Determinism Enhancement of AFDX Networks via Frame Insertion and Sub-Virtual Link Aggregation

Meng Li, Michaël Lauer, Guchuan Zhu, Yvon Savaria

Determinism Enhancement of AFDX Networks via Frame Insertion and Sub-Virtual Link Aggregation

Meng Li, Michaël Lauer, Guchuan Zhu, and Yvon Savaria

***Abstract*—AFDX is a standard proposed to implement deterministic networks by providing predictable performance guarantees. The determinism is enforced through the concept of Virtual Link, which defines a logical unidirectional connection between End Systems. Although an upper bounded end-to-end delay can be obtained by using analysis based on, e.g., Network Calculus, frame arrival uncertainty in destination End-System is a source of non-determinism that introduces a problem with respect to real-time fault detection. In this paper a mechanism based on frame insertion is proposed to enhance the determinism of frame arrival within AFDX networks. In order to mitigate network load increase due to frame insertion, a Sub-Virtual Link aggregation strategy, formulated as a multi-objective optimization problem, is introduced. In addition, a brute force algorithm, a greedy algorithm, and a greedy algorithm with pre-processing have been developed to find solutions to the optimization problem. Experiments are carried out and the reported results confirm the validity and applicability of the developed approaches.**

***Index Terms*—AFDX Networks, Determinism, Sub-Virtual Link Aggregation, Optimization.**

I. Introduction

AVIONICS Full Duplex Switched Ethernet (AFDX) has been proposed to meet increasing requirements of high speed, high reliability, and low cost avionics communication systems. This technology is standardized in ARINC 664 Part 7 [1] and is deployed in many current and future aircrafts such as Airbus A380, A350, A400M, Boeing B787, Comac ARJ21, and Bombardier CS100.

AFDX is a specialization of Ethernet whose purpose is to provide a more deterministic network with predictable performance guarantees. This determinism is enforced mainly through the concept of Virtual Link (VL), inspired by the concept of asynchronous transfer mode (ATM). As stated in the standard, a VL is a conceptual communication link, which defines: (1) a logical unidirectional connection from one source End-System (ES) to one or more destination ESs; (2) a maximum bandwidth allocated to this connection. Essentially, two mechanisms are used to ensure that the bounded data transmission bandwidth is respected. At the ingress of the network, i.e. end-systems, traffic shaping is used to control the flow for each VL in accordance with the so-called Bandwidth Allocation Gap (BAG), which defines the minimum time interval between successive frames in a VL. In the switches, traffic policing is used to protect the network from babbling-idiot failures. Furthermore, as the routes of the VLs are statically defined off-line, the network offers a consistent performance guarantee. In addition, AFDX is composed of two independent and redundant networks, which provides the high reliability required for ensuring its determinism.

AFDX networks aim at providing a guaranteed service with a firm, mathematically provable, upper bound on end-to-end frame transit delay. Hence the end-to-end delay analysis is considered as a pivotal issue among the mandatory certifications. Much work has been dedicated to evaluate the delay upper bounds. The theoretical methods, including network calculus [2]–[6], trajectory approach [7]–[11] and response time analysis [12] are applied to the worst-case transmission delay analysis. Scheduling schemes for ESs and switches are proposed to improve the end-to-end delay [13]–[16]. Furthermore, simulation and modeling approaches are implemented to evaluate end-to-end delays and to provide experimental upper bounds [17]–[20]. With the upper bounded delay, the minimum interval between successive frames in destination ES becomes deterministic.

Nevertheless, there still exist some sources of non-determinism in AFDX networks. First, being an asynchronous protocol, a global time cannot be defined or used throughout the network. Note that the asynchronism is a feature of this network, which has been chosen in order to provide robustness in communications and to facilitate the design of applications using the network. A second source of non-determinism is related to fault detection in the destination ESs. Indeed, the AFDX standard does not force a VL to transmit frames if there is no data to transmit, even though the VL is available. This means that destination ESs cannot detect one or several consecutive frame losses (due to frame corruptions or device malfunctions on both redundant networks) until a valid frame arrives. For safety-critical applications, this raises a serious issue in terms of determinism and reliability. The motivation of this paper is then to enhance the determinism of AFDX networks by proposing a solution to frame arrival uncertainty.

The proposed solution is based on the idea of inserting filler frames in a VL when its source is silent. This allows destination ESs of the VL to detect a fault if a frame is missing from the periodical pattern obtained with filler frames. Obviously, this mechanism does not affect the maximum bandwidth reserved for a VL and the worst-case performance of a regulated VL. However, inserting filler frames will increase network load and the average bandwidth used by a VL. In order to mitigate the impact on the overall network performance, we leverage a feature described in the AFDX standard, namely Sub-Virtual Link (Sub-VL) aggregation. We show that Sub-VLs aggregation in source ESs allows optimizing the bandwidth utilization of VLs. A Sub-VL aggregation strategy, formulated as a multi-objective optimization problem aimed at minimizing the overhead due to filler frame insertion and the delay introduced by Sub-VL aggregation, is then presented. It is worth noting that the proposed formulation can be applied to the generic Sub-VL aggregation problem in AFDX network design and to the extent of our knowledge, little work is dedicated to the optimization of Sub-VL aggregation. The

viability and the applicability of the proposed strategy are demonstrated through numerical simulations. Several algorithms used to reach or approach an optimal solution are developed, including a brute force algorithm (an exhaustive search), a greedy algorithm, and a greedy algorithm with preprocessing. Note that in this paper, the impact on bandwidth due to filler frame insertion and its optimization with Sub-VL aggregation are only considered at the source ES level. The impact on the global network will be studied in future work. It is proposed in [21] to aggregate messages into super-messages in source operating system (OS) partitions defined in the ARINC 653 standard to minimize bandwidth consumption. However, the proposed aggregation is not related to optimizing frame insertion. Moreover, the problem we consider is tackled in a network layer where message aggregation cannot be performed.

The contributions of this paper are:

- a mechanism based on frame insertion that enables real-time fault detection in destination ESs, thus enhancing the determinism of the network;
- a Sub-VL aggregation strategy that mitigates the network load increase due to frame insertion while simultaneously minimizing the delay introduced by Sub-VL aggregation.

The remaining of the paper is organized as follows. Section II describes issues related to Sub-VL aggregation in AFDX networks and the non-determinism in VL transmission. Section III presents a mechanism for determinism enhancement in AFDX networks. Then, in Section IV the problem of Sub-VL aggregation is formulated and effective algorithms for resolving the corresponding multi-objective optimization problem are developed. In Section V, experimentations are carried out to validate the feasibility of the proposed mechanism and to evaluate the obtained performance. Finally, some concluding remarks and directions for future research are provided in Section VI.

II. Sub-VL Aggregation and Non-Determinism in VL Transmission

We present in this section the mechanism for Sub-VL aggregation in AFDX networks and formulate the delay due to this operation. The non-determinism issue in VL transmission will be discussed, which leads to a suggestion for determinism enhancement.

A. Sub-VL Aggregation

One of the main objectives of Sub-VL aggregation is to improve the bandwidth utilization efficiency. According to the ARINC 664-part 7 standard, a VL can be composed of one or up to four Sub-VLs. Each Sub-VL has a dedicated First-In, First-Out (FIFO) queue. The Sub-VL FIFO queues are read out on a round-robin (RR) basis, as shown in Fig. 1, by the VL FIFO queue [1]. After aggregation, the frames are sent according to the BAG of the VL.

Essentially, a Sub-VL can be dedicated to a source flow from OS partitions, which can be periodic or sporadic. In either case, to allocate bandwidth for each VL, represented by the BAG, the system integrator must set the following two parameters:

- $l^{\max}$: the maximum frame size (MFS) of the source flow;
- T: the minimum time interval between two consecutive frames.

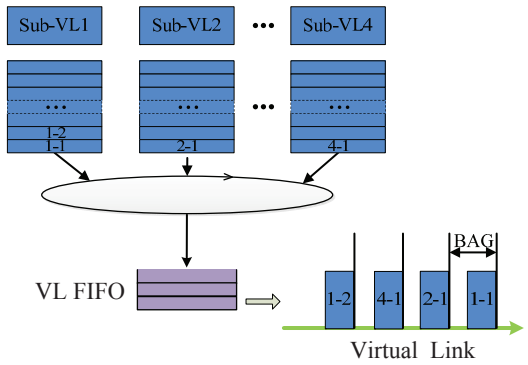


Fig. 1: Sub-VL aggregation mechanism.

To illustrate how Sub-VL aggregation may optimize bandwidth utilization, we consider the following example in which different types of source data are encapsulated in VLs for transmission. The processing capacity of the source ES is determined by the bandwidth reserved for VLs, which is parameterized by the BAG and the MFS. Suppose for instance that each Sub-VL has a period T=15ms and a MFS $l^{\max}$=1518 Bytes. The simplest configuration is to take every Sub-VL as a VL. Then the VL has the same MFS as the Sub-VL. According to the standard, the BAG should be a power of 2 multiplied by 1ms and selected from the set {1ms, 2ms, 4ms, 8ms, 16ms, 32ms, 64ms, 128ms}. In addition, since no frame should be lost due to buffer overflow, the BAG should be smaller than or equal to T. Thus in our example, to accommodate a source flow of period T=15ms, the BAG of the VL should be 8ms. In Ethernet transmission, an overhead of 20Bytes (Interframe Gap+Preamble+Start Frame Delimiter) should be added into the size of VLs. Then the reserved bandwidth for each VL is equal to $(l^{\max} + 20) \times 8/\text{BAG}$ =1.538Mbps. Suppose that the physical link operates at 100Mbps. Without considering the jitter at the output, the source ES can transmit at most $\lfloor 100/1.538 \rfloor$=65 VLs. This means that it can manage up to 65 Sub-VLs with a period T=15ms. However, the real bandwidth utilization is $(l^{\max} + 20) \times 8/T$=0.82Mbps. Hence, nearly 50 percent bandwidth for every VL is wasted in this example. During transmission, the VLs are frequently in the idle state. Consequently, if more source data are added without aggregation, another source ES is required for this configuration. Instead, if we aggregate three Sub-VLs into one VL, the MFS of the VL does not change. If Sub-VLs with suitable data rate are available, the BAG for an aggregated VL can become 4ms (this can be shown using the model presented below). For each VL, the reserved bandwidth becomes $(l^{\max} + 20) \times 8/\text{BAG}$=3.076Mbps. In this configuration, one source ES can manage at most $\lfloor 100/3.076 \rfloor$=32 VLs aggregating in total 96 Sub-VLs. Therefore, without any additional hardware, the processing capability of the source ES

can be improved by around 48%, leading to a better bandwidth utilization.

B. Computation of the BAG of Aggregated Flows and the Delay due to Sub-VL Aggregation

Consider the aggregation of n Sub-VLs, $1 \leq n \leq 4$, into one VL. Each Sub-VL$_i$ is characterized by its minimum time interval T_i and MFS $l_i^{\max}$. The frame rate of Sub-VL$_i$ is bounded by $\rho_i = 1/T_i$. Then the maximum arrival frame rate (AFR) of Sub-VLs in a VL is $\rho = \sum_{i=1}^{n} \rho_i$.

Let $L_{\max}$ be the MFS of VL:

$$L_{\max} = \max_{1 \leq i \leq n} \{l_i^{\max}\}. \tag{1}$$

Denote by $r = 1/\text{BAG}$ the maximum frame rate in a VL. Obviously, to guarantee that no frame will be blocked due to Sub-VL aggregation, there should be $r \geq \rho$. Moreover, the BAGs must be chosen from the set $\{2^k\}_{k=0}^{7}$ (ms). Therefore, for an appropriate bandwidth allocation, the BAG should be the one with maximum value that meets all the constraints, that is:

$$\text{BAG} = \max_{k=0,\ldots,7} \left\{ 2^k \leq \left(\sum_{i=1}^{n} \rho_i \right)^{-1} \right\}, \tag{2}$$

which can be expressed equivalently as:

$$\text{BAG} = 2^{\min\left(\left\lfloor \log_2 \left(\sum_{i=1}^{n} \rho_i \right)^{-1} \right\rfloor, 7 \right)}. \tag{3}$$

Then the required frame transmission rate (RFTR) for the VL is 1/BAG.

As several Sub-VL queues share the same VL, a frame in a specific Sub-VL$_i$ queue may be delayed due to the RR scheduling. Let D_{SVL_i} be the worst-case queuing delay of Sub-VL$_i$ introduced by Sub-VL aggregation. D_{SVL_i} can be analyzed by using the formulation presented in [22]. Suppose that the Sub-VL is dedicated to one source flow and denoted by $|\text{VL}|$ the cardinal number of Sub-VLs belonging to the VL. Let q be the number of packets that are ready for transmission in the Sub-VL$_i$ queue. Sub-VL$_i$ shares VL$_k$ with the other Sub-VLs. Then D_{SVL_i} can be calculated as:

$$D_{SVL_i} = \max_{q=1,2,\ldots} \left[w_i(q) - (q-1)T_i \right], \tag{4}$$

where

$$w_i(q) = (q-1)\text{BAG}_k + \sum_{\substack{j \neq i \\ 1 \leq j \leq |VL_k|}} \left(\left\lfloor \frac{(q-1)T_i}{T_j} \right\rfloor + 1 \right) \text{BAG}_k. \tag{5}$$

Obviously, Sub-VL aggregation may introduce extra delay, although it helps improving bandwidth utilization efficiency of VLs. This should be taken into account in network design. The trade-off between traffic load and delay due to Sub-VL aggregation is considered in Section IV.

C. Non-Determinism in VL Transmission

In AFDX networks, the idle state in frame transmission can be introduced by the mismatch between the BAG and the period for periodic source flows or by the arrival uncertainty for sporadic source flows. In either cases, frame arrival at destination ESs is undetermined, which might be confused with a fault due to frame loss. To illustrate the issue, in the example shown in Fig. 2, the frame P3 is assumed to be lost in transmission. The destination ES will not detect the fault until the reception of frame P4, because the destination ES does not know when the next frame will arrive. Under this protocol, the destination ES cannot distinguish between transmission silence and data loss. This is indeed a source of non-determinism.

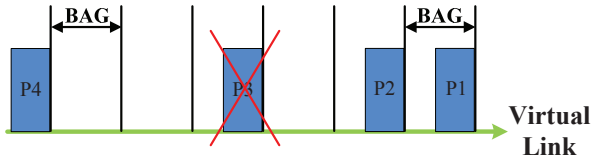


Fig. 2: Destination End System cannot detect the loss of frame P3 until it receives P4.

Inspired by synchronous transmission schemes, this problem can be solved by inserting frames in source ES to ensure that the VL has one frame and always one frame to transmit in every BAG. With this mechanism, the determinism of frame arrival can be improved. Note that the improvement in terms of determinism is at the expense of a possible average delay increase, as we can observe in synchronous transmission schemes, such as TDMA. Note also that as frame insertion is performed at VL level, there is no impact on the worst-case performance. Nevertheless, this operation will increase the actual network load. Therefore, it is of practical interest to minimize the number of inserted frames via appropriate Sub-VL aggregation schemes.

III. Determinism Enhancement with Frame Insertion

In this section, we address the mechanism suggested for frame insertion in VLs to tackle the non-determinism issue related to frame arrival uncertainty. Sub-VL aggregation is then incorporated in this mechanism to mitigate load increase due to frame insertion. We also calculate the required bandwidth after frame insertion and formulate a measure for load increases. These formulations will be used in the resolution of the multi-objective optimization problem studied in Section IV.

A. Frame Insertion in VL

To make transmission deterministic, filler frames are inserted to guarantee that the source ES sends a frame in every BAG. As shown in Fig. 3, an empty flag and a multiplexer are introduced to implement such a mechanism. If it is the time for transmission and there is no frame in VL FIFO, the empty flag is triggered. Then the multiplexer takes a filler frame from filler frame controller and forwards it into the VL sequence. Otherwise, the multiplexer outputs the data

frame from the VL FIFO queue. After every transmission, the multiplexer is halted until the end of the current BAG. Since frame insertion may be needed only if the VL FIFO queue is empty, the VLs are in general not strictly periodic, which is in accordance with the expected behavior of VLs in AFDX networks. Note that, depending on the level of criticality required by specific applications, frame insertion may be performed with a predefined interval bigger than 1 BAG. This would allow reducing the network load while still ensuring determinism. Nevertheless, the formulation presented below can be adapted to this case.

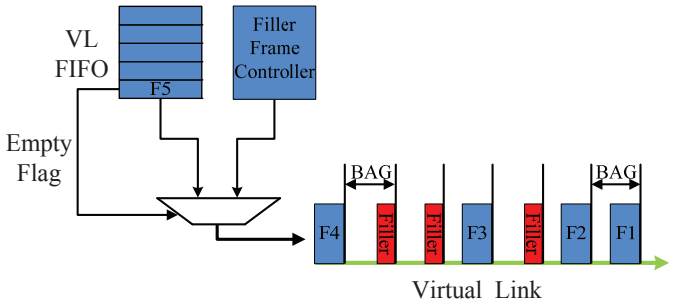


Fig. 3: Proposed mechanism for enhancing the determinism of AFDX networks.

In this mechanism, the size of the filler frame L_{filler} can be 64bytes, the minimum value specified in the AFDX protocol. This would ensure that the filler frame will have no impact on the MFS of the VL. Furthermore, it is obvious that frame insertion will not change the BAG of the VL. As BAG and MFS are the parameters of a regulated VL utilized in worst-case performance analysis, frame insertion in a VL stream has no impact on the worst-case end-to-end delay of the VL, D_{worst}, measured from the VL regulator to the destination ES. Note that the best case end-to-end delay of the VL, D_{best}, is the sum of technology latencies and transmission time, which is determined by the route and MFS of each VL. Since the filler frame has no impact on the route and MFS, D_{best} is unchanged with frame insertion. With filler frame insertion in source ES, it is ensured that there is one frame departing from source ES within every BAG of each VL. Therefore, it is guaranteed that when the destination ES finishes one reception, it must receive another frame within the time interval, $[(\text{BAG} - D_{worst} + D_{best})^+, \text{BAG} + D_{worst} - D_{best}]$ as shown in Fig. 4. By notation, $(x)^+ := \max(x, 0)$.

Nevertheless, filler-frame-based determinism enhancement is achieved at the expense of network load increase. In the follows, we try to mitigate this problem by Sub-VL aggregation.

B. Frame Insertion Based on Sub-VL Aggregation

To optimize network load, it is possible to aggregate several source data flows into a single VL, thus limiting the number of filler frames. As illustrated in Fig. 5, the Sub-VL FIFO queues are read into VL FIFO with a RR sequence and then a possible frame insertion follows. If a frame either from the Sub-VLs or from the filler frame controller is sent to the VL, the multiplexer is halted until the BAG ends.

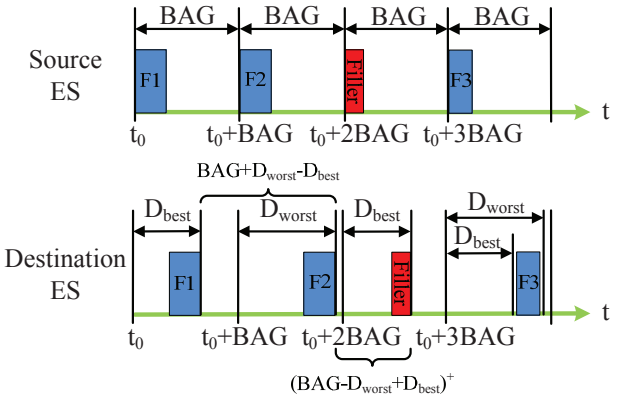


Fig. 4: Reception time interval in destination ES with frame insertion.

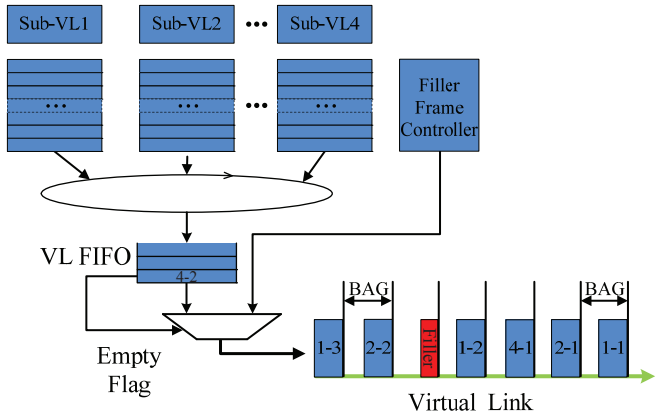


Fig. 5: Frame insertion based on Sub-VL aggregation.

Let us recall that a VL is characterized by two main parameters: the MFS and the BAG computed from (1) and (3), respectively.

C. Bandwidth Requirement with Frame Insertion

Suppose that Sub-VL$_i$ has the frame rate ρ_i and the MFS $l_i^{\max}$. If a VL is formed by only one Sub-VL, then the frame rate of VL is $r = 1/\text{BAG}$, and with frame insertion the AFR in total is $\rho = \rho_i$. Therefore, the rate difference, $r - \rho$, denotes the number of inserted frames per time unit. Then the required bandwidth is given by:

$$BW = l_i^{\max} \times \rho_i + L_{filler} \left(\text{BAG}^{-1} - \rho_i \right), \tag{6}$$

and the reserved bandwidth is $l_i^{\max}/BAG$.

If a VL aggregates n Sub-VLs, the AFR in total is $\rho = \sum_{i=1}^{n} \rho_i$. Therefore the required bandwidth is:

$$BW = \sum_{i=1}^{n} (l_i^{\max} \times \rho_i) + L_{filler} \left(\text{BAG}^{-1} - \sum_{i=1}^{n} \rho_i \right), \tag{7}$$

and the reserved bandwidth is $L_{\max}/BAG$.

The second part on the right-hand side of (6) and (7) is the bandwidth increase after frame insertion, which is related to the rate difference, $r - \rho$. Hence, one objective for optimal Sub-VL aggregation is to minimize the bandwidth increase.

IV. Optimal Sub-VL Aggregation

In order to prevent buffer overflows in the ESs, the RFTR of a VL has to be higher or equal to the AFR of the Sub-VLs it carries. In the case where the arrival period of the source falls into $\{2^k\}_{0\leq k\leq 7}$, the AFR and the RFTR for VL will be identical. Otherwise, frame insertion is required. Either way, there is no idle BAG for VL and the rate difference is equivalent to the load increase in unit time. Hence, an appropriate scheme for Sub-VL aggregation is the one that allows minimizing the sum of rate difference for all VLs, so that the overhead due to frame insertion is the minimum. Meanwhile the delay introduced by Sub-VL aggregation should also be optimized. Therefore, the Sub-VL aggregation should be modeled as a multi-objective optimization problem.

Essentially, there are three main constraints in Sub-VL aggregation:

- Sub-VLs for aggregation should have the same source and destination ESs.
- A VL contains at most 4 Sub-VLs.
- The sum of AFR after Sub-VL aggregation cannot exceed $r_{\max}$ (1K frame/s).

A. Formulation of Optimal Sub-VL Aggregation

Let $S = \{\text{Sub-VL}_1, \text{Sub-VL}_2, ..., \text{Sub-VL}_N\}$ be a set of N Sub-VLs. Denote by ρ_i the frame rate of Sub-VL$_i$. For all Sub-VLs, AFR in total is $\rho_0 = \sum_{i=1}^{N} \rho_i$.

Let $\text{VL}_j = \{\text{Sub-VL}_{j_1}, \ldots, \text{Sub-VL}_{j_n}\}$, $\forall \text{Sub-VL}_{j_k} \in S$, $k = 1, \ldots, n$. We can then formulate the Sub-VL aggregation problem as the partitioning of S into m non-empty VLs with $[\lceil N/4 \rceil \leq m \leq N]$.

Let $\mathcal{P} = \bigcup_{i\in\mathcal{I}} \mathcal{P}_i$ be the set of all admissible partitions, where $\mathcal{I}$ is the index set. In combinatorics, the number of possible partitions in which a set of N elements can be split is bounded by the so-called Bell number [23], denoted B_N. However, the partitioning of Sub-VLs has other constraints such as subset size. Hence, the cardinal of $\mathcal{P}$ may be significantly smaller than B_N. Then the ith partition can be denoted by

$$\mathcal{P}_i := \left\{\text{VL}_1^i, \text{VL}_2^i, \ldots, \text{VL}_j^i, \ldots, \text{VL}_m^i\right\} \vdash S, \tag{8}$$

where VL_j^i is an aggregation of Sub-VLs and the symbol " $\vdash$ " stands for "is a partition of." Denote by BAG_j^i the BAG of VL_j^i. For VL_j^i, the RFTR is $r_j^i = 1/\text{BAG}_j^i$. According to (3), r_j^i is given by:

$$r_j^i = \frac{1}{\text{BAG}_j^i} = 2^{-\min\left(\left\lfloor \log_2\left(\sum_{1\leq k\leq \left|\text{VL}_j^i\right|} \rho_k\right)^{-1} \right\rfloor, 7\right)}, \tag{9}$$

where ρ_k is the frame rate of Sub-VL$_k$ belonging to VL_j^i. The RFTR R_i for the partition $\mathcal{P}_i$ is

$$\begin{aligned} R_i &= \sum_{\text{VL}_j^i\in\mathcal{P}_i} r_j^i \\ &= \sum_{\text{VL}_j^i\in\mathcal{P}_i} 2^{-\min\left(\left\lfloor \log_2\left(\sum_{1\leq k\leq \left|\text{VL}_j^i\right|} \rho_k\right)^{-1} \right\rfloor, 7\right)}. \end{aligned} \tag{10}$$

The rate difference, y_i, representing the wasted bandwidth, can be expressed as:

$$\begin{aligned} y_i &= R_i - \rho_0 \\ &= \sum_{\text{VL}_j^i\in\mathcal{P}_i} 2^{-\min\left(\left\lfloor \log_2\left(\sum_{1\leq k\leq \left|\text{VL}_j^i\right|} \rho_k\right)^{-1} \right\rfloor, 7\right)} - \rho_0. \end{aligned} \tag{11}$$

Obviously, R_i varies according to the partition, and so does y_i. To reduce the load increase, we need to minimize the rate difference y_i in (11). Note that for a given set of Sub-VLs, ρ_0 is a constant. Hence, minimizing the rate difference is equivalent to obtain the minimum R_i.

The delays introduced by Sub-VL aggregation can be measured in different ways that will influence the optimization procedure. For example, if the maximum worst-case delay among all the VLs is chosen as the cost function, then the result will tend to be the lowest allowed delay for all the Sub-VLs. A less pessimist choice is the use of the average worst-case delay of all the Sub-VLs:

$$D_{\mathcal{P}_i} = \frac{1}{N} \sum_{\text{VL}_j^i\in\mathcal{P}_i} \sum_{1\leq k\leq \left|\text{VL}_j^i\right|} D_{SVL_k}, \tag{12}$$

which is the second cost function considered in the resolution of optimal Sub-VL aggregation problems in the present work.

Let $G_1(\mathcal{P}_i) = R_i$ and $G_2(\mathcal{P}_i) = D_{\mathcal{P}_i}$. Optimal Sub-VL aggregation amounts then to solving the following multi-objective optimization problem:

$$\min_{\mathcal{P}_i\in\mathcal{P}} G(\mathcal{P}_i) = \left[G_1(\mathcal{P}_i), G_2(\mathcal{P}_i)\right]^T; \tag{13}$$

$$\text{s.t. : } 1 \leq i \leq B_N; \tag{14}$$

$$\lceil N/4 \rceil \leq j \leq N; \tag{15}$$

$$1 \leq k \leq 4; \tag{16}$$

$$\sum_{1\leq k\leq \left|\text{VL}_j^i\right|} \rho_k \leq r_{\max}; \tag{17}$$

where i is the partition index, N is the total number of Sub-VLs, and k represents the index of a Sub-VL in a VL. In AFDX networks, the sum of AFRs of the aggregated Sub-VLs cannot exceed $r_{\max}$ (1K frame/s), the maximum rate of a single VL.

Since Sub-VL aggregation can introduce extra delay, an optimal solution to this problem can only be achieved in the sense of Pareto by considering the possible trade-off between these two objectives (see, e.g., [24]).

B. Lexicographic Method for Optimal Sub-VL Aggregation

In order to find a Pareto optimal solutions, system designers should impose design preferences [24]. In the considered problem, the primary objective is to reduce the load increase based on which we will try to minimize the delay introduced by Sub-VL aggregation. The lexicographical optimization method is suitable for this setup. More precisely, minimizing the rate difference is solved first. Then, a δ-constraint is imposed to control the trade-off between load increase and the delay

due to Sub-VL aggregation. The corresponding lexicographic formulation can be given as follows:

$$\min_{\mathcal{P}_i \in \mathcal{P}} \; G_l(\mathcal{P}_i); \tag{18}$$

$$\text{s.t. : (14)} - \text{(17)};$$

$$G_1(\mathcal{P}_i) \leq (1+\delta)\, G_1(\mathcal{P}_1^*), \text{ for } l = 2; \tag{19}$$

$$l = 1, 2;$$

where $\mathcal{P}_1^*$ represents the optimum of the first objective function. δ is a nonnegative value that can be varied to tighten the constraint. Note the multi-objective functions are solved in sequence to find the Pareto optimal points [24].

To illustrate the main property of the above multi-objective optimization problem, we consider an example of 8 Sub-VLs having different periods as shown in Table I. By using exhaustive enumeration, all possible solutions regarding the first objective can be obtained as shown in Fig. 6. Meanwhile, we can construct the so-called Pareto front for this multi-objective optimization problem (see Fig. 7). Note that in Fig. 7, many points are overlapped in R_i-$D_{\mathcal{P}_i}$ plane.

TABLE I: Parameters of Sub-VLs

Sub-VL	1	2	3	4	5	6	7	8
Period (ms)	10	25	30	40	60	80	100	125

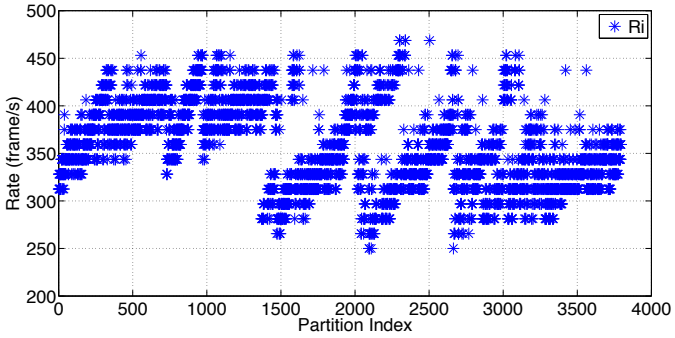


Fig. 6: RFTR with parameters in Table I.

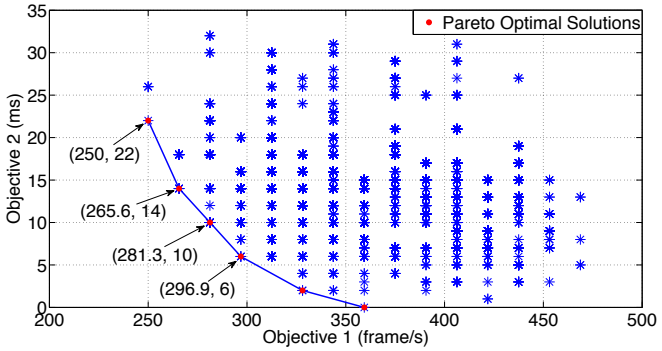


Fig. 7: All possible solutions and Pareto front.

For $\delta = 0$, the point with an RFTR of 250 frame/s and an average worst-case delay of 22ms will be the Pareto optimal solution, which has the minimum RFTR among all possible solutions, and the delay introduced by Sub-VL aggregation for the partition is the smallest under the δ-constraint.

If we relax the δ-constraint, for example let $\delta = 20\%$, then more partitions with RFTR$\leq$ 300 frame/s can be included in the set of candidate solutions. In this case, the solution point with (296.9, 6) is found to be the Pareto optimal solution. In this case, the introduced delay is reduced at the expense of network load increase. More details about this example are discussed later in Section V-B.

C. Algorithms for Sub-VL Aggregation

The multi-objective optimization problem can be solved in two iterations. The first iteration is to obtain $\mathcal{P}_1^*$ with the minimum RFTR $\mathcal{R}^*$. The second iteration is to find the partition with minimum delay introduced by Sub-VL aggregation under (14)-(17) and (19). According to (10), R_i is a discrete and nonlinear function with respect to the partition. Therefore, the objective function $G_1(\mathcal{P}_i)$ is not convex and might not admit a unique global minimum, as shown in Fig. 6. Similarly, $G_2(\mathcal{P}_i)$ is also nonlinear and non-convex. Therefore, the solution for the multi-objective optimization problem might not be unique. Indeed the considered problem is a special multiple knapsack problem, which is NP-Complete. Furthermore, it is very different from standard multiple knapsack problems when considering the trade-off between load increase and the delay introduced by Sub-VL aggregation. In this paper, three algorithms are applied to solve the multi-objective optimization problem.

1) Brute Force Algorithm: One possible and accurate method to obtain the optimal solution is the brute force algorithm. The optimal solution can be found by an exhaustive enumeration of all solutions. In the first iteration, an optimal solution $\mathcal{P}_1^*$ that leads to the minimum rate difference is obtained. Then by adding the δ-constraint in the second iteration, we can get a global optimal result for the multi-objective optimization problem. However the computational complexity grows exponentially with the number of Sub-VLs. Note that more efficient algorithms, such as branch-and-bound algorithm [25] and the greedy algorithm [26] [27] [28], can be explored to address the optimization problem with large size, when applying the proposed formulation to real-life applications. In present work, we consider to use the greedy algorithm, which is well known and computationally efficient.

2) Greedy Algorithm: The strategy behind the greedy algorithm is to make local optimal choices at every step of each iteration with the hope of finding a globally optimal result. The heuristics used in the developed algorithm is that for each step, the optimization strategy is to select one aggregation of Sub-VLs, through which R_i or $D_{\mathcal{P}_i}$ can be reduced the most. Let $\mathcal{V}$ be the candidate set of VLs composed of 2 to 4 Sub-VLs. Suppose that $\forall v \in \mathcal{V}$, the corresponding RFTR does not exceed the rate limit $r_{\max}$. Since the Sub-VLs cannot be reused, all VLs, containing one or more selected Sub-VLs, are removed from the candidate set at the end of each step. The greedy algorithm stops and gives the local optimal solution when the candidate set is empty.

As an example, we consider a set of 3 Sub-VLs characterized by their periods: {6ms, 20ms, 40ms}. The candidate

set $\mathcal{V}$ is {{6ms, 20ms}, {6ms, 40ms}, {20ms, 40ms}, {6ms, 20ms, 40ms}}. For the subset {6ms, 20ms}, the RFTRs before and after aggregation are 312.5 frame/s and 250 frame/s, respectively. Therefore, the reduction after aggregation is 62.5 frame/s. Furthermore, let us denote by $D_v = \sum_{1\leq i\leq |v|}(D_{SVL_i})$ the total introduced worst-case delay of the Sub-VLs in one VL. In this case, D_v is 8ms. Accordingly, the rate difference and the introduced delay for the other aggregations are calculated and listed in Table II.

TABLE II: Sub-VL Aggregation Candidates

Subsets in $\mathcal{V}$ (ms)	RFTR without Aggregation(frame/s)	RFTR (frame/s)	Δ (frame/s)	D_v (ms)
{6, 20}	312.5	250	62.5	8
{6, 40}	281.25	250	31.25	8
{20, 40}	93.75	125	−31.25	16
{6, 20, 40}	343.75	250	93.75	24

In the first iteration, the aggregation of {6ms, 20ms, 40ms} is selected in the first step and all the VLs comprising these Sub-VLs are removed from the candidate set. Then the greedy algorithm stops as the candidate set becomes empty. In the second iteration, the candidate set is sorted first in ascending order for the delay introduced by Sub-VL aggregation, and then by descending order for the reduction. Suppose that all subsets in $\mathcal{V}$ are allowed with the δ-constraint. Therefore, {6ms, 20ms} with a total delay of 8ms and a reduction of 62.5 frame/s is selected. Consequently, all the candidates are removed from the candidate set because they share one or more selected Sub-VLs. Then the greedy algorithm stops as the candidate set becomes empty. In this case, the local optimal solution is {{6ms, 20ms}, {40ms}}. The rigorous formulation of the developed greedy algorithm is described below.

We define a function $\Delta : \mathcal{V} \to \mathbb{R}$ which gives, for each VL, the gain obtained with the aggregation of its Sub-VLs. Denoting by $r_b(v)$ the sum of the frame rate required by the Sub-VLs within a VL before aggregation and by $r_a(v)$ the RFTR after aggregation, we have for all $v \in \mathcal{V}$:

$$\begin{aligned}\Delta(v) &= r_b(v) - r_a(v) \\ &= \sum_{1\leq i\leq |v|} 2^{-\min\left(\left\lfloor \log_2 \rho_i^{-1}\right\rfloor, 7\right)} \\ &\quad - 2^{-\min\left(\left\lfloor \log_2\left(\sum_{1\leq i\leq |v|}\rho_i\right)^{-1}\right\rfloor, 7\right)}.\end{aligned} \tag{20}$$

Note that the gain $\Delta(v)$ can be either positive, negative or null depending on the Sub-VLs in v. We define a subset of $\mathcal{V}$, $F \subset \mathcal{V}$, such that $\forall v_1, v_2 \in F$, $v_1 \cap v_2 = \emptyset$. Then the local minimum of the first objective function, $\mathcal{P}_1^*$, can be achieved by maximizing the gain, $\sum_{v\in F}\Delta(v)$. Meanwhile, we can compute the total worst-case delay of the Sub-VLs, D_v.

Then the local optimal solution for the multi-objective optimization problem can be obtained by minimizing the delay introduced under the added δ-constraint in the second iteration. The developed greedy algorithm is summarized in Algorithm 1.

Algorithm 1 Greedy Algorithm

Input: S, δ;
Output: $(R_{\min}, D_{\min})$ and corresponding partition;
Initial: $F_1 = \emptyset$, $S_1' = S$, $F_2 = \emptyset$, $S_2' = S$;

```
Construct V, the set of all possible VLs;
For each v ∈ V, compute the gain Δ(v) and the extra delay D_v;
Discard the VLs with the negative or null gain;
Sort the VLs by decreasing Δ(v) and insert them in a list L_1;
Sort the VLs first by ascending D_v and then by decreasing Δ(v) and insert them in a list L_2;
while L_1 ≠ ∅ do
    Add L_1[1] (the first VL with biggest gain in current L_1) to the solution F_1, S_1' = S_1' ∩ L_1[1];
    Remove all VLs from L_1 which contain any Sub-VL of L_1[1], including L_1[1] itself;
end while
Obtain P_1^* = F_1 ∪ S_1' and then calculate R^*;
while L_2 ≠ ∅ do
    if r_a(L_2[1]) / Σ_{1≤i≤|L_2[1]|} ρ_i ≤ (1+δ) R^* / Σ_{1≤i≤|S|} ρ_i then
        Add L_2[1] to the solution F_2, S_2' = S_2' ∩ L_2[1];
        Remove all VLs from L_2 which contain any Sub-VL of L_2[1], including L_2[1] itself;
    else
        Remove L_2[1] from L_2;
    end if
end while
Obtain P_min = F_2 ∪ S_2' and then calculate (R_min, D_min);
Output (R_min, D_min) and P_min.
```

Although the greedy algorithm cannot guarantee to find the global optimum, it is much less time consuming compared to the brute force algorithm. An experiment implemented with Matlab® for a set of 100 Sub-VLs with randomly generated period for all the Sub-VLs shows that the execution can be terminated within minutes. More detailed results of experiments will be presented in Section V.

3) Greedy Algorithm with Pre-processing: The complexity of the greedy algorithm is related to the search space size. It happens that when grouping some Sub-VLs together, the equivalent period is a power of 2. In this case, the search space can be reduced if we perform these aggregations first and remove them from set of Sub-VLs.

For example, the periods of Sub-VL$_1$, Sub-VL$_2$ and Sub-VL$_3$ are, respectively, 5ms, 5ms and 10ms. To aggregate these three Sub-VLs into one VL, the equivalent period is 2ms. According to (3), the BAG for this VL is 2ms. In this case, no filler frame insertion is needed.

Based on the above analysis, a greedy algorithm with pre-processing is developed. This algorithm has two steps. The first step is to find special cases mentioned above that perfectly fill VLs among the Sub-VL set. The second step is the use of the greedy algorithm in Section IV-C2 to find the local optimal result with the reduced Sub-VL set.

In summary, the brute force algorithm can reach the global

optimal solution, but it is suitable only for small number of considered Sub-VLs. The greedy algorithm and its variation with pre-processing are suitable for large size problems, although they may lead to local optimums. The performance of these algorithms are illustrated in the next section.

V. Performance Evaluation

In this section, we perform numerical simulations of the proposed mechanism to verify its feasibility. Then the performance of different optimization algorithms for Sub-VL aggregation is evaluated using different configurations.

A. Validation of Frame Insertion Mechanism

In Section III-B, a mechanism with frame insertion based on Sub-VL aggregation is put forward. The feasibility of such a mechanism is verified in this section by numerical simulations using Matlab® and TrueTime [29]. In the considered example, there are three Sub-VLs whose parameters are listed in Table III. Based on (3), the BAG for the aggregated VL is 4ms.

TABLE III: Parameters of Sub-VLs

Sub-VL	Period (ms)	Max Jitter (ms)	Frame Size (byte)
Sub-VL1	10	3	80
Sub-VL2	60	18	180
Sub-VL3	25	7.5	130

First, Matlab® is used to simulate the aggregation and regulation in source ES. In this simulation, we use unit frame size. The simulation results are shown in Fig. 8.

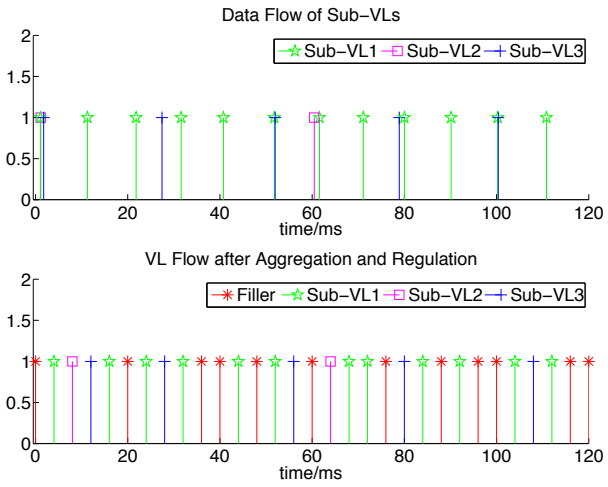
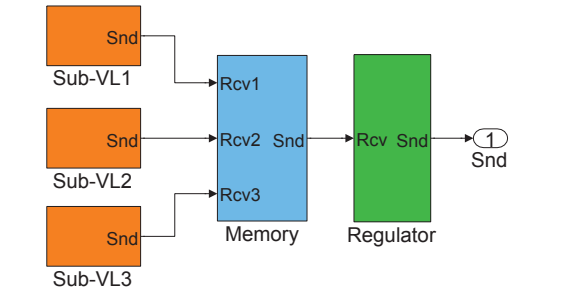
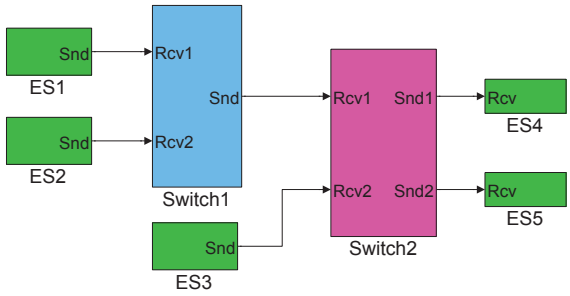


Fig. 8: Matlab® simulation result of frame insertion based on Sub-VL aggregation.

The jitter of each Sub-VL is considered in the simulation. Due to the jitter, the accurate positions of filler frames cannot be determined in advance. As shown in Fig. 5, the empty flag acts as a trigger. When there is nothing from Sub-VLs to transmit during the period of 1 BAG, the empty flag is triggered. Then the filler frame is forwarded into VL. The



(a) Sub-VL aggregation and VL regulation model



(b) AFDX simulation system configuration

Fig. 9: AFDX simulation system based on TrueTime.

simulation result is shown in Fig. 8. The red line with "∗"-mark indicates the inserted frames. The other three different colors and shapes represent the Sub-VLs. The frames of Sub-VLs and inserted frames are regulated according to the BAG. They are transmitted every 4ms. According to the simulation results, the proposed mechanism is feasible.

Furthermore, a TrueTime simulation is set up to implement the proposed mechanism. TrueTime is a more accurate timing simulator based on Matlab®/Simulink, which can model data transmission using different network protocols and task execution in real-time kernels [29]. A simple AFDX network shown in Fig. 9 is set up. Sub-VL aggregation and frame insertion are added into the model of ES1. The simulated system can perform a real-time data transmission.

After Sub-VL aggregation, frame insertion, and VL regulation, all frames are forwarded into a VL sequence. Frame insertion function is implemented in the VL regulator model. The simulation results of Sub-VL aggregation are shown in Fig. 10. The data flow without insertion is also presented for comparison. As stated in Section III-A, we can guarantee the determinism of frame arrival with frame insertion. The real-time simulation results confirms the feasibility of such a mechanism.

Using this structure, the end-to-end delay analysis can be performed. In addition, real-time fault detection can also be executed. We can set a probability of data loss in switches. With the expected deterministic reception on destination ESs, it is easy to detect some classes of faults such as lost frames in real-time. Furthermore, this AFDX network simulation system is extensible to more complex network configurations, which allows carrying out additional fault injection and fault analysis.

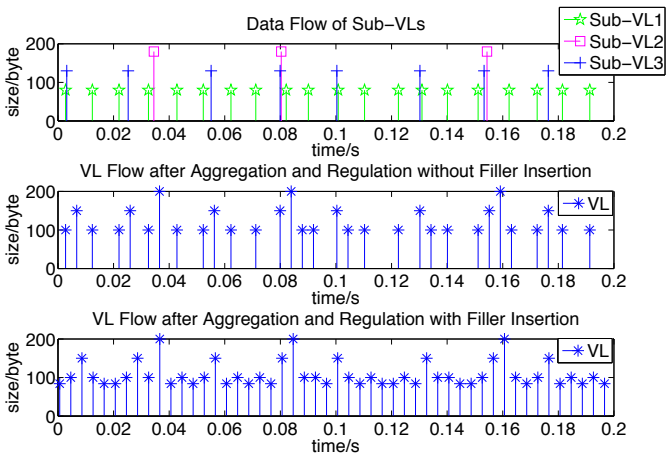


Fig. 10: Simulation results produced with TrueTime.

B. Evaluation of Sub-VL Aggregation Strategies

In the follows, comparison of different aggregation schemes is performed to evaluate the developed optimal Sub-VL aggregation strategies.

The considered system is a network with 8 Sub-VLs studied in Section IV.B whose parameters are given in TABLE I. According to (7), L_{filler} is smaller or equal to $l_i^{\max}$. Hence, the load increase percentage measured in frame is equal or greater than the load increase measured in bit. In this study, the traffic increase is measured as an increased frame rate. In this example, the AFR is a total ρ_0 of 245.5 frame/s. If every Sub-VL is transmitted by one VL, the RFTR is 359.4 frame/s. As there is no aggregation, the introduced delay is zero. Whereas, the load increase is about 46.4% compared to the arrival frames after frame insertion.

Note that some Sub-VL aggregation schemes may lead to poor performance. For example, in the above considered problem, the partition, VL_1={Sub-VL1, Sub-VL2, Sub-VL6, Sub-VL8} and VL_2={Sub-VL3, Sub-VL4, Sub-VL5, Sub-VL7}, will result in as much as 52.75% frame rate increase and 18ms introduced average delay.

To solve the multi-objective optimization problem, the brute force Sub-VL aggregation strategy is applied, in which all admissible partitions are traversed to obtain the global optimal solution under the δ-constraint. In the considered example, the Pareto optimal solution with the constraint of δ=20% is given in Table IV. The overall load increase is about 20.9%, which is better than a 46.4% load increase observed when no aggregation is performed and filler packets are inserted. As presented in Section IV-B, the introduced average delay, 6ms, is minimal for the partitions under the δ-constraint.

When the greedy algorithm is used in this example, the search is terminated by a local optimal solution. When δ=20%, the local optimal partition is {{Sub-VL1, Sub-VL4}, {Sub-VL2, Sub-VL5}, {Sub-VL3, Sub-VL6}, {Sub-VL7}, {Sub-VL8}}. The corresponding performance is given in TABLE V. In this case, the load increase is only 14.56%, which is much better than the scheme without Sub-VL aggregation. Furthermore, the introduced average delay is 10ms, It can be observed from Fig. 7 that this solution is Pareto optimal.

TABLE IV: Performance Obtained with the Brute Force Algorithm

Sub-VL Aggregations	AFR (frame/s)	RFTR (frame/s)	Excess Frame Rate in Percent	D_v (ms)
{1, 4}	125	125	0	16
{2, 5}	56.7	62.5	10.29%	32
{3}	33.3	62.5	87.5%	0
{6}	12.5	15.6	25%	0
{7}	10	15.6	56.25%	0
{8}	8	15.6	95%	0
Total	245.5	296.9	20.94%	48

However, the greedy algorithm cannot guarantee to find the Pareto optimal solution. It obtains a local optimal solution for the optimization problem considering the trade-off between the two objectives.

TABLE V: Performance of the Greedy Algorithm

Sub-VL Aggregations	AFR (frame/s)	RFTR (frame/s)	Excess Frame Rate in Percent	D_v (ms)
{1, 4}	125	125	0	16
{2, 5}	56.7	62.5	10.29%	32
{3, 6}	45.8	62.5	36.46%	32
{7}	10	15.6	56.25%	0
{8}	8	15.6	95%	0
Total	245.5	281.3	14.56%	80

It can be observed that in this example, the period of a VL aggregating Sub-VL1 and Sub-VL4 is the power of 2. The situation is the same when we aggregate Sub-VL2, Sub-VL6 and Sub-VL7. We can then apply the greedy algorithm with pre-processing. In the first step, we get a reduced set listed in TABLE VI. In the second step, the greedy algorithm is executed over the reduced set. When δ=20%, a local optimal solution of {{Sub-VL1, Sub-VL4}, {Sub-VL2, Sub-VL6, Sub-VL7}, {Sub-VL3, Sub-VL5}, {Sub-VL8}} is obtained. The load increase and the introduced average delay for this partition are 265.6 frame/s and 18ms, respectively. The greedy algorithm with pre-processing provides system designers with an additional option in network tuning.

TABLE VI: Set Obtained by First Step of Pre-processing Greedy Algorithm

Sub-VL	3	5	8
Period(ms)	30	60	125
Max Jitter(ms)	9	18	37.5

In order to validate the performance with different configurations, many instances with randomly generated periods in the [1ms, 200ms] interval were analyzed. The δ-constraint is set to 0 and 10%, respectively. The results are shown in Fig. 11, Fig. 12 and Fig. 13. For each parameter set considered, three algorithms are applied to obtain global/local optimal solutions for 1000 instances. The avg./worst/best performances for different algorithms are obtained. Compared with the

results without aggregation, the solutions using aggregation strategies are much better with respect to load mitigation, even in the worst case. The overhead load in the network due to frame insertion is reduced. It is worth noting that the brute force algorithm cannot finish in a reasonable time with 50 Sub-VLs. For this case, only the greedy algorithm and the greedy algorithm with pre-processing are applied to find the local optimal solutions. Although the solutions may not be Pareto optimal in a strict sense, they are much better than the scheme without Sub-VL aggregation.

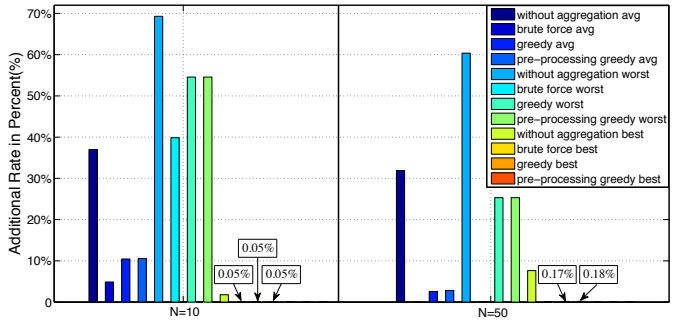


Fig. 11: Evaluation of the load increase for 10 and 50 Sub-VLs (N=10 and N=50), $\delta = 0$.

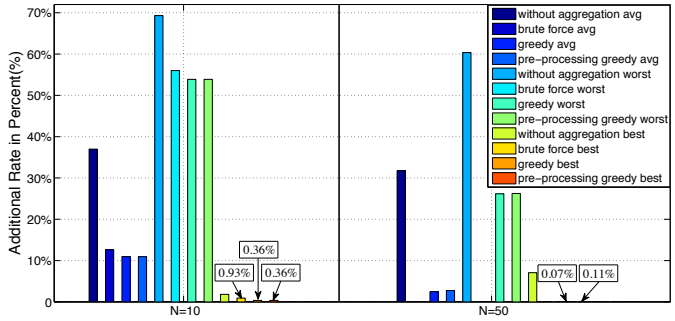


Fig. 12: Evaluation of the load increase for 10 and 50 Sub-VLs (N=10 and N=50), δ =10%.

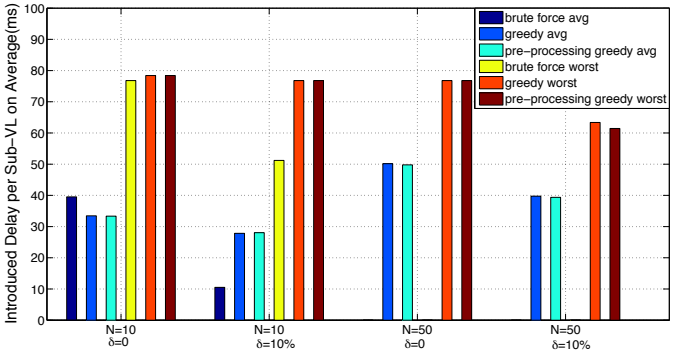


Fig. 13: Average or worst value of the average delay introduced by Sub-VL aggregation for the specified parameter.

VI. Concluding Remarks

In this paper, a mechanism for frame insertion is proposed to enhance the determinism of AFDX networks with respect to frame arrival uncertainty. In order to reduce the load increase due to frame insertion, a strategy for Sub-VL aggregation is developed, which is formulated as a multi-objective optimization problem considering the trade-off between load increase and the delay introduced by Sub-VL aggregation. Three algorithms are proposed and investigated to solve the Sub-VL aggregation optimization problem. Simulations are carried out to illustrate the feasibility of the proposed filler packet insertion method and to validate the performance of developed algorithms. The results show that the load increase can be dramatically reduced and the delay introduced by Sub-VL aggregation can be mitigated with a relaxed δ-constraint. Finally, the framework of multi-objective optimization can be extended to incorporate more design considerations.

It is worth noting that the focus of the present work is put on the configuration in source ESs. However the Sub-VL aggregation with frame insertion may have an impact on the entire network. This may raise challenges regarding the practical application of the proposed mechanism. However, it is interesting to note that the work presented in [30] shows that for a case-study composed of a flight management system, the temporal behavior of avionics functions is not significantly affected even when the worst-case network delay has been increased by 400%. Nevertheless, the impact of Sub-VL aggregation with frame insertion has to be carefully evaluated against the overall performance requirements for specific applications in the design of AFDX networks.

References

[1] ARINC 664, *Aircraft Data Network Part 7 Avionics Full-Duplex Switched Ethernet Network*. AERONAUTICAL RADIO INC., 2009.

[2] R. L. Cruz, "A calculus for network delay, part I: Network elements in isolation," *IEEE Trans. Inf. Theory*, vol. 37, no. 1, pp. 114 –131, Jan. 1991.

[3] R. L. Cruz, "A calculus for network delay, part II: Network analysis," *IEEE Trans. Inf. Theory*, vol. 37, no. 1, pp. 132 –141, Jan. 1991.

[4] J.-Y. Le Boudec and P. Thiran, *Network calculus: a theory of deterministic queuing systems for the internet*. Springer, 2001, vol. 2050.

[5] M. Boyer and C. Fraboul, "Tightening end to end delay upper bound for AFDX network calculus with rate latency FIFO servers using network calculus," in *Proc. of IEEE WFCS*, May 2008, pp. 11–20.

[6] J.-L. Scharbarg, F. Ridouard, and C. Fraboul, "A probabilistic analysis of end-to-end delays on an AFDX avionic network," *IEEE Trans. Ind. Informat.*, vol. 5, no. 1, pp. 38–49, Feb. 2009.

[7] H. Bauer, J.-L. Scharbarg, and C. Fraboul, "Applying and optimizing trajectory approach for performance evaluation of AFDX avionics network," in *Proc. of IEEE ETFA*, Sep. 2009, pp. 1–8.

[8] H. Bauer, J.-L. Scharbarg, and C. Fraboul, "Worst-case end-to-end delay analysis of an avionics AFDX network," in *Proc. of DATE*, Mar. 2010, pp. 1220–1224.

[9] H. Bauer, J.-L. Scharbarg, and C. Fraboul, "Improving the worst-case delay analysis of an AFDX network using an optimized trajectory approach," *IEEE Trans. Ind. Informat.*, vol. 6, no. 4, pp. 521–533, Nov. 2010.

[10] N. Hu, T. Lv, and N. Huang, "Applying Trajectory approach for computing worst-case end-to-end delays on an AFDX network," *Procedia Engineering*, vol. 15, pp. 2555–2560, 2011.

[11] H. Bauer, J.-L. Scharbarg, and C. Fraboul, "Worst-case backlog evaluation of avionics switched ethernet networks with the trajectory approach," in *Proc. of the 24th ECRTS*, Jul. 2012, pp. 78–87.

[12] M. Tawk, G. Zhu, Y. Savaria, X. Liu, J. Li, and F. Hu, "A tight end-to-end delay bound and scheduling optimization of an avionics AFDX network," in *Proc. of the 30th IEEE/AIAA DASC*, Oct. 2011, pp. 1–19.

[13] Y. Hua and X. Liu, "Scheduling design and analysis for end-to-end heterogeneous flows in an avionics network," in *Proc. of IEEE INFOCOM*, Apr. 2011, pp. 2417–2425.

[14] X. Zhang, X. Chen, L. Zhang, G. Xin, and T. Xu, "End-to-end delay analysis of avionics full duplex switched ethernet with different flow scheduling scheme," in *Proc. of IEEE ICCSNT*, vol. 4, 2011, pp. 2252–2258.

[15] Y. Hua and X. Liu, "Scheduling heterogeneous flows with delay-aware deduplication for avionics applications," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 9, pp. 1790–1802, 2012.

[16] J. Li, H. Guan, J. Yao, G. Zhu, and X. Liu, "Performance enhancement and optimized analysis of the worst case end-to-end delay for AFDX networks," in *Proc. of IEEE GreenCom*, Nov. 2012, pp. 301 –310.

[17] H. Charara, J.-L. Scharbarg, J. Ermont, and C. Fraboul, "Methods for bounding end-to-end delays on an AFDX network," in *Proc. of the 18th ECRTS*, 2006, pp. 193–202.

[18] J. Zhang, D. Li, and Y. Wu, "Modelling and performance analysis of AFDX based on petri net," in *Proc. of the 2nd ICFCC*, vol. 2, May 2010, pp. V2–566–V2–570.

[19] M. Boyer, N. Navet, and M. Fumey, "Experimental assessment of timing verification techniques for AFDX," in *Proc. of ERTS, Toulouse, France*, 2012.

[20] M. Adnan, J.-L. Scharbarg, J. Ermont, and C. Fraboul, "An improved timed automata model for computing exact worst-case delays of AFDX periodic flows," in *Proc. of the 16th IEEE ETFA*, 2011, pp. 1–4.

[21] A. Al Sheikh, O. Brun, M. Chéramy, and P.-E. Hladik, "Optimal design of virtual links in AFDX networks," *Real-Time Systems*, pp. 1–29, 2012. [Online]. Available: http://dx.doi.org/10.1007/s11241-012-9171-z.

[22] J. J. Gutiérrez, J. C. Palencia, and M. G. Harbour, "Response time analysis in AFDX networks with sub-virtual links and prioritized switches," *XV Jornadas de Tiempo Real*, 2012.

[23] "Bell number," http://mathworld.wolfram.com/BellNumber.html.

[24] R. T. Marler and J. S. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and Multidisciplinary Optimization*, vol. 26, no. 6, pp. 369–395, Apr. 2004.

[25] P. M. Narendra and K. Fukunaga, "A branch and bound algorithm for feature subset selection," *IEEE Trans. Comput.*, vol. 26, no. 9, pp. 917–922, Sep. 1977.

[26] G. B. Dantzig, *Linear programming and extensions*. Princeton University Press, 1998.

[27] A. Vince, "The greedy algorithm and coxeter matroids," *Journal of Algebraic Combinatorics*, vol. 11, no. 2, pp. 155–178, Mar. 2000.

[28] A. Vince, "A framework for the greedy algorithm," *Discrete Applied Mathematics*, vol. 121, no. 1-3, pp. 247–260, Sep. 2002.

[29] A. Cervin, D. Henriksson, B. Lincoln, J. Eker, and K. Arzen, "How does control timing affect performance? Analysis and simulation of timing using Jitterbug and TrueTime," *IEEE Control Syst. Mag.*, vol. 23, no. 3, pp. 16–30, Jun. 2003.

[30] M. Lauer, J. Ermont, F. Boniol, and C. Pagetti, "Latency and freshness analysis on IMA systems," in *Proc. of the 16th IEEE ETFA*, 2011, pp. 1–8.