TUNABLE LEAST SERVED FIRST

A New Scheduling Algorithm with Tunable Fairness

Pablo Serrano, David Larrabeiti, and Ángel León
Universidad Carlos III de Madrid
Departamento de Ingeniería Telemática
Av. Universidad 30, E-28911 Leganés, Madrid, Spain
{pablo,dlarra,aleon}@it.uc3m.es

Abstract At high transmission speeds, complexity of implementation for fair queuing disciplines can impose a bottleneck to the overall system performance. Available scheduling algorithms set a fixed trade off between fairness and complexity, fairer systems involving more operations per packet and vice versa. In this paper first a new fair queuing scheme is proposed, with almost the same fairness and complexity propierties achieved so far by most used algorithms. Later on a tunable parameter is introduced, which allows the modification of the above mentioned trade off between fairness and complexity depending on working conditions and thus enlarging the field of application of the scheduler.

Keywords: Fair scheduling, tunable fairness.

Introduction

In past years, round-robin-like disciplines were the most established schemes of scheduling. This was mainly because only one class of traffic (computer data) was supported, with no quality of service (QoS) guarantees, and because packet transmission time was large compared to round trip time (and thus reactive congestion control could be used to control traffic sources). The emergence of high-speed data networks has changed these conditions, leading researchers to investigate on new directions for traffic control. First-in-first-out (FIFO) queuing disciplines have been proved to be no good nor to provide QoS guarantees nor fairness in the event of congestion. Three are the main properties required for a queuing discipline ([Bensaou et al., 2001]):

- Simplicity: the processing overhead must be orders of magnitude smaller than the average packet transmission time.
- Robustness (isolation): well-behaved traffic flows cannot be hurt by misbehaving flows.

- Fairness: all bandwidth should be given to active traffic flows proportional to their weight.

Some examples of unfair queuing disciplines are: FIFO, where a session can increase its share of service by just presenting more demand, or Round robin (RR) ([Hahne, 1986]), where a flow with higher mean packet size will obtain more bandwidth at the expense of another flow with equal packet injection rate, but less sized packets.

Most Fair Queuing (FQ) algorithms aim to approach the fairness provided by the generalized processor sharing (GPS) algorithm [Parekh and Gallager, 1993]. GPS, also called fluid fair queuing (FFQ), is a theoretical algorithm based on the assumption that traffic is infinitesimally divisible, and hence can be served by infinitesimally small quanta (leading to the ideal situation where bits from different packet flows are transmitted concurrently). But in real switched networks packets are not divisible, and a flow seizes all channel during the transmission time (so no other flow is given service), which prevents the implementation of an absolute fairness system ([Golestani, 1994]).

Zhang's *virtual clock* scheme [Zhang, 1990] can be considered pioneering, due to the introduction of a metric (a virtual clock), an effective tool for both formulating fairness and representing the progress of work in the queuing system. But virtual clock-based algorithms (WFQ [Demers et al., 1990], W2FQ [Bennett and Zhang, 1996]) involves considerable computational complexity. Most of the recent research in reducing the processing requirement of the scheduler has concentrated on modifying the basic WFQ paradigm. One exception is the DRR [Shreedhar and Varghese, 1995] algorithm and its variants, which achieve lower complexity at the expense of lower fairness. In [Chaskar and Madhow, 2003] modifications to the weighted round-robin approach are discussed, with performance characteristics similar to those of W2FQ but with lower complexity of implementation (for fixed packet sizes).

This paper proposes and analyses a new paradigm of fair squeduling, Least Served First (LSF), and extends it with a parameter which tunes the trade off between complexity per packet and fairness (T-LSF). LSF serves at any instant the flow which has received less service since it arrived to the scheduler, while T-LSF deals with *groups* of flows that have received *similar* service. This tunable fairness (and complexity) enlarges the field of application of a scheduler: instead of imposing the design criterium a priori (based on the expected number of flows, line speeds, ...), it is possible to tune a parameter a posteriori based on *real* working conditions. On maximum fairness configuration it possesses similar characteristics than those of SCFQ [Golestani, 1994].

The remainder of the paper is organized as follows. Section 1 deals with the network model, giving definitions about fairness and the algorithm. Section 2 presents the basic version of LSF, without tunable fairness/complexity, and analyses its performance. On Section 2.4 we present T-LSF, with the tunable parameter N that reduces (increases) complexity (unfairness). Finally, on Section 3 we draw our conclusions and point out future lines of research.

1. Preliminaries

The network node we consider is modeled as a multiplexer fed by a superposition of M traffic flows, sharing a link of capacity C. Each flow i, $i = 1, \ldots, M$ is associated a counter W_i and a bandwidth share r_i, with $\sum_{i=1}^{M} r_i = C$. A flow i is said to be backlogged at time t if a packet from flow i is being served or it is waiting to be served at that time. A flow i is backlogged during (t_1, t_2) if it is backlogged during all the interval. $B(t_1, t_2)$ is the set of sessions which are backlogged during the entire interval (t_1, t_2). According to a *sorted* flow list (based on the values of the W_i), the LSF algorithm selects the next packet to be transmitted and updates the value of the counters. $S_i(t_1, t_2)$ is the amount of traffic served during (t_1, t_2) to flow i, while $W_i(t_1, t_2)$ is the difference of the values for the counter W_i (i.e. $W_i(t_1, t_2) = W_i(t_2) - W_i(t_2)$). Throughout the paper, and until explicitly claimed, t_n is the time when a packet n has finished its transmission.

The notion of fairness in this paper is defined according to the criterium of proportional rate sharing: if a traffic flow i is not active, its bandwidth share r_i should be allocated to the other active flows in a fashion proportional to their share. Under the fluid flow assumption, a scheduling algorithm is said to be fair if and only if

$$\forall\, t_1, t_2\ \forall i, j : i, j \in B(t_1, t_2), \left| \frac{S_i(t_1, t_2)}{r_i} - \frac{S_j(t_1, t_2)}{r_j} \right| = 0$$

$S_i(t_1, t_2)/r_i$ is defined as the *normalized service received* by flow i. In a non-fluid network model, where traffic flows are served by a non negligible quantum of variable size, the aim is to give a bound for the above substraction:

$$\left| \frac{S_i(t_1, t_2)}{r_i} - \frac{S_j(t_1, t_2)}{r_j} \right| \leq FI$$

FI is defined as the *fairness index* (also called *proportional fairness index* [Chaskar and Madhow, 2003] or *relative fairness* [Zhou and Sethu, 2002]) of the scheduling discipline. The smaller the FI, the fairer the scheduling algorithm. On the other hand, the *absolute fairness* is defined as the difference on normalized service received between a flow served by the queuing discipline under study, and the same flow in a GPS environment. Although this is a better measurement of the performance of the algorithm, in situations with a high number of flows both fairness values remain close to each other ([Zhou and Sethu, 2002]), and thus we are able to keep focus on FI.

The main features of most common available scheduling algorithms are summarized on Table 1, and compared to LSF and T-LSF, i, j being any two

Table 1. Fairness Bounds and Computational Complexity per Packet

Algorithm	Fairness Bound	Complexity
SCFQ	$L_i^{MAX}/r_i + L_j^{MAX}/r_j$	$O(log(M))$
W2FQ	$L_i^{MAX}/r_i + L_j^{MAX}/r_j$	$O(log(n))$
DRR	$L_i^{MAX} + L_i^{MAX}/r_i + L_j^{MAX}/r_j$	$O(1)$
LSF	$2 \cdot max\{L_i^{MAX}/r_i\}$	$O(log(M))$
T-LSF	$2 \cdot max\{L_i^{MAX}/r_i\} \cdot \; 1 + \frac{1}{N-1}$	$O(log(N))$

flows, M the number of flows and n the number of packets in the system, and N the configurable parameter for T-LSF, $N \in \{2, 3, \ldots, M\}$ (although usually $N << M$).

2. Least Served First

2.1 General LSF Algorithm

Consider a system with M flows. A flow i, $i \in \{1, \ldots, M\}$ is associated a bandwidth share r_i and a counter W_i. This counter is used to store an estimation of the normalized service received by flow i. In order to maximize fairness, the algorithm aims to minimize the maximum difference in service received by any two flows: the flow with minimum W_i is always the next flow to be served.

LSF Algorithm

- Initialization.

$$W_i := 0, \; i = 1, \ldots, M$$

- Operation.

  1 Transmit the head-of-line (HOL) packet of flow j with minimum W_i and size $L_{HOL(j)}$, i.e.
  $j := arg\ min^1\{W_i\}$

  2 Once departed packet from flow j update W_j,

$$W_j := W_j + L_{HOL(j)}/r_j$$

  3 Goto (1).

- Flow arrival. When a flow k is backlogged, it is initialized to the maximum of $\{W_i\}$,

$$W_k := max\,\{W_i\}$$

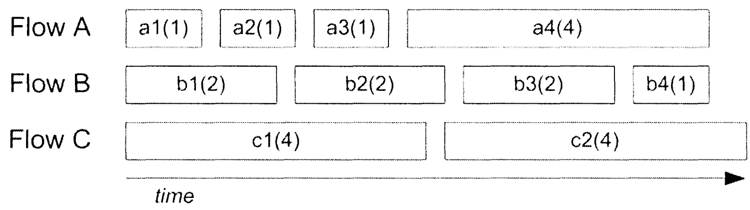

Figure 1. Example with three flows

Table 2. Example for LSF

Packet Served		a1	b1	c1	a2	a3	b2	a4
W_A	0	1	1	1	2	3	3	7
W_B	0	0	2	2	2	2	4	4
W_C	0	0	0	4	4	4	4	4

In order to illustrate the behaviour of LSF with an example, suppose a system with three flows, each of them backlogged with packets of different sizes. This situation is represented on Fig. 1 (each packet is given an identifier, with its size in parenthesis). All flows start with $W_i = 0$, and $r_i = r_j = 1, \forall i, j$. The scheduler algorithm will serve packets as shown on Table 2[2]. Due to the behaviour of the algorithm, the maximum difference between any two normalized service counters ($|W_i - W_j$) at any packet departure is equal to the maximum packet length in the system, which is the base for the analysis of the fairness of LSF.

2.2 Fairness Analysis

In this section we show that LSF achieves a fairness bound close to the one obtained in [Demers et al., 1990; Bennett and Zhang, 1996; Golestani, 1994]. This is stated on the following Theorem, proved through a sequence of four lemmas.

THEOREM 1 (FAIRNESS INDEX OF LSF) *For any two flows* $i, j \in B(t_1, t_2)$ *and* $p \in \{1, \ldots, M\}$

$$\left| \frac{S_i(t_1, t_2)}{r_i} - \frac{S_j(t_1, t_2)}{r_j} \right| \leq 2 \cdot max_p \left(\frac{L_p^{MAX}}{r_p} \right)$$

For ease of notation, we are going to define the quantity K_p as:

$$K_p = max_p \left(\frac{L_p^{MAX}}{r_p} \right), p \in \{1, \ldots, M\}$$

In order to demonstrate the Theorem, we will first bound the difference between W_i and W_j (at two different t_n) for any two flows i and j (by Lemma 2, Lemma 3 and Corollary 4). Then we will extend it for the normalized service received, S_i/r_i (via Corollary 5). At last, we will allow the comparaison between any two time instants -not only departure times- via Lemma 6, concluding the proof.

LEMMA 2 *For any two flows $i, j \in B(t_n, t_m)$, if $W_i(t_n) = W_j(t_n)$ then*

$$|W_i(t_n, t_m) - W_j(t_n, t_m)| \leq K_p$$

Proof: *Because $W_i(t_n) = W_j(t_n)$, we have $|W_i(t_n, t_m) - W_j(t_n, t_m)| = |(W_i(t_m) - W_j(t_m)) - (W_i(t_n) - W_j(t_n))| = |W_i(t_m) - W_j(t_m)|$. With $p \in \{1, \ldots, M\}$ this difference is bounded by:*

$$|W_i(t_m) - W_j(t_m)| \leq \left| max \{W_p\}_{t_m} - min \{W_p\}_{t_m} \right|$$

With $max\{W_p\}_{t_m}$ ($min\{W_p\}_{t_m}$) being the maximum (minimum) of all active counters at t_m. Because $m \geq n$, and by writing $m = n + k$, we can proceed by induction on k:

- $k = 1$:
  $|max\{W_i\}_{t_{n+1}} - min\{W_i\}_{t_{n+1}}| \leq K_p$
  The demonstration is trivial: $W_i(t_n) = W_j(t_n)$, and in t_{n+1} only one packet has departed, thus the difference in t_{n+1} is bounded by the maximum of all possible increments (which is K_p).
- $k \Rightarrow k + 1$:
  for ease of notation (and without loss of generality), we can assume $n = 0$, so $t_{n+k+1} = t_{k+1}$. Then we have $|max\{W_i\}(t_k) - min\{W_i\}(t_k)| \leq K_p$, and we are going to prove it for $k + 1$. We need to consider all possible cases:
  - $min\{W_i\}(t_k) = min\{W_i\}(t_{k+1})$ *and* $max\{W_i\}(t_{k+1}) \geq max\{W_i\}(t_k)$
    This only happens if $max\{W_i\}(t_k) = min\{W_i\}(t_k)$, an thus we are in the case $k = 1$.
  - $max\{W_i\}(t_{k+1}) = max\{W_i\}(t_k)$ *and* $min\{W_i\}(t_{k+1}) \geq min\{W_i\}(t_{k+1})^3$.
    In this case, $|max\{W_i\}(t_{k+1}) - min\{W_i\}(t_{k+1})| = |max\{W_i\}(t_k) - min\{W_i\}(t_{k+1})| \leq |max\{W_i\}(t_k) - min\{W_i\}(t_k)| \leq K_p$.
  - $max\{W_i\}(t_{k+1}) \geq max\{W_i\}(t_k)$ *and* $min\{W_i\}(t_{k+1}) \geq min\{W_i\}(t_k)$
    The only possible case is $max\{W_i\}(t_{k+1}) = min\{W_i\}(t_k) + \frac{L_k}{r_k}$, L_k being the size of the packet transmitted at t_k and r_k the flow associated with it. Then $|max\{W_i\}(t_{k+1}) - min\{W_i\}(t_{k+1})| \leq |max\{W_i\}(t_{k+1}) - min\{W_i\}(t_k)| = |min\{W_i\}(t_k) + \frac{L_k}{r_k} - min\{W_i\}(t_k)| \leq K_p$

LEMMA 3 *For any two flows $i, j \in B(t_n, t_m)$, if $|W_i(t_n) - W_j(t_n)| \leq K_p$ the following inequality holds:*

$$|W_i(t_n, t_m) - W_j(t_n, t_m)| \leq 2 \cdot K_p$$

Proof: $|W_i(t_n, t_m) - W_j(t_n, t_m)| = |(W_i(t_m) - W_j(t_m)) - (W_i(t_n) - W_j(t_n))|$. *Because* $|A - B| \leq |A| + |B|$, *we can bound* $|W_i(t_n, t_m) - W_j(t_n, t_m)| \leq |(W_i(t_m) - W_j(t_m))| + |(W_i(t_n) - W_j(t_n))|$. *The first term is bounded by* K_p *by Lemma 2, and the second term bounded by hypothesis.*

COROLLARY 4 *For any two flows* $i, j \in B(t_n, t_m)$,

$$|W_i(t_n, t_m) - W_j(t_n, t_m)| \leq 2 \cdot K_p$$

Proof: *On* t_n, *there are two possibilities for the relation between* W_i *and* W_j: *either* $W_i(t_n) = W_j(t_n)$, *or* $W_i(t_n) \neq W_j(t_n)$. *The first case is bounded via Lemma 2 by* K_p. *The second case is bounded via Lemma 3: in LSF any flow* i *who begins backlogged its counter* W_i *is imposed to initialize from the maximum of active counters. By Lemma 2, the maximum difference between any two active counters is* K_p, *which is the hypothesis of Lemma 3.*

COROLLARY 5 *For any two flows* $i, j \in B(t_n, t_m)$,

$$\left| \frac{S_i(t_n, t_m)}{r_i} - \frac{S_j(t_n, t_m)}{r_j} \right| \leq 2 \cdot K_p$$

Proof: *First we are going to define the predicate* $isMin$

$$isMin(W_l, \{W_p\}) \begin{cases} 1, & l = arg\ min\{W_p\} \\ 0, & l \neq arg\ min\{W_p\} \end{cases}$$

Then we can $W_i(t_m)$ *as*

$$W_i(t_m) = W_i(t_n) + \sum_{k=1}^{m} (isMin(W_i, \{W_p\}) \times \frac{L_i^k}{r_i}$$

When the predicate $isMin$ *equals 1 is when packet* i *is the next to be served; thus we have*

$$W_i(t_m) - W_i(t_n) = \sum_{k=1}^{m} (isMin(W_i, \{W_p\}) \times \frac{L_i^k}{r_i} \triangleq \frac{S_i(t_n, t_m)}{r_i}$$

And then the bound of Corollary 4 on $|W_i(t_n, t_m) - W_j(t_n, t_m)|$ *applies directly to* $|S_i(t_n, t_m)/r_i - S_j(t_n, t_m)/r_j|$

LEMMA 6 *For any two flows* $i, j \in B(t_1, t_2)$ *(*t_1, t_2 *being any two instants of time, and not just packet departure instants), the following inequality holds:*

$$\left| \frac{S_i(t_1, t_2)}{r_i} - \frac{S_j(t_1, t_2)}{r_j} \right| \leq 2 \cdot K_p$$

Proof: *For a* continuous t_k, *we define* t_{k+} *as the instant when the next packet leaves the system after* t_k, *and* t_{k-} *when the previous packet has left the system. Thus we have either* $i, j \in B(t_{1-}, t_{2+})$ *or* $i, j \notin B(t_{1-}, t_{2+})$.

1 $i, j \in B(t_{1-}, t_{2+})$. *Because the difference is bounded for* discretized t_n, *it is bounded for the intervals* (t_{1-}, t_{2+}), (t_{1-}, t_{2-}), (t_{1+}, t_{2+}) *and* (t_{1+}, t_{2-}). *Depending on whether* i *of* j *transmit, and on relative values of* S_i *and* S_j, *it is easy to bound the difference between them (using the* pinching *or* sandwich theorem *[de Burgos, 1995]). The detailed demonstration for this case is given on the Appendix.*

2 $i, j \notin B(t_{1-}, t_{2+})$. *We have to consider* t_1 *and* t_2 *nearness.*

- t_2. *If* i $(j) \notin B(t_2, t_{2+})$, *then no packet from* i (j) *left the system at* t_{2+} *and thus* $S_{i(j)}(t_1, t_2) = S_{i(j)}(t_1, t_{2-})$.
- t_1. *If both* $i, j \notin B(t_{1-}, t_1)$, *none of them receives service at* t_1 *and* $S_{i,j}(t_1, t_2) = S_{i,j}(t_{1-}, t_2)$.
  If flow i *was receiving service on* t_1, *flow* j *will behave as any other flow* k, $k \in B(t_{1-}, t_2)$, *which received service just before* t_{1-}, *and whose difference on service is bounded. If no other flow* k *is available, flow* j *will initialize with* $W_j(t_1) = W_i(t_{1-})$ *and we could extend* $S_j(t_1, t_2)$ *to* $S_j(t_{1-}, t_2)$ *without any obstacles.*

2.3 Complexity

The initializing/flow arrival phase does not involve any significant number of operations. Packet departure, on the other hand, besides a fixed number of operations (a multiplication and an addition) requires the management of a sorted list. This list does not need to be sorted at each packet departure, because only the served flow may change its placement. Thus the required complexity for LSF is $O(log(M))$, M being the number of flows at the system. All possible improvements discussed in [Bensaou et al., 2001] (independence of counters, flow insertion while a packet is served) are also applicable.

2.4 Tunable LSF

Tunable LSF Algorithm For N batches,

- Initialization. Initialize counters and insert all flows into the first batch,

$$W_i := 0,\ batch_1 \leftarrow i, i = 1, \ldots, M$$

- Operation.

  1 Serve a flow j on the first batch (with any scheduling discipline, e.g. round robin)

Table 3. Example for T-LSF, N=2, PQ

Packet Served		a1	a2	a3	a4	b1	b2	c1	*shift*
W_A	0	1	2	3	*7*	*7*	*7*	*7*	3
W_B	0	0	0	0	0	2	*4*	*4*	0
W_C	0	0	0	0	0	0	0	*4*	0

2 Once departed packet from flow j, proceed to update W_j,

$$W_j = W_j + L_j/r_j$$

3 Assign j to a batch k, $k = 1 \ldots N$,

$$(k-1) \cdot \frac{K_p}{N-1} \leq W_i < k \cdot \frac{K_p}{N-1} \Rightarrow batch_k \leftarrow i$$

4 Goto (1) until $batch_1$ becomes empty ($i \notin batch_1, \forall i, i = 1, .., M$).

5 Proceed to *normalize* the $\{W_i\}$,

$$W_i := W_i - \frac{K_p}{N-1}, i = 1, \ldots, M$$

6 Reassign flows to batches (shift from $batch_n$ to $batch_{n-1}$, $n = 1, \ldots, N$)

7 Goto (1)

- Flow arrival. When a new flow k arrives to the system, its counter is initialized to the maximum value of W_i, and it is placed on $batch_N$.

$$W_k := max\,\{W_i\}\ ,\ batch_N \leftarrow k$$

An example of T-LSF operation with the situation of Fig. 1 with $N = 2$ it is shown on Table 3. A "priority queuing" discipline is considered inside the batches. When a counter W_i exceeds the maximum packet size ($K_p = 4$) the flow is moved to the second batch (packets $a4$, $b2$ and $c1$ for each flow). After first batch is emptied (departure of $c1$), all counters are shifted K_p. For this case, the maximum difference between any two counters is less than twice K_p.

2.5 Fairness Analysis of T-LSF

LSF bounds the maximum difference between any two counters W_i and W_j by K_p. From this quantity is obtained the fairness index FI, twice this value. Tunable LSF bounds that difference by twice the value. Following the same steps of Section 2.2, the fairness index (which is a function of N, the number of batches) is

$$FI(N) = 2 \cdot K_p \cdot \left(1 + \frac{1}{N-1}\right)$$

2.6 Complexity

Complexity of LSF is related to the number of elements of a list. By modifying this number, more unfairness is allowed in order to decrease computational burden. This way, in situations where $M >> 1$ and computational burden imposes an appreciable bottleneck, choosing a $N << M$ will decrement the complexity at the expense of losing fairness. It should be noted that with $M = N$, T-LSF performs *worse* than LSF, because the comparisons are made with thresholds that need not to coincide with the counter values $\{W_i\}$.

3. Conclusions and Future Work

In this paper we proposed a new family of fair scheduling algorithms, with the main newness of a tunable trade-off between fairness and complexity. No fixed size nor any other additional hypothesis is assumed. For the most complex case the fairness is close to the one obtained by similar complexity algorithms. The flexibility of our algorithm allows the establishment of a trade off between complexity and fairness once real working conditions are known. We believe this is a new path of research in fair queuing. On future work we will continue analyzing the features of T-LSF, providing bounds for delay. We will also provide a full comparison between it and the other algorithms, both by theoretical analysis and simulation.

Acknowledgments

This work has been partly supported by the European Union under the e-Photon/ONe Project (FP6-001933) and by the Spanish Research Action CICYT CAPITAL (MEC, TEC2004-05622-C04-03/TCM). We also thank the reviewers of this paper for their valuable comments.

Appendix: Proof of Part (1) of Lemma 6

Taking into account that $i, j \in B(t_{1-}, t_{2+})$, and the difference is bounded for any pair of *discretized* t_k. We consider all possible cases (for ease of notation, we omit the $r_{i,j}$ in the demonstration):

- Neither i nor j transmitted at t_{1-} or t_{2-}. So $S_{i,j}(t_1, t_2) = S_{i,j}(t_{1-}, t_{2+})$ which is bounded by Lemma 6.
- Only flow i transmits at t_{1-} but not at t_{2-}. Then $S_i(t_{1+}, t_2) < S_i(t_1, t_2) < S_i(t_{1-}, t_2)$. Because $S_j(t_{1+}, t_2) = S_j(t_1, t_2) = S_j(t_{1-}, t_2)$, we have $|S_i(t_{1-}, t_2) - S_j(t_1, t_2)| \leq$

K_p and $|S_i(t_{1+}, t_2) - S_j(t_1, t_2)| \leq K_p$, thus we conclude $|S_i(t_1, t_2) - S_j(t_1, t_2)| \leq K_p$.

- Only flow i transmits at t_{2-} but not at t_1. The demonstration is analogous to the previous case.
- Flow i transmits at both t_{1-} and t_{2-}. The demonstration is again analogous, starting with $S_i(t_{1+}, t_{2-}) < S_i(t_1, t_2) < S_i(t_{1-}, t_{2+})$.
- Flow i transmits at t_{1-} and flow j transmits t_{2-}. We consider two cases:
  - Suppose $S_i(t_1, t_2) > S_j(t_1, t_2)$. Then $S_i(t_1, t_2) - S_j(t_1, t_2) < S_i(t_1, t_{2-}) - S_j(t_1, t_{2-}) < S_i(t_{1-}, t_{2-}) - S_j(t_{1-}, t_{2-}) < K_p$.
  - If $S_j(t_1, t_2) > S_i(t_1, t_2)$. Then $S_j(t_1, t_2) - S_i(t_1, t_2)$ $< S_j(t_{1+}, t_2) - S_i(t_{1+}, t_2) < S_j(t_{1+}, t_{2+}) - S_i(t_{1+}, t_{2+}) < K_p$.

Notes

1. throughout the paper and in order to ease the analysis, we assume a minimum function with a proper tie-breaker
2. in case of ties, the priority is $A > B > C$
3. $min\{W_i\}(t_{k+1}) < min\{W_i\}(t_k)$ is not possible

References

Bennett, J. C. R. and Zhang, H. (1996). Wf2q: Worst-case fair weighted fair queueing. In *In Proc. IEEE INFOCOM 96, San Francisco, CA, Mar. 1996.*

Bensaou, B., Tsang, D.H.K., and Chan, King Tung (2001). Credit-based fair queueing (cbfq): a simple service-scheduling algorithm for packet-switched networks. In *IEEE/ACM Trans. Netw., Volume: 9 , Issue: 5 , Oct. 2001 Pages:591 - 604.*

Chaskar, Hemant M. and Madhow, Upamanyu (2003). Fair scheduling with tunable latency: a round-robin approach. *IEEE/ACM Trans. Netw.*, 11(4):592–601.

de Burgos, Juan (1995). *Calculo infinitesimal de una variable*. McGraw-Hill, Madrid.

Demers, A., Keshav, S., and Shenker, S. (1990). Analysis and simulation of a fair queueing algorithm. In *Journal of Internetworking Research and Experience, pages 3-26, October 1990. Also in Proceedings of ACM SIGCOMM89, pp 3-12.*

Golestani, S. (1994). A self-clocked fair queueing scheme for broadband applications. In *Proceedings of IEEE INFOCOM 94, pages 636-646, Toronto, CA, June 1994.*

Hahne, E. (1986). Round robin scheduling for fair flow control. In *Ph.D. thesis, Dept. Elect. Eng. And Comput. Sci., M.I.T., Dec. 1986.*

Parekh, Abhay K. and Gallager, Robert G. (1993). A generalized processor sharing approach to flow control in integrated services networks: the single-node case. *IEEE/ACM Trans. Netw.*, 1(3):344–357.

Shreedhar, M. and Varghese, George (1995). Efficient fair queueing using deficit round robin. In *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication*, pages 231–242. ACM Press.

Zhang, L. (1990). Virtual clock: a new traffic control algorithm for packet switching networks. In *Proceedings of the ACM symposium on Communications architectures & protocols*, pages 19–29. ACM Press.

Zhou, Y. and Sethu, H. (2002). On the relationship between absolute and relative fairness bounds. In *IEEE Comm. Letters, vol. 6, no. 1, pp. 37–39, Jan. 2002.*