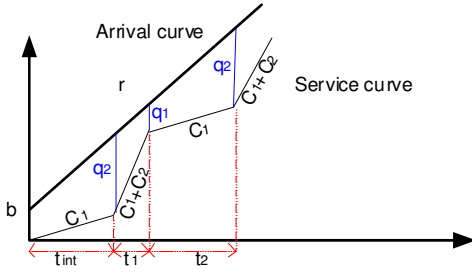
Fig. 5: Service curve of DLB

Fig. 5 shows the service curve under the critical cumulative arrival curve. Notice that SL is always on service unless the bucket is empty.

***Sufficient Condition* for R-(m,k)-firm Constraint**

The *sufficient condition* for *R-(m,k)-firm* constraint of packet model can be given as follows:

Condition (1) $\quad q_1 \geq \dfrac{C_1}{C_2} \geq \dfrac{m}{k-m}$
Condition (2) $\quad$ If $b<q_2$ then $$\max\left(\frac{q_2-1}{C_1}S, \left(\frac{q_2-q_1}{C_1+C_2}+\frac{q_1}{C_1}\right)S\right) \leq \Delta$$ else $$\max\left(\frac{q_2-1}{C_1}S, \left(\frac{b-q_1}{C_1+C_2}+\frac{q_1}{C_1}\right)S\right) \leq \Delta$$

Condition (1) ensures the (m,k) factor: Let $Q_{SL}(t)$ represent the throughput quantity through SL during

]0,t], and $Q_{DL}(t)$ represent the throughput quantity through DL. The following equation must hold $\frac{Q_{SL}(t)}{Q_{DL}(t)} \geq \frac{C_1}{C_2} \geq \frac{m}{k-m}$, since SL is always on service unless the bucket is empty, while DL is on service during the switch is open. Thus in any k packets passed through SL and DL, there are at least m units passing through SL. so that it should be provided that

$$\frac{C_1}{C_2} \geq \frac{m}{k-m} \qquad (1)$$

Furthermore, we must take into account the granularity of the packet. In case of that TVB of DL has just taken one packet when the bucket height is q_1+1, then the switch will be closed. Therefore, the service time of TVB of DL will continue S/C_2, and SL will be on service at least $\frac{q_1}{C_1}S$. The service process should be that SL is always on service during DL does, then $\frac{q_1}{C_1} > \frac{1}{C_2}$ (2). This deduces that $q_1 \geq \frac{C_1}{C_2} \geq \frac{m}{k-m}$. We set q_1 as the minimum value such as: $q_1 = \left\lceil \frac{m}{k-m} \right\rceil$, and it is clear that $q_2>q_1$.

Condition (2) ensures the transmission delay factor of R-(m,k)-firm:

The first case is that the switch of DL is open, and if $b>q_2$, the bust causes the maximum bucket load. Then SL and DL service the workload together until the height decreases to q_1. After, SL service only until the burst is finished. We know that results in the delay of the service for the burst, denoted by:

$$\left(\frac{b-q_1}{C_1+C_2}+\frac{q_1}{C_1}\right)S \qquad (3)$$

in the first case, if $b<q_2$, the height must decrease once the switch is open, so the maximum delay can be denoted by:

$$\left(\frac{q_2-q_1}{C_1+C_2}+\frac{q_1}{C_1}\right)S \qquad (4)$$

The second case is that the height is near to q_2, but the switch is still closed. SL will service alone the workload until it empties the bucket, and this case results in the delay of service, denoted by:

$$\frac{q_2-1}{C_1}S \qquad (5)$$

Maximum delay must be no more than $\triangle$ in any case, so that we conclude as condition (2). Hereto, *R-(m,k)-firm* real-time constraint can be absolutely guaranteed by DLB.

Numeric application

To show the numerical application, audio-CBR streams' parameters are considered, such as CD quality stereo audio stream which produces a high bit rate of 1.4Mbit/s [13]. Such that the flow is bounded by (r,b)=(1.4Mbps, 2kbit). We consider the packet size is S=144byte (it is a normal packet size in audio transmission). As an example, we assume that such a stream is under R-(3,5)-firm constraint and with $\triangle$=20ms as the per-flow deadline.

For example, at a moment, only 1.2Mbps bandwidth is available when the upper stream arrives. Obviously, it is not possible to guarantee the deadlines of all packets.

Then we can configure this DLB as C_1=1.008Mbps, C_2=0.672Mbps, q_1=2, q_2=5. With this configured DLB, *R-(m,k)-firm* is guaranteed $t_{delay} \le \dfrac{q_2 - 1}{C_1} S$ =4.6ms $\le \triangle$=20 (reason of sufficient condition (2)).

However, for guaranteeing all packet deadlines under delay of 20ms, it requires $r+b/t_{delay}$ = 1.5Mbit/s of bandwidth [12]. With DLB, it guarantees *R-(m,k)-firm* constraint in providing 4.6ms delay, but it requires only 1.008Mbit/s of bandwidth. Clearly, in this example, DLB is robust under smaller bandwidth.

4. Performance comparison

As shown above, DLB can provide deterministic *R-(m,k)-firm* guarantee to *(r,b)-bounded* flows. To show the performance of DLB faced to more general flows, we simulated DLB as well as TD and RED in case of a Poisson flow of the rate $\lambda=1$ packet per time unit, with equal packet size. Assume also that the source generating this flow can tolerate up to 33% of loss. To simulate their behaviors during overload period, the server capacity is set to be less than the average arrival rate of the source. For equal comparison, the same queue size and server capacity are given to DLB, TD and RED (parameters are shown in Fig 6).

Figure 6 is a sample sequence of the served and dropped packets, denoted by 1 and 0, respectively. As seen, DLB avoids consecutive losses, while TD and RED do not. Additionally, DLB performs also well in terms of queue length and queuing delay, such as shown in Table 1.

	DLB	RED	TD
Mean queue length	4.7	4.3	9.0
Mean delay (s)	4.8	4.5	8.7
Drop rate	22%	21%	20%

Table 1: Comparison of DLB, RED and DT.

5. Applications to Internet QoS

DLB can be easily implemented in place of the traditional leaky bucket. So this approach can find many applications in the current network to provide the adaptive controllable QoS according to the *R-(m,k)-firm* model.

In *Diffserv*, DLB can be implemented in each priority dropping queues, such as AF1, AF2, AF3 and AF4 in Fig.7. The classifier puts a stream into a certain priority-dropping queue according to its requirement (if it can be admitted). On the other hand, DLB can act as the classifier such that packets passing through SL will be entered in EF queue. The packets passing through the DL will be entered into best effort queue. Then, a stream is split and real-time constraint can be guaranteed by EF queue. In this way, it is clear that EF queue can admit more streams than without dropping.

DLB pattern (q₁=3, q₂ = 6, queue size = 9, C₁ = 0.8, C₂ = 0.4)
0110110110110110110110110110110110110111111111111111111110110110110110110110110111111111111011011
0110110110110110110110111111111111111111111111111110110110110110110110110110110110110110110101101
TD pattern (queue size = 9, C = 0.8)
1011011111000011000111111111111111111111111111110111010100101111111110111111010011111010100101 0
0001111100111111111111111111111111111111111111111110110110011010001111111011111101111111110
RED pattern (wq = 0.2, max_p = 0.34, minth = 3, maxth = 6, queue size = 9, C = 0.8)
11101001000011111111111111111111111111111110101111111100110011110111111111111111111111111111111
111111011001011101111101111111111111111111110100001110001100010101111111111111111111111101010

Fig. 6: Loss Packet patterns of DLB, TD and RED

Fig. 7: DLB implemented in *DiffServ*

In MPLS model, when packets pass through the previous router, they will be marked in the label if the switch of DLB was open. This ensures that in this slice of stream, no more packets will be dropped in the next hop (router). Nevertheless, the dropping can still be done in the slices where the label is not marked.

In one word, DLB is very flexible and could be compatible with almost all the current network schemes.

6. Conclusion

The main contribution of this paper resides in the proposition of one novel real-time constraint model named *Relaxed (m,k)-firm* constraint for better handling graceful QoS degradation than RED and TD in networks during congestion. In order to achieve this graceful QoS degradation guarantee, we proposed a mechanism named Double-leaks bucket (DBL) to deterministically guarantee this real-time constraint.

Numerical examples have shown that DBL can provide *R-(m,k)-firm* with high utilization of resource. We have shown how *R-(m,k)-firm* constraint and DLB can be easily implemented in *Diffserv* and MPLS. In fact, since DLB is developed from the traditional leaky bucket, DLB has a very good perspective, and it is compatible with almost all exiting current networks.

Our ongoing work consists in developing a platform in Linux-based software router to show its practical implementation.

References

[1] El-Gendy, M.A., A. Bose, K.G. Shin, "Evolution of the Internet QoS and support for soft real-time applications", proceedings of the IEEE, Vol.91, No.7, pp1086-1104, July 2003.

[2] Floyd, S., and Jacobson, V., "Random Early Detection Gateways for Congestion Avoidance". In ACM/IEEE Transactions on Networking, 3(1), August 1993.

[3] M. Hamdaoui and P. Ramanathan, "A dynamic priority assignment technique for streams with *(m,,k)-firm* deadlines", IEEE Transactions on Computers, 44(4), pp1443–1451, Dec.1995.

[4] F. Wang and P. Mohapatra, "Using differentiated services to support Internet telephony", Computer communications, Vol.24, Issue18, pp1846-1854, Dec. 2001.

[5] Striegel A., G. Manimaran, "Dynamic Class-Based Queue management for scalable media servers", Journal of systems and software, vol.66, No.2, pp.119-128, May 2003.

[6] Koubâa, A., Song, Y.Q., "Loss-Tolerant QoS using Firm Constraints in Guaranteed Rate Networks", 10th IEEE Real-Time and Embedded Technology and Applications (RTAS'2004), Toronto (Canada) 25-28 May 2004.

[7] G. Quan and X. Hu, "Enhanced Fixed-priority Scheduling with (m, k)-firm Guarantee", Proc. Of 21st IEEE Real-Time Systems Symposium, pp.79-88, Orlando, Florida, (USA), November 27-30, 2000.

[8] Mok, A.K. and W. R. Wang, "Window-Constrained Real-Time Periodic Task Scheduling", 22nd IEEE Real-Time Systems Symposium (RTSS'01), pp15-24, London, England, December 03 - 06, 2001.

[9] J. LI. Sufficient Condition for Guaranteeing *(m,k)-firm* Real-Time Requirement Under NP-DBP-EDF Scheduling. Technical report No. A03-R-452, Stage de DEA, LORIA, Jun, 2003.

[10] Richard West, Yuting Zhang, Karsten Schwan and Christian Poellabauer, "Dynamic Window-Constrained Scheduling of Real-Time Streams in Media Servers", IEEE Transactions on Computers, Volume 53, Number 6, pp. 744-759, June 2004

[11] Yuting Zhang, Richard West and Xin Qi, "A Virtual Deadline Scheduler for Window-Constrained Service Guarantees", in Proceedings of the 25th IEEE Real-Time Systems Symposium (RTSS), December 2004.

[12] Le Boudec, J.Y. and P. Thiran, Network Calculus: A Theory of Deterministic Queueing Systems For The Internet, Online Version of the Book of Springer Verlag – LNCS 2050, July 2002.

[13] http://ntrg.cs.tcd.ie/undergrad/4ba2.01/group10/bandwidth.html