The impact of buffer finiteness on the loss rate in a priority queueing system

Evaluation of Surplus Round Robin Scheduling Algorithm

Dessislava Nikolova and Chris Blondia

University of Antwerp, Department of Mathematics and Computer Science
Performance Analysis of Telecommunication Systems Research Group
Middelheimlaan 1, B-2020 Antwerpen, Belgium
and
Interdisciplinary institute for BroadBand Technology (IBBT)
{dessislava.nikolova, chris.blondia}@ua.ac.be

Keywords: Packet scheduling algorithms, Surplus round robin, Deficit round robin, Latency, Fairness.

Abstract

In multi-service packet networks the packet scheduling algorithm plays a key role in delivering guaranteed service to different flows. In this article we evaluate surplus round robin (SRR) scheduling algorithm. We apply the Latency-Rate (LR) servers theory in order to obtain bounds on the latency and delay provided by the algorithm. We compare the performance characteristics of SRR with the popular deficit round robin (DRR) algorithm to conclude that SRR can provide better performance than DRR.

1 INTRODUCTION

In packet switched networks traffic flows form different types of applications and users are contending for the network resources. In order to guarantee the different QoS requirements of the the network switches should support per flow queues. A packet scheduling algorithm arbitrates the transmissions of the packets from the queues on a link. Important requirements for a scheduling algorithm are its ability to provide end-to-end delay guarantee, fairness and its low complexity.

In [1] and [2] a general model, called $Latency - Rate$ (LR) servers, was developed for the analysis of traffic scheduling algorithms. The model introduce the notion of latency, which bounds the time that a flow has to wait until it begins receiving service at its guaranteed rate. From the latency one can obtain upper bounds on the delay, backlog in the queue and burtsiness of the output traffic.

A broad number of scheduling algorithms are available in the literature. Some of them like WFQ[3], VC[4], $STFQ$[5], WF^2Q+[6], generally classified as sorted priority algorithms, maintain a sorted queue. This allows them to serve the sessions regardless of the arrival order of the packets but based on a parameter of the session - the guaranteed bandwidth for example. The algorithms try to simulate the order in which the packets would leave the server if served under ideal general processor sharing (GPS) discipline. Sorted-priority scheduling disciplines generally provide low delay bound on the traffic but have high complexity, which depends on the number of contending flows.

Other schedulers, like DRR[7], SRR[8] (further specified in the context of striping protocol [9]), $Nested\ DRR$[10], ERR[11], $EBRR$[12], use round robin (RR) order for servicing the flows which results in O(1)complexity of the algorithm. One of the most popular algorithm is named Deficit Round Robin (DRR) and is described in [7]. It is also implemented in several high-speed routers usually combined with strict priority queuing [13], [14]. According to the DRR algorithm the flows are selected for service in a round robin order. When a flow becomes backlogged it is added to a list of backlogged flows. To the different flows are assigned allowances in a round they can send, where the allowance, called quantum, is expressed in bytes. The quantums indicate the amount of service in a round a flow should receive. A counter for each flow keeps track of the remaining service for the round. Each time a packet is transmitted the counter for the flow is reduced with the size of the packet. The decision whether or not a packet should be transmitted depends on the packet size. If the transmission of the packet would cause the counter to become negative the service of the flow for the round ends and the scheduler moves on to serve the next flow in the round robin order. The deficit of service in a round, indicated with the value of the counter at the end of the service of a flow, is compensated in the next round.

In [2] the fairness and latency bound were derived for WFQ, VC, STFQ, and DRR. Lower bounds on the latency and fairness of DRR were later derived in [15], [16]. The fairness and delay bounds for ERR, EBRR and Nested DRR can be found in the articles in which these algorithms were described.

In this paper we specify more detailed the Surplus Round Robin (SRR) algorithm previously described in [8] and [9]. We also point out certain implementation issues, which have to be taken into account if SRR is to be applied as fair scheduling mechanism. We evaluate theoretically its performance and derive fairness and latency bound using the LR servers model.

The paper is organized as follows. In the next section we describe and discuss implementation issues of the SRR algorithm. In section 3 we analyze the performance of the SRR and derive bounds on the latency and calculate the fairness by using the LR servers model. In the section 4 we discuss the obtained results and compare them with the bounds available for DRR. In the last section conclusions are drawn.

2 SURPLUS ROUND ROBIN

In this section we describe and further specify the SRR algorithm. Consider a number of flows contending for a link, each flow i having a corresponding queue i where the packets belonging to this flow are stored. To each flow is assigned weight w_i and a quantum Q_i proportional to that weight. The quantum indicates the portion of the resources in a round robin cycle a flow should get. If the minimum quantum, say Q_{min} is chosen not less than the maximum packet length in the network then the algorithm has $O(1)$ complexity. For the quantums we can write

$$Q_i = w_i Q_{min} \tag{1}$$

To each flow i is associated a counter called surplus counter SC_i, which indicates the amount of service the flow should still receive in a round. The flows are serviced in a round robin order. Each round a flow is served once. When a flow is picked for service its SC_i is increased with its quantum. After a packet is sent SC_i is decreased with the size of the packet. Packets are transmitted from the flow as long as SC_i is positive. If in a round a flow is served surplus amount i.e. SC_i has some non-positive value, it is penalized by this amount in the next round, regardless whether the queue became empty or not after the packet was transmitted. The last condition is important because of the unfairness that might arise if a flow is not penalized. Consider the scenario, where the packets of flow i arrive in bursts of $Q_i + SC_i$ bytes each round. When these bytes are transmitted the queue of the flow becomes empty. If the scheduler does n't penalize for the next round the flow will be allowed to send each round above its quantum. Thus the SRR algorithm can not be implemented as a straightforward extension of DRR but it can be realized for example by implementing the algorithm as in [12] with more than one backlogged list, keeping a round counter, and another variable per flow to track the penalty rounds. From implementation point of view the main advantage of the algorithm is that, unlike DRR, the SRR does n't need information about the length of the next packet waiting in the queue connected with the flow to make the scheduling decision.

In the next section we analyze the SRR algorithm and derive latency and fairness bounds.

3 ANALYTICAL RESULTS

Consider N contending flows for a shared link with rate r and a SRR scheduler, which arbitrates the packet transmissions from the flows. Let r_i be the reserved rate for flow i and r_{min} designate the smallest of r_i. The weight assigned to flow i is can be chosen proportional to the reserved rate

$$w_i = \frac{r_i}{r_{min}} \tag{2}$$

Assigned in this way the weights guarantee the reserved rates provided that $\sum_{i=1}^{N} r_i \le r$, which we are going to presume in the following analysis.

A frame f is one round robin cycle amongst the backlogged flows. Thus it can be expressed as

$$f = \sum_{i=1}^{N} Q_i = \sum_{i=1}^{N} w_i Q_{min} \tag{3}$$

Replacing the weights from (2) the frame size can be bounded

$$f = \sum_{i=1}^{N} \frac{r_i}{r_{min}} Q_{min} \le \frac{r}{r_{min}} Q_{min} = F \tag{4}$$

where F is the maximum frame size. Replacing (2) in (1) and rewriting the rhs. of (4) as $\frac{Q_{min}}{r_{min}} = \frac{F}{r}$ we obtain for the quantum for flow i

$$Q_i = \frac{r_i}{r_{min}} Q_{min} = \frac{r_i}{r} F \tag{5}$$

In order to derive latency bound we will need bounds on the surplus counter.

Lemma 3.1. *The surplus counter of a backlogged flow i at the end of a round is bounded by*

$$-(L_{max} - 1) \le SC_i \le 0$$

where L_{max} indicates the maximum packet length.

Proof. According to the SRR definition the scheduler will continue servicing a flow as long as its SC is positive. Thus if a flow is backlogged at the end of service $SC_i \leq 0$ otherwise the service will continue. If after servicing a packet with length L_i the SC becomes negative the server will proceed with serving another flow. For the service counter at the end of the round we can write $SC_i = SC_i^- - L_i$, where SC_i^- is the value of the counter before L_i is served. This value is bounded by $SC_i^- \geq 1$. Thus $SC_i = SC_i^- - L_i \geq -(L_i - 1) \geq -(L_{max} - 1)$, which completes the proof. □

3.1 Latency bound of SRR

To derive the latency we apply the theory of $Latency - Rate$ (LR) servers described in [2]. Basic concept in the LR servers is the notion of flow's busy period. This is defined as the maximum interval of time during which the flow would remain backlogged if served at its reserved rate. A flow is considered backlogged if there is at least one packet from this flow in the system. For a discussion on the difference between backlogged and busy period see [2]. Consider $W_i(\tau, t)$ the amount of service received by flow i in the time interval (τ, t). The latency of a scheduling algorithm S is defined in [2] as the minimum non-negative constant θ_i^S that satisfies

$$W_i(t_0, t) \geq max(0, r_i(t - t_0 - \theta_i^S)) \tag{6}$$

where t_0 is a start of a busy period and t is any time instance within this busy period. In the same reference was shown that if (6) is true for one backlogged period than the value of the latency obtained is an upper bound on the latency of the server θ_i^S. It can be easily determined whether the bound obtained from the backlogged period is a tight bound by giving an example where the offered service is equal to the bound. From the latency of a server one can easily obtain a bound on the maximum delay D_{max} for a leaky-bucket shaped traffic source with parameters (σ_i, r_i). This is given by

$$D_{max} \leq \frac{\sigma_i + \theta_i^S}{r_i} \tag{7}$$

If only latency values at packet departure times are consider a tighter bound on the maximum delay can be obtained. The latency considered only at packet departure times is referred to as SBI and is defined in [6].

Theorem 3.2. *The SRR scheduler belongs to the class of the LR-servers, with an upper bound on the latency θ_i, for flow i given by*

$$\theta_i^{SRR} \leq \frac{F - Q_i + (N-1)(L_{max} - 1)}{r}$$

Proof. To show that the SRR scheduler belongs to the class of LR servers we need to show that the service received by a flow, served by an SRR scheduler, is given by (6). As was discussed earlier the latency can be bounded based on the offered service in any one backlogged period. The service received by a backlogged flow i in one round, say h starting at time t_{h-1} is given by

$$W_i(t_{h-1}, t_h) = Q_i + SC_i(h-1) - SC_i(h) \tag{8}$$

where $SC(k-1)$ is the penalty for the surplus the flow received the previous round. Consider a backlogged period starting in round k_0 in which the flow does n't have to be penalized i.e $SC_i(k_0 - 1) = 0$. An example for such period is the first backlogged period of the flow. Let the start time of round k_0 is t_0. Using equation (8) over k consecutive backlogged rounds after k_0 the work is given by

$$W_i(t_0, t_k) = kQ_i + SC_i(k_0 - 1) - SC_i(k_0 + k) \tag{9}$$

Both expressions are the same as the ones for DRR (see [2]) but as the bounds on the SC are different from the ones on the deficit counter the work done for a backlogged period differs for the two schedulers. For any time t' after flow i received service in round $k_0 + k$ the total service for the backlogged period can be upper bounded by

$$W_i(t_0, t') \leq W_i(t_0, t_k) \leq kQ_i + (L_{max} - 1) \tag{10}$$

In obtaining this bound we have used that there is no penalty in the k_0 round and the left bound on the counter from lemma 3.1.

For any time $t \leq t_k$

$$W_i(t_0, t) \geq W(t_0, t_{k-1}) + W_i(t_{k-1}, t) \geq W(t_0, t_{k-1}) \tag{11}$$

where the last inequality comes from the fact that t can be any time in a round where the flow has n't received any service yet. Using equation (9) to express the service received for $k-1$ rounds and considering the upper bound on the surplus from lemma 3.1 we obtain lower bound on the work in the backlogged period (t_0, t)

$$W_i(t_0, t) \geq (k-1)Q_i \tag{12}$$

The total work done by the server for the time interval (t_0, t) can be written as the sum of the amount of service received from all flows $W_s(t_0, t) = \sum_{j=1}^{N} W_j(t_0, t)$ and can be bound considering (10) and replacing W_i with the minimum value from (12) for the considered period

$$\begin{aligned} W_s(t_0,t) &\leq \sum_{j=1;j\neq i}^{N} W_j(t_0,t_k) + (k-1)Q_i \\ &\leq \sum_{j=1;j\neq i}^{N-1} kQ_j + L_{max} - 1 + kQ_i - Q_i \\ &\leq kF - Q_i + (N-1)(L_{max}-1). \end{aligned} \quad (13)$$

Expressing k and considering that for constant rate server $W_s(t_0,t) = r(t-t_0)$

$$k \geq \frac{r(t-t_0)}{F} + \frac{Q_i - (N-1)(L_{max}-1)}{F} \quad (14)$$

We now replace the value of k in (12) and considering 5, obtain

$$\begin{aligned} &W_i(t_0,t) \geq \\ &\geq \left\{ \frac{(r(t-t_0)}{F} + \frac{Q_i - (N-1)(L_{max}-1)}{F} - 1 \right\} Q_i \\ &\geq r_i(t - t_0 - \frac{-Q_i + (N-1)(L_{max}-1)}{r} - \frac{Q_i}{r_i}) \end{aligned} \quad (15)$$

. Simplifying further the expression we obtain for the service received by a backlogged session i in the time interval (t_0,t)

$$\begin{aligned} &W_i(t_0,t) \geq \\ &\geq max(0, r_i(t - t_0 - \frac{F - Q_i + (N-1)(L_{max}-1)}{r})) \end{aligned} \quad (16)$$

In order to show that this is a tight bound we will present an example. In the case of the SRR algorithm it is a rather obvious one because the upper latency bound coincides with the maximum start up delay. Consider a flow which starts service in a round when there are N-1 other backlogged flows in the system. Before being considered for service the scheduler will service the other N-1 flows i.e. it has to wait N-1 flow to receive service given by (8)

$$\begin{aligned} t - t_0 &\leq \frac{\sum_{j=1}^{N-1} Q_j - SC(k-1) + SC(k)}{r} \\ &\leq \frac{F - Q_i + (N-1)(L_{max}-1)}{r}, \end{aligned} \quad (17)$$

in the last equality used that $F = \sum_{i=1}^{N} Q_i$ and lemma 3.1. With this we concluded the proof. □

3.2 Fairness of SRR

To calculate the fairness of the SRR algorithm we use the modified in [2] definition of the Golestiani's fairness

$$\left| \frac{W_i(\tau,t)}{r_i} - \frac{W_j(\tau,t)}{r_j} \right| \leq \Phi^S, \quad (18)$$

where the interval $(\tau,t]$ is such that the two compared connections i,j are continuously backlogged.

Equation (12) bounds the work received in an interval (t_0,t) where t_0 is the start of a round where the flow is not penalized. However, in order to calculate the fairness we need a bound on the service for any interval (τ,t). To obtain it we consider the lower bound on the SC form lemma 3.1 in equation (8) giving

$$W_i(\tau,t) \geq (k-1)Q_i - (L_{max}-1). \quad (19)$$

If two connections are continuously backlogged over a certain interval then their round robin order relative to each other under the SRR scheduling disciplines remains the same over the interval. The moment τ is at some time instant in round k_0 before flow i is considered for service. Flow i receives service in round k_0 before flow j and in round k the moment t is taken after flow i receives service and before flow j is start being served. Replacing the minimum bound from equation (19) and the maximum bound from equation (10) on the service received from two backlogged flows in equation (18) one obtains

$$\left| \frac{W_i(\tau,t)}{r_i} - \frac{W_j(\tau,t)}{r_j} \right| \leq \frac{Q_j}{r_j} + \frac{L_{max}-1}{r_i} + \frac{L_{max}-1}{r_j}. \quad (20)$$

Taking into account (5) we can write for the fairness

$$\Phi^S = \frac{F}{r} + \frac{L_{max}-1}{r_i} + \frac{L_{max}-1}{r_j}. \quad (21)$$

4 DISCUSSION

Several latency bounds have been reported in the literature for DRR [2], [15], [16]. The one reported in [15] is the lowest bound and is

$$\theta_i^{DRR} = \frac{(W-w_i)Q_{min} + (N-2)(m-1)}{r} + \frac{m-1}{r_i}. \quad (22)$$

In this equation $(W-w_i)Q_{min} = F - Q_i$ and the authors in [15] use m to denote any packet size, which in the considered model is bounded by L_{max}.

The SRR latency bound according to theorem 3.2 is lower than the lowest latency bound reported for the DRR algorithm with factor $\frac{L_i-1}{r_i}$. The difference is inverse proportional to the reserved rate thus it will be more significant for flows with small reserved rates. On figure 1 we compare the two latencies based on a practical example from broadband access cable network. Consider an output link with rate $r = 40Mb/s$ shared by 100 flows. The minimum reserved rate r_{min} is 8 Kb/s and the maximum packet size is 1518 bytes. Let the

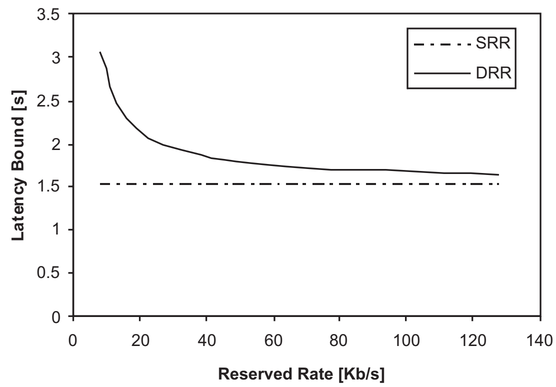

Figure 1. Comparison of the latency bounds of SRR and DRR.

minimum quantum equal the maximum packet size then the length of frame expressed from 5 as $F = \frac{Q_{min}}{r_{min}} = 1.518s$. The figure demonstrates that SRR has lower latency bound than the one for DRR.

The fairness value for SRR given by equation (21) is the same as the one reported in [16].

5 CONCLUSIONS

In this article we analyzed the SRR scheduling algorithm. By using the LR servers model we derived latency bound and demonstrated that it is lower than the one obtained for the popular DRR. The fairness was calculated and the results demonstrated that it achieves the same fairness as DRR.

REFERENCES

[1] D. Stiliadis and A. Varma, 1998, *"Latency-Rate Servers: A General Model for Analysis of Traffic Scheduling Algorithms"*, IEEE/ACM Transactions on networking, 6, (October).

[2] D. Stiliadis, 1996, *"Traffic Scheduling in Packet-Switched Networks: Analysis, Design, and Implementation"*, Ph.D. Dissertation, University of California at Santa Cruz, USA.

[3] A. Demers, S. Keshav, and S. Shenkar, 1990, *"Analysis and simulation of a fair queuing algorithm"*, Internet. Res. and Exper., vol 1.

[4] Lixia Zhang, 1990,*"Virtual Clock: A New Traffic Control Algorithm for Packet Switching Networks,"* in SIGCOMM Symposium on Communications Architectures and Protocols, (Philadelphia, Pennsylvania, September), pp. 19–29, .

[5] P. Goyal, H. Vin, and H. Cheng 1997, *"Start-time Fair Queuing: A Scheduling Algorithm for Integrated Services Packet Switching Networks"*, IEEE/ACM Transactions on Networking, 5, no.5, (June).

[6] J.C.R. Bennett and H. Zhang, 1997, *"Hierarchical Packet Fair Queueing Algorithms"*, IEEE/ACM Transactions on networking, 5, no.5, (October).

[7] Shreedhar M., and Varghese George , 1995, *"Efficient Fair Queuing using Deficit Round Robin"*, in Proc. SIGCOMM'95.

[8] S. Floyd, *"Notes on CBQ and Guaranteed Service"*, Unpublished Notes, http://www.icir.org/floyd/cbq.html.

[9] H. Adisshu, G. Parulkar, and G. Varhese, 1996, *"A Relaible and Scalable Striping Protocol"*, in Proc. ACM SIGCOMM, (August).

[10] S. Kanhere and H. Sethu, 2001, *"Fair, Efficient and Low Latency Packet Scheduling Using Nested Deficit Round Robin"*, in Proceedings of the IEEE Workshop on High Performance Switching and Routing, (Dallas, USA).

[11] S. Kanhere, H. Sethu, and Alpa Parekh, 2002, *"Fair and Efficient Packet Scheduling Using Elastic Round Robin"*, IEEE Transactions on Parallel and Distributed Systems, 13, no.3, (March).

[12] L. Lenzini, E Mingozzi, and G. Stea, 2004, *"Eligibility-Based Round Robin for Fair and Efficient Packet Scheduling in Wormhole Switching Networks"*, IEEE Transactions on Parallel and Distributed Systems, 15, no.3, (March).

[13] www.cisco.com

[14] www.juniper.net

[15] S. Kanhere and H. Sethu, 2002, *"On the latency Bound of Deficit Round Robin"*, in Proceedings of the Internation Conference on Computer Communications and Networks, (Miami, Florida, USA, October).

[16] L. Lenzini, E Mingozzi, and G. Stea, 2002, *"Aliquem: a Novel DRR Implementation to Achieve Better Latency and Fairness at O(1) Complexity"*, in Proc. 10th Int. Workshop on Quality of Service (IWQoS), (Miami Beach, FL, May): 77-86.