



**HAL**  
open science

# WCTT analysis of avionics Switched Ethernet Network with WRR Scheduling

Aakash Soni, Xiaoting Li, Jean-Luc Scharbag, Christian Fraboul

► **To cite this version:**

Aakash Soni, Xiaoting Li, Jean-Luc Scharbag, Christian Fraboul. WCTT analysis of avionics Switched Ethernet Network with WRR Scheduling. 26th International Conference on Real-Time and Network Systems (RTNS 2018), Dec 2018, Poitiers, France. pp.213-222. hal-03619600

**HAL Id: hal-03619600**

**<https://hal.science/hal-03619600>**

Submitted on 25 Mar 2022

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



## Open Archive Toulouse Archive Ouverte

OATAO is an open access repository that collects the work of Toulouse researchers and makes it freely available over the web where possible

This is an author's version published in:

<http://oatao.univ-toulouse.fr/24788>

### Official URL

DOI : <https://doi.org/10.1145/3273905.3273925>

**To cite this version:** Soni, Aakash and Li, Xiaoting and Scharbarg, Jean-Luc and Fraboul, Christian *WCTT analysis of avionics Switched Ethernet Network with WRR Scheduling*. (2018)  
In: 26th International Conference on Real-Time and Network Systems (RTNS 2018), 10 December 2018 - 12 December 2018 (Poitiers, France).

Any correspondence concerning this service should be sent to the repository administrator: [tech-oatao@listes-diff.inp-toulouse.fr](mailto:tech-oatao@listes-diff.inp-toulouse.fr)

# WCTT analysis of avionics Switched Ethernet Network with WRR Scheduling

Aakash SONI  
ECE Paris/ IRIT Toulouse, France  
aakash.soni@irit.fr

Jean-Luc SCHARBARG  
IRIT-ENSEEIH Toulouse, France  
Jean-Luc.Scharbarg@enseeiht.fr

Xiaoting LI  
ECE Paris, France  
xiaoting.li@ece.fr

Christian FRABOUL  
IRIT-ENSEEIH Toulouse, France  
Christian.Fraboul@enseeiht.fr

## ABSTRACT

Nowadays, a real-time embedded system often has to cope with flows with different criticality levels. Such a situation is forecasted for next generation avionics networks, in order to better use communication resources. Indeed, the present situation, where the Avionics Full Duplex switched Ethernet (AFDX) network is reserved for critical avionics flows, leads to a very lightly loaded network.

Quality of Service (QoS) mechanisms such as service disciplines are mandatory in order to cope with the heterogeneous constraints of the different kinds of flows. Weighted Round Robin (WRR) is envisioned in the context of avionics.

Worst-case traversal time (WCTT) analysis is mandatory to ensure that temporal constraints of flows are met. Network Calculus (NC) is a popular solution for this analysis and results exist for WRR. They lead to pessimistic upper bounds.

The main contribution of this paper is to show how existing NC results for WRR can be applied and improved in the context of an avionics configuration. The resulting analysis is evaluated on an industrial size configuration.

## KEYWORDS

WRR, Switched Ethernet, Network Calculus, WCTT, AFDX

## 1 INTRODUCTION

Avionics Full Duplex switched Ethernet (AFDX) has become the de facto standard for the transmission of avionics flows. These flows are scheduled in each switch output port, using a First In First Out (FIFO) service discipline.

Due to certification constraints, end-to-end latencies of flows have to be upper bounded. Thus a worst-case traversal time analysis (WCTT) is mandatory. The most popular approach for this analysis is based on network calculus [1, 5]. Alternative approaches have been proposed, based on trajectories [1], Compositional Performance Analysis [14, 15] and Forward Analysis [2]. All these approaches compute pessimistic upper bounds [1] for two main reasons. First, flows are over-approximated and/or the service provided by the network elements is under-approximated. Second the scheduling of flows by source end systems is not considered. This leads to a very lightly loaded AFDX network. Typically, less than 10 % of the available bandwidth is used for the transmission of avionics flows on an AFDX network embedded in an aircraft [5].

Adding quality of service mechanisms is a classical solution to improve network resource utilization. Such a mechanism is already available in existing AFDX switches: a static priority queuing discipline with two priority levels is implemented in each output port. This facility is not used in current aircrafts. However, it has been shown in [8] that these priorities can be assigned to avionics flows in such a way that the overall upper bound on end-to-end latencies is significantly reduced.

Nevertheless the most promising solution to improve network resource utilization consists in sharing these resources between avionics and additional less/not critical flows. In this context, a static priority queuing discipline with two priority levels is not sufficient. Thus more complex solutions are envisioned for future avionics switches, such as a mixed static priority / round robin discipline. The general idea is to assign the highest priority(ies) to the most critical flows and the lowest priority(ies) to the other flows. Thus these later flows share the remaining bandwidth, based on a round robin policy.

The goal of a round robin solution is to share bandwidth between flow classes in a controlled manner. Different algorithms have been proposed. Both Deficit Round Robin (DRR) [4, 9, 11, 16] and Weighted Round Robin (WRR) [7, 10, 13] are envisioned. DRR shares the bandwidth between flows, based on number of transmitted bytes, while WRR considers transmitted frames. Therefore WRR achieves less fair sharing than DRR, but it is simpler. DRR has

been studied in the context of avionics in [17]. In this paper, we focus on WRR.

As previously mentioned, a WCTT analysis is mandatory in the context of avionics, in order to upper bound end-to-end latencies. Results exist for WRR [7]. They are based on Network Calculus [3].

The main contributions of this paper are to apply these results in the context of avionics, to identify sources of pessimism of this WCTT analysis and to propose an improved solution. An evaluation on an industrial size configuration shows that the proposed optimizations significantly tighten latency upper bounds.

The paper is organized as follows. Section 2 presents the network architecture as well as the Weighted Round Robin service discipline. Section 3 show the direct application of existing NC results for WRR WCTT analysis and identifies some sources of pessimism. Section 4 and 5 propose an optimization of the approach. The evaluation is summarized in Section 6. Section 7 concludes the paper and gives some directions for future work.

## 2 NETWORK MODEL

### 2.1 Network Architecture

In this paper, we consider a real-time switched Ethernet network architecture. We focus on the context of avionics, but the work can be easily extended to other contexts. The network architecture is composed of end systems interconnected by switches via full-duplex links. Thus there are no collisions on links. Each link offers a bandwidth of  $R$  Mbps in each direction. An end system is connected to exactly one switch port. A switch forwards incoming frames to output port buffers, based on a statically defined forwarding table. This forwarding process introduces a switching latency upper bounded by  $sl$ .

Frames forwarded to the same output port compete in order to be transmitted on the corresponding output link. This competition is managed by a scheduler supporting a service discipline. In the case of a FIFO discipline, competing frames are stored in a single queue and served following a First Come First Served policy. When a Fixed Priority (FP) or Round Robin (RR) discipline is implemented, a set of queues (one per traffic class) is associated to each output port. Each competing frame is stored in the queue corresponding to its traffic class. Queues are served, based on the considered service discipline. In a given queue, frames are served following a First Come First Served policy.

In this paper, a Weighted Round Robin (WRR) scheduler is implemented in each switch output port. The WRR scheduler will be presented in Section 2.2.

Each end system generates a set of sporadic flows. A flow  $v_i$  is defined by the following features:

- the minimum duration  $T_i$ , a.k.a. bandwidth allocation gap (BAG) in AFDX, between the generation of two consecutive frames of  $v_i$  by its source end system,
- the minimum and maximum sizes  $l_i^{min}$  and  $l_i^{max}$  of frames from  $v_i$ ,
- the static multi-cast path of  $v_i$ , in the form of a tree where the root is the source end system, leaves are the destination end systems and other nodes are switches,
- the traffic class  $C_i$  of  $v_i$ .

Figure 1 shows an example of a network configuration. It includes 12 end systems  $e_1$  to  $e_{12}$  interconnected by 3 switches,  $S_1$  to  $S_3$ . 18 flows  $v_1$  to  $v_{18}$  are transmitted on this network. They are all unicast. Thus each of them has a single destination end system. Table 1 summarizes flow features. In this example, we assume a constant frame size for each flow ( $l_i^{min} = l_i^{max}$  for  $1 \leq i \leq 18$ ). Therefore, the transmission duration  $d_i$  of one frame of a flow  $v_i$  is constant. It can be computed by:

$$d_i = \tau_{bit} \times l_i^{min} \times 8 = \tau_{bit} \times l_i^{max} \times 8$$

where  $\tau_{bit}$  is the duration for the transmission of a bit.

As an example, assuming  $R = 100$  Mbs, we have  $\tau_{bit} = 0.01 \mu s$ . Thus  $d_8 = 0.01 \times 200 \times 8 = 16 \mu s$ .

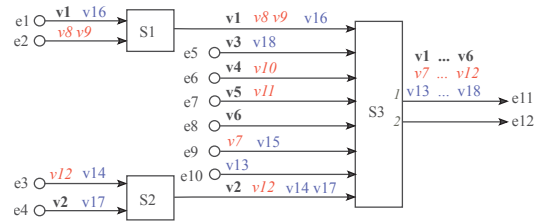


Figure 1: Switched Ethernet network

Table 1: Network Flow Configuration

Flows $v_i$	$T_i$ ( $\mu sec$ )	$l_i^{max}$ (bytes)	$l_i^{min}$ (bytes)	Class $C_x$
$v_1, v_2, v_4$	512	200	200	$C_1$
$v_3, v_5$	768	200	200	$C_1$
$v_6$	896	200	200	$C_1$
$v_7, v_9$	896	200	200	$C_2$
$v_8$	768	200	200	$C_2$
$v_{10}, v_{11}, v_{12}$	512	200	200	$C_2$
$v_{13}, v_{18}$	512	200	200	$C_3$
$v_{14}$	768	200	200	$C_3$
$v_{15}, v_{16}, v_{17}$	896	200	200	$C_3$

### 2.2 Weighted Round Robin

In this section, we briefly recall WRR scheduling policy. A more detailed description can be found in [10, 13]. We then introduce the main feature of a flow latency scheduled by WRR..

#### 2.2.1 Overview of WRR scheduling policy.

WRR was designed for the fair sharing of available network bandwidth between a set of traffic classes. The main motivation was to develop a service discipline that provides adequate congestion control even in the presence of ill-behaved sources. This is achieved by allocating bandwidth and buffer space in a fair manner, which automatically ensures that ill-behaved sources can get no more than their fair share.

WRR is an approximation of Generalized Processor Sharing (GPS) [6, 13]. GPS guarantees a percentage of the bandwidth to each active

traffic class in each output port. A traffic class is active in an output port  $h$  as long as it has pending frames in  $h$ .

More formally, if we assume  $n$  traffic classes  $C_1, \dots, C_n$ , a traffic class  $C_x$  ( $1 \leq x \leq n$ ) is allocated a percentage  $\phi_x^h$  of the bandwidth in output port  $h$ , with

$$\sum_{1 \leq x \leq n} \phi_x^h = 100$$

When all classes  $C_1, \dots, C_n$  are active in port  $h$ , each class  $C_x$  ( $1 \leq x \leq n$ ) receives exactly its allocated percentage of bandwidth  $\phi_x$ . When some classes are inactive in port  $h$ , their allocated percentage of bandwidth is shared between active classes. In that case, active classes get more than their allocated bandwidth.

GPS cannot be directly implemented, since on a link, frames are transmitted sequentially. Thus, it is impossible to respect the percentages of allocated bandwidth for an arbitrarily small interval. Therefore, approximations of GPS are implemented. The idea is to schedule frame transmissions in such a way that it leads to a bandwidth sharing similar to GPS in the (more or less) long term.

WRR implements such an approximation by allowing up to a number of frame transmissions for each class in each round.

Algorithm 1 shows the basic principle of WRR scheduler at a switch output port  $h$ . The scheduler selects queues in a round robin order. Counter  $i$  corresponds to the currently selected queue. Empty queues are ignored in each round (line 3). Each non empty queue is assigned a weight of  $w_i^h$  in each round (line 4). It corresponds to the maximum number of frames pending in the currently selected queue which can be transmitted in the current round. Therefore, packets from the queue are sent in FIFO order as long as the queue is not empty and the number of sent packets is less than or equal to  $w_i^h$  (lines 5-9). If the queue becomes empty before  $w_i^h$  frames are transmitted, remaining transmission opportunities are lost (they cannot be deferred to the following round).

<p><b>Input:</b> Weight Per flow: <math>w_1^h \dots w_n^h</math> (Integer)</p> <p><b>Data:</b> Counter: <math>i</math> (Integer)</p> <p><b>Data:</b> Packet counter: <math>x</math> (Integer)</p> <pre> 1 while true do 2   for <math>i = 1</math> to <math>n</math> do 3     if <math>notempty(i)</math> then 4       <math>x \leftarrow w_i^h</math>; 5       while (<math>notempty(i)</math>) and (<math>x \neq 0</math>) do 6         send(head(<math>i</math>)); 7         removeHead(head(<math>i</math>)); 8         <math>x \leftarrow (x - 1)</math>; 9       end 10    end 11 end </pre>
--

Algorithm 1: WRR Algorithm

The weight  $w_i^h$  of class  $C_i$  queue in output port  $h$  is computed off-line. It has to lead to a good approximation of the percentage  $\phi_i$  of bandwidth allocated to  $C_i$  in port  $h$ . A classical solution is to consider the average frame size  $l_i^{avg,h}$  of class  $C_i$  flows crossing  $h$ . Based on this average frame size, a value  $\nabla_i^h$  indicating the

proportions of  $C_i$  frames which can be transmitted in one round is computed. It is obtained by dividing the percentage of bandwidth  $\phi_i$  envisioned for  $C_i$  by  $l_i^{avg,h}$ :

$$\nabla_i^h = \frac{\phi_i}{l_i^{avg,h}}$$

Most of the time, this computation leads to non integer values for  $\nabla_i^h$ . Since integer values are needed for weights  $w_i^h$ ,  $\nabla_i^h$  values have to be transformed. In this paper, we divide each  $\nabla_i^h$  value by the smallest  $\nabla_x^h$  ( $1 \leq x \leq n$ ) or a divider of this smallest value and we consider the closest integer for  $w_i^h$ :

$$w_i^h = \text{round}\left(\frac{\nabla_i^h}{\min_{1 \leq x \leq n} \nabla_x^h}\right)$$

where  $\text{round}(a)$  denotes the integer which is closest to  $a$ .

Let us consider an example where flows from 3 classes  $C_1, C_2$  and  $C_3$  compete at a switch output port  $h$  controlled by a GPS scheduler. For each class  $C_i$ , the average frame size  $l_i^{avg,h} = 100$  bytes. Let us assume that the class  $C_1$  belongs to critical flows and it should get 50 % of the available bandwidth. The other two classes  $C_2$  and  $C_3$  have 33% and 17% share of bandwidth respectively. In this case, the proportion of frames of class  $C_1$  that can be transmitted in one round is:

$$\nabla_{C_1}^h = \frac{\phi_{C_1}}{l_{C_1}^{avg,h}} = \frac{0.5}{100 * 8}$$

Similarly for  $C_2$  and  $C_3$ , we have:

$$\nabla_{C_2}^h = \frac{0.33}{100 * 8}, \quad \nabla_{C_3}^h = \frac{0.17}{100 * 8}$$

Thus, the weight assigned to each class should be:

$$w_{C_1}^h = 3, \quad w_{C_2}^h = 2, \quad w_{C_3}^h = 1$$

### 2.2.2 WRR latency.

A WRR scheduler serving  $n^h$  traffic classes at a given output port  $h$  transmitting  $W^h = \sum_{j=1 \dots n} w_j^h$  frames in each round, guarantees each class a fraction of output link bandwidth  $R$ . It therefore defines a long-term service rate  $\rho_x^h$  to class  $C_x$  at output port  $h$ . This long-term service rate is proportionate to the weights assigned to the flow classes and the sum of all the long-term service rates assigned to each class cannot be greater than the link rate  $R$ .

The service for the class under study  $C_x$  is minimized when it serves the smallest possible frame  $l_{C_x}^{min,h}$  while the service for the competing classes  $C_j$  is maximized when it serves the largest possible frame  $l_{C_j}^{max,h}$ . Thus in worst-case the long-term service rate  $\rho_x^h$  to class  $C_x$  at output port  $h$  is given by:

$$\rho_x^h = \frac{w_x^h \times l_{C_x}^{min,h}}{(w_x^h \times l_{C_x}^{min,h}) + \sum_{j=1, j \neq x}^{n^h} (w_j^h \times l_{C_j}^{max,h})} \times R \quad (1)$$

This is also shown by the example in Figure 2, where flows from two classes  $C_1$  and  $C_2$  are being served at the link rate 100 Mbps. Each class is allowed to serve at most two frames in each round based on the assigned weight  $w_{C_i} = 2$  (where  $i = 1, 2$ ). Let us observe the service received by class  $C_1$  in the given two cases.

In case 1, as the each class is served 2 frame of maximum frame length 200 bytes, in each round, total 800 bytes are served. Thus, the service received by class  $C_1$  flow in each round is  $\frac{400}{800} \times 100 = \frac{100}{2}$  Mbps. In case 2, where the considered class  $C_1$  serves frames of minimum frame length 100 bytes and the other class  $C_2$  serves frame of maximum frame length 200 bytes, the service received by class  $C_1$  flows is minimized to  $\frac{200}{600} \times 100 = \frac{100}{3}$  Mbps.

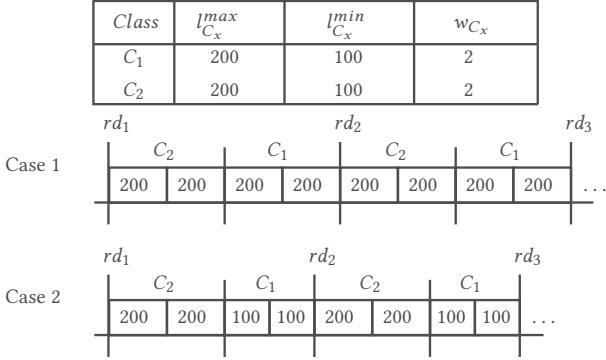


Figure 2: WRR long-term service

The WRR scheduler latency  $\Theta_x^h$  experienced by  $C_x$  flows at output port  $h$  is defined as the maximum delay before  $C_x$  flows receive their first service after their arrival at the port  $h$ . If  $C_x$  queue is empty and a  $C_x$  packet arrives at a time when it just misses its turn to receive service, it has to wait for the next round while all the other active classes  $C_j (j \neq x)$  are served. This delay is maximized when class  $C_x$  has to wait for all the other classes  $C_j$  received their maximum service ( $w_j^h \times l_{C_j}^{max,h}$ ). Thus, the WRR scheduler latency  $\Theta_x^h$  for class  $C_x$  in  $h$  is given by:

$$\Theta_x^h = \frac{\sum_{j \neq x, j=1}^{n^h} (w_j^h \times l_{C_j}^{max,h})}{R} \quad (2)$$

Let us compute this latency for flow  $v_1$  from class  $C_1$  at output port  $S_3^1$  in the network configuration in figure 1. Three traffic classes are considered ( $n^{S_3^1} = 3$ ).  $C_1$  includes flows  $v_1$  to  $v_6$  (in black and bold font, while  $C_2$  includes flows  $v_7$  to  $v_{12}$  (in red and italics font in Figure and  $C_3$  includes flows  $v_{13}$  to  $v_{18}$  (in blue and regular font in Figure 1), as listed in Table 2. Each class  $C_x$  is assigned one third of the bandwidth. Since all the frames have the same size, classes are allocated equal weight of  $w_x^h = 2$ . Thus, the service rate for any class  $C_x$  can be computed by Equation (1):

$$\rho_x^{S_3^1} = \frac{1}{3} \times 100 \text{ Mbits/s}$$

Figure 3 shows a possible scenario for WRR scheduling in port 1 of switch S3 (port  $S_3^1$ ). All the flows in Figure 1 cross this port. The WRR scheduler assumes the configuration as shown in Table 2

In the scenario in Figure 3, there are no pending frames before time  $t_0$  in output port  $S_3^1$ . At that time, eight frames arrive: one belonging to class  $C_2$  (from flow  $v_7$ ) and one belonging to class  $C_3$  (from flow  $v_{13}$ ) and six belonging to class  $C_1$  (from flows  $v_6$ ,

$v_5$ ,  $v_4$ ,  $v_3$ ,  $v_2$  and  $v_1$ ) and in this order. Since there are no pending frames before  $t_0$ , the class which gets service at  $t_0$  is the one which is selected by the round robin process. In Figure 3, we assume that class  $C_2$  is served first. Thus, at  $t_0$ ,  $C_2$  gets service to send at most  $w_2^{S_3^1} = 2$  packets. Since there is one packet from  $v_7$  waiting in  $C_2$  queue, it is transmitted. Meanwhile, two more frames (one from  $v_8$  of  $C_2$  and one from  $v_{14}$  of  $C_3$ ) have arrived at  $S_3^1$ . Since frame from  $v_8$  has arrived before the end of service of class  $C_2$  and as per the assigned weight the class  $C_2$  can transmit one more packet, the frame from  $v_8$  is served in the same round robin cycle  $r_1$ . At this moment, since class  $C_2$  has no more pending frames and it has already received service of its predefined weight, the scheduler moves to service of next active class  $C_3$ . The class  $C_3$  get service to send at max  $w_3^{S_3^1} = 2$  packets. The frames from class  $C_3$  flows  $v_{13}$  and  $v_{14}$  are now transmitted. Frames from flows  $v_9$ ,  $v_{10}$ ,  $v_{11}$ ,  $v_{12}$ ,  $v_{15}$  and  $v_{16}$  have arrived and they have been buffered into their respective class queues. The next active class  $C_1$  can now transmit at most  $w_1^{S_3^1} = 2$  packets. The two  $C_1$  head-of-line packets (from  $v_6$  and  $v_5$ ) are served in the current round. Since all the assigned weight to each class is now consumed, the remaining flows in the buffer will be served in the subsequent rounds. Frame transmissions go on in the same manner. As defined above, the delay between the arrival of class  $C_1$  flow  $v_1$  at  $t_0$  and the service of first packet of class  $C_1$  gives the latency of WRR scheduler, from equation (2) we have:

$$\Theta_{C_1}^{S_3^1} = \frac{\sum_{j=C_2, C_3} (w_j^{S_3^1} \times l_{C_j}^{max,S_3^1})}{100} = \frac{(2+2) \times (200 \times 8)}{100} = 64 \mu sec$$

Thus, the scenario in figure 3 maximizes  $\Theta_{C_1}^{S_3^1}$ .

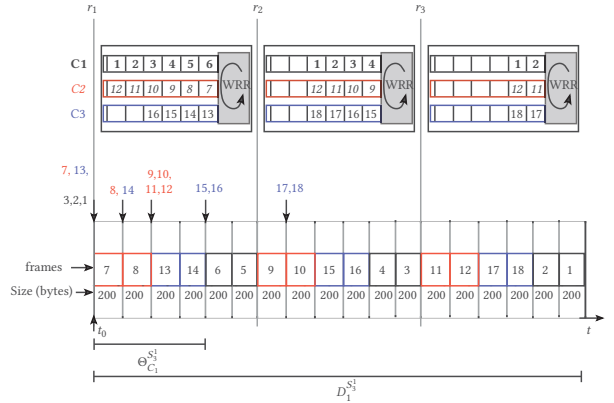


Figure 3: WRR rounds at output port  $S_3^1$

### 3 WRR WCTT ANALYSIS

In this section, we focus on the Network Calculus (NC) approach for worst-case traversal time analysis in switched Ethernet network. The NC theory is based on the  $(\min, +)$  algebra. It has been proposed for worst-case backlog and delay analysis in networks [3]. Formulas for WRR have been given in [3, 7]. It models traffic by arrival curves

**Table 2: WRR scheduler configuration**

Class $C_x$	Flows $v_i$	$w_x$ (byte)	$l_{C_x}^{max}$ (byte)	$l_{C_x}^{min}$ (byte)
$C_1$	$v_1$ to $v_6$	2	200	200
$C_2$	$v_7$ to $v_{12}$	2	200	200
$C_3$	$v_{13}$ to $v_{18}$	2	200	200

and network elements by service curves. Upper bounds on buffer size and delays are derived from these curves.

### 3.1 Arrival Curve

The traffic of a flow  $v_i$  at an output port  $h$  is over-estimated by an arrival curve, denoted by  $\alpha_i^h(t)$ . The leaky bucket is a classical arrival curve for a sporadic traffic:

$$\alpha_i^h(t) = r_i \times t + b_i, \text{ for } t > 0 \text{ and } 0 \text{ otherwise.}$$

with flow arrival rate  $r_i = \frac{l_i^{max}}{T_i}$  and burst  $b_i = l_i^{max}$ , where  $T_i$  is the minimum inter-frame arrival time of flow  $v_i$ .

Hence, arrival curve of a flow  $v_i$  at its source end system  $e_k$  is:

$$\alpha_i^{e_k}(t) = \frac{l_i^{max}}{T_i} \times t + l_i^{max}, \text{ for } t > 0 \text{ and } 0 \text{ otherwise.}$$

It means that  $v_i$  is allowed to send at most one frame of maximum length  $l_i^{max}$  bits every minimum inter-frame arrival time  $T_i$   $\mu$ s.

Any flow  $v_i$  can be modeled in a similar manner at any switch output port  $h$  it crosses. However, since a frame of flow  $v_i$  can be delayed by other frames before it arrives at port  $h$ , a jitter  $J_i^h$  has to be introduced. It is the difference between the worst-case delay and the best-case delay for a frame of flow  $v_i$  from its source end system to port  $h$  [1].

Since flows of class  $C_x$  are buffered in their class queue and scheduled by FIFO policy, an overall arrival curve is used to constrain the arrival traffic of class  $C_x$  at port  $h$ . It is denoted by  $\alpha_{C_x}^h$  and calculated by:

$$\alpha_{C_x}^h(t) = \sum_{i \in \mathcal{F}_{C_x}^h} \alpha_i^h(t) \quad (3)$$

where  $\mathcal{F}_{C_x}^h$  is the set of  $C_x$  flows crossing port  $h$ .

### 3.2 Service Curve

According to NC, the full service provided at a switch output port  $h$  with a transmission rate of  $R$  (bits/s) is defined by:

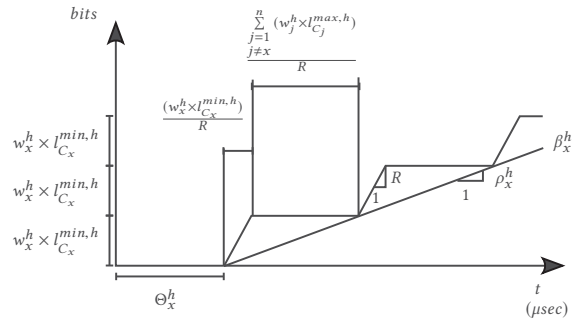
$$\beta^h(t) = R[t - sl]^+$$

where  $sl$  is the switching latency of the switch, and  $[a]^+$  means  $\max\{a, 0\}$ .

As derived in [7], the full service is shared by all WRR classes at an output port  $h$  and each class  $C_x$  has a predefined service rate  $\rho_x^h$  based on its assigned weight  $w_x^h$  as explained in Section 2.2 equation (1). Besides a reduced service rate, each class  $C_x$  could experience a WRR scheduler latency  $\Theta_x^h$  before receiving a service for the first time. The scheduler latency can be calculated by Equation (2). Therefore, based on the NC approach, the residual service  $\beta_{C_x}^h$  to each class  $C_x$  is given by:

$$\beta_{C_x}^h(t) = \rho_x^h[t - \Theta_x^h - sl]^+ \quad (4)$$

The actual service curve is a staircase one, as a flow alternates between being served and waiting for its WRR opportunity. This curve is shown in Figure 4 by dashed line. As shown in Section 2.2, the considered class  $C_x$  has to wait for  $(n-1)$  interfering classes  $C_j$  ( $j = 1$  to  $n$  and  $j \neq x$ ), in the worst-case each interfering class  $C_j$  transmits at most  $(w_j^h \times l_{C_j}^{max, h})$  bytes, this adds a delay of  $\Theta_x^h$ , then the class  $C_x$  receives its first service of minimum  $(w_x^h \times l_{C_x}^{min, h})$  bytes, which gives the first stair. The transmission continues in the same manner in each round, which gives the next stairs. For computation reason, NC approach employs the convex curve represented by equation (4) which is an under-estimated approximation of actual staircase curve. The staircase curve and its approximated convex curve are shown in Figure 4.



**Figure 4: NC service curve for WRR scheduler**

### 3.3 Delay bound

According to NC, the delay experienced by a  $C_x$  flow  $v_i$  constrained by the arrival curve  $\alpha_{C_x}^h(t)$  in a switch output port  $h$  offering a strict WRR service curve  $\beta_{C_x}^h(t)$  is bounded by the maximum horizontal difference between the curves  $\alpha_{C_x}^h(t)$  and  $\beta_{C_x}^h(t)$ . Let  $D_i^h$  be this delay. It is computed by:

$$D_i^h = \sup_{s \geq 0} (\inf \{ \tau \geq 0 | \alpha_{C_x}^h(s) \leq \beta_{C_x}^h(s + \tau) \}) \quad (5)$$

Therefore, the end-to-end delay upper bound of a  $C_x$  flow  $v_i$  is denoted by  $D_i^{ETE}$  and it is calculated by:

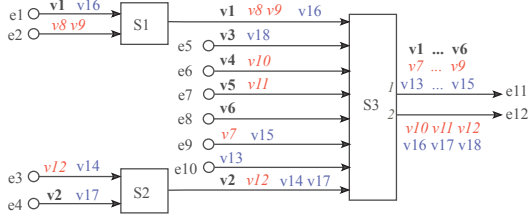
$$D_i^{ETE} = \sum_{h \in \mathcal{P}_i} D_i^h \quad (6)$$

where  $\mathcal{P}_i$  is the path followed by the flow  $v_i$ . For flow  $v_1$  of class  $C_1$ , based on the equation (5) and (6), the delay bound at port  $S_3^1$  is  $D_1^{S_3^1} = 406.35 \mu$ s and the end-to-end delay is  $D_1^{ETE} = 602.85 \mu$ s.

### 3.4 Pessimism in delay computation

The WRR service discussed in previous section is based on a pessimistic assumption that, at an output port  $h$ , at any moment there is always some amount of traffic from each active class  $C_i$  and that each interfering class consumes its maximum service in each round robin cycle. This assumption can be too pessimistic in some cases where there is not enough traffic from different classes to consume full service in each round robin cycle.

Let's illustrate this pessimism with the example in Figure 5. The network configuration considered in this example is similar to that shown in Figure 1 except that the traffic at port  $S_3^1$  is reduced in the example in Figure 5, i.e. the flows  $v_{10}, v_{11}, v_{12}, v_{16}, v_{17}$ , and  $v_{18}$  which were traversing  $S_3^1$  in Figure 1 are now passing through  $S_3^2$  in Figure 5. Therefore the traffic from class  $C_2$  and  $C_3$  at  $S_3^1$  is reduced. We assume the same scheduler configuration as shown in Table 2.



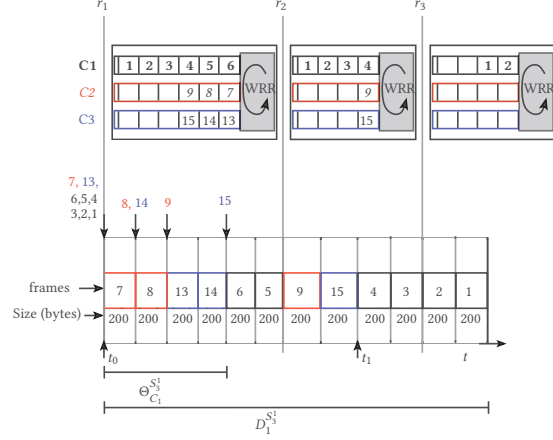
**Figure 5: Switched Ethernet network with reduced traffic at  $S_3^1$**

Let us consider a delay scenario experienced by flow  $v_1$  of class  $C_1$  at port  $S_3^1$ . A possible case of WRR service at  $S_3^1$  in Figure 5 is shown in Figure 6. In this case, in the first round  $r_1$ , the class  $C_1$  flows experience a WRR latency  $\Theta_{C_1}^{S_3^1} = 64\mu\text{sec}$ , which is similar to that in previous example. In round  $r_2$ , class  $C_2$  can serve at most  $w_2^{S_3^1} = 2$  packets. However, there is only one frame from class  $C_2$  flow  $v_9$ . The class  $C_2$  serves its only available frame from flow  $v_9$  and the scheduler moves to next active class  $C_3$ . Similarly for class  $C_3$ , only frame from  $v_{15}$  is served and the scheduler moves to next active class  $C_1$ . At time  $t_1$ , the classes  $C_2$  and  $C_3$  have served all their frames and have no more frames to further delay  $C_1$  flows in next round. Since there are no other active classes, class  $C_1$  is served at full server capacity in successive round. In such case, the staircase service and its approximation considered in previous section will be too pessimistic in delay computation.

However, the scenario in Figure 6 might not lead to the worst case. In next section, we show how to upper bound the effective impact of interfering classes in order to reduce the pessimism introduced by the service curve.

#### 4 IMPROVED APPROACH FOR DELAY COMPUTATION USING NETWORK CALCULUS

As illustrated in Section 3.4, in worst case delay computation for a flow  $v_i$  of class  $C_x$ , a pessimism might be introduced when considering maximum service of interfering classes, as the interfering classes might generate too few traffic to consume all their allocated service. In this section we show how to upper bound the traffic of these interfering classes that impact the worst-case delay computation for class  $C_x$  flows. When the computed upper bound is smaller than the maximized service for interfering classes, the difference between the two can be safely removed from the worst-case delay of  $C_x$  flow. Following steps are considered in the approach.



**Figure 6: WRR rounds at output port  $S_3^1$**

- (1) First, use the network calculus approach shown in previous section to compute worst-case delay  $D_i^h$  for class  $C_x$  flow  $v_i$  at port  $h$ .
- (2) Then, determine the maximized service of interfering class  $C_y$  considered by the NC approach used to calculate the worst case delay in step 1. We call this maximized service as *service load*  $SL_y^h(D_i^h)$  for class  $C_y$  at a node  $h$  between 0 and  $D_i^h$ .
- (3) Then, calculate an upper bound on interfering class  $C_y$  traffic being served between 0 and  $D_i^h$  at node  $h$ . We call this upper bound as *effective maximum load*  $L_y^{max,h}(D_i^h)$  of a class  $C_y$  at a node  $h$  between 0 and  $D_i^h$ .
- (4) Finally, if  $L_y^{max,h}(D_i^h) < SL_y^h(D_i^h)$ , compute the difference between  $SL_y^h(D_i^h)$  and  $L_y^{max,h}(D_i^h)$  which can be safely removed from the worst-case delay  $D_i^h$  of  $C_x$  flow  $v_i$  at node  $h$ . The new reduced delay obtained is called as improved delay  $D_{i,opt}^h$  for  $C_x$  flow  $v_i$  at node  $h$ .

Step 1 is presented in Section 3. The three other steps are detailed in the following paragraphs.

##### 4.1 Service load

The NC approach in Section 3 considers a maximized service for an interfering class  $C_y$  that can be described in two parts. First part is of interval of  $(0, \Theta_x^h]$  that correspond to the latency delay  $\Theta_x^h$  before  $C_x$  is served for the first time. Second part is the interval  $(\Theta_x^h, +\infty)$  which is divided into multiple identical intervals of length  $t_N$ . In each interval, the NC approach assumes that  $C_y$  gets a service of at most  $w_y^h \times I_{C_y}^{max,h}$  bytes.

Thus the maximized service load  $SL_y^h(t)$  is defined as follows:

$$SL_y^h(t) = \begin{cases} 0 & t < \Theta_x^h \\ (w_y^h \times I_{C_y}^{max,h}) \left( 1 + \left\lfloor \frac{(t - \Theta_x^h)}{t_N} \right\rfloor \right) & \Theta_x^h \leq t \end{cases} \quad (7)$$



The length of the each interval  $t_N$  can be given by:

$$t_N = \frac{(w_x^h \times l_{C_x}^{max,h})}{R} + \frac{\sum_{j=1, j \neq x}^{n^h} (w_j^h \times l_{C_j}^{max,h})}{R}$$

where  $C_j$  is active in  $h$ .

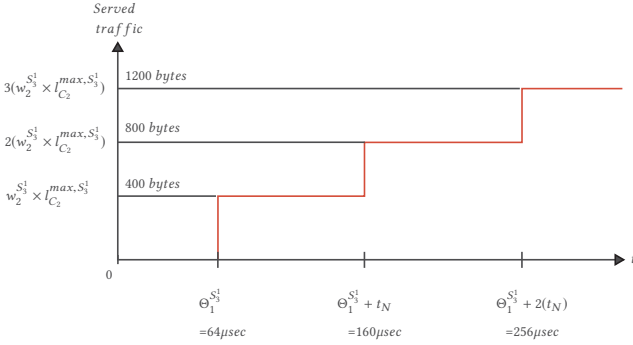


Figure 7: Served load of class  $C_2$  at  $S_3^1$

It should be noticed that, for a given interval, the corresponding load is taken into account at the end of the interval. Thus  $SL_y^h(t)$  gives an under-bound of the maximized service load considered in the NC approach. The obtained step function is shown in Figure 7.

In the example in Figure 5, from NC computation, we obtain  $D_1^{S_3^1} = 397.99 \mu\text{sec}$ . Thus, the maximized load of an interfering class  $C_y$  in the service duration  $t = D_1^{S_3^1}$  of class  $C_1$  flow  $v_1$  is:

$$SL_{C_2}^{S_3^1}(D_1^{S_3^1}) = SL_{C_3}^{S_3^1}(D_1^{S_3^1}) = 1600 \text{ bytes}$$

## 4.2 Upper bound on interfering class traffic

As shown in Section 3, NC arrival curves gives an overestimated traffic of a flow, thus, in a given duration  $t$ , an overall arrival curve of an interfering class  $C_y$  at node  $h$  can be used to compute effective maximum load  $L_y^{max,h}(t)$ . Since the delay of a  $C_x$  flow  $v_i$  packet in node  $h$  is upper bounded by  $D_i^h$ , only packets of interfering class  $C_y$  arriving within a duration  $D_i^h$  will get a chance to delay a given  $C_x$  packet. Thus  $C_y$  load that can delay a given  $C_x$  packet is upper bounded by:

$$L_y^{max,h}(D_i^h) = \alpha_{C_y}^h(D_i^h) \quad (8)$$

where  $\alpha_{C_y}^h(t)$  is the overall arrival curve of the class  $C_y$  flows at node  $h$ , which can be computed by Equation (3).

In the example in Figure 5, we have  $D_1^{S_3^1} = 397.99 \mu\text{sec}$ . Thus, we have:

$$L_{C_2}^{max,S_3^1}(D_1^{S_3^1}) = 918 \text{ bytes}$$

Similarly, for  $C_3$  flows, we have:

$$L_{C_3}^{max,S_3^1}(D_1^{S_3^1}) = 981 \text{ bytes}$$

## 4.3 Service limitation to the load

If the upper bound calculated in Section 4.2 is smaller than the service load (Section 4.1) considered by the NC approach, the difference between them can be safely removed from the delay  $D_i^h$  of  $C_x$  flow  $v_i$  packets in node  $h$ . Thus, for each interfering class  $C_y$  we can remove the following value:

$$\max\left(SL_y^h(D_i^h) - L_y^{max,h}(D_i^h), 0\right) \quad (9)$$

For  $n^h - 1$  interfering classes, the improved delay  $D_{i,opt}^h$  for  $C_x$  flow  $v_i$  in node  $h$  is given by:

$$D_{i,opt}^h = D_i^h - \frac{\sum_{y \in 1 \dots n^h, y \neq x} \max\left(SL_y^h(D_i^h) - L_y^{max,h}(D_i^h), 0\right)}{R} \quad (10)$$

Therefore, the end-to-end delay upper bound of a class  $C_x$  flow  $v_i$  can be computed by:

$$D_{i,opt}^{ETE} = \sum_{h \in \mathcal{P}_i} D_{i,opt}^h \quad (11)$$

In the example in Figure 5,  $D_{1,opt}^{S_3^1}$  is  $293.95 \mu\text{sec}$  and  $D_{1,opt}^{ETE}$  is  $461.14 \mu\text{sec}$ , which gives 23.5% improvement in ETE delay computation as compared to the  $D_1^{ETE} = 602.85 \mu\text{sec}$ .

## 5 FURTHER IMPROVEMENTS

### 5.1 Serialization

The overall arrival curve  $\alpha_{C_x}^h$  for a class  $C_x$ , at a node  $h$ , discussed in Section 3 is obtained by summing the arrival curves of all  $C_x$  flows in  $h$ . This operation assumes that one frame from each flow arrives exactly at the same time in  $h$ . This assumption may not be true in some cases. For example, in Figure 1,  $C_2$  flows  $v_8, v_9$  share the link between  $S_1$  and  $S_3$ . Thus, they cannot arrive in port  $S_3^1$  at the same time. In this case, they are serialized on the link.

This serialization effect has been already integrated in the NC approach for FIFO [1]. The basic idea is to consider that the largest packet among all the flows sharing the link arrives first while the packets from the other flows arrives at the speed of the link in decreasing order of their packet lengths.

We adapt this approach to WRR schedulers by considering serialization for each class separately. Therefore, the serialization effect can be integrated as in [1], on a class by class basis. These serialized arrival curves are then directly used for the worst-case delay computation.

### 5.2 Flow scheduling

To reduce the effective traffic in a switched Ethernet network, each end system introduces a temporal separation between its individual flows. The scheduling of flows emitted by given end system is characterized by the assignment of offsets which constrain the arrival of flows at output ports. [12] proposed integration of offset in NC for First-In-First-Out (FIFO) scheduler. We use a similar approach for WRR schedulers. The basic idea is that, the temporally separated flows cannot arrive at an output port at the same time. Such flows can be aggregated as a single flow.

Flow scheduling at a given end system is characterized by the offset assignment which constrain the flow release time and, consequently, their arrivals at a switch output port. [12] defines *definite offset*  $O_{d,m}^{e_i}$  as the release time of the first frame of a flow  $v_m$  at its source end system  $e_i$ , and *relative offset*  $O_{r,m,n}^h$  at an output port  $h$  as the minimum time interval between the arrival time of a frame  $f_m$  from a benchmark flow  $v_m$  in a port  $h$  and the arrival time of a following frame  $f_n$  from flow  $v_n$  in the same port  $h$ .

For each flow, *definite offsets* are fixed and are defined at the respective end systems. The *relative offset* depends on the flow traffic at the considered output port. [12] propose a definite offset computation algorithm at end system, however, the aggregation technique can work with any offset assignment algorithm.

At an output port  $h$ , the *relative offset*  $O_{r,m,n}^h$  computation assumes that a frame  $f_m$  from benchmark flow  $v_m$  experiences its maximum delay  $D_{max,m}^{e_i,h}$  between its emission at source end system  $e_i$  till its arrival at output port  $h$ , while a frame  $f_n$  from flow  $v_n$  experiences its minimum delay  $D_{min,n}^{e_i,h}$ . Thus,  $O_{r,m,n}^h$  is given by:

$$O_{r,m,n}^h = O_{r,m,n}^{e_i} + D_{min,n}^{e_i,h} - D_{max,m}^{e_i,h} \quad (12)$$

Since  $f_m$  and  $f_n$  are serialized, hence, the *Relative Offset* between  $f_m$  and  $f_n$  cannot be less than the transmission time  $tr_m$  of  $f_m$ . Thus, we have:

$$O_{r,m,n}^h = \max \{O_{r,m,n}^h, tr_m\}$$

In WRR, the effect of offset is introduced per class, as each class gets its own dedicated bandwidth. Now we show a step by step approach to compute an overall arrival curve of aggregated flows. Let us consider the example in Figure 5.

First we make subsets based on the flows sharing the same source end system. At output port  $S_3^1$ , the class  $C_2$  flows  $v_8$  and  $v_9$  share the same source end system  $e_2$ , hence, they are assigned to a same subset  $SS_1$ . The flow  $v_7$  from class  $C_2$  emitted from a different source end system  $e_9$  is assigned to an another subset  $SS_2$ .

Next, we aggregate the flows of each subset as one flow and characterize its arrival curve  $\alpha_{SS_i}^h$ .

$$\alpha_{SS_1}^{S_3^1} = \max \{ \alpha_{v_8\{v_9\}}^{S_3^1}, \alpha_{v_9\{v_8\}}^{S_3^1} \}$$

where,  $\alpha_{m\{n\}}^h$  is the arrival curve obtained when flow  $v_m$  arrives before flow  $v_n$  at output port  $h$ , with temporal separation of  $O_{r,m,n}^h$ , this kind of arrival curve is shown in Figure 8.

Since, there is only one flow  $v_7$  in subset  $SS_2$ , the aggregated arrival curve is same as the arrival curve of flow  $v_7$ .

$$\alpha_{SS_2}^{S_3^1} = \alpha_{v_7}^{S_3^1}$$

These aggregated arrival curves are show in figure 9.

The overall arrival curve is the sum of the arrival curve of each subset, i.e.  $\alpha_{C_2}^{S_3^1} = \sum_{j=1}^2 \alpha_{SS_j}^{S_3^1}$ .

The obtained overall arrival curve can be used to compute delay using equation (5). It can also be used to calculate effective maximum load  $L_{C_y}^h(t)$  of interfering class  $C_y$  to improve the delay computation using equation (10). Figure 10 shown an overall arrival curve for class  $C_2$  flows at  $S_3^1$ .

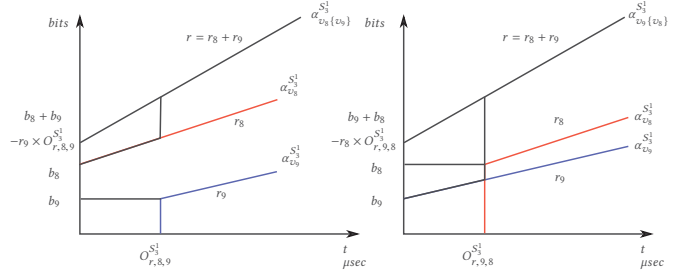


Figure 8: Possible arrival curves for subset  $SS_1$

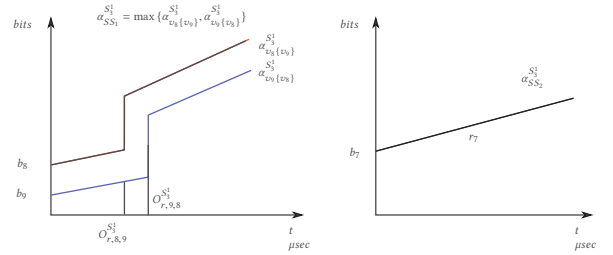


Figure 9: Aggregated arrival curves at  $S_3^1$

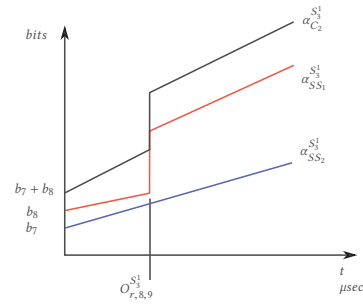


Figure 10: Aggregated arrival curves at  $S_3^1$

## 6 EVALUATION

In this section, we compare the classical approach shown in Section 3 with our improved approach (improved NC WRR) shown in 4. Note that all the results shown in this paper consider serialization: the bursts in each output port can be limited, since flows arriving from the same input link are serialized and, consequently, they cannot arrive at the same time.

### 6.1 Evaluation on illustrative example configuration

We consider the two examples discussed in Figures 1 and 5, the end-to-end delay for each flow in the two examples are calculated using the classical and the improved NC approach and are shown in Figure 11. Based on the compared results, for example in Figure 1, our improved NC approach gives better results with average gain 8.12%, and maximum gain 17.5%, whereas for example in Figure 5, average gain is 17.06% and maximum 26.78%.

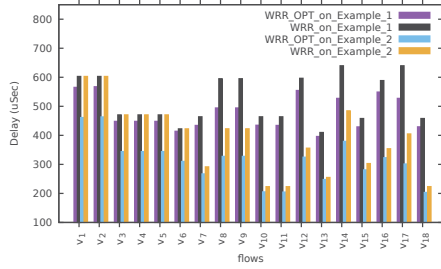


Figure 11: End-to-end delay in network example 1 & 2

## 6.2 Evaluation on industrial configuration

Now we consider an industrial-size configuration, it includes 96 end systems, 8 switches, 984 flows, and 6276 paths (due to VL multi-cast characteristics). The flows are characterized into three WRR classes: critical flows, multimedia flows and best effort flows. Table 3 shows the WRR scheduler configuration at each output port.

Table 3: WRR Scheduler Configuration for Industrial Network

Class	Number of Flows	Max Frame Length (byte)	$w_x$	T (msec)	Category
C <sub>1</sub>	718	475	4	4 - 128	Critical
C <sub>2</sub>	194	971	2	2 - 128	Multimedia
C <sub>3</sub>	72	1535	1	2 - 128	Best-effort

For the given industrial network configuration, Figure 12 shows a comparison between classical NC approach and improved NC approach, the average improvement of the E2E delay bound computed in the given industrial configuration is 32.7% and a maximum gain of 54%. The Figure 13 shows the similar comparison with integration of flow scheduling through offset.

In Figure 12 and 13, for each path  $\mathcal{P}_x$  of each flow  $v_i$ , the upper bound  $D_{i,NC}^{\mathcal{P}_x}$  computed by the classical NC approach is taken as the reference value and it is normalized to 100. Then the upper bound  $D_{i,opt}^{\mathcal{P}_x}$  of improved NC approach is normalized as

$$D_{i,opt,norm}^{\mathcal{P}_x} = \frac{D_{i,opt}^{\mathcal{P}_x}}{D_{i,NC}^{\mathcal{P}_x}} \times 100$$

For illustration purpose, the paths are sorted in increasing order of  $D_{i,opt,norm}^{\mathcal{P}_x}$ .

## 7 CONCLUSIONS

In this paper, we consider an avionics switched Ethernet network enhanced in order to share the available bandwidth between flows with different criticality levels. Flows are scheduled in switch output ports, based on a Weighted Round Robin service discipline. First, we present a classical network calculus approach for the Worst-Case Traversal Time analysis of such a network. Then we exhibit sources of pessimisms in this classical analysis and we propose several improvements in order to mitigate this pessimism. We show on

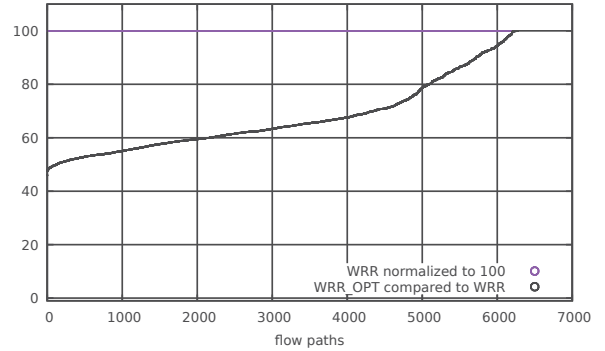


Figure 12: Classical NC WRR vs. Improved NC WRR

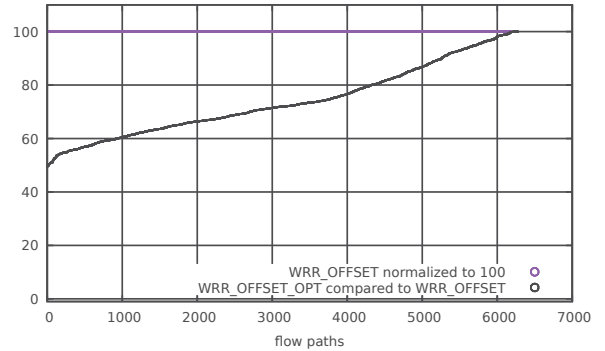


Figure 13: Classical NC WRR with OFFSET vs. Improved NC WRR with OFFSET

an industrial-size case study that these improvements significantly tighten worst-case end-to-end latencies.

Other scheduling policies are envisioned, such as Deficit Round Robin [16]. It provides a better approximation of GPS, at the price of a higher complexity. A comparison between both solutions, including complexity issues, is needed. Such a comparison has to take into account features and constraints of additional (less/non critical) flows. The precise characterization of these flows is still in progress.

## REFERENCES

- [1] Henri Bauer, Jean-Luc Scharbag, and Christian Fraboul. 2010. Improving the Worst-Case Delay Analysis of an AFDX Network Using an Optimized Trajectory Approach. *IEEE Trans. Industrial Informatics* 6, 4 (Nov 2010).
- [2] Nassima Benammar, Frédéric Ridouard, Henri Bauer, and Pascal Richard. 2018. Forward End-to-End Delay for AFDX Networks. *IEEE Trans. Industrial Informatics* 14, 3 (2018), 858–865.
- [3] Jean-Yves Le Boudec and Patrick Thiran. 2012. *Network Calculus: a theory of deterministic queuing systems for the internet*. Vol. 2050. LNCS.
- [4] Marc Boyer, Nicolas Navet, Marc Fumey, Jörn Miggié, and Lionel Havet. 2013. Combining static priority and weighted round-robin like packet scheduling in AFDX for incremental certification and mixed-criticality support. *5TH EUROPEAN CONFERENCE FOR AERONAUTICS AND SPACE SCIENCES (EUCASS)* (2013).
- [5] Hussein Charara, Jean-Luc Scharbag, Christian Fraboul, and Jerome Ermont. 2006. Methods for bounding end-to-end delays on an AFDX network. *Real-Time Systems. 18th Euromicro Conference on. IEEE* (July 2006), 10.

- [6] A. Demers, S. Keshav, and S. Shenker. 1989. Analysis and Simulation of a Fair Queueing Algorithm. *SIGCOMM Comput. Commun. Rev.* 19, 4 (Aug. 1989). <https://doi.org/10.1145/75247.75248>
- [7] Jean-Philippe Georges, Thierry Divoux, and Éric Rondeau. 2011. Network calculus: application to switched real-time networking. In *Proceedings of the 5th International ICST Conference on Performance Evaluation Methodologies and Tools*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 399–407.
- [8] Tasnim Hamza, Jean-Luc Scharbag, and Christian Fraboul. 2014. Priority assignment on an avionics switched Ethernet Network (QoS AFDX). In *10th IEEE Workshop on Factory Communication Systems WFCS, Toulouse, France, May 5-7*. 1–8.
- [9] S. Salil Kanhere and Harish Sethu. 2002. On the latency bound of Deficit Round Robin. *Computer Communications and Networks, Proceedings. Eleventh International Conference on. IEEE*, p. 548–553. (October 2002), 7.
- [10] M. Katevenis, S. Sidiropoulos, and C. Courcoubetis. 1991. Weighted round-robin cell multiplexing in a general-purpose ATM switch chip. *IEEE Journal on Selected Areas in Communications* 9, 8 (Oct 1991), 1265–1279. <https://doi.org/10.1109/49.105173>
- [11] L. Lenzini, E. Mingozzi, and G. Stea. 2002. Aliquem: a novel DRR implementation to achieve better latency and fairness at  $O(1)$  complexity. *IEEE Tenth IEEE International Workshop on Quality of Service (2002)*.
- [12] Xiaoting Li, Jean-Luc Scharbag, and Christian Fraboul. 2010. Improving end-to-end delay upper bounds on an AFDX network by integrating offsets in worst-case analysis. (Sept 2010), 1–8.
- [13] A. K. Parekh and R. G. Gallager. 1993. A generalized processor sharing approach to flow control in integrated services networks: the single-node case. *IEEE/ACM Transactions on Networking* 1, 3 (Jun 1993), 344–357. <https://doi.org/10.1109/90.234856>
- [14] Jonas Rox and Rolf Ernst. 2010. Formal timing analysis of full duplex switched based Ethernet architectures. In *Proc. of the SAE aerotech congress and exhibition*.
- [15] Jonas Rox and Rolf Ernst. 2013. Compositional performance analysis with improved analysis techniques for obtaining viable end-to-end latencies in distributed embedded systems. *STTT* 15, 3 (2013), 171–187.
- [16] Madhavapeddi Shreedhar and George Varghese. 1996. Efficient fair queueing using deficit round-robin. *IEEE/ACM Transactions on networking, vol. 4, no 3, p. 375-385* (1996), 11.
- [17] Aakash Soni, Xiaoting Li, Jean-Luc Scharbag, and Christian Fraboul. 2018. Integrating Offset in Worst Case Delay Analysis of Switched Ethernet Network With Deficit Round Robin. *IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)* (2018).