Efficient End-to-End Communication Services for Mixed Criticality Avionics Systems

Ying Fang
McGill University
Montreal, Canada
yingfang@cs.mcgill.ca

Yu Hua
Huazhong Univ. of Sci. and Tech.
Wuhan, China
csyhua@hust.edu.cn

Xue Liu
McGill University
Montreal, Canada
xueliu@cs.mcgill.ca

***Abstract*—Avionics Full Duplex Switched Ethernet (AFDX) is a new data network standard for modern avionics systems. Avionics systems are typical mixed criticality systems, where safety-critical and non-safety-critical applications co-exist. Traditional AFDX leverages best-effort redundant transmissions for flows to improve the communication reliability, resulting in the increase of end-to-end delays with weak reliability. In order to bridge the gap between high reliability requirements and short end-to-end delays, we propose a novel scheme, named HERSA (Holistic End-to-End Redundant Scheduling on AFDX). The idea behind HERSA is to leverage a differentiated transmission mode to distinguish safety-critical and non-safety-critical flows, as well as an efficient credit-based scheduling scheme to reduce the serving size for each flow. The salient feature of HERSA is to obtain $O(1)$ complexity and offer the efficient end-to-end transmission services and the guaranteed reliability. We implement HERSA on a real test-bed. Experimental results demonstrate the efficiency and efficacy of HERSA.**

I. Introduction

Modern avionics architectures aim to support multiple functions on a shared, single and fault-tolerant platform. These functions have different levels of criticality and can be classified into safety-critical and non-safety-critical. For safety-critical systems, reliable real-time communication links are essential, say Avionics Full Duplex Switched Ethernet (AFDX). AFDX is a new data network standard building upon the protocol specifications of IEEE 802.3 [1] and ARINC 664 [2]. It has been used in many state-of-the-art aircrafts, such as Airbus A380/A350, Boeing 787, etc., and becomes an important communication standard in avionics applications.

A standard AFDX system consists of avionics subsystems, source/destination end systems and interconnect networks. Avionics subsystems, such as the flight control computer and the global positioning, indicate traditional avionics subsystems on an aircraft. The interconnect networks generally consist of a network of switches forwarding frames to their appropriate destinations. The end systems provide an interface between avionics subsystems and the interconnect networks to guarantee real-time and reliable data transmission by using Virtual Links (VLs). According to the AFDX specification [2], we can identify a VL by setting its 16-bit ID, Bandwidth Allocation Gap (the minimum interval between two consecutive frames in the VL) and the largest length of VL frames. In order to guarantee transmission reliability, the AFDX network provides redundancy management on VLs.

In modern avionics application, the amounts and types of data to be transmitted significantly increase, while the reliability requirements of these flows need to be guaranteed. *How to provide reliable and efficient communication for mixed criticality systems has been a non-trivial and important problem*. Specifically, we need to deal with two main challenges in avionics applications.

Challenge 1: Best-effort Redundant Transmissions. The design goal of an AFDX network is to offer high reliability for data transmission. Hence, based on the standard AFDX specification, the flows need to be redundantly transmitted to protect the network from mechanical failures in a network [3], [4]. Transmitting each frame in parallel over two independent networks increases the transmission overheads. The redundant management unit at the destination end system identifies and deletes duplicate frames, which also increases the computation complexity. Full redundant transmission for all mixed criticality network flows consumes too much bandwidth. The traffic congestion in practice exacerbates the bandwidth efficiency. This causes longer latency for data transmission and increases the possibility of overflows in the queue buffer. The reliability goal in practice is difficult to achieve.

Challenge 2: Inefficient Scheduling Schemes. An AFDX network contains multi-type heterogeneous flows with different transmission requirements, such as avionics, multimedia and best-effort data. Avionics flows involve flight-critical data, thus requiring the highest reliable and real-time services. Best-effort data traffics are related with non-critical avionics applications, such as file transmission and data backup, thus requiring the lowest reliable and real-time services. In a practical avionics application, most frames in the avionics flows are short frames [5] [6]. As a result, compared with other heterogeneous QoS networks, short frames in the AFDX network are preferred to send.

Traditional scheduling schemes, such as First-In-First-Out (FIFO) [7] and Static-Priority (SP) [8] [9], are inefficient to deal with heterogeneous flows since they heavily depend on static "one-size-fit-all" schemes. Sorted priority schedulers, such as Self-Clocked Fair Queuing (SCFQ) [10], transmit packets in some priority order. Sorted priority algorithms generally have good latency and fairness properties, but incur high complexity due to the priority sorting process [11]. Real-world environments need to design simple algorithms that have a constant number of operations per packet transmission. Round robin algorithms, such as Weighted Round Robin (WRR) [12], Deficit Round Robin(DRR) [13] [14] [15], Elastic Round Robin (ERR) [16] [17], Nested DRR (NDRR) [11], etc., cyclically serve for different flows. Each flow is served for up to a given quantum in per round. The round robin algorithms, hence, have O(1) worst-case per packet complexity.

978-1-4799-4852-9/14/$31.00

However, DRR has high latency and poor fairness when the flows with different rate requirements are scheduled. NDRR modifies DRR by leveraging multiple frames inside each DRR frame, thus resulting in a lower latency bound. However, in per inner round of NDRR, all flows receive the same amounts of services in spite of their different QoS requirements. Moreover, an avionics packet size is generally much smaller than the minimum serving size (i.e., a given quantum) in an inner round. If the minimum serving size for avionics packets can be reduced, the aggregated delay for all flows can be smaller, especially for large amounts of simultaneously arriving avionics flows.

In order to address the above challenges, Holistic End-to-End Redundant Scheduling on AFDX (HERSA) scheme offers a differentiated transmission mode and smaller serving sizes to alleviate the traffic congestion and reduce the queuing delays. This paper makes the following contributions:

Guaranteed Reliability. Our scheme only transmits safety-critical flows in a redundant manner, while other flows are transmitted non-redundantly. Hence, the number of the transmitted frames is significantly reduced and we obtain the lightweight network load. Moreover, the non-safety-critical flows cause less congestion, compared with the case in the redundant transmissions. Therefore, the idea of differentiated transmission mode helps provide the guaranteed reliability for the mixed criticality network flows in the avionics applications.

Low End-to-End Transmission Latency. In order to meet different transmission requirements from heterogeneous flows, we design HERSA by providing a flow service proportional to its guaranteed rate. In order to reduce the queuing delay of each flow, in HERSA, we decrease the serving size of each flow in each service opportunity by limiting the sum (in bytes) of virtual packets. The virtual packet refers to the packet whose length is multiplied by a factor that increases with the number of transmitted packets. Hence, the service interval for each flow decreases.

Prototype Implementation. To examine the performance of our proposed scheme, we conduct simulations on a real test-bed and evaluate end-to-end delays for heterogeneous flows by using FIFO, DRR, NDRR and HERSA. All the settings and configurations on the test-bed can simulate a typical AFDX network. The results demonstrate the efficacy and efficiency of HERSA.

The rest of the paper is organized as follows. Section II presents the backgrounds in the differentiated transmission. Section III presents a new switching algorithm, as well as its latency bound and fairness. Section IV presents theoretical analysis of the AFDX network performance based on the HERSA. Section V shows the evaluation results and analysis via a simulation prototype. Section VI presents the related work. Section VII concludes the paper.

II. End-to-End Delays Under Differentiated Transmission Mode

A. Redundant End-to-End Delays Analysis

In order to offer reliable services, AFDX provides a redundancy scheme to protect the network against failures from a network. The redundancy management is configurable. In a redundant transmission manner, the same frame is transmitted with identical sequence number through two independent networks, e.g., networks A and B. Therefore, in general, the destination end system will receive two copies of the identical packet. The destination end system only accepts the first valid frame and discards the redundant one. The parameter $SkewMax$ represents the maximum time between two received copies. This value usually depends on the network topology and should be provided by the network manager.

Let D_{before} be the delay before the packet arriving at the destination end system. Before-destination delay D_{before} can be split into three parts, including the delay at the source end system, the delay on transmission links, and the delay on switches, i.e., $D_{before} = D_{source} + D_{transmit} + D_{switch}$. Specifically, D_{source} is the processing delay of generating constrained flow at the source end system. It comes from selecting the transmitted frame from a virtual link queue, assigning the sequence number, duplicating the frame and transmitting the frame and its copy on the physical links. $D_{transmit}$ is the transmission delay over the links. We usually treat it as a constant when the network configuration is determined. Moreover, D_{switch} is the switching delay from source to destination end systems. It can be divided into technological delay and queuing delay. The technological delay is bounded by a small constant for specified hardware capacity [18], and the queuing delay depends on scheduling algorithms. In this paper, we focus on the queuing delay.

Let D_{dest} be the processing delay at the destination end system.

Corollary 2.1: The upper bound of the end-to-end delay of a flow transmitting through an AFDX network redundantly, $E_r(D)$, is given by

$$E_r(D) = \begin{cases} D^1_{before} + SkewMax + D_{dest}, & \text{if } \Delta_2 > SkewMax \\ D^2_{before} + D_{dest}, & \text{if } \Delta_2 < SkewMax \end{cases}$$

where D^1_{before} and D^2_{before} respectively represent the before-destination delays of original and redundant flows, Δ_2 indicates the arriving time difference of the two copies at the destination end system, and $SkewMax$ is maximum time between receiving the two copies.

Proof: The upper bound of end-to-end delays rely on the values of $SkewMax$ and Δ_2. If Δ_2 is larger than $SkewMax$, the first frame spends a D_{dest} for processing data and then waits for a $SkewMax$ to complete the end-to-end transmission. Otherwise, the end-to-end transmission completes when the second frame is discarded and the upper bound is the sum of before-destination delay of the redundant flow and the processing delay at the destination. ■

B. Non-redundant End-to-End Delays Analysis

In a non-redundant transmission manner, a flow is transmitted through either the network A or B. Hence, the destination end system avoids de-duplicating copies.

Corollary 2.2: The upper bound of the end-to-end delay of a flow transmitting through an AFDX network non-redundantly, $E_{nr}(D)$, is given by

$$E_{nr}(D) = D_{before} + D_{dest}.$$

Proof: For a flow that is not transmitted redundantly, the upper bound of the end-to-end delay can be split into four parts: the delay at the source end system, the delay on the transmission link, the delay on the switches, and delay at the destination end system, i.e., the delay before the packet arriving at the destination end system, as well as the processing delay at the destination end system. ■

III. Scheduler Design for HERSA

A. Scheduling Design

DRR is a popular Round Robin scheduler with $O(1)$ complexity. In DRR, each flow is assigned a quantum proportional to its share of bandwidth. For each flow, a deficit counter indicates the remaining quantum in the current round. In each round, a flow can receive the amount of services up to the sum of its quantum size and the deficit value. The latency and fairness of DRR are not optimal.

To improve the latency of DRR, NDRR scheduler has been proposed [11]. In NDRR, each outer round of DRR is split into smaller inner rounds and a version of DRR is executed over these inner rounds. During each inner round, the maximum of services is the sum of the smallest quantum (the quantum value of the flow with the lowest reserved rate) and the deficit value. Since the smallest quantum is larger than or equal to the maximum packet size, it is generally much larger than the small avionics packet size. Furthermore, the heterogeneous flows with the same minimum size decrease the quality of service.

In order to leverage the input traffic pattern and decrease the serving size of flows, HERSA splits each inner round of NDRR into one or more smaller micro rounds. In each micro round, the minimum serving size is related with the packet size. Each flow has a credit counter ($Allowance$). At the beginning of each micro round, $Allowance$ is initialized to the smallest quantum. The packet can be transmitted only if the packet size is smaller than or equal to $Allowance$. After successfully transmitting a packet, $Allowance$ decreases by the virtual size of the transmitted packet. The virtual size equals to the real packet size multiplied by a factor that increases with the number of transmitted packets. As a result, different flows have various serving sizes in one service opportunity.

In the outer rounds, HERSA characterizes flow i by the quantum Q_i, the reserved rate ρ_i , the reserved quantum UQ_i, and the weight ω_i. The smallest rate of all active flows is represented as ρ_{min}. Q_{min} is the quantum value of the flow with the lowest reserved rate. ω_i is the weight assigned to flow i, and is given by,

$$\omega_i = \frac{\rho_i}{\rho_{min}}. \quad (1)$$

Note that $\omega_i \geq 1$. The quantum Q_i in HERSA is given by,

$$Q_i = \omega_i Q_{min}. \quad (2)$$

Three deficit counters, ODC_i, IDC_i and MDC_i depict the differences in the amounts of data actually to be sent, and the amounts that should have been sent in each outer round, inner round and micro round respectively. During each outer round, the scheduler attempts to serve flow i with the quantum of Q_i. During each inner round, it tries to serve flow i with the quantum of Q_{min}. During each micro round, the serving size is related with packet sizes, and up to the quantum of Q_{min}.

In order to implement HERSA, we maintain three lists of active flows, ActiveList(0), ActiveList(1) and ActiveList(2). The flows in the ActiveList(0) have completed services for the current outer round. ActiveList(1) maintains a list of flows that have completed services for the current inner round. ActiveList(2) maintains another list of flows that are served in the current micro round.

The HERSA scheduler can be divided into three parts: Initialize, Enqueue and Dequeue. In the Initialize process, all of the deficit counters are set to 0. A flow whose queue is empty and therefore is not in any of the three lists is called an inactive flow. When an inactive flow i becomes active, it should be added into the ActiveList(2) and UQ_i should be set to $(Q_i - Q_{min})$, while MDC_i should be set to Q_{min}. This process calls Enqueue routine. The Dequeue routine schedules packets out of queues. The pseudo-code implementation of Dequeue is shown as Figure 1.

The current inner round in progress ends when the ActiveList(2) becomes empty. In this case, if the ActiveList(1) is not empty, all flows in the ActiveList(1) have to participate in the next inner round. ActiveList(1) and ActiveList(2) are exchanged. Otherwise, the current outer round in progress ends and all flows in ActiveList(0) have to participate in the next outer round. ActiveList(0) and ActiveList(2) are exchanged. When the ActiveList(2) is not empty, HERSA serves the first flow in the list, say flow i. When transmitting a packet from flow i, the scheduler first checks whether the $Allowance_i$ is larger than or equal to the virtual packet size. If so, the packet is transmitted. MDC_i decreases by the actual size (in bytes) of the transmitted packet, while $Allowance_i$ decreases by the virtual packet size. Otherwise, the packet can not be transmitted and the current micro round service for flow i ends. In this case, if MDC_i is larger than or equal to the packet size, flow i should be added to the tail of the ActiveList(2) to participate in the next micro round. If the packet size is larger than MDC_i, but smaller than the sum of MDC_i and UQ_i, flow i ends the current inner round services. The value of MDC_i is assigned to IDC_i to record the deficit quantum in the inner round. Flow i is added to the tail of the ActiveList(1).

The quantum of service in the next inner round can have a $min\{UQ_i, Q_{min}\}$ compensation. If the packet size is larger than the sum of MDC_i and UQ_i, the scheduler ends the current outer round services for flow i and adds flow i to the tail of ActiveList(0). The sum of MDC_i and UQ_i is assigned to ODC_i to record the deficit quantum in the current outer round. The next outer round obtains a Q_i service compensation, UQ_i hence becomes Q_i. At the beginning of the first micro round in the next outer round, HERSA adds $min\{UQ_i, Q_{min}\}$ to MDC_i and reduces UQ_i by the same value.

To select the next packet to be transmitted, it is unnecessary for HERSA to sort or search process. In HERSA, the number of operations needed to select the next packet is a constant. Hence, the work complexity of the HERSA scheduler is O(1).

```
Dequeue Algorithm
while TRUE do
  if SizeOfActiveList(2)==0 then
    if SizeOfActiveList(1)>0 then
      MoveActiveListFromTo(1, 2);
    else
      if SizeOfActiveList(0)> 0 then
        MoveActivelistFromTo(0, 2);
      end if
    end if
  end if
  if SizeOfActiveList(2)> 0 then
    i = HeadOfActiveList(2);
    RemoveheadOfActiveList(2);
    Allowance_i = Q_min;
    count = 0;
    repeat
      Transmitpacket(Queue(i));
      count++;
      Allowance_i -= count * LengthOftransmittedPacket;
      MDC_i = MDC_i - LenthOftransmittedPacket
    until IsEmpty(Queue(i)) == TRUE) or
    Allowance_i < (count + 1) * LenthOfPacketAtHead(i)
    if IsEmpty(Queue(i) == TRUE then
      MDC_i = 0;
      IDC_i = 0;
      ODC_i = 0;
      DecrementSizeOfActiveList(2);
    else
      if UQ_i + MDC_i < LengthOfPacketAtHead(i) then
        AddQueueToActiveList(0);
        IncrementSizeOfActiveList(0);
        DecrementSizeOfActiveList(2);
        ODC_i = MDC_i + UQ_i;
        IDC_i = ODC_i;
        UQ_i = Q_i - min{UQ_i, Q_min};
        MDC_i = IDC_i + min{UQ_i, Q_min};
      else
        if MDC_i > LengthOfPacketAtHead(i) then
          AddqueueToActiveList(1);
          IncrementSizeOfActiveList(1);
          DecrementSizeOfActiveList(2);
          IDC_i = MDC_i;
          UQ_i = UQ_i - min{UQ_i, Q_min};
          MDC_i = MDC_i + min{UQ_i, Q_min};
        else
          AddQueueToActiveList(2);
        end if
      end if
    end if
  end if
end while
```

Fig. 1. Dequeue algorithm.

B. Latency Bound of HERSA

In order to analyze the performance of HERSA, we derive its upper bound on the scheduling latency. Let $ODC_i(k)$ be the deficit quantum of flow i following its k-th outer round. $IDC_i(k,s)$ is the deficit quantum of flow i following the s-th inner round inside the k-th outer round. $MDC_i(k,s,u)$ is the deficit quantum of flow i following the u-th micro round on the s-th inner round inside the k-th outer round. $(k,s,0)$ represents the beginning of the first micro round in the (k,s)-th inner round. Since MDC_i obtains a compensation at the beginning of a new inner round, the equation is

$$MDC_i(k,s,0) = IDC_i(k,s-1) + min\{UQ_i, Q_{min}\}. \quad (3)$$

$(k,0)$ represents the first inner round inside the k-th outer round. Note that $IDC_i(k,0) = ODC_i(k-1)$.

The bytes sent by flow i during the (k,s,u)-th micro round, $Sent_i(k,s,u)$, is given by,

$$Sent_i(k,s,u) = MDC_i(k,s,u-1) - MDC_i(k,s,u). \quad (4)$$

Note that $u \geq 1$.

The bytes sent by flow i during the (k,s)-th inner round, $Sent_i(k,s)$, is given by,

$$Sent_i(k,s) = Q_{min} + IDC_i(k,s-1) - IDC_i(k,s). \quad (5)$$

Note that $s \geq 1$.

The bytes sent by flow i during the k-th outer round, $Sent_i(k)$, is given by,

$$Sent_i(k) = \omega_i Q_{min} + ODC_i(k-1) - ODC_i(k). \quad (6)$$

Note that $k \geq 1$, $ODC_i(0) = 0$.

Corollary 3.1: For all i, the following representation holds for each execution of HERSA: $0 \leq ODC_i(k) < m$, $0 \leq IDC_i(k,s) < m$, $0 \leq MDC_i(k,s,u) < m + Q_{min}$, where m represents the biggest size in bits of those packets that are actually served.

Proof: Initially, $ODC_i(0) = 0$, $IDC_i(0,0) = 0$ $MDC_i(0,0,0) = 0$. $ODC_i(k)$ only changes its value when flow i completes the k-th outer round of service. The outer round ends when the sum of $UQ_i(k)$ and $MDC_i(k)$ is less than the length of the packet to be transmitted. Since the packet length is no larger than m, naturally $ODC_i(k)$ is less than m. Likewise, we obtain that $IDC_i(k,s)$ is less than m. $MDC_i(k,s,u)$ changes its value when the (k,s,u)-th micro round ends. In one service opportunity, at least one packet can be sent in each service opportunity since Q_{min} is larger than the maximum packet size, i.e., $Sent_i(k,s,u) > 0$. According to Equations (3) and (4), the maximum value of $MDC_i(k,s,u)$ should be $MDC_i(k,s,1)$, which is less than $Q_{min} + m$. ■

Let τ_i be the time instant that flow i becomes active and coincides with the beginning of some outer round k_0. Let $\tau_i^{(k,\lceil\omega_i\rceil,last)}$ be the time instant that flow i receives the last micro round on the $(k,\lceil\omega_i\rceil)$-th inner round of service. $\lceil\omega_i\rceil$ means the minimum integer that is larger than or equal to ω_i. Let $Sent_i(t_1,t_2)$ be the amount of service received by flow i during the time interval (t_1,t_2). At the time instance $\tau_i^{(k,\lceil\omega_i\rceil,last)}$, let G_0 denote the set of flows in the AtiveList(0). Let G_1 and G_2 represent the set of flows in the ActiveList(1) and the ActiveList(2) respectively. Flow i is not included in any of the three lists.

In order to achieve the worst-case scheduling latency, we assume that all n flows are active in the time interval of $(\tau_i, \tau_i^{(k,\lceil\omega_i\rceil,l)})$ and flow i is the last to receive services in a micro round.

Theorem 3.2: The HERSA scheduler belongs to the class of Latency-Rate servers, with an upper bound on the latency, Θ_i^b for flow i, given by,

$$\begin{aligned}\Theta_i^b = \frac{1}{r}\Big(&\sum_{j\in G_0}\{\omega_j Q_{min}\} + \sum_{j\in G_1}\{\lceil\omega_i\rceil Q_{min}\} + \sum_{j\in G_2}\{\lceil\omega_i\rceil Q_{min}\}\Big)\\ &+ \Big(\frac{1}{r} - \frac{1}{\rho_i}\Big)(Q_{min} - m + 1) + \frac{1}{r}(n-1)(m-1)\\ &+ \Big(\frac{1}{r} - \frac{1}{\rho_i}\Big)(\lceil\omega_i\rceil - 1)Q_{min}) + \Big(\frac{1}{r} - \frac{1}{\rho_i}\Big)(Q_{min} - m + 1).\end{aligned}$$

Proof: Let $t(k-1)$ be the time to indicate the completion oft the $(k+k_0-2)$-th outer round. To analyze the latency bound, the first step is to obtain an upper bound on the size of the time interval $(\tau_i, \tau_i^{(k,\lceil\omega_i\rceil,last)})$. The time interval can be split into the following two sub-intervals:

1) (τ_i, t_{k-1})*:* The sub-interval includes $(k-1)$ outer rounds starting at round k_0. Let W be the sum of all the flow weights and r be the link rate. Summarizing Equation (6) over all n flows and over $(k-1)$ outer rounds beginning with the outer round k_0, we obtain

$$t_{k-1}-\tau_i = \frac{W}{r}(k-1)Q_{min} + \frac{1}{r}\sum_{j=1}^{n}\{ODC_j(k_0-1) - ODC_j(k_0+k-2)\}.$$

2) $(t_{k-1}, \tau_i^{(k,\lceil\omega_i\rceil,last)})$*:* In this sub-interval, we consider flow i, as well as flows in the sets G_0, G_1 and G_2.

Flows belonging to the set G_0 have completed the (k_0+k-1)-th outer round. The total bytes sent by these flows during the sub-interval are

$$\sum_{j\in G_0} Sent_j(t_{k-1}, \tau_i^{(k,\lceil\omega_i\rceil,last)}) = \sum_{j\in G_0}\{\omega_j Q_{min}\} + \sum_{j\in G_0}\{ODC_j(k_0+k-2) - ODC_j(k_0+k-1)\}.$$

Flows in the set G_1 have completed the $\lceil\omega_i\rceil$-th inner round of service inside the (k_0+k-1)-th outer round. The total bytes sent by these flows during the sub-interval are

$$\sum_{j\in G_1} Sent_j(t_{k-1}, \tau_i^{(k,\lceil\omega_i\rceil,last)}) = \sum_{j\in G_1}\{\lceil\omega_i\rceil Q_{min}\} + \sum_{j\in G_1}\{ODC_j(k_0+k-2) - IDC_j(k_0+k-1, \lceil\omega_i\rceil)\}.$$

Flows in the set G_2 have completed the $(k_0+k-1, \lceil\omega_i\rceil, last)$-th service opportunity. Since the last inner round may get a compensation less than Q_{min}, using Equations (4) and (5), we obtain

$$\begin{aligned}&\sum_{j\in G_2} Sent_j(t_{k-1}, \tau_i^{(k,\lceil\omega_i\rceil,last)})\\ &\le \sum_{j\in G_2}\{\lceil\omega_i\rceil Q_{min}\} + \sum_{j\in G_2}\{ODC_j(k_0+k-2)\}\\ &- \sum_{j\in G_2}\{MDC_j(k_0+k-1, \lceil\omega_i\rceil, last)\}.\end{aligned}$$

The total bytes sent by flow i during the sub-interval are given by

$$\begin{aligned}&Sent_i(t_{k-1}, \tau_i^{(k,\lceil\omega_i\rceil,last)})\\ &= (\lceil\omega_i\rceil-1)Q_{min} + ODC_i(k_0+k-2)\\ &- IDC_i(k_0+k-1, \lceil\omega_i\rceil-1) + L(i),\end{aligned}$$

where L(i) represents the bytes sent by flow i during the micro round one through $last$ on the $(k_0+k-1, \lceil\omega_i\rceil)$-th inner round, which is given by, $L(i) = MDC_i(k, \lceil\omega_i\rceil, 0) - MDC_i(k, \lceil\omega_i\rceil, last)$. Since the $(k, \lceil\omega_i\rceil, last)$-th micro round is the last service opportunity on the $(k, \lceil\omega_i\rceil)$-th inner round, $MDC_i(k, \lceil\omega_i\rceil, last)$ must be less than m. Hence, we have

$$L(i) \ge MDC_i(k, \lceil\omega_i\rceil, 0) - m + 1 \ge Q_{min} - m + 1.$$

Note that $ODC_i(k_0-1) = 0$ since flow i becomes active at the k_0-th outer round, according to Corollary 3.1, summarizing all active flows over the two periods, we obtain

$$\begin{aligned}&\tau_i^{(k,\lceil\omega_i\rceil,l)} - \tau_i \le\\ &\frac{W}{r}(k-1)Q_{min} + \frac{1}{r}\sum_{j\in G_0}\{\omega_j Q_{min}\}\\ &+ \frac{1}{r}\sum_{j\in G_1}\{\lceil\omega_i\rceil Q_{min}\} + \frac{1}{r}\sum_{j\in G_2}\{\lceil\omega_i\rceil Q_{min}\}\\ &+ \frac{1}{r}L(i) + \frac{1}{r}(\lceil\omega_i\rceil-1)Q_{min}\\ &+ \frac{1}{r}(n-1)(m-1) - \frac{1}{r}IDC_i(k_0+k-1, \lceil\omega_i\rceil-1).\end{aligned} \tag{7}$$

During the time interval under consideration, flow i receives services over $(k-1)$ outer rounds, $(\lceil\omega_i\rceil-1)$ inner rounds and $last$ micro rounds,

$$\begin{aligned}&Sent_i(\tau_i, \tau_i^{(k,\lceil\omega_i\rceil,l)}) = (k-1)\omega_i Q_{min} + ODC_i(k_0-1)\\ &- IDC_i(k_0+k-1, \lceil\omega_i\rceil-1) + (\lceil\omega_i\rceil-1)Q_{min} + L(i).\end{aligned} \tag{8}$$

Note that the reserved rates are proportional to the weights, using Equations (7) and (8), we obtain

$$\begin{aligned}&Sent_i(\tau_i, \tau_i^{(k,\lceil\omega_i\rceil,l)}) \ge max\Bigg\{0, \rho_i\Bigg(\tau_i^{(k,\lceil\omega_i\rceil,l)} - \tau_i\\ &- \frac{1}{r}\Big(\sum_{j\in G_0}\{\omega_j Q_{min}\} + \sum_{j\in G_1}\{\lceil\omega_i\rceil Q_{min}\}\\ &+ \sum_{j\in G_2}\{\lceil\omega_i\rceil Q_{min}\}\Big) - \Big(\frac{1}{r}-\frac{1}{\rho_i}\Big)(Q_{min}-m+1)\\ &- \frac{1}{r}(n-1)(m-1) - \Big(\frac{1}{r}-\frac{1}{\rho_i}\Big)Q_{min}\\ &- \Big(\frac{1}{r}-\frac{1}{\rho_i}\Big)(\lceil\omega_i\rceil-1)Q_{min}\Bigg)\Bigg\}.\end{aligned} \tag{9}$$

The latency bound reaches the upper bound when $IDC_i(k_0+k-1, \lceil\omega_i\rceil-1) = m-1$ and $L(i) = Q_{min}-m+1$.

It is proved in [19] that the latency bound of a scheduler can be determined by considering only those periods during which a flow is continuously active. From the Lemma 7 in [19], we know that HERSA is a Latency-Rate server with a latency less than or equal to Θ_i^b given by,

$$\begin{aligned}\Theta_i^b &= \frac{1}{r}\Big(\sum_{j\in G_0}\{\omega_j Q_{min}\} + \sum_{j\in G_1}\{\lceil\omega_i\rceil Q_{min}\} + \sum_{j\in G_2}\{\lceil\omega_i\rceil Q_{min}\}\Big)\\ &+ \Big(\frac{1}{r}-\frac{1}{\rho_i}\Big)(Q_{min}-m+1) + \frac{1}{r}(n-1)(m-1)\\ &+ \Big(\frac{1}{r}-\frac{1}{\rho_i}\Big)(\lceil\omega_i\rceil-1)Q_{min}) + \Big(\frac{1}{r}-\frac{1}{\rho_i}\Big)(Q_{min}-m+1).\end{aligned}$$

■

Theorem 3.3: For any execution of the HERSA scheduler, the inequality $FM < M + 2m$ holds, where M is the size of the largest packet that may potentially arrive.

Proof: FM is defined to be the maximum value of $FM(t_1, t_2)$ over all the possible executions of the fair queueing scheme and all possible intervals (t_1, t_2) in an execution [13]. $FM(t_1, t_2)$ is the maximum of $(\frac{Sent_i(t_1,t_2)}{\omega_i} -$

$\frac{Sent_j(t_1,t_2)}{\omega_j}$), over all pairs of flows i, j that are backlogged in the interval (t_1, t_2)[13]. Since HERSA works in rounds, time here can be measured in terms of rounds.

For any time interval (t_1, t_2), during which flow i receives the (x_1, y_1, z_1)-th micro round of service starting from the (x_0, y_0, z_0)-th micro round, we derive

$$\begin{aligned} Sent_i(t_1,t_2) &= (x_1-x_0)Q_i + (y_1-y_0)Q_{min} \\ &\quad + MDC_i(x_0,y_0,z_0-1) - MDC_i(x_1,y_1,z_1) \\ &\le (x_1-x_0)Q_i + (y_1-y_0+1)Q_{min} + m. \end{aligned} \quad (10)$$

For a flow j in the set G_0, the minimum bytes sent from flow j are

$$\begin{aligned} Sent_j(t_1,t_2) &= (x_1-x_0-1)Q_j + (\omega_j-y_0)Q_{min} \\ &\quad + MDC_j(x_0,y_0,z_0) - ODC_j(x_1-1) \\ &\ge (x_1-x_0-1)Q_j + (\omega_j-y_0)Q_{min} - m. \end{aligned} \quad (11)$$

As $\frac{Q_i}{\omega_i} = \frac{Q_j}{\omega_j} = Q_{min}$, using Equations (10) and (11), we derive

$$|\frac{Sent_i(t_1,t_2)}{\omega_i} - \frac{Sent_j(t_1,t_2)}{\omega_j}| \le Q_{min} + 2m. \quad (12)$$

The Equation follows due to $\omega_i \ge 1$ and $\omega_j \ge 1$.

For the flow j in the sets G_1 and G_2, $|\frac{Sent_i(t_1,t_2)}{\omega_i} - \frac{Sent_j(t_1,t_2)}{\omega_j}| \le Q_{min} + 2m$.

When Q_{min} is set to M, we derive $FM \le M + 2m$. The fairness of the HERSA scheduler is identical to that of the NDRR scheme[11]. ■

IV. AFDX Communications based on HERSA

In HERSA, we distinguish safety-critical flows from non-safety-critical flows in avionic applications using different transmission modes. To guarantee the communication reliability of the safety-critical flows, we transmit them redundantly. To reduce the network load, we transmit non-safety-critical flows through either networks A or B. In order to show the advantages of HERSA, we compare HERSA with the NDRR scheduler that schedules full redundant flows in the same applications.

Assumption 4.1: There are N flows in the AFDX network, among which n_s flows are safety-critical. Each network has n switches with the same capacity and configuration.

A. Safety-critical Flows

For a safety-critical flow i, its end-to-end delay bound is related with the arrival time difference on the two networks and the system parameter $SkewMax$. The value of $SkewMax$ is the number of switches crossed by the transmitted frames and usually provided by the system administrator. The arrival time difference is equal to the before-destination delay difference.

In this paper, we make use of leaky bucket function as arrival curve, i.e., $\alpha(t) = \sigma_i + \rho_i t$, where σ_i is the burst and ρ_i is the rate. From the Theorem 4 in [19], the maximum switching delay D_i^n of flow i in a network of latency-rate servers, consisting of n servers in series, is bounded as

$$D_i^n \le \frac{\sigma_i}{\rho_i} + \sum_{j=1}^{n} \Theta_i^j, \quad (13)$$

where Θ_i^j is the latency bound of the j-th switch in the network for flow i.

In HERSA, for safety-critical flows, the original frames are transmitted through the network A, while the redundant frames are sent through the network B. There are n_1 and n_2 flows on the networks A and B respectively. ω_{nsafe} is the total weight of the non-safety-critical flows. On the network A, l_1 flows have weights smaller than ω_i, and the sum of their weights is ω_i^{l1}. On the network B, l_2 flows have weights smaller than ω_i, and the sum of their weights is ω_i^{l2}. Using the Theorem 3.2 and Equation (13), the switching delay difference of flow i on the two networks is bounded by

$$\begin{aligned} D^i_{switch_2} - D^i_{switch_1} &= n\{\frac{1}{r}Q_{min}(\omega_i^{l2} - \omega_i^{l1}) \\ &+ \frac{1}{r}Q_{min}\lceil\omega_i\rceil(n_2 - n_1 + l_1 - l_2) + \frac{1}{r}(n_2 - n_1)(m-1)\}, \end{aligned} \quad (14)$$

where $D^i_{switch_1}$ and $D^i_{switch_2}$ are the switching delays on the original network and the redundant network respectively. Let Δ_1 be the transmission interval between the original frame and its redundant copy at the source end system bounded by 0.5ms. The arrival time difference Δ_2 in HERSA is given by, $\Delta_2 = D^i_{switch_2} - D^i_{switch_1} + \Delta_1$. This arrival time difference indicates the principles for flow configurations: The number of non-safety-critical flows and the sum of their weights on one network should be approximate to those on the other network. Otherwise, safety-critical flows wait long for the arrival of their copies. The possibility of overflows and the computation complexity hence increase at the end system.

Let $D'^i_{switch_1}$ and $D'^i_{switch_2}$ be the switching delays on the networks A and B respectively in the NDRR scheduler. It is proved in Ref. [11] that the NDRR scheduler has an upper bound on the latency Θ_i for flow i given by,

$$\begin{aligned} \Theta_i &= \frac{1}{r}\sum_{j\in G_{over}}\{\omega_j M\} + \frac{1}{r}\sum_{\substack{j\in(n-G_{over})\\ j\neq i}}\{\lceil\omega_i\rceil M\} \\ &+ (\lceil\omega_i\rceil - 1)M(\frac{1}{r} - \frac{1}{\rho_i}) + \frac{1}{r}(n-1)(m-1) \\ &+ (m-1)(\frac{1}{\rho_i} - \frac{1}{r}). \end{aligned} \quad (15)$$

The switching delay difference is zero when the two networks transmit the same flows. Therefore, the arrival time difference in NDRR is only related with Δ_1.

If $D^i_{switch_2} - D^i_{switch_1} + \Delta_1 < SkewMax$, the end-to-end delays of transmitting any packet of flow i under both schemes come from the redundant network. Note that safety-critical flows are transmitted in a redundant manner in HERSA, their delays at the source/destination end systems and on the links are the same as those in the NDRR scheduler. The end-to-end delay difference ΔE_r between HERSA and NDRR hence is equal to the switching delay difference, and is given by,

$$\begin{aligned} \Delta E_r &= D'^i_{switch_2} - D^i_{switch_2} \\ &= n\{\frac{1}{r}M\frac{\omega_i^{l1} - \omega_i^{l2} + \omega_{nsafe}}{2} + \frac{1}{r}(m-1)(N-n_2)\}, \end{aligned}$$

where $\omega_i^{l1} - \omega_i^{l2}$ depicts the difference of the weights of non-safety-critical flows on the two networks. The weights difference is smaller than the sum of weights of all non-safety-critical flows on the two networks, i.e., $\omega_i^{l1} - \omega_i^{l2} + \omega_{nsafe} > 0$. In addition, from the definitions of m, N and n_2, $(m-1)(N-n_2) > 0$, we determine $\Delta E_r > 0$.

If $D^i_{switch_2} - D^i_{switch_1} + \Delta_1 > SkewMax$ and $\Delta_1 > SkewMax$, the end-to-end delays in HERSA and DRR come from the original network. According to Corollary 2.1, the end-to-end delay difference is given by,

$$\begin{aligned}\Delta E_r &= D'^i_{switch_1} - D^i_{switch_1} \\ &= n\{\frac{1}{r}M\frac{\omega_i^{l2} - \omega_i^{l1} + \omega_{nsafe}}{2} + \frac{1}{r}(m-1)(N-n_1)\} > 0.\end{aligned}$$

If $D^i_{switch_2} - D^i_{switch_1} + \Delta_1 > SkewMax$, but $\Delta_1 < SkewMax$, the end-to-end delays in HERSA and NDRR come from the original network and the redundant network respectively. Since the delay difference at the source end system is Δ_1, the end-to-end delay difference is given by,

$$\Delta E_r = D'^i_{switch_1} - D^i_{switch_1} + \Delta_1 - SkewMax.$$

When $n\{\frac{1}{r}M\frac{\omega_i^{l1} - \omega_i^{l2} + \omega_{nsafe}}{2} + \frac{1}{r}(m-1)(N-n_2)\} > Skewmax - \Delta_1$, the end-to-end delays of safety-critical flows under the HERSA scheme turn out to be much shorter than those under the NDRR scheme.

B. Non-safety-critical Flows

In HERSA, for a non-safety-critical flow j, it is transmitted only on the network A. There are l_{j1} flows whose weights are smaller than ω_j on the network A, and the sum of their weights is ω_j^{lj1}. On the network B, l_{j2} flows have weights smaller than ω_j, and the sum of their weights is ω_j^{lj2}. The switching delay $D^j_{switch_1}$ of the flow j, is given by,

$$\begin{aligned}D^j_{switch_1} = \frac{\sigma_j}{\rho_j} &+ n\{\frac{1}{r}M\omega_j^{lj1} + \frac{1}{r}\lceil\omega_j\rceil M(n_1 - l_{j1} - 1) \\ &+ (\frac{1}{r} - \frac{1}{\rho_j})(M - m + 1) + \frac{1}{r}(n_1 - 1)(m-1) \\ &+ (\frac{1}{r} - \frac{1}{\rho_j})(\lceil\omega_j\rceil - 1)M\}.\end{aligned}$$

In NDRR, the end-to-end delay of the flow is related with Δ_1 and $SkewMax$. The switching delay of the flow j on the original network, $D'^j_{switch_1}$, is given by,

$$\begin{aligned}D'^j_{switch_1} &= n\{\frac{1}{r}M(\omega_j^{lj1} + \omega_j^{lj2}) + \frac{1}{r}\lceil\omega_j\rceil M(N - l_{j1} - l_{j2} - 1) \\ &+ (\frac{1}{r} - \frac{1}{\rho_j})(M - m + 1) + \frac{1}{r}(N-1)(m-1) \\ &+ (\frac{1}{r} - \frac{1}{\rho_j})(\lceil\omega_j\rceil - 1)M\} + \frac{\sigma_j}{\rho_j}.\end{aligned}$$

Note that the frames of the flow are not duplicated at the end systems in HERSA, the delays at the source/destination end systems are shorter than those in NDRR. According to Corollaries 2.1 and 2.2, we obtain

$$\begin{aligned}\Delta E_r &> D'^j_{switch_1} - D^j_{switch_1} + min(\Delta_1, SkewMax) \\ &= n\{\frac{1}{r}M\omega_j^{lj2} + \frac{1}{r}M\lceil\omega_j\rceil(N - l_{j2} - n_1) \\ &+ \frac{1}{r}(m-1)(N-n_1)\} + min(\Delta_1, SkewMax) > 0.\end{aligned}$$

This equation proves that the end-to-end delays of non-safety-critical flows in HERSA are also shorter than those in NDRR.

In summary, the HERSA scheme is more efficient than the traditional NDRR scheme that schedules full redundant flows.

TABLE I. THE NUMBERS OF HETEROGENEOUS FLOWS (VLS) UNDER DIFFERENT BAG VALUES.

BAG (ms)	No. of Avionics	No. of Multimedia	No. of Data
2	75	10	5
4	120	30	10
8	260	70	15
16	80	200	140
32	50	240	170
64	10	30	180
128	5	20	80

V. PERFORMANCE EVALUATION

In this section, we present the experimental results from the implementations of multiple schedulers, including FIFO (baseline), DRR, NDRR and HERSA, to schedule heterogeneous flows. Current AFDX protocol only supports the FIFO scheduling scheme.

The experiments are configured according to the specifications of AFDX networks [2]. The experiments evaluate the end-to-end delays of different scheduling schemes. Figure 2 shows the network configuration for the experiments, which include 10 source end systems, 1 destination end system and 4 switches. In order to make meaningful comparisons, we also implemented FIFO, DRR, NDRR and HERSA. We use 16-bit values as virtual link IDs in the entire AFDX network for the Ethernet frames.

A. Experiment Setup

In the experiments, we use three types of heterogeneous flows, i.e., avionics, multimedia (video & audio) and best-effort data, to evaluate the end-to-end delays under our scheme. Tables I and II demonstrate the Bandwidth Allocation Gap (BAG) and frame sizes of the three flows. These settings come from a synthetic scenario to simulate a real AFDX network. All links are set to 100Mbps. All BAGs are smaller than 128ms and most frames are shorter than 1000 bytes according to the AFDX specification [2] and the experiences from real-world environments [20]. In general, avionics VLs demonstrate the heaviest workloads among the three flows.

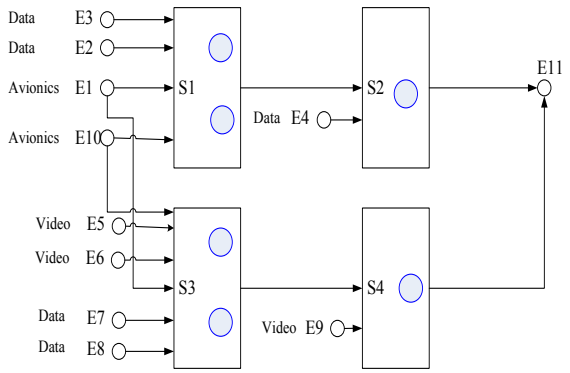


Fig. 2. An AFDX prototype configuration.

We evaluate end-to-end delays of heterogeneous flows that are periodically generated. In the context of HERSA design, the "periodicity" is interpreted as a period for transmitting a periodic message. The experiments define the periodicity to be 40ms. In this way, periodic flows with BAGs smaller than 40ms can be replayed every 40ms. We replay other flows in the time

TABLE II. THE NUMBERS OF HETEROGENEOUS FLOWS (VLS) UNDER FRAME LENGTH VALUES.

Frame Length (bytes)	No. of Avionics	No. of Multimedia	No. of Data
0-100	275	5	10
100-200	145	10	20
200-400	130	25	30
400-600	20	70	140
600-800	15	80	150
800-1000	10	235	130
>1000	5	175	120

interval of their own BAG values. The experiments leverage different amounts of the transmitted VLs to show variable loads. The BAG values are from 2 to 128 with exponential growth of 2.

B. End-to-end Delays

Figure 3 shows the experimental results when executing FIFO, DRR, NDRR and HERSA for scheduling avionics, multimedia (video & audio) and best-effort data. Except HERSA, other schedulers transmit redundant heterogeneous flows. HERSA only redundantly transmits avionics flows due to their high reliability requirements. The end-to-end delays are evaluated with the increased numbers of VLs. These scheduling schemes cause different end-to-end delays due to their different scheduling strategies for heterogeneous flows. HERSA obtains the best performance since it can significantly alleviate the network congestion and decrease re-transmission probability.

For different scheduling schemes, we observe that the delay of FIFO is the longest while that of HERSA is the shortest. FIFO produces on average 4.72 times longer delay than HERSA. FIFO equally treats heterogeneous flows that have different deadlines of hard real-time transmission, which leads to the longest delays among the schedulers. DRR scheduling algorithm obtains smaller latency since it uses different quanta to differentiate heterogeneous flows. Compared with DRR, NDRR leverages a nested set of multiple frames inside each DRR frame, thus resulting in a significantly lower latency bound, while maintaining the $O(1)$ complexity and fairness as DRR. Unlike them, HERSA obtains the smallest latency. The reason is that it has smaller service unit, which can be adjusted according to the input traffic lengths.

For scheduling heterogeneous flows, the evaluation of end-to-end delays involves in avionics, multimedia (video & audio) and best-effort data VLs. Their performance depends upon frame sizes and the scheduling scheme. First, as shown in Figure 3(a), we observe that the average and maximum latencies in avionics flows are respectively 6.1ms and 11.7ms in HERSA, 21.2ms and 26.3ms in NDRR, 28.3ms and 41.8ms in DRR and 38.5ms and 55.2ms in FIFO. The HERSA scheduler can guarantee the hard real-time and reliability requirements for avionics flows.

Moreover, as shown in Figure 3(b), the average and maximum latencies of multimedia flows are respectively 7.5ms and 12.8ms in HERSA, 27.8ms and 38.7ms in NDRR, 36.2ms and 55.6ms in DRR and 47.6ms and 74.1ms in FIFO. HERSA delivers substantial performance improvements due to its partially redundant transmission policy and shorter service opportunity gap. The amounts of the transmitted multimedia flows in HERSA decrease, and in the meantime these heterogeneous flows do not need to wait for their redundant copies in the destination end systems, thus significantly reducing the end-to-end delays.

In addition, as shown in Figure 3(c), the average and maximum latencies of data flows are 11.5ms and 20.3ms in HERSA, 33.7ms and 50.9ms in NDRR, 49.8ms and 68.2ms in DRR and 60.3ms and 91.7ms in FIFO. Compared with FIFO, DRR, and NDRR, the end-to-end latency of the data flows in HERSA is also much shorter. The reason is that HERSA alleviates the number of data to be re-transmitted, and further mitigates network congestion.

Figure 4 shows the end-to-end delays of transmitting redundant flows for all schedulers. Compared with partially redundant transmission, we observe that the delays increase. Specifically, for avionics flows, the average and maximum delays are 42.2ms and 60.5ms in FIFO, 30.6ms and 51.8ms in DRR, 25.3ms and 32.6ms in NDRR, and 6.8ms and 15.6ms in HERSA. The reason is that redundant transmission for avionics flows causes heavier network load, thus aggravating network congestion.

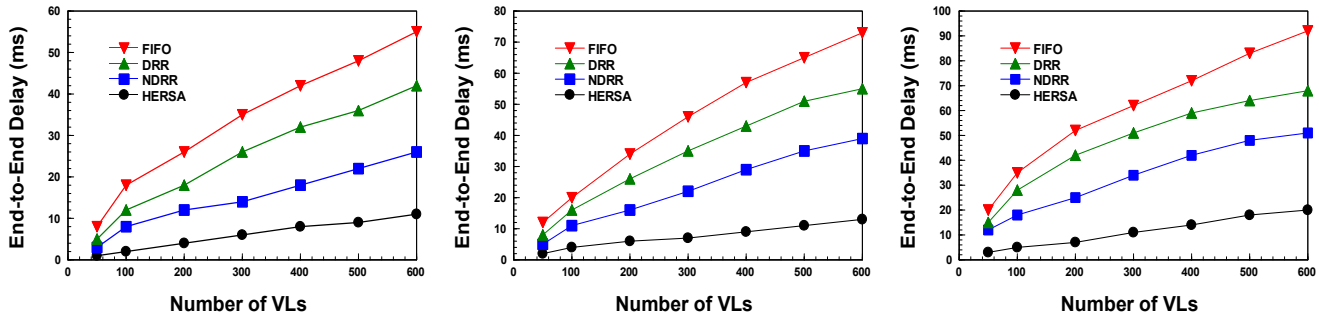
Moreover, as shown in Figure 4(b), the average and maximum latencies of multimedia flows are respectively 8.1ms and 18.6ms in HERSA, 30.2ms and 47.4ms in NDRR, 39.5ms and 62.9ms in DRR and 53.2ms and 79.7ms in FIFO. HERSA delivers substantial performance improvements due to its shorter service opportunity gap.

In addition, as shown in Figure 4(c), the average and maximum latencies of data flows are 12.8ms and 28.2ms in HERSA, 36.6ms and 58.2ms in NDRR, 52.7ms and 81.6ms in DRR and 71.3ms and 101.2ms in FIFO. Compared with FIFO, DRR, and NDRR, the end-to-end latency of data flows in HERSA is also much shorter. The reason is that HERSA reduces the minimum service size, thus decreasing the waiting delay for all flows.

For different scheduling policies under the same scenario, we observe that the delay of HERSA is the shortest while that of FIFO is the longest. FIFO fails to serve for different flows with different quanta, thus leading to the longest delays among the scheduler. The DRR scheduler allocates large quanta to high-rate flows, which leads to low-rate flows waiting long for one service opportunity. Compared with DRR, NDRR leverages a smaller inner round, in which a flow can receive one service opportunity, hence improving the low-rate flows' performance. However, the minimum serving size in one inner round is still too large for small size packets. HERSA takes advantages of input traffic and reduces the minimum serving size for all the flows, especially for flows containing small packets. Since a large fraction of packets in avionics flows can be short, HERSA effectively reduces the queuing time for all flows.

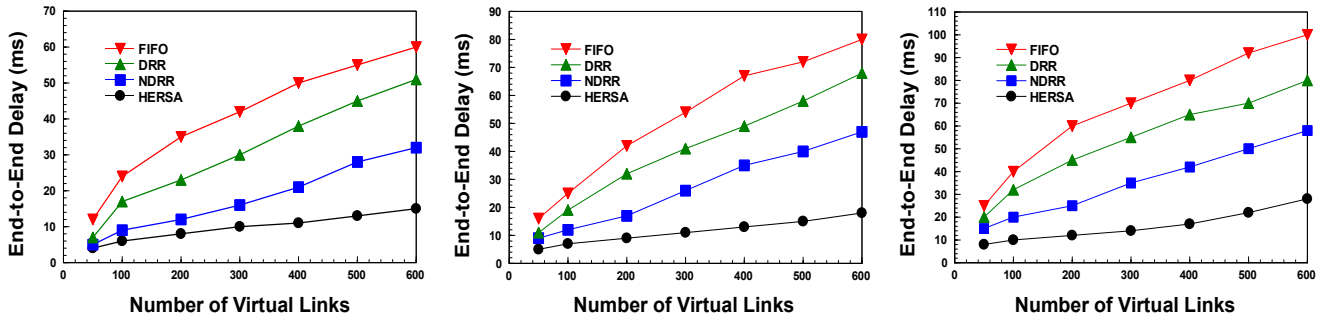
C. Retransmission Rate

The number of retransmitted data is an important metric to evaluate the end-to-end transmission performance. In order to provide efficient reliability, data are generally retransmitted when packet loss or timeout occurs. The fault model refers to the international standard IEC61508 [21], which can identify various levels of integrity or system reliability. In each level,



(a) Avionics flows. (b) Multimedia flows. (c) Data flows.

Fig. 3. End-to-end delays of heterogeneous flows under HERSA, DRR, NDRR, and FIFO schedulers.



(a) Avionics flows. (b) Multimedia flows. (c) Data flows.

Fig. 4. End-to-end delays of fully redundant flows under multiple schedulers.

this standard describes the probability of system-level failure in time unit. Moreover, in the context of AFDX design, the probability of failure p_i of each flow i can be computed from the sizes of the flows (i.e., $SIZE_i$) and bit error rates.

Figure 5 shows the retransmission rates for scheduling avionics flows and the data loss rate for scheduling multimedia and best-effort data flows. The retransmission and data loss rates can be interpreted as the same semantics to evaluate the transmission performance. Avionic flows use retransmission and other flows use directly data loss in the end-to-end transmission.

We observe that compared with FIFO, DRR, and NDRR schedulers, HERSA can significantly reduce retransmission rate, i.e., 3.6% for avionics flows, and decrease the data loss rate, i.e., 5.2% for multimedia and data flows. The reason is that it decreases the service unit using the input traffic characteristics and allows each flow to receive service as soon as possible, which shortens the end-to-end delays substantially. HERSA hence decreases the number of time-out packets and the possibility of overflow in the queue buffer. Moreover, HERSA also reduces the number of packets waiting in the queues of intermediate switches, hence mitigating packet loss.

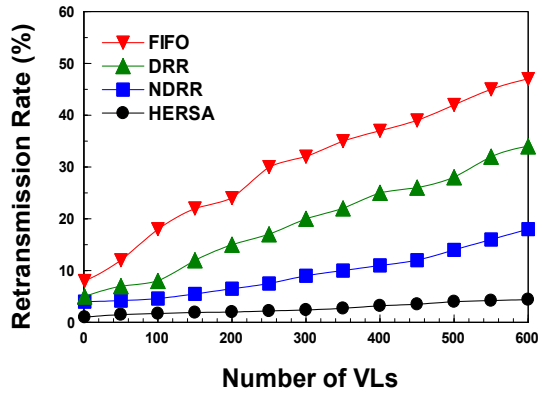
VI. RELATED WORK

Mixed-criticality systems in avionics application bring new challenges to the AFDX network communications.

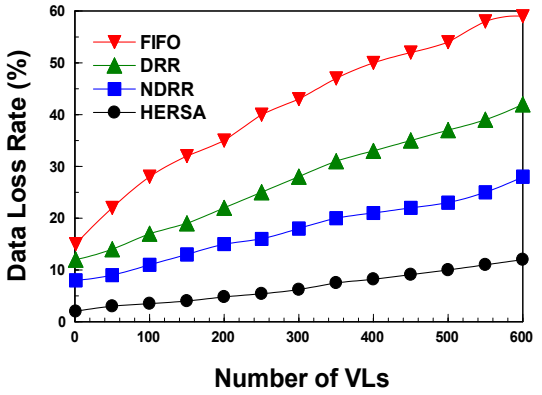
Scharbarg et al. [20] analyze the real-time of AFDX networks upon switches that run FIFO queuing. The result shows that FIFO scheduler couldn't support heterogeneous flows successfully since multiple flows share queues identically. Bauer et al. [9] get the end-to-end delay bound of static priority scheduler by applying trajectory approach. However, both FIFO and static priority scheduler fail to support heterogeneous flows well. In order to support multiple flows transmission, the authors in [22] try to combine static priority and weighted round-robin to improve schedulability. Likewise, Li et al. [23] propose the scheduling policy of fixed priority combined with Round Robin (FP-RR) to improve the transmission delay. Hua et al. [6] select modified deficit round robin scheduler to better serve heterogeneous flows. In order to further improve the transmission efficacy, they simplify the de-duplication operation at the end destination system[4].

Rao et al. [24] apply TDMA crossbar real-time switch architecture into the AFDX network designment and derive the end-to-end delay bounds using network calculus. Henri Bauer et al.[25] focus on AFDX switch outports backlog evaluation using trajectory approach. Their work shows that smaller buffer size can be used in each AFDX switch outport. Trajectory approach [26] is also used to measure the end-to-end delay besides three common methods, i.e., network calculus, model checking and queuing networks simulation, in the AFDX systems.

The work reviewed above focus on how schedulers help improve the end-to-end delays of heterogeneous flow. Unlike their work, our work observes the traffic transmission modes in the AFDX network, and demonstrates that the differentiated transmission mode contributes to reduce the end-to-end delay while still guaranteeing the reliability. Furthermore, our proposed algorithm efficiently schedules heterogeneous flows, while incurring low complexity.



(a) Retransmission rates.



(b) Data loss rate.

Fig. 5. Transmission performance of heterogeneous flows.

VII. Conclusion

In order to improve the latency in the avionics network for the mixed criticality systems, we propose an efficient and novel scheme, HERSA, to address the problems of data transmission and inefficiency scheduling. HERSA uses a differentiated transmission mode and an efficient scheduler to schedule heterogeneous flows. We obtain the tight latency bound and fairness of HERSA by theoretically analyzing the working principle of switches. With the latency bound of HERSA scheduler, we study the end-to-end delays for mixed criticality flows using the Latency-Rate servers. Moreover, we implement HERSA on a real test-bed. Experimental results demonstrate the efficiency and efficacy of our proposed scheme and show that the performance improvements come from better scheduling and a light-load network configuration.

References

[1] H. Frazier, "The 802.3z Gigabit Ethernet Standard," *IEEE Network*, vol. 12, no. 3, pp. 6–7, May/June 1998.

[2] ARINC664, "Aircraft Data Network, Part 7 Avionics Full Duplex Switched Ethernet (AFDX) Network," *ARINC 05-005/ADN-39*, 2005.

[3] J. Yao, G. Zhu, X. Liu, and A. Jou, "Optimal bandwidth allocation for non-critical traffics in afdx network," *Proc. IEEE Conference on Industrial Electronics and Applications*, pp. 1727–1733, 2012.

[4] Y. Hua and X. Liu, "Scheduling Heterogeneous Flows with Delay-Aware Deduplication for Avionics Applications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 9, pp. 1790–1802, 2012.

[5] H. Charara, J.-L. Scharbarg, J. Ermont, and C. Fraboul, "Methods for bounding end-to-end delays on an AFDX network," *Proc. Euromicro Conference on Real-Time Systems*, 2006.

[6] Y. Hua and X. Liu, "Scheduling design and analysis for end-to-end heterogeneous flows in an avionics network," *Proc. INFOCOM, 2011.*

[7] M. Boyer and C. Fraboul, "Tightening end to end delay upper bound for afdx network calculus with rate latency fifo servers using network calculus," *Proc. WFCS*, 2008.

[8] F. Ridouard, J.-L. Scharbarg, and C. Fraboul, "Probabilistic upper bounds for heterogeneous flows using a static priority queueing on an afdx network," *Proc. IEEE International Conference on Emerging Technologies and Factory Automation*, pp. 1220–1227, 2008.

[9] 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 systems*, vol. 48, no. 1, pp. 101–133, 2012.

[10] S. J. Golestani, "A self-clocked fair queueing scheme for broadband applications," *Proc. INFOCOM*, 1994.

[11] S. S. Kanhere and H. Sethu, "Fair, efficient and low-latency packet scheduling using nested deficit round robin," *Proc. HPSR, 2001.*

[12] M. Katevenis, S. Sidiropoulos, and C. Courcoubetis, "Weighted round-robin cell multiplexing in a general-purpose atm switch chip," *Selected Areas in Communications*, vol. 9, no. 8, pp. 1265–1279, 1991.

[13] M. Shreedhar and G. Varghese, "Efficient fair queuing using deficit round-robin," *IEEE Transactions on Networking*, vol. 4, no. 3, pp. 375–385, 1996.

[14] H. M. Chaskar and U. Madhow, "Fair scheduling with tunable latency: a round-robin approach," *IEEE/ACM Transactions on Networking*, vol. 11, no. 4, pp. 592–601, 2003.

[15] L. Lenzini, E. Mingozzi, and G. Stea, "Tradeoffs between low complexity, low latency, and fairness with deficit round-robin schedulers," *IEEE/ACM Transactions on Networking*, vol. 12, no. 4, pp. 681–693, 2004.

[16] S. S. Kanhere, H. Sethu, and A. B. Parekh, "Fair and efficient packet scheduling using elastic round robin," *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, no. 3, pp. 324–336, 2002.

[17] S. S. Kanhere and H. Sethu, "Low-latency guaranteed-rate scheduling using elastic round robin," *Computer Communications*, vol. 25, no. 14, pp. 1315–1322, 2002.

[18] Y.-C. Jenq, "Performance analysis of a packet switch based on single-buffered banyan network," *IEEE Journal on Selected Areas in Communications*, vol. 1, no. 6, pp. 1014–1021, 1983.

[19] D. Stiliadis and A. Varma, "Latency-rate servers: a general model for analysis of traffic scheduling algorithms," *IEEE/ACM Transactions on Networking*, vol. 6, no. 5, pp. 611–624, 1998.

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

[21] Function safety of electrical/electronic/ programmable electronic safety-related system, *IEC 61508, http://www.iec.ch.*

[22] M. Boyer, N. Navet, M. Fumey, J. Migge, L. Havet, and T. Avionics, "Combining static priority and weighted round-robin like packet scheduling in AFDX for incremental certification and mixed-criticality support," *Proc. EUCASS*, 2013.

[23] J. Li, H. Guan, J. Yao, G. Zhu, and X. Liu, "Performance Enhancement and Optimized Analysis of the Worst Case End-to-End Delay for AFDX Networks," *Proc. GreenCom*, 2012.

[24] L. Rao, Q. Wang, X. Liu, and Y. Wang, "Analysis of TDMA crossbar real-time switch design for AFDX networks," *Proc. INFOCOM*, 2012.

[25] H. Bauer, J. Scharbarg, and C. Fraboul, "Worst-case backlog evaluation of avionics switched ethernet networks with the trajectory approach," *Proc. Euromicro Conference on Real-Time Systems (ECRTS)*, pp. 78–87, 2012.

[26] H. Bauer, J. Scharbarg, and C. Fraboul, "Improving the worst-case delay analysis of an afdx network using an optimized trajectory approach," *IEEE Transactions on Industrial Informatics*, vol. 6, no. 4, pp. 521–533, 2010.