Quantum assignment for QoS-aware AFDX network with Deficit Round Robin

Aakash Soni, Jean-Luc Scharbarg, Jérôme Ermont

Quantum assignment for QoS-aware AFDX network with Deficit Round Robin

Aakash Soni, Jean-Luc Scharbarg, Jérôme Ermont
University of Toulouse - IRIT - INPT/ENSEEIHT
Toulouse, France
firstname.lastname@enseeiht.fr

ABSTRACT

Avionics Full Duplex switched Ethernet (AFDX) is the de facto standard for the transmission of critical avionics flows. It is a specific switched Ethernet solution based on First-in First-out (FIFO) scheduling. Timing constraints have to be guaranteed for such critical flows. The worst-case traversal time analysis introduces some pessimism, leading to a very lightly loaded network: typically less than 10 % of the bandwidth is used. One solution to improve the utilisation of the network is to introduce Quality of Service (QoS) mechanisms. First, it can decrease worst-case delays for the most constrained avionics flows. Second less/non critical additional flows can be transmitted on the network with bounded impact on avionics ones. Deficit Round Robin (DRR) is such a QoS mechanism and it is envisioned for future avionics networks. An optimised WCTT analysis has been proposed for DRR on AFDX, based on network calculus. With DRR, the flow set is divided into classes and each class is allocated a quantum. In each round, transmissions are managed, based on these quanta. Thus delays are significantly impacted by quanta. The contribution of this paper is to propose an efficient quantum assignment for a set of critical avionics flow classes and at most one additional class with less/non critical flows.

CCS CONCEPTS

• **Networks** → *Network performance modeling*; *Network performance analysis*;

KEYWORDS

Quantum Allocation, Deficit Round Robin, WCTT analysis, AFDX, Switched Ethernet, Quality of Service.

ACM Reference format:
Aakash Soni, Jean-Luc Scharbarg, Jérôme Ermont. 2019. Quantum assignment for QoS-aware AFDX network with Deficit Round Robin. In *Proceedings of 27th International Conference on Real-Time Networks and Systems, Toulouse, France, November 6–8, 2019 (RTNS 2019),* 10 pages.
https://doi.org/10.1145/3356401.3356421

RTNS 2019, November 6–8, 2019, Toulouse, France

ACM ISBN 978-1-4503-7223-7/19/11…$15.00
https://doi.org/10.1145/3356401.3356421

1 INTRODUCTION

Up to now, Quality of Service (QoS) mechanisms are not used in practice in the context of avionics. The de facto standard is the AFDX network, which mainly implement a FIFO service discipline in switch output ports. Actually, two priority levels are available, but they are rarely used. Different approaches have been proposed for Worst-case traversal time analysis in the context of avionics, e.g. Network Calculus (NC) [2, 3] with successful implementation to certify A380 and A350 AFDX backbone[1]. The pessimism of WCTT analysis as well as the fact that worst-case scenarios have a very low probability to occur lead to a very lightly loaded 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. One solution to improve the utilisation of the network is to introduce Quality of Service (QoS) mechanisms. Deficit Round Robin (DRR) is such a mechanism and it is envisioned for future avionics networks. The goals are, first to better use the bandwidth for avionics flows by assigning flows with very different timing constraints to different classes, second to allow the transmission of less/non critical additional flows.

Deficit Round Robin (DRR) was proposed in [10] to achieve fair sharing of the capacity of a server among several flow classes. It is based on quanta which are allocated to classes per round. The main interest of DRR is its simplicity of implementation. As long as specific allocation constraints are met, it can exhibit $O(1)$ complexity. A comparison of DRR scheduler with First-In-First-Out (FIFO) and Static Priority (SP) scheduler used in AFDX network is shown in [6]. The end-to-end delay (ETE) bounds are computed and the paper shows the comparatively better performance of DRR scheduler over FIFO and SP scheduler, given an optimised network configuration. Another DRR implementation is proposed in [4], which combines the DRR with SP scheduling, to improve schedulability and make more efficient use of hardware resources. A detailed analysis and improvement of DRR latency bound for homogeneous flows is given in [7]. Some mathematical errors of [7] are pointed out and corrected in [8]. Analysis of a server with DRR scheduler using NC method is first discussed in [5] which also proposes improvement in DRR latency. [5] generalises the analysis to network with heterogeneous flows. An optimised WCTT analysis for DRR on AFDX is proposed in [11].

The assignment of flows to classes and the quantum allocated to each class has a major impact on the end-to-end latencies of flows. In this paper, we assume that flows have been assigned to classes and we focus on credit allocation. This problem has been addressed in the context of non critical flows with Weighted Round Robin (WRR) [9]. To the best of our knowledge, no paper considered the situation of a network shared by critical and non critical flows.

In this paper, we propose an efficient quantum allocation for an AFDX network implementing DRR and shared by a set of critical flow classes and at most one less/non critical flow class.

The rest of the paper is organised as follows. Section 2 summarises the context of the paper, i.e. the network model, DRR main features and worst-case analysis as well as the problem statement. Section 3 presents the proposed quantum assignment algorithm. Sections 4 and 5 illustrate the algorithm on a small example and evaluate it on a realistic case study. Section 6 concludes the paper and gives some direction for future works.

2 CONTEXT

In this section, we present the network model, which considers DRR scheduling, and we state the problem.

2.1 Network Model

In this paper, we consider an avionics switched Ethernet network (AFDX). It is composed of a set of end systems, interconnected by switches via full-duplex links.

End-systems are the source and destination of a set of flows. These flows are statically defined as virtual links (VL). They are forwarded by switches, based on a statically defined forwarding table. The forwarding process introduces a switching latency (sl).

Each flow is shaped at the end system. A minimum interval BAG_i is guaranteed between any two consecutive frames of flow v_i. Each flow v_i is also constrained by a maximum frame length (l_i^{max}) and a minimum frame length (l_i^{min}). The path $\mathcal{P}_i$ followed by a flow v_i in the network is statically defined.

Figure 1 shows an example of an AFDX network. Four switches (S_1 to S_4) interconnect nine end systems (e_1 to e_9). Twenty flows (v_1 to v_{20}) are transmitted on this network. Each switch output port has a set of buffers controlled by a Deficit Round Robin (DRR) scheduler. The maximum link bandwidth is R = 100 Mbits/s. Flow features are given in Table 1.

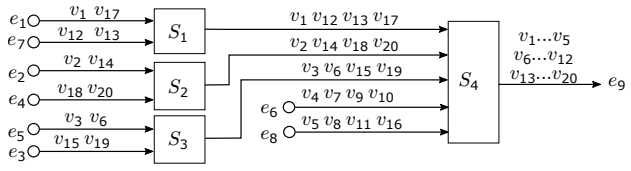

Figure 1: AFDX Network Example

In DRR scheduling, a set of queues is associated to each output port. Each queue belongs to a traffic class C_i. The flow frames are stored in queues corresponding to their traffic class. The DRR scheduler serves the queues in rounds. In a given queue, frames are served following a First Come First Served policy. The flows in Figure 1 are divided into three traffic classes C_1, C_2 and C_3. The classes C_1 and C_2 belongs to critical flows with an upper limit on delay constrains 3100 and 3200 μsec. The class C_3 belongs to non-critical flows with no delay constraints.

Table 1: VL parameters in network example (Figure 1)

Flow	BAG_i ($msec$)	l_i^{max} ($bytes$)	l_i^{min} ($bytes$)	Delay constraint (μsec)	Class
v_1, v_2, v_3	8	1500	800	3100	Critical (C_1)
v_4, v_5	8	990	800		
v_{12}, v_{13}, v_{18}	16	1000	800	3200	Critical (C_2)
v_{20}, v_{15}, v_{19}	16	990	800		
v_{17}, v_{14}, v_6, v_7	32	1000	800	-	Non critical (C_3)
$v_8, v_9, v_{10}, v_{11}, v_{16}$	32	990	800		

2.2 Overview of DRR scheduler

DRR scheduler was designed in [10] to achieve a better quality of service by fair sharing of available network bandwidth among the flows. Basically DRR is a variation of Weighted Round Robin (WRR) which allows fair sharing of bandwidth among variable length flow packets. DRR service is divided into rounds. In each scheduling round, active classes are served sequentially. A class is said to be active when it has some pending frames in its queue. Algorithm 1 shows DRR implementation at a switch output port h serving n traffic classes.

DRR scheduler assigns quantum Q_i to each active class C_i. Q_i represents the number of bytes allocated to C_i in each round. The unused fraction of quantum in a given round is deficit Δ_i for the next round. Thus, the total amount of credit that C_i can get in any round is $Q_i + \Delta_i$. Initially each class deficit is set to 0 (lines 1-3). Each active class queue is served in round robin order (lines 4-16). Empty queues are ignored in each round (line 6). The non-empty queues are credited by their quantum Q_i added to the previous deficit Δ_i (line 7). Selected class can send frames as long as its queue is not empty and the deficit is larger than the head-of-line packet (line 8-12). On each transmission of frame, the credit is reduced by the frame size. If the queue becomes empty, the deficit is reset to 0 (lines 13-14). Then the next class is selected to be served.

Figure 2 illustrates DRR scheduling on the configuration in Figure 1. Each class is assigned a quantum of 1500 bytes. One frame of VLs v_1, v_3, v_5, v_6, v_7, v_{12}, v_{13}, v_{14}, v_{17}, v_{18}, v_{20} with maximum frame size arrive at the output port of S_4 simultaneously and they are arbitrarily buffered in this order. We arbitrarily assume that class C_2 is selected first. Since its initial deficit is 0, it gets a credit of 1500 (its quantum). Frame from v_{12} is first in the queue and its size (1000 bytes) is less than the credit. Thus it is transmitted. Remaining credit is 500, which is not enough for the transmission of next C_2 pending frame (from v_{13}). Therefore, scheduler moves to class C_3 with a credit of 1500. It allows the transmission of v_6 frame, but not v_7 one and the remaining credit is also 500. Then C_1 frame from v_5 is transmitted with a remaining credit of 510. Then C_2 is selected again. It gets a credit of 2000 (500 of deficit plus 1500 of quantum). It allows the transmission of v_{13} and v_{18} frames, leading to a remaining credit of 0, and so on.

Algorithm 1: DRR Algorithm

Input: Per class quantum: $Q_1^h \dots Q_n^h$ (Integer)
Data: Per class deficit: $\Delta_1^h \dots \Delta_n^h$ (Integer)
Data: Counter: i (Integer)

```
for i = 1 to n do
    Δ_i^h ← 0 ;
end
while true do
    for i = 1 to n do
        if isNotEmpty(C_i) then
            Δ_i^h ← Q_i^h + Δ_i^h;
            while (isNotEmpty (C_i)) and (headFrameSize(C_i) ≤ Δ_i^h) do
                send(headFrame(C_i));
                Δ_i^h ← Δ_i^h − headFrameSize(C_i);
                remove(headFrame(C_i));
            end
            if isEmpty(C_i) then
                Δ_i^h ← 0
    end
end
```

2.3 DRR WCTT analysis

A WCTT analysis is mandatory in order to guarantee that critical flows never exceed their deadline. Such a WCTT analysis has been proposed for DRR in [5] and optimised for AFDX in [11]. Both approaches are based on network calculus (NC). In this section, we give a very general overview of these approaches (only the details which are needed for the rest of the paper).

The NC theory is based on the (min, +) algebra [3]. It models the traffic and network elements by piecewise linear curves called arrival curves and service curves respectively.

The arrival curve of a flow represents an over-estimation of the traffic of this flow at any instant t. For a VL v_i at its source end system es_x, it is:

$$\alpha_i^{es_x} = \frac{l_i^{max}}{BAG_i} \times t + l_i^{max} \; for \; t > 0 \; and \; 0 \; otherwise$$

l_i^{max} is the burst (one frame) while $\frac{l_i^{max}}{BAG_i}$ is the long-term rate.

A frame of v_i can experience jitter since it can be delayed by other frames before it arrives at a switch output port h. Thus the arrival curve of v_i in h is obtained by shifting the arrival curve of v_i in es_x to the left by this jitter [2].

A switch output port h with maximum service rate R and switching latency sl is modelled by a service curve:

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

where $[a]^+$ means $max(a, 0)$.

In a DRR scheduler, the bandwidth is shared by all the classes at each output port h. Therefore each class C_x receives a fraction ρ_x^h of R, based on its assigned quantum Q_x^h:

$$\rho_x^h = \frac{Q_x^h}{\sum_{1 \le j \le n} Q_j^h} \times R \tag{1}$$

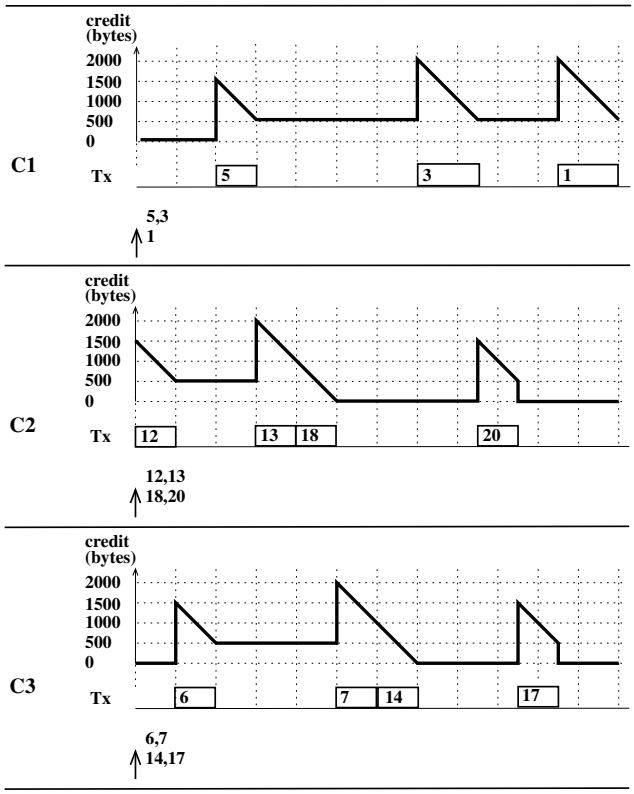

Figure 2: DRR rounds at S_4 output port

Moreover, class C_x experiences the DRR scheduler latency Θ_x^h: it might have to wait before being served for the first time (while the other classes are served at their maximum capacity which is $\sum_{j \neq x} (Q_j^h + \Delta_j^{max,h})$) and, when it is served for the first time, it might get less than its allocated fraction ρ_x^h. This latency Θ_x^h can be illustrated with the example in Figure 2. Let us consider class C_1. Its first frame (from v_5) is served after one frame from C_2 and one frame from C_3, i.e. 160 μs. In the first round, frames from $v_5, v_{13}, v_{18}, v_7, v_{14}$ are served, leading to 5500 transmitted bytes. Thus C_1 gets $\frac{1500}{5500} \times 100 = 27.27$ Mb/s. Based on ρ_x^h, it should have received 33.33 Mb/s.

Therefore the residual service curve for class C_x at port h is given by

$$\beta_x^h(t) = \rho_x^h \times [t - \Theta_x^h - sl]^+ \tag{2}$$

The delay computation for class C_x in each output port h is bounded by the maximum horizontal difference between the overall arrival curve of C_x flows in h (sum of individual arrival curves) and the service curve for class C_x. These curves are illustrated in Figure 3. The service curve is an under-approximation of the actual staircase service curve of C_x. Indeed C_x alternates periods when it get no service and periods when it gets full service at rate R. Such a staircase curve does not fit within NC, since it is not convex. The considered under-approximation is convex.

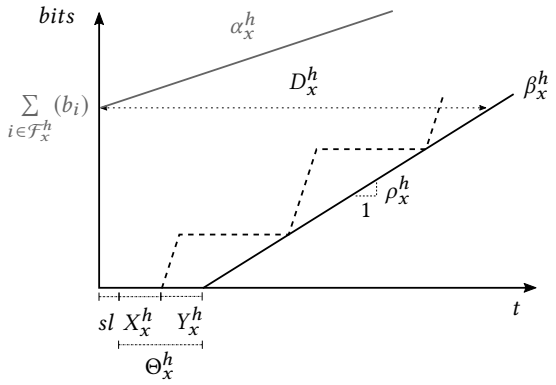

Figure 3: Arrival and Service Curves

The basic approach in [5] implement this delay computation. It does not make any assumption on the volume of competing traffic from other classes. It comes to consider that competing classes are always active. It might not be the case. Thus the approach in [11] optimises the one in [5] by quantifying the maximum amount of traffic of each competing class within the delay bound computed by [5].

2.4 Problem statement

Our goal is to allocate the quanta on an AFDX network implementing DRR scheduling so that:

- every critical frame respects its deadline,
- the delay of non critical frames is minimised.

Therefore the goal is to assign as few bandwidth as possible to critical classes in order to maximise the bandwidth assigned to the non critical class. As previously mentioned, we assume a set of $n-1$ critical classes $C_1 \dots C_{n-1}$ and one single non critical one C_n. We also assume that a given class is assigned the same quantum in all switch output ports. The quantum Q_x assigned to a class C_x has to allow the transmission of any frame of class C_x. Therefore, we must have

$$Q_x \geq Max_x \; for \; 1 \leq x \leq n \tag{3}$$

where Max_x is the largest frame size among class C_x frames.

If $\mathcal{V}_x$ is the set of class C_x VLs, we have

$$Max_x = \max_{v_j \in \mathcal{V}_x} l_j^{max} \tag{4}$$

In the next section we propose an algorithm that solves this quantum allocation problem. This algorithm assumes the basic WCTT computation in [5].

3 QUANTUM ASSIGNMENT

Since the quantum of a given class is the same in all switch ports, we omit node h in notations in the rest of the paper.

Let us define Q as the sum of quanta allocated to classes, i.e.

$$Q = \sum_{1 \leq i \leq n} Q_i \tag{5}$$

If we focus on one class C_x, it receives a quantum Q_x while the other classes shares $Q - Q_x$. First, we show that the worst-case end-to-end latency of a given class C_x depends on the quantum Q_x allocated to C_x as well as on $Q - Q_x$, but it does not depend on the distribution of $Q - Q_x$ among the other classes.

Lemma 3.1. *Given Q_x and Q, the worst-case delay for C_x computed by network calculus approach in [5] is the same for any quantum distribution among competing classes.*

Proof. The quanta only impact the service curve for class C_x in each switch output port. Thus we show that this service curve depends on Q_x and Q, but not on quantum distribution. The service curve for class C_x in a node is defined by equation 2. It depends on the fraction ρ_x of bandwidth allocated to C_x and the DRR scheduler latency Θ_x.

ρ_x is defined by equation 1, which can be rewritten

$$\rho_x = \frac{Q_x}{Q} \times R \tag{6}$$

Thus ρ_x only depends on Q_x and Q.

Θ_x is defined in [5] as the sum of two delays X_x and Y_x, where,

$$X_x = \frac{\sum_{1 \leq j \leq n, j \neq x} (Q_j + \Delta_j^{max})}{R} \tag{7}$$

$$Y_x = \frac{Q_x - \Delta_x^{max} + \sum_{1 \leq j \leq n, j \neq x} Q_j}{R} - \frac{Q_x - \Delta_x^{max}}{\frac{Q_x}{Q} \times R} \tag{8}$$

where Δ_i^{max} is the maximum deficit for class C_i, i.e. $Max_i - 1$.

Thus, we have

$$\begin{aligned}
\Theta_x &= X_x + Y_x \\
&= 2 \times \frac{\sum_{\substack{1 \leq j \leq n \\ j \neq x}} Q_j}{R} + \frac{\sum_{\substack{1 \leq j \leq n \\ j \neq x}} \Delta_j^{max}}{R} + \frac{Q_x - \Delta_x^{max}}{R} \left(1 - \frac{Q}{Q_x}\right) \\
&= 2 \times \frac{\sum_{\substack{1 \leq j \leq n \\ j \neq x}} Q_j}{R} + \frac{\sum_{\substack{1 \leq j \leq n \\ j \neq x}} \Delta_j^{max}}{R} + \frac{Q_x - \Delta_x^{max}}{R} \left(\frac{Q_x - Q}{Q_x}\right) \\
&= 2 \times \frac{\sum_{\substack{1 \leq j \leq n \\ j \neq x}} Q_j}{R} + \frac{\sum_{\substack{1 \leq j \leq n \\ j \neq x}} \Delta_j^{max}}{R} + \frac{Q_x - \Delta_x^{max}}{R} \left(-\frac{\sum_{\substack{1 \leq j \leq n \\ j \neq x}} Q_j}{Q_x}\right) \\
&\left[since \sum_{\substack{1 \leq j \leq n \\ j \neq x}} Q_j = Q - Q_x\right] \\
&= \frac{\sum_{\substack{1 \leq j \leq n \\ j \neq x}} Q_j}{R} + \frac{\sum_{\substack{1 \leq j \leq n \\ j \neq x}} \Delta_j^{max}}{R} + \frac{\Delta_x^{max}}{R} \left(\frac{\sum_{\substack{1 \leq j \leq n \\ j \neq x}} Q_j}{Q_x}\right)
\end{aligned}$$

Therefore Θ_x depends on

- the link rate R and the maximum deficits Δ_i^{max} for each class, which does not depend on quanta,
- the quantum Q_x assigned to class C_x under study,

- the sum of the quanta assigned to competing classes $\sum_{1 \le j \le n, j \ne x} Q_j$, but not the individual Q_j values.

□

Based on lemma 3.1 the quantum assignment can be done in the following manner.

- the sum Q of quanta allocated to classes is fixed,
- we calculate the minimum portion of Q that has to be allocated to C_1 so that all the VLs in C_1 respect their deadlines (we do not need to know the quanta that will be assigned to other classes),
- we do the same for $C_2 \dots C_n$.

Q value has to be large enough to lead to the valid values for each quantum (equation 3). This can be illustrated with the example in Figure 1. Let us consider three cases:

case 1: If Q = 4000, on computing the delay bound with NC approach we observe that the minimum quantum required by C_1 and C_2 to respect their delay constraints are 1282 and 1119 bytes and the residual quantum for C_3 is 1599 bytes (which is 34.52 % of Q). However, this is not a valid case since the quantum assigned to C_1 should be at least 1500 bytes (equation 3).

case 2: Now, let us assume a higher value of Q = 6000. In this case, the minimum quantum required by C_1 and C_2 are 2356 and 1995 bytes. The residual quantum for C_3 is 1649 bytes (27.48%). The percentage share of quantum is clearly reduced as compared to the previous case. In the next case, let us try a value of Q somewhere between that in case 1 and case 2.

case 3: Let Q = 4500 bytes. In this case, the minimum quantum required by C_1 and C_2 are 1507 and 1322 bytes, which gives the residual quantum for C_3 as 1671 bytes, thus, the percentage share of quantum for C_3 is increased to 37.13%.

This small example shows that the ratio between computed quanta depends on Q. Since one goal is to minimise the response time for non critical flows (class C_n), we want to get the valid Q value that maximises the percentage of Q which is not assigned to critical classes. Next lemma states that the best Q value is the smallest valid one.

Lemma 3.2. *When each critical class is assigned the minimum quantum which guarantees that no deadline is missed, based on the WCTT analysis in [5], a smaller valid value of Q never leads to a smaller percentage of Q assigned to the non critical class C_n.*

Proof. From equation 5, we have

$$Q_n = Q - \sum_{1 \le j < n} Q_j \tag{9}$$

Therefore the percentage of Q assigned to C_n is

$$1 - \sum_{1 \le j < n} \frac{Q_j}{Q} \tag{10}$$

Let us consider two valid Q values Q' and Q'' with $Q'' = \sigma \times Q'$ with $\sigma > 1$. We have to show that the percentage of Q' assigned to C_n cannot be smaller than the percentage of Q'' assigned to C_n. Therefore we have to show that

$$\sum_{1 \le j < n} \frac{Q'_j}{Q'} \le \sum_{1 \le j < n} \frac{Q''_j}{Q''} \tag{11}$$

To show that this inequality is true, we show that

$$\frac{Q'_j}{Q'} \le \frac{Q''_j}{\sigma \times Q'} \text{ for } 1 \le j < n \tag{12}$$

Let us consider one class C_x ($1 \le x < n$). When the sum of quanta is Q', it gets a quantum of Q'_x, which is the minimum quantum it needs to meet all its deadlines. When the sum of quanta is $Q'' = \sigma \times Q'$, we have to show that

$$Q''_x \ge \sigma \times Q'_x \tag{13}$$

The worst-case delay of a C_x frame is the maximum horizontal distance between C_x traffic curve and C_x service curve. C_x traffic curve does not depend on its assigned quantum. The distance clearly decreases when Q''_x increases. Thus inequality 13 is true if moving from a quantum of Q'_x out of Q' to a quantum of $\sigma \times Q'_x$ out of $\sigma \times Q'$ does not lead to a higher service curve for C_x.

The service curve for class C_x in a node is defined by equation 2. It depends on the fraction ρ_x of bandwidth allocated to C_x and the DRR scheduler latency Θ_x.

ρ_x is defined by equation 1. Considering Q' and Q'' we have

$$\rho'_x = \frac{Q'_x}{Q'} \times R \text{ and } \rho''_x = \frac{\sigma \times Q'_x}{\sigma \times Q'} \times R$$

Thus $\rho'_x = \rho''_x$

Concerning DRR scheduling latency Θ_x, we have shown in lemma 3.1 proof that

$$\Theta'_x = \frac{\sum_{1 \le j \le n, j \ne x} Q'_j}{R} + \frac{\sum_{1 \le j \le n, j \ne x} \Delta_j^{max}}{R} + \frac{\Delta_x^{max}}{R}\left(\frac{\sum_{1 \le j \le n, j \ne x} Q'_j}{Q'_x}\right)$$

Since $Q'' = \sigma \times Q'$ and $Q''_x = \sigma \times Q'_x$, we have

$$\begin{aligned}\Theta''_x &= \frac{\sum_{1 \le j \le n, j \ne x} \sigma \times Q'_j}{R} + \frac{\sum_{1 \le j \le n, j \ne x} \Delta_j^{max}}{R} + \\ &\quad \frac{\Delta_x^{max}}{R}\left(\frac{\sum_{1 \le j \le n, j \ne x} \sigma \times Q'_j}{\sigma \times Q'_x}\right) \\ &= \sigma \times \frac{\sum_{1 \le j \le n, j \ne x} Q'_j}{R} + \\ &\quad \frac{\sum_{1 \le j \le n, j \ne x} \Delta_j^{max}}{R} + \frac{\Delta_x^{max}}{R}\left(\frac{\sum_{1 \le j \le n, j \ne x} Q'_j}{Q'_x}\right)\end{aligned}$$

Since $\sigma > 1$, we have $\Theta''_x > \Theta'_x$. Thus the service curve for C_x when considering Q'' and Q''_x is under the service curve for C_x when considering Q'_x and Q'. This phenomenon is illustrated in Figure 4. When σ increases, the service curve is shifted to the right, leading to a higher horizontal distance with traffic curve and, consequently, a higher worst-case delay. □

Based on lemmas 3.1 and 3.2, the quantum assignment is implemented by algorithm 2. The basic idea of the algorithm is to start from an initial value *ValInit* for the sum of quanta Q (line 1). It computes for each critical class C_i the minimum portion Q_i of Q that insures that no C_i frame misses its deadline (line 9, function *QuantumMin*). The process stops as soon as the unused portion *QResid* of Q is smaller than the computed Q_i value (line 10-11) or

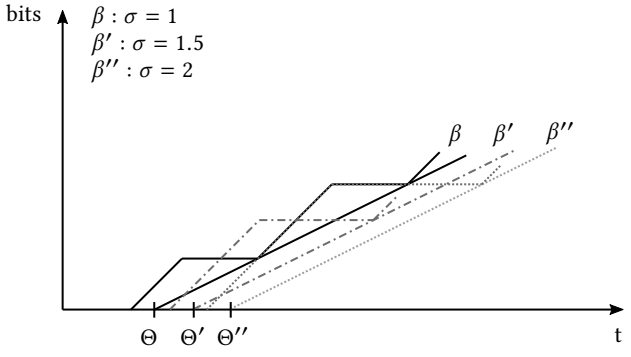

Figure 4: Service curves

all the critical classes have been treated. In the later case, the non critical class gets a quantum of *QResid* and a solution has been obtained. This solution might be either not valid if condition 3 is not respected (at least one class has a quantum which is smaller than its maximum frame size) or improvable if the quantum of every class exceeds its maximum frame size by more than a configured (small) value ϵ: we tolerate an ϵ difference, since it might be tricky to reach a valid solution where at least one class has its maximum frame size as quantum. In both cases (not valid or improvable solution), we adapt the sum of quanta Q (lines 25-26): we increase it if the solution is not valid, we reduce it if the solution is improvable. As soon as a non improvable valid solution is found, the algorithm returns it and stops.

QuantumMin function (line 9) implements a binary search. It applies the WCTT analysis in [5] on values not greater than Q, until it gets the smallest value with no deadline miss.

In the next section, we illustrate Algorithm 2 on the example in Figure 1 and give some more details on the computation.

4 QUANTUM ASSIGNMENT EXAMPLE

The flows are divided into $n = 3$ classes based on their delay constrains. C_1 and C_2 are critical flow classes with delay constraint 3100 μsec and 3200 μsec, respectively, and C_3 is non-critical flow class with no constraint on delay (see Table 1). The maximum service rate at each output port is $R = 100$ Mb/s.

Initially, we assume a sum of quanta $Q = ValInit = 4550$ bytes (line 1). First critical class is C_1 (line 4). For the given sum of quanta, the minimum quantum value Q_i (i = 1 and 2) for each class is computed (line 8–18). The function *QuantumMin(i,Q)* computes the minimum quantum required to respect the deadline in class C_i given the quantum sum Q. This computation is based on the maximum end-to-end delay in the class, obtained thanks to the WCTT analysis in [5].

The whole process is shown in Table 2 for each class of the network in Figure 1. For each class, the process is based on a binary search. In the first step, the maximum quantum is assigned to Q_1 (4550 in our case). With this quantum, the worst-case end-to-end delay for C_1 obtained by NC approach in [5] is 1007.87 μsec. Since, this computed delay is less than the delay constraint on C_1 (3100

Algorithm 2: Quantum assignment algorithm

input : Initial sum of quanta: ValInit (Integer)
input : Per class maximum frame size: $Max_1 \ldots Max_n$ (Integer)
input : Improvement level: ϵ (Float)
output: Valid solution found: valid (Boolean)
output: Per class quantum: $Q_1 \ldots Q_n$ (Integer)
Data: Remaining quantum: *QResid*
Data: Relative distance to a non improvable solution: *min* (Float)
Data: End of the process: *Fini* (boolean)
Data: Counter: i (Integer)

```
1  Q ← ValInit ;
2  Fini ← false ;
3  while not Fini do
4      i ← 1 ;
5      QResid ← Q ;
6      valid ← true ;
7      min ← Q ;
8      while i < n and valid do
9          Q_i ← QuantumMin(i, Q) ;
10         if Q_i > QResid then
11             valid ← false ;
12         else
13             QResid ← QResid − Q_i ;
14             if min > Q_i / Max_i then
15                 min ← Q_i / Max_i ;
16             i ← i + 1 ;
17         end
18     end
19     if not valid then
20         Fini ← true ;
21     else
22         Q_n ← QResid ;
23         if min > Q_n / Max_n then
24             min ← Q_n / Max_n ;
25         if min < 1 or min > 1 + ε then
26             Q ← Q / min ;
27         else
28             Fini ← true ;
29         end
30     end
31 end
32 if valid then
33     The non improvable solution is reached ;
34 else
35     No valid solution was found ;
36 end
```

μsec), Q_1 can be reduced. It will increase the worst-case delay for C_1 flows.

The delay computation in NC is based on the convergence of overall arrival curve and the service curve. Thus, the service rate ρ_x must be more than the arrival rate $r_x = \sum_{vi \in C_x} \frac{l_i^{max}}{BAG_i}$ of the cumulative flows in C_x. In C_1, the maximum traffic arrival rate is observed at output port of switch S_4 as 6.48 $bits/\mu sec$. If the minimum bandwidth provided to C_1 is 7.48 $bits/\mu sec$, the corresponding quantum value is 340 bytes.

The next value assumed in binary search is the mid value between 4550 and 340, thus, Q_1 = 2445. The corresponding maximum end-to-end delay is 2138.72 μsec. Then Q_1 can be further decreased, since, the computed delay is less than the delay constraint on C_1. This process is repeated until the computed delay is close to the C_1 delay constraint (and smaller). The value returned by *QuantumMin(C_1,Q)* is Q_1 = 1524 bytes.

The residual quantum is *QResid* = 4550 - 1524 = 3026 bytes (line 13 in Algorithm 2) which can be distributed among the other classes. *min* is the smallest factor which is obtained from the ratio of the quantum assigned to a class and its maximum frame length. This factor will be used to optimise the quantum sum (line 26). From class C_1, *min* = 1.016 (line 14–16).

Next, critical class is C_2. The minimum quantum obtained from *QuantumMin(C_2, Q)* is Q_2 = 1337 bytes (see Table 2). Thus, the residual quantum is *QResid* = 3026 - 1337 = 1689 bytes. Since, the minimum quantum is successfully computed for critical classes, next class is non-critical class C_3 (line 21–30). The residual quantum can be assigned to C_3 (line 22).

Based on Lemma 3.2, the goal is to find the smallest valid value of Q, which we call a non improvable value.

The factor *min* represents the convergence towards a non improvable value of sum Q of quanta. If $min < 1$, then Q is lower than the expected value and if $min > 1$, then Q is higher than the expected value. Thus, the non improvable sum of quanta can be obtained by the $\frac{Q}{min}$ = 4479 (line 26). The distribution of this new sum of quanta among each class can be done by repeating the whole process with Q = 4479. The corresponding values of quantum for each class is given in Table 2.

Table 2: Quantum assignment in the network (Figure 1).

(bytes)				(μsec)		
Q	Q_1	Q_2	Q_3	max $D_{C_1}^{E2E}$	max $D_{C_2}^{E2E}$	max $D_{C_3}^{E2E}$
4550	4550	-	-	1007.87	-	-
	2445	-	-	2138.72	-	-
	1393	-	-	3290.62	-	-
	1919	-	-	2614.62	-	-
	1656	-	-	2911.66	-	-
	1524	-	-	3088.99	-	-
		3026	-		1460.72	-
		1604	-		2746.07	-
		893	-		4236.5	-
		1248	-		3308.81	-
		1426	-		2998.9	-
		1337	-		3145.28	-
			1689			3075.09
4479	4479	-	-	1007.87	-	-
	2407	-	-	2133.06	-	-
	1371	-	-	3274.21	-	-
	1889	-	-	2600.87	-	-
	1630	-	-	2896.67	-	-
	1500	-	-	3073.2	-	-
		2979	-		1456.72	-
		1579	-		2730.81	-
		879	-		4217.1	-
		1229	-		3290.67	-
		1404	-		2982.32	-
		1316	-		3128.82	-
			1663			3060.18

5 EVALUATION

In this section, we evaluate the proposed quantum assignment algorithm on an industrial-size network.

First, we perform a WCTT analysis using Network Calculus on the original industrial configuration assuming that each switch output port is controlled by a FIFO scheduler.

Then, we introduce some additional non-critical flows to this configuration and analyse the impact on the worst-case end-to-end delays of the critical flows, still considering FIFO scheduling.

Next, we assume a DRR scheduler at each switch output port. We show how the constraints on end-to-end delays for critical flows can be guaranteed while minimising the delay for non-critical flows with DRR scheduling in presence of optimised quantum values.

5.1 Industrial Case Study

We consider an industrial-size configuration. It includes 96 end-systems, 8 switches, 984 flows, and 6276 paths (multi-cast VL) [11]. We arbitrarily distribute flows between 2 critical classes: a class C_1 for flows with small BAGs (up to 16 ms) and a class C_2 for flows with larger BAGs. Table 3 summarises the features of these classes.

Table 3: VL parameters in industrial configuration

Flow count	*BAG* Range ($msec$)	Frame length range ($bytes$)	Class
280 (1681 paths)	2–16	84–1497	Critical C_1
704 (4595 paths)	32–128	84–1535	Less-Critical C_2
Additional flows			
40 (120 paths)	4	84–963	Best-Effort C_3

In the first step, this network is assumed to use FIFO scheduling in switch output ports (classes are not considered). Worst-case

delays are computed for all the flows, using the classical network calculus approach for FIFO [2]. Obtained results are shown in Figure 5. In Figure 5, each unit on x-axis represents a flow path and y-axis represents the worst-case end-to-end delay corresponding to this path. The paths are sorted in increasing order of delays in each class.

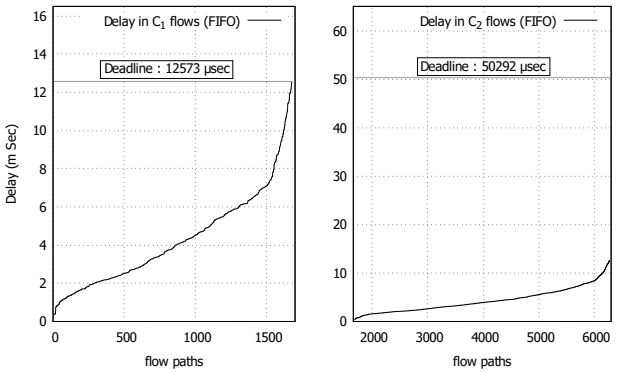

Figure 5: End-to-end delay bounds in C_1 and C_2 flows under FIFO scheduling

The maximum delays computed in the network is 12572.6 μsec. Therefore flow deadlines are met. Indeed deadlines for C_1 flows are 12573 μs, while deadlines for class C_2 flows are 50292 μs.

Next, we arbitrarily introduce some additional flows into this configuration. Actually, we consider a set of 40 C_1 like flows. The idea is to interfere with the critical flows and evaluate the impact on worst-case end-to-end delay in the network. The added flows are considered as non-critical (best-effort) flows characterised by class C_3 shown in Table 3.

The impact of additional flows on the critical ones is shown in Figure 6. Not surprisingly, the delay is increased for critical flows as they share waiting queues with non-critical flows. Moreover, there are 9 flows of C_1 which exceed their deadlines. The maximum delay in C_2 is still much lower than the delay constraint.

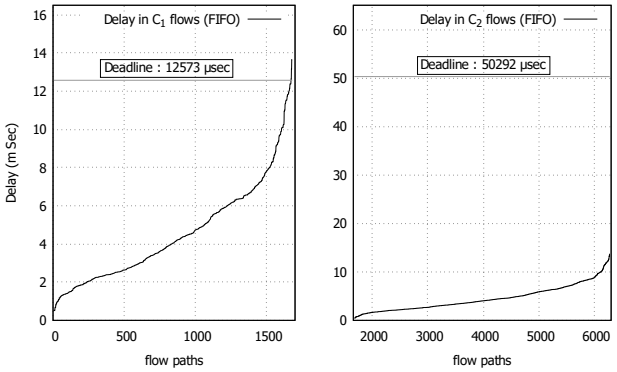

Figure 6: End-to-end delay bounds in C_1 and C_2 flows under FIFO scheduling in presence of additional flows

Now, we consider DRR scheduling at each switch output port in the given network. The DRR scheduler configuration is given in Table 4. The quantum for each class is optimised using the Algorithm 2 such that no critical flow (in C_1 and C_2) misses its deadline while maximising the percentage of bandwidth assigned to non-critical flows (C_3).

Table 4: DRR configuration

Class	Delay constraints (μsec)	Optimized Quantum (bytes)
Critical C_1	12573	3581 (Bandwidth = 56.02%)
Less-Critical C_2	4×12573	1535 (Bandwidth = 24.01%)
Best-effort C_3	-	1276 (Bandwidth = 19.97%)

Based on WCTT analysis in [5], worst-case end-to-end delays are shown in Figure 7. Quantum assignment insures that critical flows are within their expected constraints and reducing one quantum for a critical class would lead to exceeded deadlines.

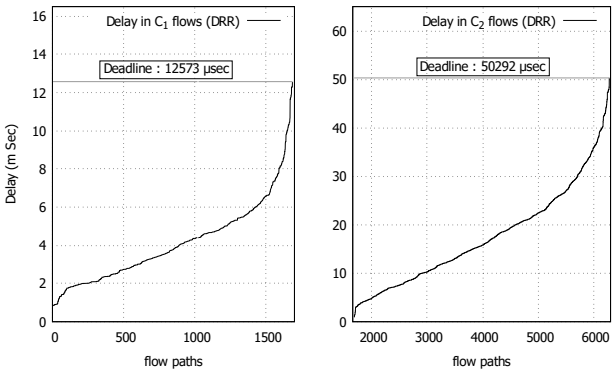

Figure 7: End-to-end delay bounds in C_1 and C_2 flows under DRR scheduling in presence of additional flows

Finally we compare the delays observed on the additional flows (C_3) in both cases (FIFO and DRR). Worst-case delays for each flow are shown in Figure 8 and summarised in Table 5. On this specific example, DRR globally leads to smaller worst-case delays for C_3 flows than FIFO. This is due to the fact that DRR reserves bandwidth to these flows while, with FIFO, they are transmitted only when there are no pending critical frames. It would be interesting to evaluate average delays for C_3 flows, for instance by simulation. We can guess that DRR would still be better than FIFO.

5.2 Discussion

The NC approach for WCTT analysis used in this paper (from [5]) can be very pessimistic as it assumes that the competing classes are always active and the traffic from each class is maximum. This

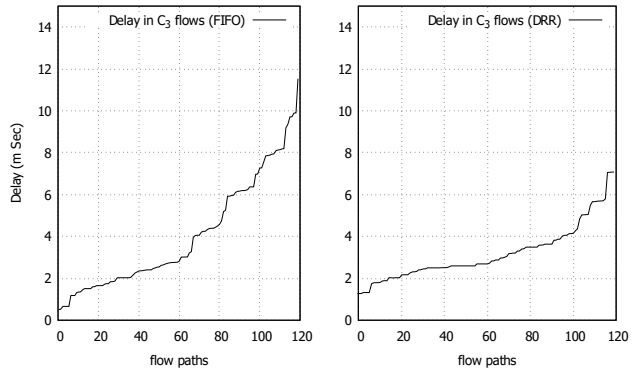

Figure 8: Comparison of end-to-end delay bounds in C_3 flows under FIFO and DRR scheduling

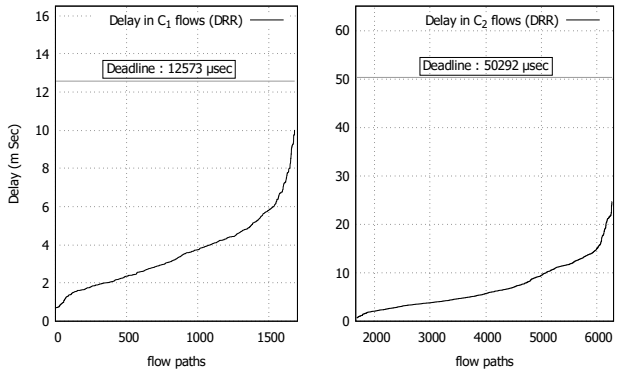

Figure 9: End-to-end delay bounds in C_1 and C_2 flows under DRR scheduling in presence of additional flows (computed by optimised NC approach)

Table 5: Performance comparison: FIFO Vs DRR scheduling

	Max delay (μsec)			Deadlines missed
	C_1	C_2	C_3	
FIFO	12572.6	12572.6	-	
FIFO (with added flows)	13659.9	11535.4	11535.4	9
DRR (with added flows)	12549.2	50215.7	7082.42	0
Deadlines	12573	50292	-	

pessimism also affects the optimised quantum computed by the algorithm presented in this paper, since it over-estimate worst-case delay.

The problem of pessimism in [5] was addressed in [11] by quantifying the maximum amount of traffic of each competing class within the delay bound computed by [5]. However, the optimised NC approach in [11] cannot be used with the quantum assignment algorithm presented in this paper as it does not respect Lemma 3.1. Indeed the optimisation in [11] depends on the quantum allocated to each competing class. Therefore the resulting WCTT analysis depends, not only on the sum of the quanta of competing classes, but also on each quantum value. Thus the algorithm proposed in this paper cannot be used.

The delay bounds computed in this paper, using [5], can be pessimistic which means that the actual delays are much lower. We evaluate the difference between both bounds on the case study. We take the optimised quanta computed earlier (Table 4) and use these quanta values in the optimised NC approach given in [11]. The results are shown in Figure 9. The maximum delay in C_1 and C_2 are 10016.9 and 24669.2 μsec respectively, which are clearly much less than what was computed by the pessimistic approach. It means that we can increase the percentage of bandwidth assigned to non critical flows without compromising the constraints of critical ones.

6 CONCLUSION

In this paper we show how the available bandwidth of an AFDX network can be efficiently shared between critical avionics flows and non critical ones. We consider a Deficit Round Robin scheduling in switch output ports. We assume a set of critical flow classes and one non critical flow class. We propose an algorithm that assign the minimum quanta to critical classes that insure that no deadlines will be missed. Thus it maximises the percentage of bandwidth assigned to non critical flows. We show that such a strategy leads to smaller worst-case delays for non critical flows. It would be interesting to show that the trend is the same for average delays. This could be done by simulation. The other advantage of the proposed solution is to take into account the different deadlines of critical flows.

In the proposed approach, the WCTT analysis in [5] is used. It is known to be potentially very pessimistic and a much less pessimistic solution has been proposed in [11]. This optimised solution is not compatible with the algorithm proposed in this paper. Therefore one future work is to adapt the algorithm to the optimised WCTT analysis. Our intuition is that it might significantly increase the algorithm complexity.

Other scheduling policies are envisioned for QoS-aware AFDX networks, such as Weighted Round Robin [12]. Another future work is to develop a similar assignment algorithm for this scheduling policy.

REFERENCES

[1] 2002 - 2005. *Aircraft Data Network, Parts 1,2,7 Aeronotical Radio Inc.* Technical Report. ARINC Specification 664.
[2] Henri Bauer, Jean-Luc Scharbarg, 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).
[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 Miggie, 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 Aeronautic and Space Sciences (EUCASS)* (2013).
[5] Marc Boyer, Giovanni Stea, and William Mangoua Sofack. 2012. Deficit Round Robin with Network Calculus. *Performance Evaluation Methodologies and Tools (VALUETOOLS), 2012 6th International Conference on (pp. 138-147). IEEE* (October 2012), 10.

[6] Yu Hua and Xue Liu. 2011. Scheduling design and analysis for end-to-end heterogeneous flows in an avionics network. *2011 Proceedings IEEE INFOCOM* (April 2011).
[7] S. Salil Kanhere and Harish Sethu. 2002. On the latency bound of Deficit Round Robin. *Computer Communications and Networks, 2002. Proceedings. Eleventh International Conference on. IEEE, 2002. p. 548-553.* (October 2002), 7.
[8] Anton Kos and Saso Tomazic. 2009. A More Precise Latency Bound of Deficit Round-Robin Scheduler. *Elektrotehnivski vestnik* 76 (January 2009), 257–262.
[9] Sho Noda and Katsunori Yamaoka. 2016. Approach to optimal WRR weight assignment method in delay-limited environment. In *13th IEEE Annual Consumer Communications Networking Conference (CCNC).* https://doi.org/10.1109/CCNC.2016.7444945
[10] Madhavapeddi Shreedhar and George Varghese. 1996. Efficient fair queuing using deficit round-robin. *IEEE/ACM Transactions on networking, 1996, vol. 4, no 3, p. 375-385* (1996), 11.
[11] Aakash Soni, Xiaoting Li, Jean-Luc Scharbarg, and Christian Fraboul. 2018. Optimizing Network Calculus for Switched Ethernet Network with Deficit Round Robin. *IEEE Real-Time Systems Symposium (RTSS)* (2018).
[12] Aakash Soni, Xiaoting Li, Jean-Luc Scharbarg, and Christian Fraboul. 2018. WCTT analysis of avionics Switched Ethernet Network with WRR Scheduling. *26th International Conference on Real-Time Networks and Systems (RTNS)* (2018).