Noname manuscript No.
(will be inserted by the editor)

Optimal Design of Virtual Links in AFDX Networks

Ahmad Al Sheikh · Olivier Brun · Maxime Chéramy · Pierre-Emmanuel Hladik

Received: date / Accepted: date

Abstract The Avionics Full Duplex Switched Ethernet (AFDX) backbone constitutes one of the major technological breakthroughs in modern avionic architectures. This network is based on routing Ethernet frames through isolated data tunnels referred to as Virtual Links (VL). VLs can be thought of as multicast trees, each serving for data transmission between one and only one end of the network to several others. Multiple VLs are deployed for exchanging data between avionic systems with a reserved amount of bandwith.

In this paper, we propose different methods to define VL characteristics and to route VLs in the network while minimizing the maximum utilization rate of the links. The proposed methods provide the basis for a more efficient design of the VLs, and have to be completed later on by the verification of the worst-case network latencies. The industrial applicability is shown on experimental results and on a representative benchmark.

Keywords AFDX · Virtual Link · Bandwidth consumption · Route optimization

1 Introduction

1.1 Resource Allocation Problems in IMA Architectures

Nowadays, aircrafts have to maintain high standards of safety and reliability in a high stress environment whilst integrating many computers, sensors, actuators, and control and display units [17,18,4,14]. Traditional avionic systems have been built according to a federated architecture ("one function - one computer"), which leads to numerous equipment boxes, one for each subsystem.

A. Al Sheikh · O. Brun · M. Chéramy · P.E. Hladik
CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France.
Univ de Toulouse, INSA, LAAS, F-31400 Toulouse, France.
E-mail: {aalsheik, brun, mcheramy, pehladik}@laas.fr

With the ever increasing number of embedded avionic functions, the avionic industry has had to abandon this design approach in favor of the Integrated Modular Avionics (IMA) [19]. IMA architectures allow the execution of avionic functions using shared computing and communication resources while respecting hard segregation constraints in order to avoid fault-propagation between applications.

At the computing level, the processing units, called modules, can host several avionic applications of different criticalities and execute them independently. Segregation constraints are enforced using complete partitioning, not only on a functional basis, but also with respect to space (spatial partitioning) and time (temporal partitioning). A partition is therefore a program unit of an application that can execute only within strictly periodic time intervals and that can only access a statically allocated memory space.

At the communication level, the Avionics Full-Duplex Switched Ethernet (AFDX) network enforces the segregation constraints by replacing the point-to-point cabling used in federated architectures with Virtual Links (VL) [2]. Each VL is dedicated to a single communication flow between a source partition and multiple receivers and uses a statically configured route in the network. It can be thought as a permanently defined virtual circuit between the source and the receivers with guaranteed timing and bandwidth allocation on the network. Traffic-shaping is used to regulate the time between two consecutive transmissions on the network by the same VL, thus controlling the bandwidth it uses, and providing traffic at a constant and deterministic rate.

Primary benefits of IMA include reduction of the weight, the power consumption and the overall complexity of the physical architecture, as well as providing greater flexibility in software design and shortened design-cycle times. However, the transition from federated to IMA architectures has given rise to complex resource allocation problems. We can distinguish two main types of resource allocation problems:

- The first is related to the multiprocessor scheduling of the strictly periodic partitions on the processing modules. Indeed, as partitions allocated to the same module share the overall execution time, the system designer has to ensure the temporal segregation of these partitions by designing a proper periodic schedule.
- The second is related to the design of the Virtual Links. Indeed, data exchanges between partitions located on different modules have to be tunneled through VLs, and thus a dedicated VL has to be configured for each communication flow. This amounts to selecting some transmission parameters, which have an impact on the timing properties and on the bandwdith allocation of the VL, in addition to finding a multicast path between the source and the receivers.

In a previous work [3], we have exposed a method to allocate and schedule partitions on the processing modules. The present paper is the logical progression from this previous work and is focused on the design of VLs in AFDX

networks, assuming a predefined multiprocessor scheduling of the avionic partitions.

1.2 Design of Virtual Links

Due to the complexity of the overall problem, the design of an IMA platform usually follows a (suboptimal) decomposition approach (refer to Figure 1). In this approach, the design of the virtual links is performed once the avionic partitions have been scheduled on the processing modules of the platform. A dedicated VL implementing its own traffic shaper has to be configured for each data exchange between partitions located on different modules. This basically requires that the system integrator answers the following questions:

- *What are the VLs to be configured ?*
- *How to set the transmission parameters of the VLs ?*
- *How to route the VLs in the AFDX Interconnect ?*

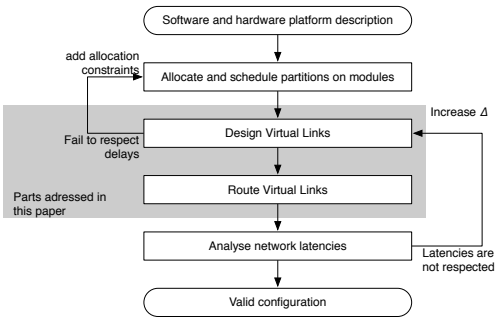

Fig. 1: Decomposition approach for the design process on IMA.

Let us first consider the second question. Two key parameters are used to configure a VL. The first one is the Bandwidth Allocation Gap (BAG), *i.e*, the minimum time interval between the starting bits of two successive AFDX frames, assuming zero jitter. In other words, the BAG defines the maximum frequency at which frames are sent in a VL. This parameter can take only discrete values that are powers of 2 from 1 to 128 ms. The second parameter is the Maximum Frame Size (MFS), *i.e*, the maximum size, in bytes, of the transmitted Ethernet frames for the VL. MFS is in the range from 64 to 1518 bytes. Together, these parameters allow to limit the transmission rate of a VL in order to prevent its traffic from interfering with the traffic of other VLs.

For each message sent by a partition to receivers on other modules, the system integrator has to set the BAG and MFS values of the VL dedicated to this message, taking into account application-level requirements in the form

of a maximum delivery time and a message size. Note that a major difficulty to guarantee the delivery time of a message is that it depends on the network traversal time, which in turn depends on the number of configured VLs, on their transmission parameters and on how they are routed in the network. However, as observed by Lauer *et al.* in [11], "the upper bounds on worst-case traversal time [...] are small compared to periods and duration of the functions" (the experimental results on industrial systems reported in [5,15] lead to the same observation). In practice, the worst-case network traversal time is in the order of few ms, whereas the maximum delivery times of the messages are comparable to partition periods, and are thus in the order of several tens or hundreds of ms. A pragmatic approach to VL design is therefore to assume an upper bound Δ on the worst-case network traversal time, to design the VL using maximum delivery times decreased by Δ, and then to check *a posteriori* that this bound is satisfied.

There are usually several feasible values of the BAG and MFS parameters that allow to guarantee the message delivery time. It is therefore convenient to use the feasible BAG and MFS values minimizing the bandwidth consumption since this will give more flexibility to add future VLs. For instance, to transmit a 100 bytes message within a maximum delivery time of 50 ms, it is possible to use a VL with (1) BAG=8 ms and MFS=64 bytes (17 bytes payload), or with (2) BAG = 32 ms and MFS=147 bytes (100 bytes payload). In these two options the temporal requirement is respected, but the bandwidth is 64 Kbps for the first option and 36.75 Kbps for the second. Although trivial, this example clearly shows the influence of the VL transmission parameters on the quality of a design.

We now turn to the question of what VLs have to be created. A dedicated VL has to be created whenever a source partition sends a message to receivers located on other modules. It may however occur that a source partition sends several messages to the same set of receivers. In this case, the system integrator can choose to use a separate VL for each message, or to aggregate all data into a single message and thus to use a single VL, and he can even choose to use an intermediate strategy. As we show in Section 3, the choice made by the system integrator can have a strong influence on the bandwidth consumption.

Finally, once the VLs are defined, the system integrator has to route them in the AFDX network. The routing problem amounts to finding one and only one multicast tree between the source and the destinations of each VL whilst guaranteeing that the amount of reserved bandwidth on each link is lower than its capacity. Since an upper-bound on the worst-case network traversal time has been assumed, message delivery times are guaranteed by design and it is therefore not suitable to minimize the network latency. A more appropriate design goal is to maximize the minimum residual capacity of the network links since ensuring a fair load distribution among network links eases the introduction of new VLs, or the modification of existing ones.

1.3 Related work

Although many works have been devoted to the design of avionic architectures, the problems addressed in this paper have been largely overlooked. To the extend of our knowledge, there is no previous work on how to define the VLs. All previous works regarding VLs focus on the analysis of the time required to transmit a frame through the network. Different methods have been proposed, such as Network Calculus [6], probabilistic analysis [15], trajectory approach [12], ILP [11], etc.

Although it seems that there is no previous work on the VL routing problem, similar problems have been considered for telecommunication networks. In the special case where we only have one VL, the problem becomes related to the computation of a minimum cost tree, known as a Steiner tree [8, 9]. This Steiner tree problem is NP-complete [16]. Another special case is obtained when each VL has a single destination, yielding a single-path routing problem. This problem is also known to be NP-complete [13]. Not much propositions can be found for routing several multicast demands at once. The authors in [16] look into finding multicast trees for multicast traffic requests that arrive one-by-one, while minimizing the maximum link utilization. Many others also propose multiobjective multicast routing algorithms for a single multicast traffic request in an already loaded network [7, 20].

1.4 Contribution

Our contribution is threefold. We first show how to set the BAG and MFS parameters of a VL so as to minimize the reserved bandwidth while transmitting the data within their maximum delivery time. We next consider the case where a source partition has to send multiple messages to the same set of receivers. We present several closed-form results and efficient numerical algorithms for aggregating messages into super-messages so as to minimize the bandwidth consumption. Finally, we propose an exact integer-linear programming formulation of the routing problem that can be used to route thousands of VLs so as to maximize the minimal residual capacity of the links. All the proposed approaches allow to reduce the bandwidth consumption and thus to get more flexibility for adding new VLs or modifying existing ones. We illustrate the benefits of the proposed design approaches on a benchmark inspired from an industrial case study.

1.5 Organization

The paper is organized as follows. Section 2 is devoted to the analysis of the optimal transmission parameters to send a single message through a VL. In Section 3, we consider the case where a source partition has to send multiple messages to the same set of receivers and analyze the optimal strategy to

minimize the bandwidth consumption. Section 4 presents an exact integer-linear programming formulation to solve the routing problem. Experimental results are presented in Section 5 and some conclusions are drawn in Section 6.

2 Optimal Transmission Parameters of a VL

2.1 Problem statement

We consider the transmission of a message from a source partition to a set of destinations (located on modules other than that of the source's). Let s denote the size of the message in bytes. The message can be fragmented into n frames, each having a header of size c bytes. The maximum size of the frame's payload is $f \in \mathbb{N}$ bytes, and it is assumed that $f_{min} \leq f \leq f_{max}$. In AFDX networks, the values of these parameters are $c = 47$ bytes, $f_{min} = 17$ bytes and $f_{max} = 1471$ bytes, so the minimum frame size is 64 bytes and the maximum one is 1518 bytes. The payload size f and the number of frames n have to be such that $n\,f \geq s$.

The delay between the transmision of two consecutive frames is $bag = 2^k$ ms, where $k \in \{0, 1, \ldots, 7\}$, so that the total delay between the transmission of the first frame and that of the last frame is $(n-1)\,bag$ ms. Let Δ denote an upper bound on the time required to transmit a frame of size 1518 bytes from one point of the netwok to another. Then the total delay between the transmission of the first frame of the message by the source partition and the reception of the last frame by the destinations is upper bounded by $(n-1)\,bag + \Delta$. It is assumed that this total delay has to be lower than a given constant $\delta + \Delta$, so the parameters n and k have to be chosen such that $(n-1)\,2^k \leq \delta$.

Since at most $c + f$ bytes are transmitted in bag seconds, the total bandwidth to be reserved for this communication is $bw = (c + f)/bag$. The problem amounts to finding the parameters n, f and bag such that the reserved bandwidth is minimized, while ensuring the end-to-end delay constraint of the message delivery. The problem can thus be stated as follows,

$$\begin{aligned} &\text{minimize } \frac{f + c}{2^k} && \text{(OPT)}\\ &\text{subject to}\\ &\qquad (n-1)\,2^k \leq \delta,\\ &\qquad n\,f \geq s,\\ &\qquad f_{min} \leq f \leq f_{max},\\ &\qquad k \in \{0, 1, \ldots, 7\},\\ &\qquad n, f \in \mathbb{N}. \end{aligned}$$

2.2 Derivation of the optimal parameters

In what follows, we introduce important Lemmas and Propositions that help in deriving the optimal parameters.

Lemma 1 *Problem* (OPT) *has a solution if and only if* $\left\lceil \frac{s}{f_{max}} \right\rceil \leq 1+\delta$.

Proof See Appendix A.1. □

In the following, we let $n_{min} = \left\lceil \frac{s}{f_{max}} \right\rceil$ and $n_{max} = 1+\delta$. We shall assume that $n_{min} \leq n_{max}$ and thus that the set $\Omega = \{n_{min}, n_{min}+1, \ldots, n_{max}\}$ of all feasible values of n is non empty. The following proposition shows how to find the optimal parameters for the Virtual Link.

Proposition 1 *Let* $f(n) = \max(\lceil \frac{s}{n} \rceil, f_{min})$, $k(n) = \min(7, \left\lfloor \log_2 \left(\frac{\delta}{n-1} \right) \right\rfloor)$, *and* $bw(n) = (f(n)+c)\, 2^{-k(n)}$ *for* $n \in \Omega$. *Let* n^* *be a minimum in* Ω *of* $bw(n)$, *i.e.,* $bw(n) \geq bw(n^*)$ *for all integers* $n \in \Omega$. *Then,* $(n^*, f(n^*), k(n^*))$ *is an optimal solution of Problem* (OPT).

Proof See Appendix A.2. □

Proposition 1 implies that finding the optimal solution of problem (OPT) reduces to finding the minimum of $bw(n)$ over the set Ω. To this end, let us introduce the sequence $\{n_q\}_{q \in \mathbb{N}}$ defined by

$$n_q = 1 + \left\lfloor 2^{-q} \delta \right\rfloor \quad , q \in \mathbb{N}, \tag{1}$$

and note that $n_q > n_{q+1}$ for all for $q \in \mathbb{N}$. These numbers will play a key role in determining the optimal solution, as will be shown below.

As an example on the above, Figure 2 shows the function $bw(n)$ for s=2000 bytes and δ=100 ms. Note that $n_{min} = 2$, $n_{max} = 101$ and $n_6 = 2$. The minimum of $bw(n)$ is 16.36 and is obtained for $n = n_6 = 2$. The frame size is $f(n_6)$=1000 bytes and the *bag* is 2^6 ms.

Figure 3 shows the function $bw(n)$ for s=2000 bytes and δ=400 ms. Here we have $n_{min} = 2$, $n_{max} = 401$ and $n_7 = 4$. The minimum of $bw(n)$ is 4.27 and is obtained for $n = n_7 = 4$. The frame size is $f(n_7)$=500 bytes and the *bag* is 2^7 ms.

We shall first state several properties of the functions $f(n)$ and $k(n)$ that will be helpful in determining the optimal strategy.

Lemma 2 *For all integers* $n \geq 1$ *and all integers* $q \in \{0, \ldots, 6\}$, *we have* $k(n) = q$ *if and only if* $n_{q+1} < n \leq n_q$. *Moreover* $k(n) = 7$ *if and only if* $n \leq n_7$.

Proof See Appendix A.3. □

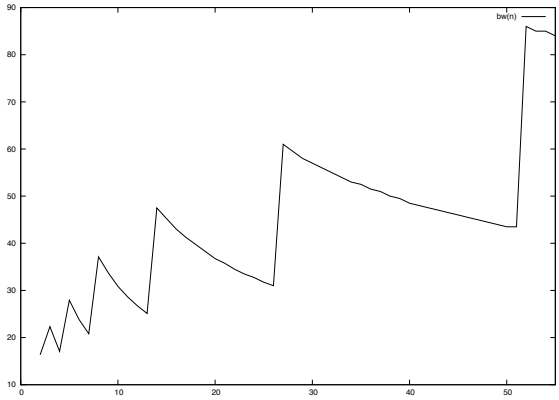

Fig. 2: Function $bw(n)$ for δ=100 ms and s=2000 bytes.

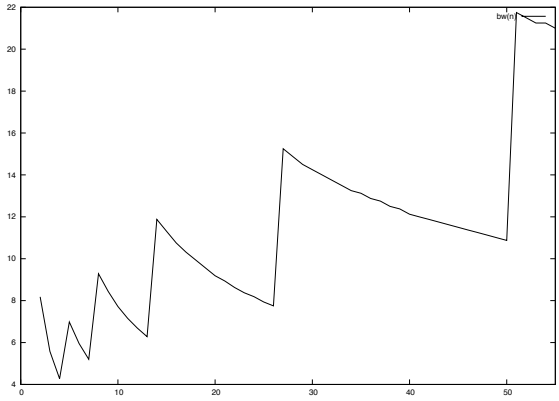

Fig. 3: Function $bw(n)$ for δ=400 ms and s=2000 bytes.

Lemma 3 *For all* $q \in \mathbb{N}$, $2n_{q+1} - 1 \leq n_q \leq 2n_{q+1}$.

Proof See Appendix A.4. □

Lemma 4 *We have (a)* $f(n) = f_{min}$ *if and only if* $n \geq \left\lceil \frac{s}{f_{min}} \right\rceil$, *and (b)* $f(n) = \left\lceil \frac{s}{n} \right\rceil$ *if and only if* $n < \left\lceil \frac{s}{f_{min}} \right\rceil$. *Therefore,* $f(n)$ *is a non-increasing function of* n *over* Ω, *which implies that* $f(n_q) \leq f(n_{q+1})$ *for all* $q \in \mathbb{N}$.

Proof See Appendix A.5. □

We now study the minimum of $bw(n)$ over Ω. We proceed in two steps. We show (a) that the minimum of $bw(n)$ is attained at a point n^* in $\{n_0, n_1, \ldots, n_7\}$, and (b) that n^* is the minimum value of the sequence $\{n_q\}_{q\in\mathbb{N}}$ in Ω. To prove (a), we first prove that this function is piecewise non-increasing.

Lemma 5 *For all $n \leq n_7$ we have $bw(n) \geq bw(n_7)$. Moreover, for all $q \in \{0, 1, \ldots, 6\}$ and for all $n \in (n_{q+1}, n_q]$, $bw(n) \geq bw(n_q)$.*

Proof See Appendix A.6. □

Using the piecewise monotonic behaviour of $bw(n)$, Lemma 6 below proves that the minimum of $bw(n)$ over Ω is necessarily attained at a point of the sequence $\{n_q\}_{q\in\mathbb{N}}$.

Lemma 6 *Let $\mathcal{J} = \Omega \cap \{n_0, n_1, \ldots, n_7\}$. There exists $n^* \in \mathcal{J}$ such that $bw(n^*) \leq bw(n)$ for all integers $n \in \Omega$.*

Proof See Appendix A.7. □

Since the minimum of $bw(n)$ is in $\{n_0, n_1, \ldots, n_7\}$, we will now compare the values of this function at these points. Lemma 7 shows that $bw(n_q) > bw(n_{q+1})$.

Lemma 7 *For all integers $q \in \{0, 1, \ldots, 6\}$, it holds that $bw(n_q) > bw(n_{q+1})$.*

Proof See Appendix A.8. □

We can now prove that the minimum of $bw(n)$ is attained by the smallest value of the sequence $\{n_q\}_{q\in\mathbb{N}}$ belonging to the set Ω. This is formally stated in the following proposition.

Proposition 2 *Let $n^* = \min\{x : x \in \mathcal{J}\}$. Then, $(n^*, f(n^*), k(n^*))$ is an optimal solution of Problem* (OPT).

Proof See Appendix A.9. □

We are now in position to state our main result.

Theorem 1 *If $s \leq f_{max}$, then $n^* = n_7$, $f^* = f(n_7)$ and $k^* = 7$ is an optimal strategy. Otherwise, $k^* = \min\left(7, \left\lfloor \log_2\left(\frac{\delta}{n_{min}-1}\right)\right\rfloor\right)$, $n^* = 1 + \lfloor 2^{-k^*}\delta \rfloor$ and $f^* = f(n^*)$ is an optimal strategy.*

Proof From Proposition 2, an optimal strategy is obtained by choosing the largest value of k in $\{0, \ldots, 7\}$ such that $n_k = 1 + \lfloor 2^{-k}\delta \rfloor \geq n_{min}$. If $n_{min} = 1$, this value of k is clearly $k = 7$. Otherwise, $k \leq \left\lfloor \log_2\left(\frac{\delta}{n_{min}-1}\right)\right\rfloor$, that is $k = \min\left(7, \left\lfloor \log_2\left(\frac{\delta}{n_{min}-1}\right)\right\rfloor\right)$. □

To summarize, the above results prove that an optimal strategy for sending the message is always obtained by sending n_7 frames and choosing $k = 7$ when the message size is lower than or equal to f_{max}. Otherwise, an optimal strategy is obtained by setting $k = \min\left(7, \left\lfloor \log_2\left(\frac{\delta}{n_{min}-1}\right)\right\rfloor\right)$, with $n_{min} = \left\lceil \frac{s}{f_{max}} \right\rceil$, and fragmenting the message into n_k frames of size $f(n_k)$ that are transmitted one after the other, separated by 2^k ms.

Before concluding this section, we present below two basic properties of the optimal solution that will be used in the following. The next lemma provides bounds on the optimal number of frames.

Lemma 8 *If $s > f_{max}$ and if $\delta \leq 2^7 (n_{min} - 1)$, $n_{min} \leq n^* \leq 2\,[n_{min} - 1]$.*

Proof See Appendix A.10. □

It is worthwhile noticing that the above lemma implies that if $f_{max} < s < 2f_{max}$, i.e., if $n_{min} = 2$, then $n^* = 2$. The following lemma is concerned with the optimal size of the frames.

Lemma 9 *If $s > f_{max}$ and if $\delta \leq 2^7 (n_{min} - 1)$, then $f^* > f_{min}$.*

Proof See Appendix A.11. □

3 Optimal strategy for sending multiple messages

3.1 Problem statement

We now consider the situation where the source partition (application program unit) has to send multiple messages to the destinations. Let $M > 1$ be the number of messages and assume that they are numbered from 1 to M. We let $\mathcal{M} = \{1, \ldots, M\}$ denote the set of messages. Message $i \in \mathcal{M}$ has a size equal to s_i bytes and has to be delivered before δ_i ms after the first bit has been transmitted[1]. In the rest of this section, the term *partition* is used in the set-theoretic sense and, unless indicated otherwise, is not to be confused with the previous description of application program units.

When we consider the transmission of multiple messages, several strategies are possible. The system designer can choose to use a separate VL for each message, or he can choose to aggregate all data into a single message and thus to use a single VL. Of course, intermediate strategies are also possible, i.e., the system designer can choose to partition the set $\mathcal{M}$ into $K \in \{2, \ldots, M-1\}$ subsets $\mathcal{A}_1, \ldots, \mathcal{A}_K$ such that $\mathcal{A}_i \cap \mathcal{A}_j = \emptyset$ for $i \neq j$ and $\cup_i \mathcal{A}_i = \mathcal{M}$, and to send all the messages belonging to each subset $\mathcal{A}_i$ as a single super-message

[1] In practice, all messages sent by a partition share the same maximum delivery time. The analysis presented here does not rely on this assumption, but it of course also applies in this case. The numerical results presented in this article consider both the case where the δ_i are equal and the case where they are different.

of size $s(\mathcal{A}_i) = \sum_{m\in\mathcal{A}_i} s_m$. Note that in this case the upper bound on the delivery time of the super-message i is $\delta(\mathcal{A}_i) = \min_{m\in\mathcal{A}_i} \delta_m$.

The question we want to answer here is that of the strategy that minimizes the reserved bandwidth. For any subset $\mathcal{A} \subset \mathcal{M}$, let $bw(\mathcal{A})$ denote the minimum bandwidth to send a super-message of size $s(\mathcal{A}) = \sum_{m\in\mathcal{A}} s_m$ with a delivery time lower than $\delta(\mathcal{A}) = \min_{m\in\mathcal{A}} \delta_m$. Note that the optimal parameters $k(\mathcal{A})$, $n(\mathcal{A})$ and $f(\mathcal{A})$ for the associated Virtual Link are given in Theorem 1. We let also $\mathcal{P}(\mathcal{A})$ denote the set of all partitions of a subset $\mathcal{A}$ of $\mathcal{M}$. The problem amounts to finding a partition $\{\mathcal{A}_i\}_{i=1,\dots,K}$ of $\mathcal{M}$ minimizing the total bandwidth $\sum_{i=1}^{K} bw(\mathcal{A}_i)$. Formally,

$$\begin{aligned} &\text{minimize} \sum_{i=1}^{K} bw(\mathcal{A}_i) && \text{(PART)}\\ &\text{subject to} \\ &\qquad \{\mathcal{A}_i\}_{i=1,\dots,K} \in \mathcal{P}(\mathcal{M}) \\ &\qquad K \in \{1,\dots,M\} \end{aligned}$$

A solution to problem PART is thus a partition of the set $\mathcal{M}$ of all messages. Note that the number of solutions correspond to the cardinality of $\mathcal{P}(\mathcal{M})$ and that it is given by the Bell number of order M:

$$|\mathcal{P}(\mathcal{M})| = \frac{1}{e}\sum_{k=0}^{\infty}\frac{k^M}{k!}, \tag{2}$$

which implies that we cannot hope finding the optimal solution by an exhaustive enumeration of all solutions for large values of M. We thus first consider some simple special cases for which closed-form results can be obtained in Section 3.2, before presenting general numerical approaches in Section 3.3 and 3.4.

3.2 Some simple special cases

In the following, we will focus on the case where the message sizes are lower than or equal to f_{max} and the upper bound on the delivery time of the super-message is strictly below 128 ms. This is formally stated in the following assumption.

Assumption 2 *For each message $m \in \mathcal{M}$, we have $f_{min} \le s(m) \le f_{max}$ and $\delta(m) < 2^7$.*

Throughout this section, this assumption is adopted. This is clearly the most interesting case, and it is also the most tractable as indicated by the following lemma.

Lemma 10 *Let $\mathcal{A} \subset \mathcal{M}$ and $\{\mathcal{A}_i\}_{i=1,\dots,K}$ be a partition of $\mathcal{A}$ such that $s(\mathcal{A}_i) \leq f_{max}$ for all i. It holds that $k(\mathcal{A}_i) = 7$, $n(\mathcal{A}_i) = 1$ for all i, and $\sum_{i=1}^{K} bw(\mathcal{A}_i) = 2^{-7}\ (s(\mathcal{A}) + Kc)$.*

Proof See Appendix B.1. □

Note that Lemma 10 implies that the total reserved bandwidth depends on the particular partition that is considered only through the number K of super-messages, as long as the size of each one is below f_{max}. We immediatly get the following corollary.

Corollary 1 *We have*

$$\frac{bw(\mathcal{A})}{\sum_{i=1}^{K} bw(\mathcal{A}_i)} = 2^{7-k(\mathcal{A})}\ \frac{f(\mathcal{A}) + c}{s(\mathcal{A}) + Kc}, \tag{3}$$

for any subset $\mathcal{A} \subset \mathcal{M}$ and any partition $\{\mathcal{A}_i\}_{i=1,\dots,K}$ of $\mathcal{A}$ such that $s(\mathcal{A}_i) \leq f_{max}$ for all i.

Using Corollary 1 we first show that as long as $s(\mathcal{A}) \leq f_{max}$, the total reserved bandwidth is minimized by sending all messages of $\mathcal{A}$ in a single Virtual Link.

Proposition 3 *For $\mathcal{A} \subset \mathcal{M}$ such that $s(\mathcal{A}) \leq f_{max}$ and a partition $\{\mathcal{A}_i\}_{i=1,\dots,K}$ of $\mathcal{A}$, it holds that*

$$\frac{bw(\mathcal{A})}{\sum_{i=1}^{K} bw(\mathcal{A}_i)} = 1 - (K-1)\ \frac{c}{s(\mathcal{A}) + Kc}, \tag{4}$$

which implies that $bw(\mathcal{A}) < \sum_{i=1}^{K} bw(\mathcal{A}_i)$ if $K \geq 2$.

Proof Since $s(\mathcal{A}) \leq f_{max}$, Lemma 10 implies that $k(\mathcal{A}) = 7$, $n(\mathcal{A}) = 1$ and $bw(\mathcal{A}) = 2^{-7}\ (s(\mathcal{A}) + c)$. Thus, with Corollary 1 we obtain $\frac{bw(\mathcal{A})}{\sum_{i=1}^{K} bw(\mathcal{A}_i)} = \frac{s(\mathcal{A})+c}{s(\mathcal{A})+Kc}$. □

Note that in practice, most partitions (program units) send a small number of short messages with a maximum delivery time that is smaller than 128 ms. In this case, we can directly apply the above proposition and assert that the total reserved bandwidth is minimized by aggregating all messages into a single super-message.

Example 1 Assume that there are $M = 6$ messages to be sent and that the sum of their sizes is 148 bytes. According to formula (4), we can reduce by about 55% the total reserved bandwidth by sending all messages as a single super-message instead of sending each one in a separate Virtual Link.

Is it always optimal to aggregate the messages? The answer is no, as indicated by the following proposition which considers the case where a partition (program unit) has to send two messages satisfying the assumptions of Lemma 10, and proves that it is optimal to send each message separately if the sum of their sizes is greater than f_{max}.

Proposition 4 *Let $\mathcal{A} \subset \mathcal{M}$ be such $n_{min}(\mathcal{A}) = 2$ and $\{\mathcal{A}_1, \mathcal{A}_2\}$ be a partition of $\mathcal{A}$ such that $s(\mathcal{A}_i) \leq f_{max}$, $i = 1, 2$. Then $bw(\mathcal{A}) \geq bw(\mathcal{A}_1) + bw(\mathcal{A}_2)$.*

Proof See Appendix B.2. □

Example 2 Assume that there are two messages to be sent with the following parameters : $s_1 = s_2 = 1024$ bytes and $\delta_1 = \delta_2 = 30$ ms. In this case, we obtain $bw(\{1\}) + bw(\{2\}) = 2\,\frac{1024+47}{2^7} = \frac{1024+47}{2^6}$. However, since $n_{min}(\{1,2\}) = 2$ we obtain $k(\{1,2\}) = \lfloor log_2(30) \rfloor = 4$ and $n(\{1,2\}) = 2$, which yields $bw(\{1,2\}) = \frac{1024+47}{2^4}$. Aggregating the messages thus requires four times more bandwidth than sending them separately.

Proposition 4 implies that aggregating messages of relatively small sizes, under Assumption 2 that is, in one super-message is not trivial if its size exceeds f_{max}. In addition, solving PART for general cases, especially when Assumption 2 does not hold, is not an easy task. Resorting to numerical algorithms becomes hence inevitable to determine optimal strategies. For this purpose, an exact branch-and-bound algorithm is first introduced in Section 3.3, then followed by a greedy one in Section 3.4.

3.3 Branch-and-bound algorithm

Problem PART can be solved using the branch-and-bound algorithm 1. As a reminder, a branch-and-bound algorithm works in two steps: the branching step consists of splitting the problem in two or more sub-problems whereas the bounding step defines a lower and an upper bound for the current sub-problem. When the lower bound meets the upper bound, it is not necessary to continue exploring from this state. The recursive algorithm 1 takes as input the current partial solution which is defined by a set of super-messages and a set of messages not yet handled. The initial state is defined by an empty set of super-messages. Starting from a partial solution, the algorithm generates another partial solution by taking a message that is not present in any super-message and merging it to a super-message or putting it alone.

In the following, we describe the details of this algorithm.

3.3.1 Initial upper bound

The branch-and-bound algorithm is started with an initial value of the upper bound. In our implementation, we have used the cost returned by the greedy algorithm described in Section 3.4. As will be shown later, this heuristic often provides near-optimal solutions. As described in Algorithm 1, the upper bound is updated each time an improving complete solution is discovered.

Algorithm 1

procedure BandB($\mathcal{S}$: set of super-messages, $\mathcal{N}$: set of remaining messages)
  if $|\mathcal{N}| > 0$ **then**
    if $lowerbound(\mathcal{N}) + bw(\mathcal{S}) \geq bw(sol)$ **then**
      return ▷ Prune.
    end if
    for $i \leftarrow 1$ **to** $|\mathcal{S}|$ **do**
      $\mathcal{C} \leftarrow \mathcal{S}$ ▷ Let $\mathcal{C} = \bigcup \mathcal{C}_j$ represent the set of super-messages.
      $\mathcal{C}_i \leftarrow \mathcal{C}_i \cup \{\mathcal{N}_1\}$ ▷ Merge $\mathcal{N}_1$ to the existing super-message $\mathcal{C}_i$.
      BandB($\mathcal{C}, \mathcal{N} \setminus \{\mathcal{N}_1\}$)
    end for
    $\mathcal{S} \leftarrow \mathcal{S} \cup \{\{\mathcal{N}_1\}\}$ ▷ Add the message alone.
    BandB($\mathcal{S}, \mathcal{N} \setminus \{\mathcal{N}_1\}$)
  else
    if $bw(sol) > bw(\mathcal{S})$ **then**
      $sol \leftarrow \mathcal{S}$
    end if
  end if
end procedure

$sol \leftarrow$ Greedy($\mathcal{M}$) ▷ Initial upper bound.
BandB({}, $\mathcal{M}$) ▷ Run the Branch and Bound.

3.3.2 Lower bound

A lower bound on the optimal cost-to-go is generated in each node of the search tree. This lower bound is based on the following conjecture.

Conjecture 1 It holds that

$$\sum_{i=1}^{K} bw(\mathcal{A}_i) > \sum_{m \in \mathcal{A}} \left\lceil \frac{s(m)}{n(m)} \right\rceil 2^{-k(m)} \quad \forall \{\mathcal{A}_i\}_{i=1,\ldots,K} \in \mathcal{P}(\mathcal{A}), \forall \mathcal{A} \subset \mathcal{M} \tag{5}$$

We show below that this conjecture holds true in an important special case. The proof is based on the following lemma.

Lemma 11 *Under assumption 2, Conjecture 1 is equivalent to*

$$bw(\mathcal{A}) > 2^{-7}\, s(\mathcal{A}) \quad \forall \mathcal{A} \subset \mathcal{M}. \tag{6}$$

Proof See Appendix C.1. □

The interest of Lemma 11 is that it provides an equivalent formulation of Conjecture 1 that is independent of a particular solution: it depends only on the sum of the sizes of the messages in the set $\mathcal{A}$ and on the minimum of their delivery times. This allows to obtain the following Proposition.

Proposition 5 *Under Assumption 2, Conjecture 1 holds true for all* $\mathcal{A} \subset \mathcal{M}$.

Proof See Appendix C.2 □

3.3.3 Variable ordering

As in most cases, ordering plays an important role in ameliorating the performance of a branch-and-bound algorithm. In our work, messages were sorted following a decreased order of sizes. Implementing this ordering scheme had a great impact on resolution times as it helped in obtaining greater lower bounds faster and hence pruning the search tree earlier (due to starting with messages of larger size).

3.4 Greedy algorithm

An initial upper bound is determined using a greedy algorithm that selects the state with the best lower bound until no messages are left (Algorithm 2). This greedy algorithm performs in $\mathcal{O}(M^2)$ so it can be used for large-scale problems.

Algorithm 2

function GREEDY($\mathcal{M}$: the set of messages)
  $\mathcal{S} \leftarrow \{\}$ ▷ The super-messages of the partial solution.
  for $\forall m \in \mathcal{M}$ **do**
    $Q \leftarrow \{\}$
    for $i \leftarrow 1$ **to** $|\mathcal{S}|$ **do**
      $\mathcal{C} \leftarrow \mathcal{S}$ ▷ Let $\mathcal{C} = \bigcup \mathcal{C}_j$ represent the set of super-messages.
      $\mathcal{C}_i \leftarrow \mathcal{C}_i \cup \{m\}$ ▷ Merge m to the existing super-message $\mathcal{C}_i$.
      $Q \leftarrow Q \cup \{\mathcal{C}\}$
    end for
    $\mathcal{S} \leftarrow \mathcal{S} \cup \{\{m\}\}$ ▷ Message alone.
    $Q \leftarrow Q \cup \{\mathcal{S}\}$
    $\mathcal{S} \leftarrow \text{argmin}_{Q_x \in Q}(bw(Q_x))$ ▷ Best partial solution.
  end for
  return $\mathcal{S}$
end function

4 Optimal Routing of the VLs

As illustrated in Figure 4, let $\mathcal{N} = \{1, \ldots, N\}$ be a set of N nodes representing the AFDX network ($n \in \mathcal{N}$ may correspond to an AFDX switch or End System). Here an End System corresponds to a processing module. Denote by $\mathcal{E}$ the set of directed edges (links) between nodes such that no edge exists between

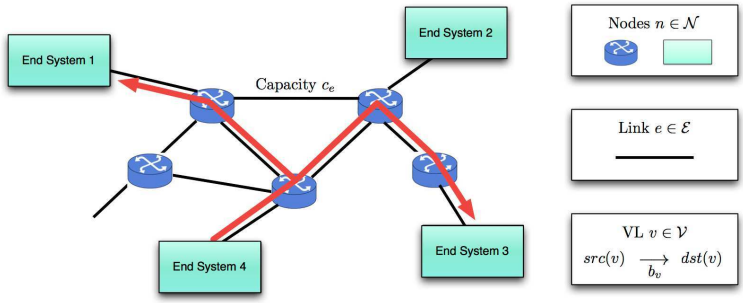

Fig. 4: Notations for modeling an AFDX network.

two End System nodes, that is to say, End Systems are interconnected through AFDX switches only. The set $\mathcal{C} = \{c_e : e \in \mathcal{E}\}$ represents link capacities.

Consider a set $\mathcal{V} = \{1, \ldots, V\}$ of V VLs to be deployed in the AFDX network. After its definition following the analysis in Section 3, VL $v \in \mathcal{V}$ is characterized by the following:

- a source node (End System) $src(v) \in \mathcal{N}$,
- a set of destination nodes (End Systems) $dest(v) \subseteq \mathcal{N} \setminus \{src(v)\}$, and the corresponding number of destinations $K_v = |dst(v)|$,
- and a bandwidth b_v.

As the VLs need to be routed in the AFDX network, the problem amounts to finding one and only one Steiner tree for each VL, thereby indicating the links that shall be traversed.

In order to ensure an increased residual capacity in the network, the optimization associated to the VL routing problem is taken to be the minimization of the maximum link load (links actually represent switch interfaces), that is,

$$\text{Minimize } \rho = \max_{e \in \mathcal{E}} \left(\frac{y_e}{c_e} \right),$$

where y_e represents the total reserved bandwidth on link e.

In this paper, we opted for an exact node-link formulation [1] based on Mixed Integer Linear Programming (MILP). Thus, for every node $n \in \mathcal{N}$, let $\Gamma^+(n) \subseteq \mathcal{E}$ denote the set of incoming links to n and $\Gamma^-(n) \subseteq \mathcal{E}$ the set of outgoing links from n as can be seen in Figure 5.

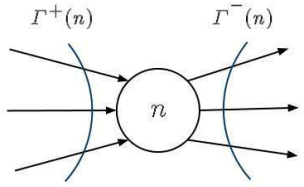

Fig. 5: Links to and from node n.

The formulation is hence as follows,

$$\min \rho$$

s.t.

$$\sum_{e\in\Gamma^+(n)} x_v^e - \sum_{e\in\Gamma^-(n)} x_v^e = h_{n,v} \qquad ,\forall n, v, \tag{7}$$

$$y_v^e\, M \geq x_v^e \qquad ,\forall v, e, \tag{8}$$

$$y_v^e \leq x_v^e \qquad ,\forall v, e, \tag{9}$$

$$\sum_{e\in\Gamma^+(n)} y_v^e \leq 1 \qquad ,\forall n, v, \tag{10}$$

$$y_e = \sum_{v\in\mathcal{V}} y_v^e\, b_v \qquad ,\forall e, \tag{11}$$

$$y_e \leq c_e\, \rho \qquad ,\forall e, \tag{12}$$

$$y_v^e \in \{0, 1\} \qquad ,\forall v, e, \tag{13}$$

$$x_v^e \in \{0, \ldots, K_v\} \qquad ,\forall v, e, \tag{14}$$

where x_v^e can be thought of as the number of destinations VL v is addressing via link e (*cf.* Figure 6). The equality in constraint (7) indicates that the difference in the number of addressed destinations by VL v between before and after entering a node n should be equivalent to $h_{n,v}$ where,

$$h_{n,v} = \begin{cases} -K_v & \text{if } n = src(v), \\ 1 & \text{if } n \in dst(v), \\ 0 & \text{otherwise.} \end{cases}$$

Evidently the source node (End System) should address all of the destinations. A destination node (End System) should be the last in the chain as it cannot re-route information, and only receives what is addressed to it from VL v and hence the value of $h_{n,v} = 1$. Finally, an intermediate node (switch) should transfer whatever is addressed by the VLs from its input links to its output links.

The constant value M in (8) is considered as a large number, *e.g.*, a value greater than the maximum number of destinations in the VLs. Consequently,

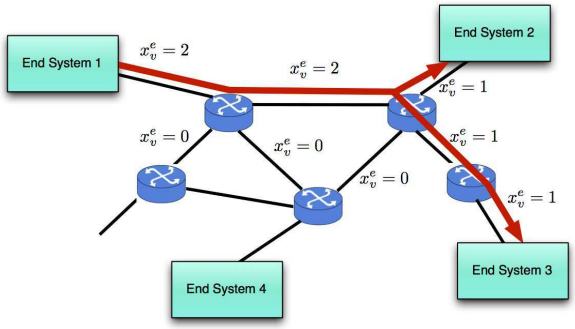

Fig. 6: An example on the number of destinations x_v^e addressed by a VL v on the links, following a given multicast tree between End System 1 (source) and End Systems 2 and 3 (destintations).

the boolean y_v^e indicates whether link e is traversed by VL v or not, i.e. if $x_v^e > 0$ then $y_v^e = 1$ indicating the passage of v in this link, otherwise it can be either 0 or 1 but will be set to 0 due to constraint (9). Constraint (10) ensures that each VL is routed along a tree where no node is visited more than once. Constraints (11) and (12) define the total reserverd bandwidths and the maximum utilization rates on the links, respectively. Constraints (13) and (14) represent the domains for the decision variables y_v^e and x_v^e, respectively.

As a result of the preceding node-link formulation, Steiner trees for the various VLs can be assigned based on the decision variables y_v^e.

5 Results

5.1 Message Aggregation

The optimal strategy for sending multiple messages (*cf.* Section 3) was first compared to the strategy which consists of sending each message separately in a different VL ("1/VL"). In addition, the strategy of grouping all messages together and defining one unique VL was also considered ("All in 1"). Furthermore, the results of the greedy algorithm from Section 3.4 are presented to demonstrate the efficiency of this algorithm.

To compare these different strategies, several experiment sets, with 100 instances each, were considered. Each instance consisted of n randomly generated messages, defined by their sizes (in bytes) and their maximum delivery times δ (in ms). The message sizes were chosen uniformly from the set $\{16, 32, 64, 128, 256, 512, 1024\}$ bytes, thereby favoring small sizes (as the mean is at 290 bytes). As observed in [15], adopting small sizes is more representative for messages in real-world instances. In addition, values for maximum delivery times (δ) were chosen to respect assumption 2.

Table 1: Average relative gap in % to the optimal solution, for $n = 10$ and δ variable.

δ	{30, 60, 100}	{60, 100, 120}	30	60	100
1/VL	12.12	11.58	12.19	10.54	10.69
Greedy	.06	.31	<.01	<.01	.01
All in 1	282.62	96.93	293.57	98.98	.90

Table 2: Average relative gap in % to the optimal solution, for $\delta \in \{30, 60, 100\}$ and n variable.

n	5	10	15
1/VL	10.35	12.12	11.64
Greedy	.03	.06	.06
All in 1	176.81	282.62	301.61

Tables 1 and 2 demonstrate the relative gap, in percentage, between the optimal solution of PART and the various strategies, while varying δ and n respectively.

We observe that the strategy of placing all messages in a single VL gives severly unoptimized solutions, unless the maximum delivery time is large enough. Indeed, with an average message size of 290 bytes, instances with only 5 messages can have a total size which exceeds f_{max}. Hence, in most cases and similar to proposition 4, it is not optimal to aggregate the messages in a single VL. This is also verified in Table 2, where the relative gap to the optimal solution exceeds 170% when grouping all messages in a single VL, so as to repect the maximum delivery times. As the total size exceeds f_{max} and the minimum value of the maximum delivery times has to be respected, it becomes favorable to choose a small *bag* value, which maximizes the bandwidth in the process.

We can also observe that the strategy of sending each message separately in different VLs has a better performance at average, where the total bandwidth consumption is approximatively 12% greater than the optimal solution. However, as shown in Figure 7, the relative gap between the solution obtained by sending one message per VL and the optimal one highly depends on the set of messages considered. A closer look at the extreme cases shows that using a separate VL for each single message leads to acceptable performances when the message sizes are close to f_{max}, but that it leads to a significant waste of bandwidth when message sizes are small.

Finally, the greedy algorithm performs very well in all cases. The worst solution obtained, among all the tested instances (1400 instances in all, including ones with a greater number of messages and not presented in this paper), is only 2.6% above the optimal solution. Thus, this algorithm could replace the optimal algorithm when the number of messages becomes large (> 20-25 messages).

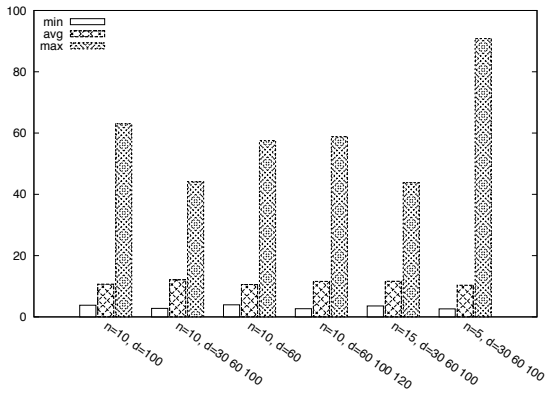

Fig. 7: Relative gap to the optimal solution in % when using one VL per message.

5.2 Routing

Assuming that the messages have been aggregated, and the VLs have been defined, we hereafter demonstrate the performance of the node-link formulation presented in Section 4 for routing the different VLs in the network. For solving the associated MILPs, the linear program solver CPLEX [10] from IBM ILOG was used. As for the topology, and for depicting as much as possible the AFDX network found on board aircrafts, the one shown in Figure 8 was considered. This topology consists of 7 AFDX switches and 6 End Systems. The End Systems represent the group of sources and destinations (e.g. partitions) for VLs.

All links are full duplex at 100Mbps. Given $nbVL$ VLs, all End Systems were evenly attributed a number of VLs in which they act as a source (e.g. if $nbVL = 12$ then each End System in our topology would act as source for 2 VLs). For each VL, the destinations were chosen so that 1 to 5 End Systems (other than the source) would be uniformly selected. Bandwidth requirements were chosen based on an exponential distribution with an average of 125Kbps (so that for large examples link capacities are not exceeded). Several examples were generated with $nbVL = \{100, 300, 500, 700, 1000\}$.

For each case of $nbVL$, 100 instances were generated and solved following the exact node-link formulation. In all of the cases, optimal solutions for routing the VLs where obtained under 10 seconds.

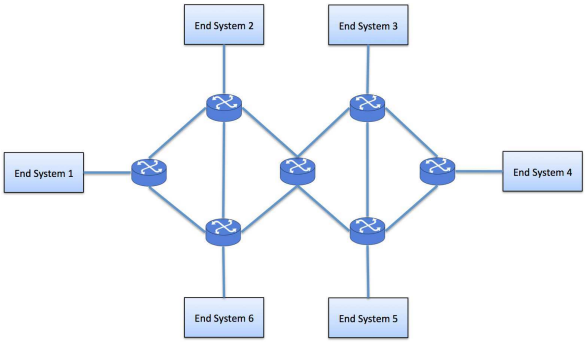

Fig. 8: Topology considered for experimentations. Set of 7 AFDX switches and 6 End Systems.

5.3 Benchmark

In order to validate the benefit of both the message aggregation and routing approaches in industrial applications, a benchmark was generated. The topology was considered similar to that of Figure 8, except that each End System was replaced with 12 separate ones depicting different processing modules, and connected to the same switch. Each module itself hosted a set of 12 partitions (application program units), giving a total of 864 partitions in the system.

Messages were first generated in such a way that each partition would be a source to a number of messages uniformly chosen from 2 to 9. Each message was destined to partitions uniformly chosen among all possible partitions (the number of destinations was limited between 1 and 43). Message sizes and maximum delivery times were chosen from the sets $\{16, 32, 64, 128, 256, 512, 1024\}$bytes and $\{30, 60, 100\}$ms respectively. All these considerations lead to a total number of 5305 messages.

The optimal strategy for message aggregation lead to the definition of 1568 distinct super-messages, and hence VLs, in a couple of seconds. This reduction in the total number of VLs as compared to sending each of the 5305 messages in a separate VL, reduced the total demand in the network by about 10% from 114.3Mbps to 103.5Mbps.

Furthermore, routing the arising VLs using the exact node-link formulation, with an objective of minimizing the maximum link utilization rates, gave a solution at about 39% maximum utilization in a couple of minutes. To demonstrate the importance of obtained results, we need to consider the strategy that might be followed in avionics. The messages would be first placed in separate VLs, which are then routed to obtain a feasible solution while minimizing route lengths, for example. Doing so will not only give an increased total bandwidth requirement (aforementioned 10%), but also a routing scheme with a maximum link utilization rate of about 53%.

This clearly shows that, using our strategies (aggregation and link utilization rate minimization) leads to more flexibility in the network. That is, residual capacities are increased in the network so as to allow any future evolution, that is the introdution of new messages or VLs. In worst-case scenarios this might not be possible if messages are not efficiently aggregated and link utilization rates are not decreased.

6 Conclusion

In AFDX networks, bandwidth is a valuable resource that should not be wasted. We have shown how to compute the optimal transmission parameters of a VL so as to minimize the bandwidth consumption while transmitting the data within their maximum delivery times. We have also shown that the aggregation of messages destined to the same set of receivers can lead to significant bandwidth savings, and we have presented efficient algorithms to compute the optimal aggregation strategy. Finally, we have also proposed an exact integer-linear programming formulation of the VL routing problem allowing to maximize the residual capacity of the network links, thereby providing more flexibility for adding new VLs or modifying existing ones. The proposed methods provide the basis for a more efficient design of the VLs, and have to be completed later on by the verification of the worst-case network latencies.

A Proofs of the results in Section 2.1

A.1 Proof of Lemma 1

Assume that n is fixed. Then there exists at least one value of $f \in \mathbb{N}$ such that $\max\left(f_{min}, \lceil \frac{s}{n} \rceil\right) \leq f \leq f_{max}$ if and only if $\lceil \frac{s}{n} \rceil \leq f_{max}$, i.e., $n \geq \left\lceil \frac{s}{f_{max}} \right\rceil$. Similarly, there exists at least one value of $k \in \{0, 1, \ldots, 7\}$ such that $(n-1)2^k \leq \delta$ if and only if $n \leq 1 + \delta$. We conclude that the problem has a solution (n, f, k) if and only if we can find at least one value of $n \in \mathbb{N}$ such that both conditions are satisfied, i.e., $\left\lceil \frac{s}{f_{max}} \right\rceil \leq 1 + \delta$ holds.

A.2 Proof of Proposition 1

Assume that the value of $n \in \Omega$ is fixed. The minimum value of f is $f(n) = \max(f_{min}, \lceil \frac{s}{n} \rceil)$ and the maximum value of k is $k(n) = \min(7, \left\lfloor \log_2 \left(\frac{\delta}{n-1} \right) \right\rfloor)$. Thus, the minimum bandwidth for this fixed value of n is $bw(n)$. The optimal value of n is the value n^* which minimizes $bw(n)$ in the interval Ω.

A.3 Proof of Lemma 2

Assume first that $k(n) = q < 7$. We have $q \leq \log_2\left(\frac{\delta}{n-1}\right) < q+1$ if and only if $2^q \leq \frac{\delta}{n-1} < 2^{q+1}$. This is equivalent to $n > 1 + 2^{-(q+1)}\delta$ and $n \leq 1 + 2^{-q}\delta$. Since n is integer, this is equivalent to $n_{q+1} < n \leq n_q$. Assume now that $k(n) = 7$. Note that $7 \leq \left\lfloor \log_2\left(\frac{\delta}{n-1}\right)\right\rfloor$ is equivalent to $2^7 \leq \frac{\delta}{n-1}$, which holds if and only if $n \leq 1 + \lfloor 2^{-7}\delta \rfloor = n_7$.

A.4 Proof of Lemma 3

From the definition of n_{q+1}, we have $2^{-(q+1)}\delta < n_{q+1} \leq 1 + 2^{-(q+1)}\delta$, which yields $2^{-q}\delta < 2n_{q+1} \leq 2 + 2^{-q}\delta$. But since $2n_{q+1} \in \mathbb{N}$, $2n_{q+1} > 2^{-q}\delta$ implies that $2n_{q+1} \geq 1 + \lfloor 2^{-q}\delta \rfloor = n_q$. Moreover, since $2n_{q+1} \in \mathbb{N}$, $2n_{q+1} \leq 2 + 2^{-q}\delta$ implies that $2n_{q+1} \leq 1 + (1 + \lfloor 2^{-q}\delta \rfloor) = 1 + n_q$, i.e., $n_q \geq 2n_{q+1} - 1$.

A.5 Proof of Lemma 4

To prove (a), observe that $\left\lceil \frac{s}{n} \right\rceil \leq f_{min}$ if and only if $\frac{s}{f_{min}} \leq n$, which is equivalent to $n \geq \left\lceil \frac{s}{f_{min}} \right\rceil$. Statement (b) is just the contrapositive of (a). Finally, statement (b) implies that $f(n)$ is strictly decreasing on the interval $\left[1, \left\lceil \frac{s}{f_{min}} \right\rceil\right]$, while statement (a) implies that $f(n)$ is constant for $n \geq \left\lceil \frac{s}{f_{min}} \right\rceil$.

A.6 Proof of Lemma 5

Let $n \leq n_7$. From Lemma 2, $k(n) = 7$. Since $f(n)$ is non-increasing on the interval $[1, n_7]$, it yields $bw(n) \geq bw(n_7)$, as claimed. Let us now consider n such that $n_{q+1} < n \leq n_q$ with $q \in \{0, 1, \ldots, 6\}$. According to Lemma 2, $k(n) = q$ for $n_{q+1} < n \leq n_q$. Moreover, $f(n)$ is a non-increasing function of n. Therefore, $bw(n)$ is non-increasing on $(n_{q+1}, n_q]$ and thus $bw(n) \geq bw(n_q)$ for all integers n such that $n_{q+1} < n \leq n_q$.

A.7 Proof of Lemma 6

Observe first that $\mathcal{J} \neq \emptyset$ since $n_0 = n_{max} \in \Omega$. Let $n_{q_{max}} = \min\{x : x \in \mathcal{J}\}$ and assume on the contrary that there exists $m \in \Omega$ such that $bw(m) < bw(x)$ for all $x \in \mathcal{J}$. We have either $m \in [n_{min}, n_{q_{max}}]$ or $n_{q_{max}} < m \leq n_{max}$. If $m \leq n_{q_{max}}$, then, according to Lemma 5, we have $bw(m) \geq bw(n_{q_{max}})$, i.e., a contradiction. Thus $m > n_{q_{max}}$, which is possible if and only if $q_{max} > 0$. However, this implies that there exists $q \in \{0, 1, \ldots, q_{max} - 1\}$ such that

$m \in (n_{q+1}, n_q]$. From Lemma 5, we then have $bw(m) \geq bw(n_q)$ which is again a contradiction. Hence, there exists $n^* \in \mathcal{J}$ such that $bw(n^*) \leq bw(m)$ for all integers $m \in \Omega$.

A.8 Proof of Lemma 7

According to Lemma 2, $k(n_q) = q$ and $k(n_{q+1}) = q+1$. We thus have to show that $(f(n_q) + c)\ 2^{-q} > (f(n_{q+1}) + c)\ 2^{-(q+1)}$, i.e.,

$$2f(n_q) + c > f(n_{q+1}). \tag{15}$$

Assume $n_{q+1} \geq \left\lceil \frac{s}{f_{min}} \right\rceil$. Since $n_q \geq n_{q+1}$, Lemma 4.(a) states that $f(n_q) = f(n_{q+1}) = f_{min}$. In this case, equation (15) clearly holds, which proves the result. Assume on the contrary that $n_{q+1} < \left\lceil \frac{s}{f_{min}} \right\rceil$ which, according to Lemma 4.(b), implies that $f(n_{q+1}) = \left\lceil \frac{s}{n_{q+1}} \right\rceil$. By definition, $f(n_q) \geq \left\lceil \frac{s}{n_q} \right\rceil$, and thus

$$2f(n_q) + c \geq 2 \left\lceil \frac{s}{n_q} \right\rceil + c \geq 2\frac{s}{n_q} + c \geq \frac{2s}{2n_{q+1}} + c,$$

where the last inequality is a direct consequence of Lemma 3. We thus get $2f(n_q) + c \geq \frac{s}{n_{q+1}} + c$ and since the left-hand side is an integer value this implies that $2f(n_q) + c \geq \left\lceil \frac{s}{n_{q+1}} \right\rceil + c$. Since $f(n_{q+1}) = \left\lceil \frac{s}{n_{q+1}} \right\rceil$, it yields $2f(n_q) + c > f(n_{q+1})$. Thus equation (15) also holds in this case, which proves the result.

A.9 Proof of Proposition 2

From Lemma 7, we have $bw(y) > bw(z)$ for all $y, z \in \mathcal{J}$ such that $y > z$. Hence, $bw(n^*) < bw(y)$ for all $y \in \mathcal{J} \setminus \{n^*\}$. However, according to Lemma 6 there exists $x \in \mathcal{J}$ such that $bw(x) \leq bw(n)$ for all $n \in \Omega$, which is possible if and only if $x = n^*$. We thus have $bw(n^*) \leq bw(n)$ for all $n \in \Omega$. According to Proposition 1, this implies that $(n^*, f(n^*), k(n^*))$ is an optimal solution of Problem (OPT).

A.10 Proof of Lemma 8

Assume $s > f_{max}$ and thus $n_{min} > 1$. Using Lemma 8, we get

$$\begin{aligned} n^* &\leq 2\ [n_{min} - 1] = 2 \left[\left\lceil \frac{s}{f_{max}} \right\rceil - 1 \right] \\ &\leq 2\frac{s}{f_{max}} = 2\frac{s}{f_{min}}\frac{f_{min}}{f_{max}} < \left\lceil \frac{s}{f_{min}} \right\rceil \end{aligned}$$

where the last inequality is obtained using $f_{min} = 17 < \frac{1}{2} f_{max} = \frac{1471}{2}$. According to Lemma 4, we can conclude that that $f^* > f_{min}$.

A.11 Proof of Lemma 9

The assumptions imply that $k^* = \left\lfloor \log_2 \left(\frac{\delta}{n_{min}-1} \right) \right\rfloor$. From

$$\log_2 \left(\frac{\delta}{n_{min}-1} \right) - 1 < k^* \leq \log_2 \left(\frac{\delta}{n_{min}-1} \right),$$

we get $n_{min} - 1 \leq 2^{-k^*}\delta < 2\,[n_{min} - 1]$, and this implies that $n_{min} \leq n^* = 1 + \left\lfloor 2^{-k^*}\delta \right\rfloor \leq 2\,[n_{min} - 1]$, as claimed.

B Proofs of the results in Section 3

B.1 Proof of Lemma 10

According to Theorem 1, $n_{min}(\mathcal{A}_i) = 1$ implies that $k(\mathcal{A}_i) = 7$. Moreover, since $\delta(\mathcal{A}_i) < 2^7$, we have $n(\mathcal{A}_i) = 1 + \left\lfloor 2^{-7}\delta(\mathcal{A}_i) \right\rfloor = 1$ and thus $f(\mathcal{A}_i) = \left\lceil \frac{s(\mathcal{A}_i)}{1} \right\rceil = s(\mathcal{A}_i) \geq f_{min}$. It yields

$$\begin{aligned} \sum_{i=1}^{K} bw(\mathcal{A}_i) &= 2^{-7} \left(\sum_{i=1}^{K} f(\mathcal{A}_i) + Kc \right) \\ = 2^{-7} \left(\sum_{i=1}^{K} s(\mathcal{A}_i) + Kc \right) &= 2^{-7} \left(s(\mathcal{A}) + Kc \right). \end{aligned} \tag{16}$$

B.2 Proof of Proposition 4

Note first that $\delta(\mathcal{A}) < 2^7$ and $n_{min}(\mathcal{A}) = 2$ implies that $k(\mathcal{A}) \leq 6$. Moreover, we have $n(\mathcal{A}) = 2$ and $f(\mathcal{A}) = \left\lceil \frac{s(\mathcal{A})}{2} \right\rceil > f_{min}$ according to Lemmata 8 and 9. With Corollary 1, and the fact that $\lceil x \rceil \geq x$, we thus obtain

$$\begin{aligned} \frac{bw(\mathcal{A})}{\sum_{i=1}^{2} bw(\mathcal{A}_i)} &\geq 2^{7-k(\mathcal{A})} \frac{s(\mathcal{A})/2 + c}{s(\mathcal{A}) + 2c} = 2^{6-k(\mathcal{A})} \frac{s(\mathcal{A}) + 2c}{s(\mathcal{A}) + 2c} \\ &\geq 1. \end{aligned} \tag{17}$$

C Proofs of the results in Section 3.3

C.1 Proof of Lemma 11

We first show that the conjecture is equivalent to the following condition:

$$bw(\mathcal{A}) > \sum_{m\in\mathcal{A}} \left\lceil \frac{s(m)}{n(m)} \right\rceil 2^{-k(m)} \quad \forall \mathcal{A} \subset \mathcal{M}. \tag{18}$$

Indeed, if Conjecture 1 holds true, then clearly, since $\{\mathcal{A}\}$ is a partition of $\mathcal{A}$, assertion (18) also holds true. Conversely, if (18) holds true, then for any partition $\{\mathcal{A}_i\}_{i=1,\dots,K}$ of $\mathcal{A}$, it holds that

$$\sum_{i=1}^{K} bw(\mathcal{A}_i) > \sum_{i=1}^{K} \sum_{m\in\mathcal{A}_i} \left\lceil \frac{s(m)}{n(m)} \right\rceil 2^{-k(m)} = \sum_{m\in\mathcal{A}} \left\lceil \frac{s(m)}{n(m)} \right\rceil 2^{-k(m)}.$$

However, since we have assumed that $n_{min}(m) = 1$, we have $k(m) = 7$ and since $\delta(m) < 2^7$ this implies that $n(m) = 1$ for all $m \in \mathcal{M}$. Thus, under Assumption 2, condition (18) can be equivalently written as $bw(\mathcal{A}) > 2^{-7}\, s(\mathcal{A}), \quad \forall \mathcal{A} \subset \mathcal{M}$.

C.2 Proof of Proposition 5

According to Theorem 1, Condition 6 is trivialy satisfied if $s(\mathcal{A}) \leq f_{max}$. Assume therefore that $s(\mathcal{A}) > f_{max}$, i.e., $n_{min}(\mathcal{A}) \geq 2$. Together with $\delta(\mathcal{A}) < 2^7$, this implies that $k(\mathcal{A}) \leq 6$. Thus, with Lemma 9, we obtain $bw(\mathcal{A}) = \left(\left\lceil \frac{s(\mathcal{A})}{n(\mathcal{A})} \right\rceil + c \right) 2^{-k(\mathcal{A})}$, which implies that

$$bw(\mathcal{A}) > \left(\frac{s(\mathcal{A})}{n(\mathcal{A})} + c \right) 2^{-k(\mathcal{A})}. \tag{19}$$

Assume first that $k(\mathcal{A}) = 6$. Since $n(\mathcal{A})$ and $k(\mathcal{A})$ are solutions of problem OPT, the constraint $(n(\mathcal{A}) - 1)\, 2^{k(\mathcal{A})} \leq \delta(\mathcal{A}) < 2^7$ has to be satisfied, which is only possible if $n(\mathcal{A}) = 2$. In this case, (19) yields

$$bw(\mathcal{A}) > \left(\frac{s(\mathcal{A})}{2} + c \right) 2^{-6} = (s(\mathcal{A}) + 2c)\, 2^{-7} > 2^{-7}\, s(\mathcal{A}),$$

and the conjecture holds true. Let us now assume that $k(\mathcal{A}) < 6$. In this case, the condition $(n(\mathcal{A}) - 1)\, 2^{k(\mathcal{A})} \leq \delta(\mathcal{A}) < 2^7$ implies that $n(\mathcal{A})\, 2^{k(\mathcal{A})} \leq 2^7$, and we thus obtain from (19) that

$$bw(\mathcal{A}) > \frac{s(\mathcal{A})}{n(\mathcal{A})\, 2^{k(\mathcal{A})}} + c\, 2^{-k(\mathcal{A})} \geq \frac{s(\mathcal{A})}{2^7} + c\, 2^{-k(\mathcal{A})} > 2^{-7}\, s(\mathcal{A}),$$

which proves that the conjecture also holds true in this case.

Acknowledgements The work presented in this paper was conducted under the research project SATRIMMAP (SAfety and Time Critical Middleware for future Modular Avionics Platforms) which is supported by the French National Agency for Research (ANR).

References

1. Ahuja, R., Magnanti, T., Orlin, J., Weihe, K.: Network flows: theory, algorithms, and applications. Prentice hall Englewood Cliffs, NJ (1993)
2. Airlines electronic engineering committee (AEEC): Aircraft data network, Part 7: Avionics Full Duplex Switched Ethernet (AFDX) Network. ARINC specification 664 (2005)
3. Al-Sheikh, A., Brun, O., Hladik, P.E., Prabhu, B.: Strictly Periodic Scheduling in IMA-based Architectures. Real-Time Systems (2012). URL `http://dx.doi.org/10.1007/s11241-012-9148-y`
4. Authority, F.: 178B, Software considerations in airborne systems and equipment certification. DO-178B/ED-12B, Radio Technical Commission for Aeronautics (1992)
5. Charara, H.: évaluation des performances temps réel de réseaux embarqués avioniques. Ph.D. thesis, Institut national polytechniques de Toulouse (2007)
6. Charara, H., Scharbarg, J.L., Ermont, J., Fraboul, C.: Methods for bounding end-to-end delays on an AFDX network. In: 18th Euromicro Conference on Real-Time Systems, 2006 (2006)
7. Crichigno, J., Barán, B.: A multicast routing algorithm using multiobjective optimization. Telecommunications and Networking-ICT 2004 pp. 63–74 (2004)
8. Gilbert, E., Pollak, H.: Steiner minimal trees. SIAM Journal on Applied Mathematics **16**(1), 1–29 (1968)
9. Hwang, F., Richards, D.: Steiner tree problems. Networks **22**(1), 55–89 (1992)
10. ILOG CPLEX: `http://www.ilog.com/products/cplex/`
11. Lauer, M., Ermont, J., Boniol, F., Pagetti, C.: Latency and freshness analysis on IMA systems. In: IEEE 16th Conference on Emerging Technologies & Factory Automation (ETFA), pp. 1–8 (2011)
12. Martin, S., Minet, P.: Schedulability analysis of flows scheduled with FIFO: application to the expedited forwarding class. In: 20th International Parallel and Distributed Processing Symposium (IPDPS 2006) (2006)
13. Pióro, M., Medhi, D., service), S.O.: Routing, flow, and capacity design in communication and computer networks. Citeseer (2004)
14. SAE ARP4754: Certification considerations for highly-integrated or complex aircraft systems. Systems Integration Requirements Task Group AS-1C, ASD, Society of Automotive Engineers, Inc. (1995)
15. Scharbarg, J.L., Ridouard, F., Fraboul, C.: A Probabilistic Analysis of End-To-End Delays on an AFDX Avionic Network. IEEE Transactions on Industrial Informatics pp. 38–49 (2009)
16. Seok, Y., Lee, Y., Choi, Y., Kim, C.: Explicit multicast routing algorithms for constrained traffic engineering (2002)
17. Spitzer, C.: Digital avionics systems. McGraw-Hill Inc. (1993)
18. Spitzer, C.: The avionics handbook. CRC Press (2001)
19. Watkins, C., Walter, R.: Transitioning from federated avionics architectures to Integrated Modular Avionics. In: Proceedings of the IEEE/AIAA 26th Digital Avionics Systems Conference (DASC'07) (2007)
20. Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. evolutionary computation, IEEE transactions on **3**(4), 257–271 (1999)