A Revised Trajectory Approach for the Worst-Case Delay Analysis of an AFDX Network

XUEQIAN TANG, QIAO LI, GUANGSHAN LU, AND HUAGANG XIONG
School of Electronics and Information Engineering, Beihang University, Beijing 100191, China
Corresponding author: Qiao Li (avionics@buaa.edu.cn)

ABSTRACT AFDX (Avionics Full-Duplex Switched Ethernet) standardized as ARINC 664 is a major upgrade for safety-critical applications in avionics systems. Worst-case delay analysis of all the flows transmitted on the AFDX network is mandatory for certification reasons. Different approaches have been proposed for end-to-end (ETE) delay upper bound, such as network calculus and trajectory approach, but these methods still introduce some pessimism in the computations and overestimate the exact worst-case delay of the flows. In addition, the existing trajectory approaches may underestimate the exact worst-case delay of the flows for some corner cases. In this article, we revised the trajectory approach to make the flow analysis more accurate for the computation of the worst-case ETE delay. The results had been shown that the worst-case delay analysis of an AFDX network can be improved by using our revised trajectory approach.

INDEX TERMS Aircraft, analytical model, Ethernet networks, real-time systems, upper bound.

I. INTRODUCTION
The distributed real-time systems have widely spread in many domains, such as industrial automation and telecommunication, and their increasingly widespread use leads to a growing number of embedded functions, involving an exponential increase of exchanged data and connections [1]. In order to cope with this problem, the concept of shared computation and network resources has been developed. Real-time capable networks offering large bandwidth capacity over a shared medium have been proposed in the avionics field, such as the Avionics Full-Duplex Switched Ethernet (AFDX) [2], used as a backbone network in most of the recent civilian aircraft. Nevertheless, AFDX still exists the indeterminism problem which requires worst-case latency analysis, as guaranteed upper bounds of end-to-end (ETE) delays for messages transmitted over an AFDX network are mandatory for certification reasons.

Several approaches have been developed in order to determine guaranteed upper bounds of ETE delays, such as network calculus (NC) and trajectory approach (TA). Even though upper bounds are sufficient for certification purposes, they often imply overestimation when used for network dimensioning. Existing trajectory approaches overestimate the worst-case end-to-end delays for the messages in some corner cases and underestimate them in some other corner cases. The contribution of this paper is to propose a revised trajectory approach (RTA) for computing ETE delays in an AFDX network. It can reduce some overestimation (also called pessimism) in the computations and deal with the underestimation (also called optimism) problem, and hence improve the worst-case end-to-end delay upper bound of the flows.

This paper is organized as follows. In Section II, we present the existing worst-case ETE computation methods. The detailed context of AFDX is given in Section III. The trajectory approach is presented in Section IV. A revised trajectory approach in the context of a FIFO and FP/FIFO policy is demonstrated in section V. Finally, experiments and comparisons are conducted in Section VI. The conclusion is presented in Section VII.

II. RELATED WORK AND MOTIVATIONS
Different methods have been proposed to determine the worst-case end-to-end delay for the messages of the AFDX networks in recent years. A simulation method and
a stochastic network calculus approach have been proposed for the computation of the distribution of the delay of a given flow, but they do not cope with the worst-case analysis problem considered in this paper. As the current reference certification tool, network calculus [3] has been applied in the AFDX networks in civilian aircraft. But the pessimism of the obtained upper bounds was difficult to quantify. Although a stochastic network calculus approach [4], [5] has been proposed, the results are not yet relevant to real-world systems. The model-checking approach presented in [6] computes an exact worst-case delay of each flow. But it cannot scale up with large AFDX configurations due to the combinatorial explosion. Although some work [7] has been devoted to reducing the state space, it still does not cope with the real AFDX configurations. The holistic approach [8] has also been used to compute the local delays, but it considers a worst-case scenario (possibly unreachable) in each node visited by a flow. The forward end-to-end delay approach [1] is proposed, which computes an upper bound for the end-to-end delay for the AFDX networks based on the maximum backlog of single priority traffic.

The classic trajectory approach (CTA) has been defined in [9], [10]. CTA has been optimized in an AFDX context to provide an upper bound for the end-to-end delay encountered by any frame of a flow traversing a data network. And the optimized trajectory approach (OTA) [11], [12] has taken into account the impact of the serialization of flows sharing the same link. The pessimism of OTA has been analyzed in [13]. Recently, several research studies have focused on improving the serialization [14] and optimization [15] of OTA for the end-to-end delay analysis. Besides, the trajectory approach had also been used on deterministic scheduling in network-on-chip. In addition, an extended trajectory approach had been used to analyze deterministic delay of AVB switched ethernet networks [16]. However, a counterexample [17] has been given, which shows that OTA with current form can bring an underestimation problem in some corner cases. The source of the underestimation problem has been discussed in [17], [18] and a solution has been proposed in [18].

The limitations in the existing trajectory approaches motivate revising trajectory approach for improving worst-case ETE delays in packet networks. Compared with the existing trajectory approaches, the most significant contribution of our revised trajectory approach is to sum the maximum net delay introduced by each node on the path of the flow under study. Besides, the revised trajectory approach considers the frame constraints in the serialization effect. This improvement has shown to produce less pessimistic ETE delays for existing trajectory approaches and solve the underestimation problem at the same time.

III. THE AFDX NETWORK CONTEXT

Avionics Full-Duplex Switched Ethernet (AFDX) [2] is a switched Ethernet network taking into account avionic constraints. The ingress and egress points of the AFDX networks are called End Systems (ES). Each output port of each ES is connected to exactly one switch port by a full duplex physical link. The end-to-end traffic is mapped on logical communication channels called virtual links (VL). Fig. 1 depicts a sample AFDX configuration [1]. It is composed of nine end systems interconnected by six switches. A VL is a logical, unidirectional connection from one source ES to one or more destination ES. For example, in Fig. 1, VL $\tau_1$ is a multicast VL starting from $ES_6$ with the following paths $ES_5 – S5 – S6 – ES_9$ and $ES_5 – S5 – S4 – ES_9$. The routing of each VL is statically defined. Each VL flow is characterized by the Bandwidth Allocation Gap (BAG) [19], which is the minimum duration between two consecutive frames of the flow ($\tau_i$), as well as a maximum frame size ($L_{max}$) and minimum frame size ($L_{min}$). Each flow $\tau_i$ following the path $P_i$ has a BAG, denoted $T_i$. Each flow $\tau_i$ has a maximum release jitter $J_i$, an end-to-end deadline $D_i$, and a maximum processing time $C_i$ on each node $h$, with $h \in P_i$. Moreover, there are neither collisions nor packet losses on the links.

Each node operates traffic shaping and policing units to guarantee the BAG. An AFDX switch that operates as store-and-forward between input and output ports applies traffic policing and routing frames to one or several outputs ports. An AFDX switch has a FIFO buffer at each output port and no input buffer. The technological latency of an AFDX switch is estimated to be 16 $\mu$s. The servicing rate of an output port is constant (mostly 100 Mb/s). There is no global clock or synchronization protocol, so switches and end systems are fully asynchronous. All the above constraints make it possible to have a deterministic network based on Ethernet through an ETE delay analysis of the flows.

IV. AFDX WORST-CASE DELAY ANALYSIS WITH TRAJECTORY APPROACH

A. THE COMPUTATION OF THE TRAJECTORY APPROACH

The trajectory approach is proposed to compute a worst-case end-to-end delay of any flow transmitted on a switched Ethernet network for the First In First Out (FIFO) scheduling and non-preemptive Fixed Priority (FP) scheduling. The trajectory approach is based on the busy period concept. A busy period of level $L$ is defined by an interval $[t, t’)$ such that $t$ and $t’$ are both idle times of level $L$ and there is no idle time of level $L$ in $(t, t’)$. An idle time $t$ of level $L$ is a time such as all packets with the priority greater than or equal to $L$ generated before $t$ have been processed at time $t$. The trajectory approach identifies the busy period and recursively compute the latest start time of considered flow $\tau_i$ on its last
node. According to the trajectory approach, the worst-case end-to-end delay upper bound $R_i$ of flow $\tau_i$ is computed by the equation

$$R_i = \max_{0 \leq t} \left\{ W_{l,t}^{last_i} - t + C_i \right\} \quad (1)$$

where $last_i$ is the last visited node of flow $\tau_i$ and $W_{l,t}^{last_i}$ is a bound on the latest starting time of a packet $m$ generated at time $t$ on its last visited node. The definition of $W_{l,t}^{last_i}$ is given in [12].

$$2W_{l,t}^{last_i} = \sum_{j \in \text{sp} \cup \{i\}} \left(1 + \left(1 + \frac{t + A_{i,j}}{T_j}\right) \times C_j\right)$$

$$+ \sum_{j \in \text{chp}_i} \left(1 + \left(1 + \frac{W_{l,t}^{last_i,j} + B_{i,j}}{T_j}\right) \times C_j\right)$$

$$+ \sum_{h \in \text{Pi}_j} \left(1 + \frac{\text{Max}_{j \in \text{sp} \cup \{i\}} (C_j))}{h \in \text{Pi}_j} \times C_j\right)$$

$$+ \left(1 - 1\right) \times L_{\text{max}}$$

$$+ \sum_{h \in \text{Pi}_j} \left(\Delta_h\right)$$

$$- \sum_{h \in \text{Pi}_j} \left(\Delta_h\right)$$

$$- C_i \quad (8)$$

Term (2) corresponds to the processing time of packets from flows, crossing the flow $\tau_i$, with a fixed priority level equal to this of $\tau_i$ and transmitted in the same busy period as $m$. $A_{i,j}$ denotes the maximum difference on the delay between their source node and their first shared output port for flows $\tau_i$ and $\tau_j$. $A_{i,j}$ is calculated as follows.

$$A_{i,j} = \text{Smax}^h_i - \text{Smin}^h_i - M^h_i + \text{Smax}^h_i + J_{\text{switch}} \quad (9)$$

where $M^h_i$ denotes the earliest arrival time of the first packet that will delay the frame $m$ on the output port of the node $h$. $\text{Smax}^h_i$ and $\text{Smax}^h_i$ is the maximum and minimum time taken by a frame of flow $\tau_i$ to go from its source node to node $h$, respectively. $\text{Smax}^h_i$ is similar to $\text{Smax}^h_i$. $J_{\text{switch}}$ is the jitter associated to VL$_j$ entering a switch node $h$.

Term (3) corresponds to the processing time of packets from flows with a fixed priority level higher than this of $\tau_i$. $B_{i,j}$ denotes the maximum difference on the delay between their source node and their last shared output port for flows $\tau_i$ and $\tau_j$. $B_{i,j}$ is calculated as follows.

$$B_{i,j} = \text{Smax}^h_i - \text{Smin}^h_i - M^h_i + \text{Smax}^h_i + J_{\text{switch}} \quad (10)$$

Term (4) represents the processing time of the longest packet for each node of path $P_i$ except the last one.

Term (5) corresponds to the switching latencies along the path $P_i$. The switching latency at each switch is considered as an upper bounded constant $L_{\text{max}}$.

Term (6) corresponds to the maximum delay caused by lower-priority flows due to with a fixed priority lower than this of $\tau_i$.

Term (7) refers to the serialization effect, which is the duration between the beginning of the busy period and the arrival of the first packet coming from the preceding node in $P_i$, i.e. $h - 1$.

Term (8) is subtracted since $W_{l,t}^{last_i}$ is the start time of frame $f_i$ transmission at the output port $last_i$.

### B. PESSIMISTIC PROBLEMS IN THE TRAJECTORY APPROACH

The worst-case scenario built by CTA [9], [10] is that the arrival time of every packet joining the trajectory of the packet $m$ under study in the node $h$ is postponed in order to maximize the waiting time of $m$ in $h$. A packet joining $m$ in $h$ can not arrive before the first packet coming from the same previous node as $m$, thus the term (8) is null in [9], [10]. The pessimism is generated because the computation of CTA is done based on pessimistic assumption that all the frames of different flows arrive at the same time. Unlike CTA, OTA [11], [12] considers that all flows from different virtual links sharing a common physical link can not arrive at the same time and need to be serialized. Then the beginning time of the busy period on node $h$ is not always $M^h_i$. Compared to CTA, OTA gets a tighter end-to-end delay upper bound by considering the serialization effect. But it may still lead the pessimism problem in some corner cases. The computation of the serialization effect does not consider the BAG constraint between the two consecutive frames from the same virtual link according to the traffic policy. Moreover, it is very rare that no flow leaves the path of the flow under study in an AFDX network [20]. Some flows join the flow under study, share part of the path and then leave. This implies that the pessimism problem may occur [21]. The pessimism problem still exists in the current trajectory approaches [10], [12], [13].

### C. UNDERESTIMATION PROBLEMS IN THE TRAJECTORY APPROACH

Recently, a counterexample given in [17] shows that OTA can lead to computing underestimation delay upper bounds for some corner cases. The underestimation problem is introduced since the subtraction of serialization factors is partially overlapped with the subtraction of the time interval $t$ in (1). A solution [13] is proposed to solve this problem by taking into account the subtraction of time $t$ in the serialization factors. However, as shown in Fig. 2, an input sequence (seq$_{0}$) from an input port may contain two frame subsequences (sbq$_{01}$ and sbq$_{02}$) separated by an idle time. The existing trajectory approaches do not consider this. Actually, $M^h_i$ is neither the beginning time of the input sequence under study, nor the beginning time of the output sequence. As $M^h_i$ is an inaccurate input to the existing trajectory approaches, then the end-to-end delays computed by the existing trajectory approaches...
are pessimistic for some corner cases and optimistic in some other corner cases.

V. REVISED TRAJECTORY APPROACH
A. WORST-CASE SCENARIO ON AN INPUT PORT
Filtering and policing function is implemented by an AFDX switch to filter all frames arriving at an input port. Here, we assume that the AFDX switches implement a frame-based traffic policing algorithm. The specification of the frame-based traffic policing algorithm [2] is as follows.

1) If Frame ACount (also called ACi) is greater than the maximum frame size ($FS_{i\text{max}}$) for $\tau_i$, the frame is accepted and Frame ACi is debited by $FS_{i\text{max}}$.

2) If Frame ACi is less than $FS_{i\text{max}}$, the frame is discarded and Frame ACi is not changed.

For a single flow $\tau_i$, the worst-case scenario of an input port is depicted in Fig. 3. The minimum duration between the first frame and the second frame is $BAG - J_i$ and the minimum duration between any two other consecutive frames is $BAG$. Frames from $\tau_i$ with this time order can maximize its delay contribution in a time interval $[0, \tau_i]$.

For the aggregation flows, frames from each flow with the above time order arriving at the same input port generates the worst-case scenario. However, to compute an exact serialization effect, we must consider all possible transmission orders of the frames of the sporadic flows sharing the same physical link [20]. This will generate a huge number of scenarios at each node. To reduce the computational complexity, we simplify the computation of the serialization effect. Each aggregation flow arriving the same input port is assigned the offset equal to zero. Frames from the same flow are restricted by the arriving time order. If the transmission time interval of a frame overlaps with that of other frames, then these frames need to be serialized. For example, in Fig. 4, there are flows ($\tau_1, \tau_2, \tau_3$) arriving at the same input port. As the transmission time interval of the first frame of $\tau_1$ is identical with that of $\tau_2$ and $\tau_3$, then the first frames of $\tau_1, \tau_2$ and $\tau_3$ need to be serialized. Since there are no conflicts between other frames with the current time order, the time order of other frames keeps unchanged. Finally, the worst-case scenario of the input aggregation sequence is obtained to compute the worst-case ETE delay.

B. REVISED TRAJECTORY APPROACH
In this section, we describe our revised trajectory approach in detail. Firstly, two theorems are given as follows.

**Theorem 1:** The servicing rate of the input link is equal to that of the output link, then the maximum transition cost ($m\text{TransCost}$) from one busy period (including $n$ packets) of the input link to that of the output link is the processing time of the longest packet for the last frame.

$$m\text{TransCost} = \max_{1 \leq i \leq n} (x_i)$$  \hspace{1cm} (11)

**Proof:** As is shown in Fig. 5, there is a sequence including $n$ frames on the input link and output link of a node. The processing time of $i^{th}$ frame is denoted as $x_i$ ($1 \leq i \leq n$). The start transmission time and end transmission time of $x_i$ on the output link is denoted as $a_i$ and $b_i$. The start transmission time and end transmission time of $x_i$ on the output link is denoted as $c_i$ and $d_i$. By the definition of the busy period, there is no idle time between any two consecutive frames of an input sequence, then

$$x_i = b_i - a_{i-1}, a_i = b_{i-1}, 2 \leq i \leq n$$

$$x_i = d_i - c_{i-1}, c_i = d_{i-1}, 2 \leq i \leq n$$  \hspace{1cm} (12)

According to the store and forward strategy, the start transmission time of $i^{th}$ frame on the output link must be larger than the end transmission time of $i^{th}$ frame on the input link. Then,

$$c_i \geq b_i, 1 \leq i \leq n$$  \hspace{1cm} (13)
Thus,
\[ \sum_{i=2}^{i} x_i \geq \sum_{i=1}^{i-1} x_j \tag{14} \]

To maximize the transition cost, the first frame has to be one of the longest packets. Then the maximum transition cost \(m\text{TransCost}\) is obtained as follows.
\[ m\text{TransCost} = d_n - b_n = \max_{1 \leq i \leq n} (x_i) \tag{15} \]

Finally the theorem is proven. In fact, the transition cost for the last frame is irrelevant to the frame transmission order and is only determined by the processing time of the longest packet. To maximize the worst-case end-to-end delay of the last frame under study, one of the longest packets is firstly transmitted on the output link. The conclusion of the proof also conforms to the above Term (4).

**Theorem 2:** If the servicing rate of the input port is smaller than that of the output link, then the maximum transition cost \(m\text{TransCost}\) is the processing time of the longest packet on the output link for the last frame. If the servicing rate of the input port is larger than that of the output link, then the maximum transition cost is the accumulative processing time of all packets on the output link minus that of all packets on the input link, and plus the processing time of the longest packet on the input link for the last frame.

**Proof:** As is shown in Fig. 5, there is a sequence including \(n\) frames on the input link and output link. Let \(x_i\) and \(y_i\) \((1 \leq i \leq n)\) be the processing time of the \(i\)th frame on the input link and output link, respectively. The start transmission time and end transmission time of the \(i\)th frame on the input link is denoted as \(a_i\) and \(b_i\). The start transmission time and end transmission time of the \(i\)th frame on the output link is denoted as \(c_i\) and \(d_i\). Then,
\[ x_i = b_i - a_{i-1}, 1 \leq i \leq n \]
\[ y_i = d_i - c_{i-1}, 1 \leq i \leq n \tag{16} \]

Let \(B_{IP}\) and \(B_{OP}\) be the servicing rate of the input link and output link, respectively. We distinguish between two cases:
(i) \(B_{IP} < B_{OP}\). (ii) \(B_{IP} > B_{OP}\).
ad(i): \(B_{IP} < B_{OP}\)
If there is no idle time between any two consecutive frames on the output link, then the first frame is easily proved to be the longest packet. Then the transition cost for the last frame is obtained
\[ \text{TransCost} = \sum_{i=1}^{n} y_i + x_1 - \sum_{i=1}^{n} x_i < y_1 \tag{17} \]

If there are some idle times between two consecutive frames on the output link. Then for the last idle time \((idle)_{last}\) locating between \(x_q\) and \(x_{q+1}\), we have
\[ \text{TransCost} = \sum_{i=1}^{q+1} x_i + \sum_{i=q+1}^{n} y_i - \sum_{i=1}^{n} x_i < y_q \tag{18} \]

According to (17) and (18), it is obtained.
\[ \text{TransCost} < \max_{1 \leq i \leq n} (y_i) \tag{19} \]

For the sake of convenient calculation, \(m\text{TransCost}\) is viewed to be the processing time of the longest packet on the output link for the last frame. Then,
\[ m\text{TransCost} = \max_{1 \leq i \leq n} (y_i) \tag{20} \]

ad(ii): \(B_{IP} > B_{OP}\)
In this case, we have
\[ \text{TransCost} = \sum_{i=1}^{n} y_i + x_1 - \sum_{i=1}^{n} x_i \tag{21} \]

By (21), it is concluded that \( \text{TransCost} \) is maximized when the first frame is the longest packet. Thus,
\[ m\text{TransCost} = \max_{1 \leq i \leq n} (x_i) + \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} x_i \tag{22} \]

Hence the theorem is proven.

**Corollary 1:** If there are some idle times between two consecutive frames of an input sequence and the servicing rate of the input link is larger than that of the output link, then the maximum transition cost is the accumulative processing time of all packets on the output link minus the length of the input sequence on the input link, and plus the processing time of the longest packet on the input link for the last frame.

**Proof:** According to the proof of Theorem 2, it is obtained that the maximum transition cost is the accumulative processing time of all packets on the output link minus the length of the input sequence on the input link, and plus the processing time of the longest packet on the input link for the last frame.

**The FIFO context** As is illustrated in Fig. 2, the packet \(m\) of flow \(\tau_i\) under study is sent from the input link \(IP^h\) to the output link \(OP^h\) in a busy period \(bp^h\). Sequence \(seq_i\) \((1 \leq i \leq k)\) contains the frames of \(bp^h\) coming from \(IP^h\). \(f(h)\) is the first packet which is processed in \(bp^h\), \(p(h-1)\) is defined as the first packet which is processed in \(bp^h\) and comes from node \(h-1\). In a FIFO context (one single priority level), the frames of flow \(\tau_i\) with a fixed priority level equal to this of \(\tau_i\) under study can delay \(m\) if they arrive at the node \(h\) after the earliest time \(a_{f(h)}^h\) and before the latest time \(t + \text{Smax}^h\). Then the estimation interval on the node \(h\) is \([a_{f(h)}^h, t + \text{Smax}^h]\) for \(m\). Let \(Q^h\) be the net delay generated by all the concurrent flow on the node \(h\). Then,
\[ Q^h \left( a_{f(h)}^h \right) = t + \text{Smax}^{h+1} - \left( t + \text{Smax}^h \right) \tag{23} \]

Based on the concept of the Theorem 1 and Theorem 2, we have the following Theorem 3.

**Theorem 3:** Assume that there are \(k\) input links on the node \(h\) with the servicing rate of each input link equal to that of the output link. Each input link has a input sequence.
including \( n \) subsequences. Then the maximum net delay \( Q^h \) on the node \( h \) for the packet \( m \) is as follows.

\[
Q^h = \sum_{i=0}^{k} \sum_{j=1}^{n} sbq_i^j - \max_{1 \leq x \leq k} l_x + L_{\max}^{_{heP_{i}}} \quad \text{(24)}
\]

Proof: By to the definition of the busy period, there is no idle time within any two consecutive frames of a output sequence. It is easy to obtained as follows.

\[
t + S_{\text{max}}^{h+1} = a^h_{f(h)} + \sum_{i=0}^{k} \sum_{j=1}^{n} sbq_i^j + L_{\max}^{_{heP_{i}}} \quad \text{(25)}
\]

where \( sbq_i^j \) denotes the \( j^{th} \) subsequence from \( i^{th} \) input link. It is noted that any subsequence from any input link has no idle time between any two consecutive frames. \( seq_i \) denotes the duration of the aggregation flows from \( i^{th} \) input link and may contain some idle times. \( a^h_{f(h)} \) is computed by (26).

\[
a^h_{f(h)} = t + S_{\text{max}}^h - \max_{1 \leq x \leq k} (\min(l_x)) \quad \text{(26)}
\]

where \( l_x(0 \leq x \leq k) \) denotes the duration of the input sequence \( seq_x \) without its first packet.

So, (23) transforms as follows.

\[
Q^h = \sum_{i=0}^{k} \sum_{j=1}^{n} sbq_i^j - \max_{1 \leq x \leq k} (\min(l_x)) + L_{\max}^{_{heP_{i}}} \quad \text{(27)}
\]

Hence the theorem is proven. \( \blacksquare \)

Generally, \( a^h_{f(h)} \) is not known at the beginning of the calculation. To maximize \( Q^h \), we need to maximize interference packets and minimize the serialization effect. \( Q^h \) is viewed as a monotonically increasing function of \( a^h_{f(h)} \). Firstly, it is obviously that \( S_{\text{max}}^1 = 0 \) at the source node of the frame \( m \). Then, we continue to compute the maximum time \( t + S_{\text{max}}^2 \) at the source node of the frame \( m \). The maximum net delay \( Q_{\text{max}} \) on the source node is easily obtained. The length of the sequence \( seq_0 \) for frame \( m \) under study at the output link of the source node is the sum of frames joining the frame \( m \) under study and need to be recorded. At the second node, we consider the worst-case scenario of all the input links. The initial value of \( a^2_{f(2)} \) is set to be \( t + S_{\text{max}}^2 \). Then a search within the interval \( (t, t + S_{\text{max}}^2) \) is performed for a maximum value of \( Q^2 \). If \( Q^2 = Q_{\text{max}}^2 \) is satisfied, then the search continues to be implemented. Once \( a^2_{f(2)} \) is minimized under the condition of \( Q^2 = Q_{\text{max}}^2 \), the search stops. Repeat the above computation process until the last node visited by the frame \( m \). Finally, the worst-case end-to-end delay of \( t_i \) can be obtained by (28).

\[
R_{\text{RTA}} = \max_{0 \leq t \leq J_{i, \text{switch}}} \left\{ S_{\text{max}}^{last+1} \right\} \quad \text{(28)}
\]

The FP/FIFO context As is illustrated in Fig. 6, in a FP/FIFO context (multiple priority levels), frames of flow \( \tau _j \) with a fixed priority level higher than this of \( \tau _i \) under study can delay \( \tau _i \) if they arrive at the node \( h \) after the earliest time \( a^h_{f(h)} \) and before the latest time \( t + S_{\text{max}}^{h+1} \). \( C_m \) denotes the processing time of the frame \( m \). Then the estimation interval on the node \( h \) is \( \left[ a^h_{f(h)}, t + S_{\text{max}}^{h+1} \right) \) for \( \tau _i \). If a sequence from an input link contains some packets with a higher priority than \( m \) and some other packets with a priority equal to \( m \), then the sequence need to be separated into two sequence, such as \( hsep_1 \) and \( sep_1 \) from \( IP^1 \) in Fig. 6. In a FP/FIFO context, \( Q^h \) is calculated as follows,

\[
2Q^h \left( a^h_{f(h)} \right) = \sum_{i=0}^{k} \sum_{j=1}^{n} hsqb_i^j + \sum_{i=0}^{k} \sum_{j=1}^{n} sbq_i^j + L_{\max}^{_{heP_{i}}} + \delta^h - \max_{1 \leq x \leq k} (\min(l_x)) \quad \text{(29)}
\]

where \( hsqb_i^j \) denotes the \( j^{th} \) subsequence of the input sequence \( hsep_1 \) with a higher priority than \( m \) from \( i^{th} \) input link. It is noted that then the sequence \( hsep_1 \) and \( sep_1 \) from the same input link need to be separated into two sequence to compute \( max_{1 \leq x \leq k} l_x \). Finally, the worst-case end-to-end delay of \( \tau _i \) in a \( I_S \leq k \) FP/FIFO context can be obtained by (29)-(33).

Theorem 4: Let \( m \) be the packet of flow \( \tau _i \) generated at time \( t \). When the flows are scheduled FIFO or FP/FIFO under the worst-case scenario described in Section V, we have for any time \( t \geq J_{i, \text{switch}} \):

\[
R_i \leq R_{\text{RTA}} \quad \text{(34)}
\]

Proof: According to the definition of \( Q^h \), it is easy to obtain.

\[
S_{\text{max}}^h = \sum_{i=1}^{h-1} Q^i \quad \text{(35)}
\]

Assume that any busy period \([ S_m^h, T_m^h ]\) on each node \( h \) of the path \( P_i \) is selected for a packet \( m \) and \( R_i \) is obtained. The busy period \([ S_m^h, T_m^h ]\) on each node \( h \) is captured by RTA for
a packet $m$ and $R_{RTA}$ is obtained. Then, we have

$$Q^h = T^h_c - T^{h-1}_c \text{ or } Q^h = T^h_m - T^{h-1}_m$$  \hspace{1cm} (36)

In the first (source) node, there is no serialization effect. It trivially holds.

$$S^1_c = S^1_m, T^1_c = T^1_m$$  \hspace{1cm} (37)

In the second node, it trivially holds.

$$T^2_c \leq T^2_m$$  \hspace{1cm} (38)

In the third node, we distinguish between two cases: (i) $S^3_c > S^3_m$ or $S^3_c < S^3_m$.

ad(i): If $S^3_c > S^3_m$, the estimation interval on the third node transforms as follows.

$$t + S_{max}^i - a_{(h)}^i = T^2_c - S^3_c < T^2_c - S^3_m$$  \hspace{1cm} (39)

Then,

$$Q^3(T^2_c - S^3_c) \leq Q^3(T^2_c - S^3_m)$$  \hspace{1cm} (40)

It is easy to obtain.

$$T^3_c < T^3_m$$  \hspace{1cm} (41)

ad(ii): Although $S^3_c \leq S^3_m$, we still obtain $Q_h(T^3_c - S^3_c) \geq Q_h(T^3_m - S^3_m)$ according to RTA.

Then,

$$T^3_c < T^3_m$$  \hspace{1cm} (42)

......

In the last node, we can easy to obtain delay.

$$T^l_{last} + 1 < T^l_{last} + 1$$

$$R_i = T^l_{last} + 1 - S^l_c \leq T^l_{last} + 1 - S^l_m = R_{RTA}$$  \hspace{1cm} (43)

Thus,

$$R_i \leq R_{RTA}$$  \hspace{1cm} (44)

\section*{C. COMPARED WITH CTA AND OTA}

According to OTA, the net delay generated by all the concurrent flow on the node $h$ in a FIFO context is calculated as follows.

$$2Q^h_{OTA} = \sum_{j \in sp \cup \{i\}} \left( 1 + \left[ \frac{t + A_{i,j}}{T_j} \right] \right) \times C_j$$  \hspace{1cm} (45)

$$+ \sum_{h \in P_i \cap \{i\}} \left( \max_{j \in sp \cup \{i\}} (C_j) \right)$$  \hspace{1cm} (46)

$$+ \sum_{h \in P_i \cap \{i\}} L_{max}$$

$$+ \sum_{h \in P_i \cap \{i\}} S^h_{i}$$  \hspace{1cm} (47)

The serialization effect $\Delta_h$ given by Term (48) is obtained as follows.

$$\Delta_h = \max \{0, \Delta_{se} \}$$  \hspace{1cm} (48)

where

$$\Delta_{se} = \max_{1 \leq i \leq k}(\max_{1 \leq l \leq k}(l_i) - (\min_{1 \leq l \leq k}(l_0))$$  \hspace{1cm} (49)

With all the same input subsequences given, we compare OTA and RTA. The difference $d_1$ between $Q^h_{OTA}$ and $Q^h_{RTA}$ is given as follows.

$$d_1 = Q^h_{OTA} - Q^h_{RTA}$$

$$= \left\{ \begin{array}{ll} \max_{1 \leq i \leq k} (C_i) - \min_{1 \leq l \leq k} (C_l), & \Delta_h > 0. \\
0, & \Delta_h = 0. \end{array} \right.$$  \hspace{1cm} (50)

According to (50), $Q^h_{OTA}$ is larger than $Q^h_{RTA}$. Generally, considering the frame constraints from the same flows on the node $h$, the accumulative inference of the input sequences captured by OTA may be more larger than that of those sequences captured by RTA. Compared to OTA, RTA can compute more exact worst-case ETE delay. Besides, (50) conforms to the conclusion described in [14]. As the idle times included in the input sequence, RTA can compute more exact worst-case ETE delay than the method proposed in [14].

Compared to OTA, the net delay ($Q^h_{CTA}$) computed by CTA has Term (43-46) except Term (47). With all the same subsequences given, we compare CTA and RTA. The difference $d_2$ between the net delay ($Q^h_{CTA}$) calculated by CTA and $Q^h_{RTA}$ is given as follows.

$$d_2 = Q^h_{CTA} - Q^h_{RTA}$$

$$= \left\{ \begin{array}{ll} \max_{1 \leq i \leq k} (\min_{1 \leq l \leq k}(l_i)) - (\min_{1 \leq l \leq k}(l_0)), & \Delta_h > 0. \\
0, & \Delta_h = 0. \end{array} \right.$$  \hspace{1cm} (51)

According to (51), $Q^h_{CTA}$ is larger than or equal to $Q^h_{RTA}$. Compared with CTA, the maximum pessimism reduced by RTA is ($\max_{1 \leq l \leq k}(\min_{1 \leq l \leq k}(l_i)) - (\min_{1 \leq l \leq k}(l_0))$). Finally, the worst-case ETE delay obtained by CTA is larger than or equal to that obtained by RTA.

\section*{VI. CASE STUDY}

\subsection*{A. CASE1}

We consider the counterexample given in Fig. 7, which is discussed in [17], [18]. The flow temporal characteristics are given in Table. 1. Here, the switching latency is considered as null. The transmission priority of all flows is identical. Frame $f_1$ which is under study follows the path $P = ES_2 - S_3 - ES_7$. It can be delayed by frames of flows $f_2$, $f_3$ and $f_4$ at the output port of $S_3$. $J_{switch}$ is set to be $40 \mu s$. Let’s use the RTA to compute the ETE delay of frame $f_1$. Since $f_1$ is the only flow emitted by $ES_1$, then $S_{max} = ES_1 = 40 \mu s$ and $Q^h_{max} = 40 \mu s$. Then, we need to compute the value of $A_{i,j}$ for each flow joining the frame $f_1$. The length of the input sequence at the input port of $S_3$ is $40 \mu s$. We consider the worst-case scenario at each input port of $S_3$, as is shown in Fig. 8. We implement the search to calculate the maximum net delay on the node $S_3$ for frame $f_1$. When $A_{i,j} = 0 \mu s$, $Q^h_{max} = 140 \mu s$. Then the net delay on all the nodes of the
path followed by frame $f_1$ is accumulated to be 180 $\mu$s. This result is the same as hand built (180 $\mu$s) and CTA (180 $\mu$s), higher than OTA (160 $\mu$s) and smaller than NC (200 $\mu$s).

Based on the results of the above analysis, the underestimation problem is solved by the RTA. It’s noted that the optimism of OTA exists under the condition of $J_{\text{switch}} = 40$ $\mu$s. If $J_{\text{switch}}$ is set to be 0 $\mu$s, then the underestimation problem produced by OTA disappears. In addition, the arriving time gap between the frame $f_4'$ and $f_4$ in the worst-case scenario given by Fig. 8 [18] cannot less than 80 $\mu$s, as they do not consider the frame constraints from the same flow on the input port.

**B. CASE2**

We experiment with the configuration depicted in Fig. 1. The servicing rate of output ports is equal to 10 Mb/s for ES$1$, ES$3$, ES$4$, S$2$(to S$4$), S$3$(to ES$6$) and 100 Mb/s for other nodes. The characteristics of the flows are given in Table 2. $J_{\text{switch}}$ is set to be 0 $\mu$s. The global load of traffic encountered by some flows (e.g., $\tau_7$) on their path is strictly larger than 1, which prevents the convergence of the computation with the current TA. But it does not matter for RTA. The worst-case ETE delay computed with NC, FA [1] and RTA are presented in Table 3 and Fig. 10. The exact delay (ED) computed by hand built also presented. As shown in Fig. 11, the ETE delay bounds obtained by RTA are tighter than by NC, and FA. Compared to NC, the difference is up to 561.30% for flow $\tau_8$ (ES$9$) and $\tau_9$ (ES$9$). Compared to FA, the difference is up to 441.05% for flow $\tau_8$ (ES$9$) and $\tau_9$ (ES$9$).

**C. CASE3**

In this case study, we experiment with the topology configuration depicted in Fig. 7. The experiment network contains 100 flows, each of which is configured by a random length and BAG. The statistical information of the load is shown in Table 4. The three approaches (NC, CTA, OTA and RTA) have been applied to the network configuration. In Fig. 12, the worst-case ETE delays of flows have been computed with NC, CTA, OTA and RTA. We compare RTA with NC, CTA and OTA. For example, the worst-case end-to-end computed by NC and CTA for VL$1$ is 1922 $\mu$s and 1362 $\mu$s. With the grouping optimization, the worst-case end-to-end delay computed by OTA for VL$1$ is 1072 $\mu$s. By using RTA, the worst-case delay for is only 994 $\mu$s. Compared with NC, CTA and OTA, the gain obtained by RTA is 50.06%, 26.96% and 7.2%.

Flow $\tau_8$ (ES$9$) and $\tau_9$ (ES$9$). Compared to FA, the difference is up to 441.05% for flow $\tau_8$ (ES$9$) and $\tau_9$ (ES$9$).

**TABLE 2.** A configuration for AFDX.

| VL   | L$_{\text{max}}$(bit) | BAG($\mu$s) |
|------|------------------------|-------------|
| $\tau_1$ | 12144                | 4000        |
| $\tau_2$ | 12144                | 4000        |
| $\tau_3$ | 8256                  | 8000        |
| $\tau_4$ (ES$7$) | 8256                | 8000        |
| $\tau_5$ | 8256                  | 8000        |
| $\tau_6$ (ES$8$) | 2048                | 1000        |
| $\tau_7$ (ES$8$) | 2048                | 1000        |
| $\tau_8$ (ES$9$) | 312                  | 1200        |
| $\tau_9$ (ES$9$) | 312                  | 1200        |
| $\tau_{10}$ (ES$9$) | 2304               | 500         |
| $\tau_{11}$ (ES$9$) | 2304               | 500         |
| $\tau_{12}$ (ES$9$) | 312                 | 250         |

**TABLE 3.** Computed worst-case ETE delays.

| VL   | NC($\mu$s) | FA($\mu$s) | RTA($\mu$s) | ED($\mu$s) |
|------|------------|------------|-------------|------------|
| $\tau_1$ | 4270.48    | 3133.60    | 3050.08     | 2689.44    |
| $\tau_2$ | 3678.54    | 3270.4     | 3270.4      | 3270.4     |
| $\tau_3$ | 3404.05    | 4957.6     | 4957.6      | 4957.6     |
| $\tau_4$ (ES$7$) | 6709.98    | 5564.16    | 5491.36     | 5452.48    |
| $\tau_5$ (ES$8$) | 2955.28    | 2924.8     | 2901.76     | 2901.76    |
| $\tau_6$ (ES$9$) | 6709.98    | 5564.16    | 5491.36     | 5452.48    |
| $\tau_7$ (ES$9$) | 2955.28    | 2924.8     | 2839.68     | 2814.08    |
| $\tau_8$ (ES$8$) | 3009.81    | 2850.26    | 2835.68     | 2777.6     |
| $\tau_9$ (ES$8$) | 4447.91    | 3309.44    | 3236.64     | 3197.76    |
| $\tau_{10}$ (ES$8$) | 4447.91    | 3309.44    | 3231.28     | 3182.4     |
| $\tau_{11}$ (ES$9$) | 973.43     | 796.42     | 147.2       | 147.2      |
| $\tau_{12}$ (ES$9$) | 693.2      | 670.08     | 647.04      | 644.48     |
| $\tau_{13}$ (ES$9$) | 973.43     | 796.42     | 147.2       | 147.2      |
| $\tau_{14}$ | 3915.79    | 2789.12    | 2716.32     | 2767.44    |

**TABLE 4.** Statistical information of the load.

| BAG | load | 14% | 16% | 14% | 16% | 14% | 16% | 14% | 16% |
|-----|------|-----|-----|-----|-----|-----|-----|-----|-----|
|     | 2ms  | 14% | 16% | 14% | 16% | 14% | 16% | 14% | 16% |
|     | 4ms  | 14% | 16% | 14% | 16% | 14% | 16% | 14% | 16% |
|     | 8ms  | 14% | 16% | 14% | 16% | 14% | 16% | 14% | 16% |
|     | 16ms | 14% | 16% | 14% | 16% | 14% | 16% | 14% | 16% |
|     | 32ms | 14% | 16% | 14% | 16% | 14% | 16% | 14% | 16% |
|     | 64ms | 10% | 16% | 14% | 16% | 14% | 16% | 14% | 16% |
|     | 128ms| 10% | 16% | 14% | 16% | 14% | 16% | 14% | 16% |
X. Tang et al.: RTA for the Worst-Case Delay Analysis of an AFDX Network

on average 41.1%, 14.34%, 5.63%, less pessimistic than the upper bounds computed by NC, CTA and OTA, respectively. Different from the existing the trajectory, RTA accumulates the maximum net delay of each node visited by the packet \( m \) under study. Besides, the frame constraints from the same flow and the idle times in the input sequences are considered. Thus, RTA achieves less pessimistic upper bounds than other existing trajectory approach.

**FIGURE 9.** Worst-case ETE delay of frame \( f_1 \) by hand built.

**FIGURE 10.** Worst-case ETE delays computed by NC, FA, RTA and ED.

**FIGURE 11.** Pessimism of worst-case ETE delays computed by NC, FA, RTA.

**FIGURE 12.** Worst-case ETE delay computed by NC, CTA, OTA and RTA.

**FIGURE 13.** The difference in percentage between the worst-case ETE delay obtained by RTA and other methods (NC, CTA, OTA).

**VII. CONCLUSION**

We have proposed a revised trajectory approach for computing worst-case ETE delays of flows in an AFDX network. The proposed method is applicable to obtain exact bounds.
It can reduce some pessimism compared to current trajectory approaches and eliminate the underestimation problem introduced by OTA. Compared to other approaches (NC and FA) on a case study, the results indicated that tighter bounds can be obtained by RTA than NC, and FA. Finally, applying the RTA method to other kinds of architectures, like Time-Triggered Ethernet, would be of the highest interest.

REFERENCES

[1] N. Benammar, F. Ridouard, H. Bauer, and P. Richard, “Forward end-to-end delay for AFDX networks,” IEEE Trans. Ind. Informat., vol. 14, no. 3, pp. 858–865, Mar. 2017.
[2] Aircraft Data Network, Parts 1, 2, 7, Standard ARINC 664, Aeronautical Radio, Inc., Annapolis, MD, USA, 2005.
[3] J. Y. L. Boudec and P. Thiran, Network Calculas: A Theory of Deterministic Queuing Systems for the Internet. Berlin, Germany: Springer Science & Business Media, 2001.
[4] 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.
[5] J. Yao, J. Wu, Q. Liu, Z. Xiong, and G. Zhu, “System-level scheduling of mixed-criticality traffics in avionics networks,” IEEE Access, vol. 4, pp. 5880–5888, 2016.
[6] H. Charara, J.-L. Scharbarg, J. Ermont, and C. Fraboul, “Methods for bounding end-to-end delays on an AFDX network,” in Proc. 18th Euromicro Conf. Real-Time Syst. (ECRTS), Dresden, Germany, Jul. 2006, p. 10.
[7] M. Adnan, J.-L. Scharbarg, J. Ermont, and C. Fraboul, “An improved timed automata approach for computing exact worst-case delays of AFDX sporadic flows,” in Proc. 17th Int. Conf. Emerg. Technol. Factory Automat. (ETFA), Kraków, Poland, Sep. 2012, pp. 1–8.
[8] J. J. Gutiérrez, J. C. Palencia, and M. G. Harbour, “Holistic schedulability analysis for multipacket messages in AFDX networks,” Real-Time Syst., vol. 50, no. 2, pp. 230–269, Mar. 2014.
[9] S. Martin and P. Minet, “Schedulability analysis of flows scheduled with FIFO: Application to the expedited forwarding class,” in Proc. 20th IEEE Int. Parallel Distrib. Process. Symp., Rhodes Island, Greece, Apr. 2006, p. 8.
[10] S. Martin and P. Minet, “Worst case end-to-end response times of flows scheduled with PP/FIFO,” in Proc. Int. Conf. Netw., Int. Conf. Syst. Int. Conf. Mobile Commun. Learn. Technol. (ICNICONSMCL), Morne, Mauritius, Apr. 2006, p. 54.
[11] 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.
[12] H. Bauer, J. L. Scharbarg, and C. Fraboul, “Applying trajectory approach with static priority queuing for improving the use of available AFDX resources,” Real-Time Syst., vol. 48, no. 1, pp. 101–133, Jan. 2012.
[13] X. Li, J. Scharbarg, and C. Fraboul, “Analysis of the pessimism of the trajectory approach for upper bounding end-to-end delay of sporadic flows sharing a switched Ethernet network,” in Proc. RTNS, Nantes, France, Sep. 2011, pp. 149–158.
[14] Y. Xiong, F. He, X. Li, C. Zhao, and H. Xiong, “Improving the serialisation of Trajectory approach for switched Ethernet network,” Electron. Lett., vol. 53, no. 13, pp. 845–847, Jun. 2017.
[15] S. Liu, F. He, T. Wang, and Y. Li, “Optimization of trajectory approach in end-to-end delay analysis considering the flow offsets scheduling,” in Proc. IEEE Region Conf. (TENCON), Nov. 2016, pp. 3614–3620.
[16] X. Li and L. George, “Deterministic delay analysis of avb switched Ethernet networks using an extended trajectory approach,” Real-Time Syst., vol. 53, no. 1, pp. 121–186, Jan. 2018.
[17] G. Kemayo, F. Ridouard, H. Bauer, and P. Richard, “Optimistic problems in the trajectory approach in FIFO context,” in Proc. IEEE 18th Conf. Emerg. Technol. Factory Automat. (ETFA), Cagliari, Italy, Sep. 2013, pp. 1–8.
[18] X. Li, O. Cros, and L. George, “The trajectory approach for AFDX FIFO networks revisited and corrected,” in Proc. IEEE 20th Int. Conf. Embedded Real-Time Comput. Syst. Appl., Chongqing, China, Aug. 2014, pp. 1–10.
[19] J. Yao, X. Xin, and L. Xue, “MixCPS: Mixed time/event-triggered architecture of cyber–physical systems,” Proc. IEEE, vol. 104, no. 5, pp. 923–937, May 2016.
[20] M. Adnan, “Exact worst-case communication delay analysis of AFDX network,” Ph.D. dissertation, Dept. Inst. Res. Comput. Sci. (IRIT), Univ. Toulouse, Toulouse, France, Nov. 2013.
[21] S. Medlej, “Scalable trajectory approach for ensuring deterministic guarantees in large networks,” Ph.D. dissertation, Dept. Lab. Res. Comput. Sci., Univ. Paris-Sub, Orsay, France, Sep. 2013.