# A Feasible Configuration of AFDX Networks for Real-Time Flows in Avionics Systems

Dongha An, Hyun Wook Jeon, Kyong Hoon Kim, and Ki-Il Kim
Department of Informatics
Gyeongsang National University
Jinju-daero 501, Jinju, 660-701, South Korea
Email: {dhan,krsoftno1,khkim,kikim}@gnu.ac.kr

***Abstract*—AFDX (Avionics Full Duplex Switched Ethernet) Networks have been proposed to meet unique ADN (Aircraft Data Networks) characteristics and then standardized as a Part 7 in ARNIC 664. As for this new communication technology, some research works have been conducted to address design issues such as optimizing virtual links as well as analytic modeling including response time. Despite of their research efforts, configuration problem for both MTU (Maximum Transmission Unit) and BAG (Bandwidth Allocation Gap) over virtual links in AFDX networks remains unsolved yet. In this paper, we propose how to set MTU and BAG value on each virtual link according to both application requirements and AFDX switch constraints. We define a new problem of feasible configurations of virtual links in an AFDX switch and propose an algorithm to derive feasible BAG and MTU pairs based on the branch-and-bound technique. Throughout simulations, we evaluate the proposed algorithm and analyze the effect of parameters in AFDX networks.**

## I. INTRODUCTION

As new aircraft's demanding requirements to high available bandwidth, minimum wiring to reduce the weight and low development cost have emerged, the current three main ADNs (Aircraft Data Networks), ARNIC 429, MIL-STD-1553 and ARNIC 629 are regarded as not appropriate communication technologies to meet these demands completely. This fact implies that not only reliable and deterministic property of ADN but also implementation cost should be concerned in next generation aircraft. Consequently, from development of data networks on the Airbus 380 aircraft, a new technology, called AFDX (Avionics Full Duplex Switched Ethernet), has been implemented and then standardized for new ADN [1], [2], [3].

The AFDX was extended from original Ethernet to ensure deterministic behavior and high reliability in order to comply with the stringent requirements of ADNs. To ensure them, new functions are implemented in two ways. One is traffic control by guaranteeing the bandwidth of each application, and the other is dual redundant channel for reliability. While the former targets to limit the jitter and transmit latency, the latter transmits the same data stream over disjoint networks. To achieve this goal, virtual links have been employed between source and destination. With these virtual links, deterministic behaviors are guaranteed and all controls are ensured through them. So, determining virtual link properties and configuring network environments become network designer's great task.

System configuration parameters of virtual links include traffic scheduling, maximum jitter, and bandwidth constraints [1], [2], [3]. Among many system parameters, two are important with regard to the guarantee of real-time requirements: *BAG* (Bandwidth Allocation Gap) and *MTU* (Maximum Transfer Unit). BAG is a timeslot confining the virtual link's bandwidth by defining the minimum gap time between two consecutive frames. The range of the BAG value is between 1 and 128 msec in a form of power of 2. MTU is defined as the maximum size of message to be transmitted in each frame.

Much recent work has focused on the system analysis of AFDX networks [7], [8], [9], [10], [12]. The AFDX network analysis is done by queuing networks, network calculus, or model checking. Throughout the analysis, the impact of parameters has been analyzed, including end-to-end delays, worst-case latencies, and so on. However, only a few studies have been done on the problem of AFDX configuration such as BAG and MTU. In [4], the authors proposed how to set the transmission parameters of virtual links so as to minimize the reserved bandwidth while transmitting the data within their maximum delivery times. They first derive optimized parameters of each virtual link for a given set of messages. Then, they solve the optimization problem of multiple virtual links in order to minimize bandwidth. The weakness of this approach is that the optimized parameters found in a single virtual link cannot be feasible when they are used in finding feasible configurations of multiple virtual links in an AFDX network switch.

In this paper, we focus on finding feasible BAG and MTU parameters of virtual links in an AFDX switch for a given virtual links of messages. We define a new problem of feasible configuration of an AFDX switch, and then solve the problem using the branch-and-bound technique. The main contributions of this paper include (i) defining a problem of feasible configuration, (ii) providing an algorithm to solve the problem, and (iii) analyzing the algorithm through the simulations.

The rest of this paper is organized as follows. First, we describe the related work briefly in Section II. And then, the system model and the problem definition are provided in Section III. In Section IV, we explain the proposed algorithm. Performance evaluations are shown in Section V. Finally, conclusion and further work are followed in Section VI.

## II. RELATED WORK

In this section, we briefly introduce related work on AFDX networks. In this research area, existing technologies mainly fall into two main categories. One is for design issue and the other is for analysis modeling.

First, the authors in [4] have focused on the design of virtual links in AFDX networks. In their works, the problem domain is ranged from how to set the transmission parameters of virtual links to how to route virtual links in the AFDX interconnect. For this goal, several closed-form results and efficient numerical algorithms as well as exact integer-linear programming formulation of the routing problem are newly presented. Through above method, optimal bandwidth management is achieved, such as, minimizing reserved bandwidth and the bandwidth consumption. In another research work, modeling method for AFDX frame management was introduced to ascertain the reliability properties of design [5]. They modeled the system as a network of timed automata to indicate weakness of current AFDX frame management against faults. Moreover, they present the solution by including a priority queue at receivers. In addition to mentioned works, one of outstanding features, reliability through redundancy transmission on AFDX was analyzed by formal method in [6].

While the design issue targets to build AFDX networks, the other works have been proposed to analyze the system metric such as response time. The representative work for this goal has been proposed in [7]. The authors introduce three methods, network calculus, queuing networks simulation and model checking to evaluate bounding end-to-end delays on AFDX networks. As the previous work, they also showed that Trajectory approach which analyzes the worst-case delays throughout message flows outperforms the Network calculus method under industrial configuration [11] and reached reliable conclusion that combination of two methods could lead to an improvement of the existing analysis in [8]. However, since the previous model did not include contention in the end or switches, different analysis was given to obtain worst-case latencies and output jitter for the network messages in [9] by defining a real-time model for a communications network based on AFDX. In addition to analysis model, simulation system based on popular NS-2 was designed and implemented to evaluate the performance and analyze impact of several system parameters such as scheduling algorithm in [10].

## III. SYSTEM MODEL AND PROBLEM DEFINITION

### A. System Model

Avionics network systems consist of many components, such as sensors, LRUs (Line Replacement Units), computing units, and so on. These components communicate each other throughout AFDX switches. An AFDX message is uniquely defined by UDP source and destination ports, as shown in Figure 1. Sine we focus on real-time AFDX messages, a message flow $f_i$ is defined by ($l_i$, $p_i$), where $l_i$ is the payload of the message in bytes and $p_i$ is Message Transmit Cycle (MTC) of the message in msec. That is, a message of $l_i$ bytes is generated every $p_i$ time units and is delivered to the destination application.

A *Virtual Link (VL)* is a logical communication unit in AFDX networks. Figure 1 shows an example of AFDX networks with three virtual links among LRUs. These virtual links sharing physical links are scheduled in AFDX network switches. Furthermore, multiple applications transmit real-time messages throughout a common virtual link if their source and destination units are the same. In the example of Figure 1, two application messages are shared in the virtual link $VL_3$.

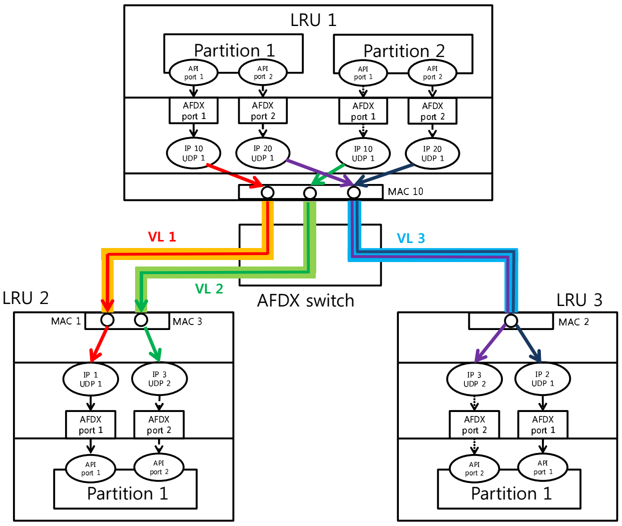


Fig. 1. An example of virtual links in an AFDX switch

A virtual link requires two important parameters other than source and destination information. The first is Bandwidth Allocation Gap (BAG) to specify a periodic frame. In AFDX switches, a BAG is defined by a value of $2^k$ msec, where $k = 0, 1, \ldots, 7$. As all BAGs are $2^k$ msec, virtual links are multiplexed in AFDX switches. The second parameter is Maximum Transfer Unit (MTU) of the message in bytes at each frame. Payloads of applications in a virtual link are transmitted within maximum MTU bytes in a single frame. If the size of a payload is greater than the MTU, it is fragmented into multiple frames. Therefore, a virtual link $VL_i$ is defined by ($BAG_i$, $MTU_i$, $F_i$) as follows.

- $BAG_i$: bandwidth allocation gap or period of $VL_i$ in a value of $2^k$ msec where $k = 0, 1, \ldots, 7$.
- $MTU_i$: maximum transfer unit or message size of $VL_i$ in bytes.
- $F_i$: a set of message flows in $VL_i$, where the $j$-th message flow is denoted as $f_{i,j} = (l_{i,j}, p_{i,j})$.

As avionics systems are hard real-time systems, it is an important issue to guarantee the schedulability both in computing units and in network flows. The virtual link scheduler in an AFDX switch plays a role in scheduling multiple virtual links. For example, the scheduling algorithm in [2] is Round Robin (RR). In this paper, we will define a new problem of finding a feasible configuration of BAG and MTU pairs of given virtual links in an AFDX switch in order to meet all real-time requirements of messages.

### B. Problem Definition

For a given virtual link $VL_i$, MTU and BAG are configured so as to meet all the real-time requirements of message flows in

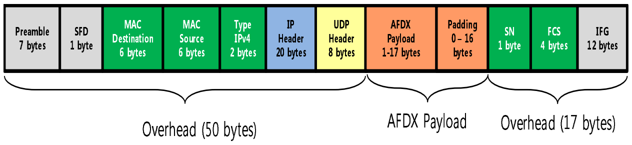


Fig. 2. The AFDX frame structure and its overhead

the link. If the payload of a message is greater than the MTU size, it is transmitted in multiple fragmented packets. Since all BAGs of VLs are harmonic, the schedulability analysis is easily derived by utilization analysis. Thus, Eqn. (1) tells the message constraint of $VL_i$ with $n_i$ messages to guarantee the real-time requirement of all message flows in the link [2].

$$\sum_{j=1}^{n_i} \frac{\lceil l_{i,j}/MTU_i \rceil}{p_{i,j}} \leq \frac{1}{BAG_i} \tag{1}$$

Let us assume that the system has $N$ VLs on an AFDX switch with $B$ bandwidth in bps. Each $VL_i$ is configured with ($MTU_i$, $BAG_i$), so that $MTU_i$ bytes are transmitted every $BAG_i$ msec. In addition, each VL message requires the overhead of 67 bytes as shown in Figure 2. Since the total bandwidth of VLs should not exceed the network bandwidth, the following bandwidth constraint should be met.

$$8 \sum_{i=1}^{n} \frac{MTU_i + 67}{BAG_i} \times 10^3 \leq B \tag{2}$$

The last constraint of virtual link scheduling is about jitter. The maximum allowed jitter on each virtual link in the ARINC 664 specification requires 500 $\mu$sec [2]. Thus, the following equation tells the jitter constraint, where 40 $\mu$sec is the typical technological jitter in hardware level to transmit an Ethernet frame.

$$40 + \frac{8 \sum_{i=1}^{n} (67 + MTU_i)}{B} \leq 500 \tag{3}$$

Now we define a problem of finding a feasible configuration of BAG and MTU pairs of virtual links of an AFDX switch. Three constraints of Eqn. (1), Eqn. (2), and Eqn. (3) should be met in order to satisfy all real-time requirements of messages in virtual links, which derives a new problem as follows.

*Definition 3.1:* For a given set of virtual links $V = \{VL_i \mid i = 1, \ldots, N\}$, the problem of **AFDX-CONF** is to determine ($BAG_i$, $MTU_i$) of each $VL_i$ so as to satisfy three constraints of Eqn. (1), Eqn. (2), and Eqn. (3), where $BAG_i \in \{1, 2, 4, 8, 16, 32, 64, 128\}$ and $MTU_i \in \{1, 2, \ldots, 1471\}$.

## IV. THE PROPOSED ALGORITHM

We solve the problem **AFDX-CONF** in two steps. The first step is to find the list of ($BAG_i$, $MTU_i$) which guarantees the schedulability of message flows in $VL_i$. Each ($BAG_i$, $MTU_i$) should be selected such that it satisfies the constraint of Eqn. (1). Then, we find the feasible solutions of a given virtual links with consideration of two constraints of Eqn. (2) and Eqn. (3).

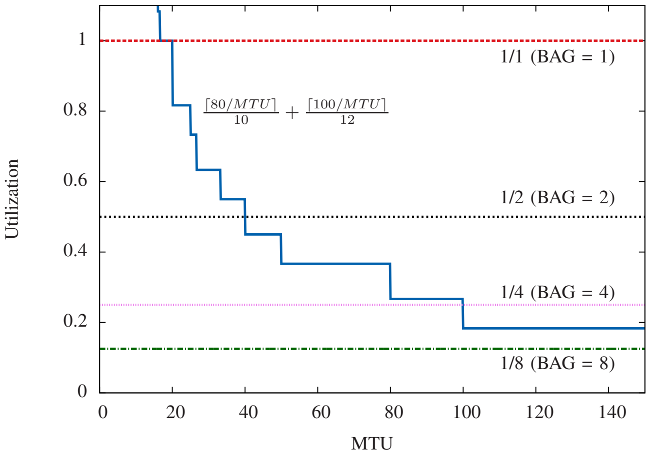


Fig. 3. An example of feasible BAG and MTU of a virtual link

### A. Schedulable BAG and MTU Pairs of a VL

Let us consider a virtual link $VL_1$ with two message flows of $f_{1,1}(80, 10)$, $f_{1,2}(100, 12)$ as an example. The values of BAG and MTU of $VL_1$ are set to satisfy Eqn. (1) in order to meet the real-time requirement of two messages. The left side of Eqn. (1) is shown in Figure 3 as a step function, while $1/BAG$ is also drawn in the figure for different BAG values.

For a given $BAG_i$, there exist many MTUs which satisfy Eqn. (1). For example, when $BAG_1$ = 1, all MTUs can be used if MTU $\geq$ 17, as shown in Figure 3. Since a longer MTU size requires more bandwidth and jitter, the smallest value should be selected. Thus, $MTU_1$ of the example $VL_1$ is 17 bytes when $BAG_1$ is 1 msec. Similarly, MTUs of $VL_1$ for BAGs with 2 msec and 4 msec are given by 40 bytes and 100 bytes in each, as shown in Figure 3.

When the MTU size is greater than the maximum payload size of messages, the required utilization is not changed. For example, the lower bound of the utilization of $VL_1$ is given by about 0.1834 at MTU = 100. This implies that there is no MTU which guarantees the schedulability of two messages if BAG is greater than or equal to 8 msec. Therefore, the feasible solutions, ($BAG_1$, $MTU_1$), of $VL_1$ are given by (1, 17), (2, 40), and (4, 100).

The pseudo-algorithm of Figure 4 describes how to obtain the set of feasible BAG and MTU pairs of a given virtual link $VL_i$. The first part of the algorithm gathers all step integers at which the utilization function begins a new piecewise constant due to the ceiling function. We denote the set of such step integers as $N_{step}$. For each message $f_{i,j}$, such step points are derived and added into $N_{step}$ (lines 1-8).

Then, for each $2^k$ value, we find the minimum MTU which satisfies Eqn. (1). (lines 9-13). We denote $s_{i,k}$ as the feasible BAG and MTU pair in case of $BAG_i = 2^k$ for a virtual link $VL_i$. For a given $n_i$ flows, the time complexity of the algorithm in Figure 4 is $O(n_i \cdot |N_{step}|)$ since we have to find and check the feasibility at each step point of messages.

```
Algorithm Find_Feasible_BAG_MUT ($VL_i$)
$N_{step} \leftarrow \varnothing$
for each message $f_{i,j}$ in $VL_i$ do
    $frag \leftarrow \lceil l_{i,j}/(\lceil l_{i,j}/p_{i,j}\rceil)\rceil$
    while $frag \geq 1$ do
        $m \leftarrow \lceil l_{i,j}/frag\rceil$
        $N_{step} \leftarrow N_{step} \cup \{m\}$
        $frag \leftarrow frag - 1$
    endwhile
endfor
for $k$ from $0$ to $7$ do
    $m_k \leftarrow$ the least $m \in N_{step}$ s.t. $\sum_{j=1}^{n_i} \frac{\lceil l_{i,j}/m\rceil}{p_{i,j}} \leq \frac{1}{2^k}$
    if $m_k \neq NULL$ then
        $s_{i,k} \leftarrow (2^k, m_k)$
endfor
```

Fig. 4. Algorithm of feasible BAG and MTU pairs of a VL

### *B. Feasible BAG and MTU Pairs of VLs*

The problem of finding feasible BAG and MTU pairs of a given set of virtual links is not trivial. For example, let us consider the example of two virtual links of Table I where the network speed ($B$) is given by 1Mbps. For each virtual link, the feasible BAG and MTU pairs are derived by the algorithm of Figure 4, as shown in the last column of Table I. Now, a new problem arises about selecting appropriate BAG and MTU pairs of two virtual links so as to meet both constraints of Eqn. (2) and Eqn. (3).

There are some tradeoffs among feasible $s_{i,k}$ of a virtual link $VL_i$. Solutions with smaller BAG provide less jitter due to smaller MTU size, while they require more bandwidth due to overhead of fragmentation. For example, if we select (1,5) and (1,6) as (BAG, MTU) of two VLs of Table I, it does not meet the bandwidth constraint of Eqn. (2). On the contrary, if (2,9) and (2,12) are selected as (BAG, MTU) of two VLs, this configuration does not meet the jitter constraint of Eqn. (3). The selection of (1,5) and (2,12) of $VL_1$ and $VL_2$ satisfies both constraints so that all messages in VLs meet their real-time requirements.

Let us denote $s_{i,k}$ as the feasible BAG and MTU pair of $VL_i$ in case of $BAG_i = 2^k$, which is derived from the algorithm of Figure 4. If there is no feasible MTU for $BAG_i = 2^k$, $s_{i,k} = \varnothing$. Then, the problem to be solved is defined as follows.

*Definition 4.1:* For a given set of virtual links $V = \{VL_i \mid i = 1, \ldots, N\}$, let us assume that a feasible pair of BAG and MTU for $BAG_i = 2^k$ is available as $s_{i,k}$. The problem of **AFBM** is to select $s_{i,k}$ of each $VL_i$ so as to satisfy both constraints of Eqn. (2) and Eqn. (3).

For a given $N$ virtual links, the exhaustive search of the problem **AFBM** takes $O(8^N)$ since each virtual link might have maximum eight solutions. In this paper, we provide a branch-and-bound algorithm to find a feasible solution for a given $N$ virtual links with their feasible BAG and MTU pairs derived by Figure 4. The proposed branch-and-bound algorithm consists of pruning condition and branch-and-bound strategy as follows.

TABLE I. AN EXAMPLE OF VIRTUAL LINKS ($B$ = 1 MBPS)

| | Flows ($f_{i,j}$) | Payload ($l_{i,j}$) | MTC ($p_{i,j}$) | Feasible BAG and MTU pairs ($s_{i,k}$) |
|---|---|---|---|---|
| $VL_1$ | $f_{1,1}$ | 200 | 80 | (1,5), (2,9), (4,17), (8,34), (16,67), (32,200) |
| | $f_{1,2}$ | 250 | 160 | |
| $VL_2$ | $f_{2,1}$ | 250 | 220 | (1,6), (2,12), (4,25), (8,50), (16,100), (32,200) |
| | $f_{2,2}$ | 200 | 40 | |

```
Algorithm Find_Feasible_Configurations ($V$)
  /* $V = \{VL_i | i = 1, \ldots, N\}$ */
1:  for $i$ from $1$ to $N$ do
2:      call Find_Feasible_BAG_MUT ($VL_i$)
3:  $S \leftarrow \varnothing$
4:  result $\leftarrow$ DFS_BandB (0, 0, 1, $S$)
5:  return $S$

  Function DFS_BandB ($B_{curr}$, $J_{curr}$, $i$, $S$)
6:  if $i = N + 1$ then return true
7:  for each $s_{i,k}$ of $VL_i$ do
8:      bandwidth $\leftarrow B_{curr} + (mtu_{i,k} + 67)/bag_{i,k}$
9:      jitter $\leftarrow J_{curr} + 67 + mtu_{i,k}$
10:     if bandwidth $\leq B/8000$ and jitter $\leq 460 \cdot B$ then
11:         result $\leftarrow$ DFS_BandB (bandwidth, jitter, $i + 1$, $S$)
12:         if result = true then
13:             $S \leftarrow S \cup \{s_{i,k}\}$
14:             return true
15:         endif
16:     endif
17: endfor
18: return false
```

Fig. 5. The proposed algorithm

- **Pruning condition:** The pruning condition is two constraints of Eqn. (2) and Eqn. (3). The algorithm examines whether the solutions in the subset satisfy both constraints. Since both bandwidth and jitter values increase with a new branch in the search tree, the algorithm stops the search of the subset which already violates one of two constraints.
- **Branch and bound strategy:** We can use the current values of total bandwidth and jitter as a branch condition. For example, a node with the least bandwidth is selected as a new branch. The algorithm finds a feasible solution when it reaches at any leaf node in the search tree.

The proposed algorithm searches a feasible solution in a leaf node in Depth-First-Search (DFS) manner. The function **DFS_BandB** in Figure 5 is the recursive implementation at level $i$ in the search tree. Two values of $B_{curr}$ and $J_{curr}$ are the total bandwidth and jitter of sub-solutions from $VL_1$ to $VL_{i-1}$. For each $s_{i,k} = (bag_{i,k}, mtu_{i,k})$, two constrains of Eqn. (2) and Eqn. (3) are checked including a new solution of $VL_i$ (lines 8-10). If either of two constraints is not satisfied, it is pruned. Otherwise, the depth-first-search is continued with two updated bound values (line 11).

When the search reaches at a leaf node, the function returns *true* (line 6). The return value of calling **DFS_BandB** is *true*, the final solution $S$ is updated as to include $s_{i,k}$ (line 13) and the function returns *true*. Thus, the problem of **AFDX-CONF** is solved by the algorithm in Figure 5. If the return value of **DFS_BandB** (0, 0, 1, $S$) is *true*, a feasible solution is stored

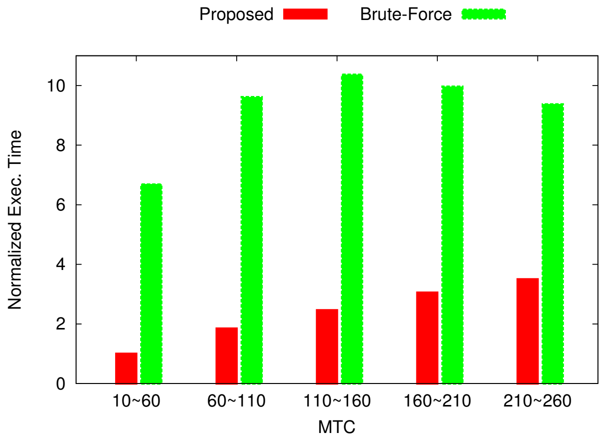


Fig. 6. Algorithm execution time

TABLE II. THE PERCENTILE OF FEASIBLE SOLUTIONS

| MTC | $10 \sim 60$ | $60 \sim 110$ | $110 \sim 160$ | $160 \sim 210$ | $210 \sim 260$ |
|---|---|---|---|---|---|
| Feasible Sets | 3.4% | 23.6% | 40.1% | 54.0% | 62.7% |

in $S$. Otherwise, the empty set is returned, which implies no feasible configuration is found for a give set of virtual links.

## V. PERFORMANCE EVALUATIONS

In this section, we show performance evaluation of the proposed algorithm. First, we evaluate the execution time of the proposed algorithm compared with the brute-force search. In the experiments, we generate five virtual links with two message flows in each virtual link. The payload of a message is randomly generated from 20 to 80 bytes. The MTC or period of a message is randomly selected among five different intervals, as shown in Figure 6. The network bandwidth is set as 6Mbps.

For each case of Figure 6, we generate 5000 random sets of five virtual links and measure the average execution time of the proposed algorithm. In order to compare the execution time, the brute-force search algorithm is also run. In Figure 6, the execution time is normalized based on that of the proposed algorithm in case of the first interval of MTC in the experiments.

As shown in Figure 6, the proposed algorithm runs about two or six times faster than the exhaustive search algorithm. Since the proposed algorithm is based on branch-and-bound technique, it runs faster. Table II shows the percentile of feasible solutions among 5000 random test cases. In case of smaller MTCs, the performance of the proposed algorithm is more than those in bigger MTCs. As shown in Table II, most of random test sets are infeasible in lower MTCs. In this case, the proposed algorithm rejects the given virtual link sets in early search steps due to the pruning condition. However, the exhaustive search algorithm tests all possible cases.

Next, we analyze the payload bound of a message to be schedulable. The MTC is varied from 10 to 100 bytes. We generate 12 messages of the same requirement. The number of virtual links is varied from 1 to 6 in order to analyze the impact of the number of virtual links. Figure 7 shows the payload bound of a message by simulating the proposed algorithm from the payload size 1 to 1471. Figure 7 shows that the message of lower payload size than the bound is guaranteed to be scheduled.

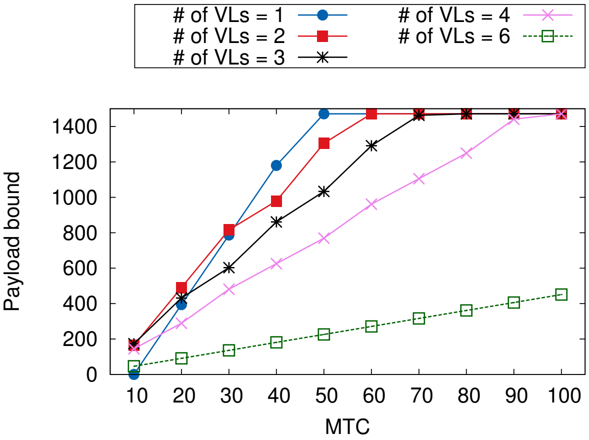


Fig. 7. Payload bounds w.r.t MTC

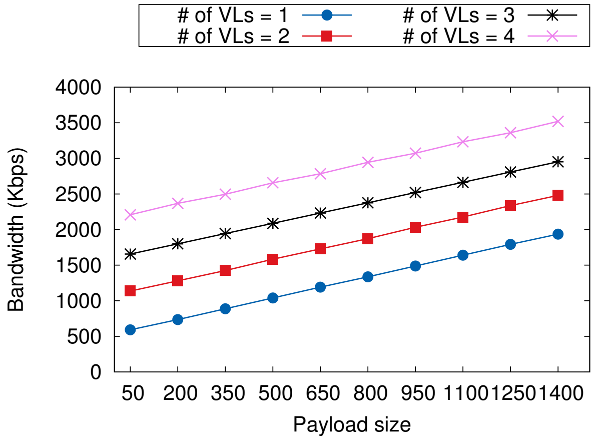
As shown in Figure 7, the schedulability of more virtual links shows generally worse than that of less virtual links for the same number of messages. It is because of jitter and bandwidth overhead of virtual links in AFDX switches. However, in case of lower MTCs, the schedulability of a single virtual link shows poor since it becomes difficult to meet the message constraint of Eqn. (1).

Let us consider the case of MTC = 100 in Figure 7. All messages of the payload size less than or equal to 1471 bytes are schedulable if $N \leq 4$. We measure the bandwidths and jitters of four different number of virtual links, as shown in Figure 8. Figure 8 implies that it is better to use a single virtual link to send 12 messages in order to reduce the total bandwidth and jitter. The remaining bandwidth can be used to transmit other non-real-time network traffic in AFDX switches.

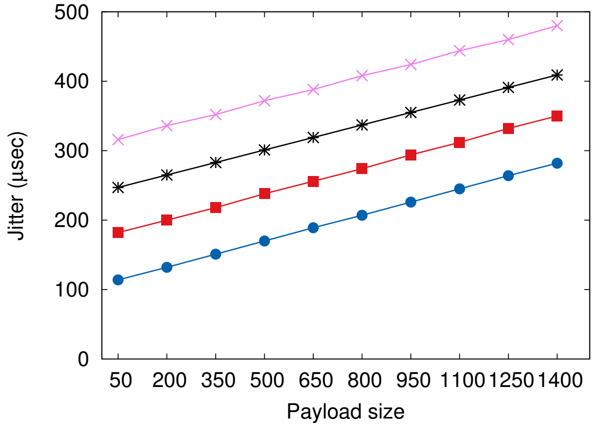
## VI. CONCLUSIONS

In this paper, we defined a new problem of feasible configurations of an AFDX switch for the purpose of meeting the real-time requirements of all messages in avionics. Two important parameters of BAG and MTU of virtual links are derived by solving the problem. The proposed algorithm first derives optimal MTUs of a virtual link for each possible BAG, and then obtains feasible BAG and MTU pairs of multiple virtual links. In the simulation results, the proposed scheme is faster than the exhaustive search algorithm. And, we also analyzed the payload bound and the effect of selection of virtual links.

Since the AFDX network configuration becomes an important issue in avionics systems, we will investigate many problems based on the results of this paper. For example, we will extend the problem into multiple AFDX switches or discuss about the routing issues through the networks.



(a) bandwidth



(b) jitter

Fig. 8. Bandwidth and jitter of feasible sets

## ACKNOWLEDGMENT

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (No. NRF-2012R1A1A1015096) and BK21+ Program.

## REFERENCES

[1] R. L. Alena, J. P. Ossenfort, K. I. Laws, A. Goforth, and F. Figueroa. Communications for integrated modular avionics. In *Proc. of 2007 IEEE Aerospace Conference*, March 2007.

[2] AFDX Tutorial. http://www.techsat.com/fileadmin/media/pdf/infokiosk/TechSAT_TUT-AFDX-EN.pdf.

[3] AFDX/ARNIC 664 Tutorial. http://www.cems.uwe.ac.uk/∼ngunton/afdx_detailed.pdf.

[4] A. Al Sheikh, O. Brun, M. Cheramy, and P.-E. Hladik. Optimal design of virtual links in AFDX networks. *Real-Time Systems*, vol. 49, no. 3, pp. 308-336, May 2013.

[5] I. Saha and S. Roy. A finite state modeling of AFDX frame management using spin. *Lecture Notes in Computer Science*, vol. 4346, pp. 227-243, 2007.

[6] J. Taubrich and R. Hanxleden. Formal specification and analysis of AFDX redundancy management algorithms. *Lecture Notes in Computer Science*, vol. 4680, pp. 436-450, 2007.

[7] H. Charara, J. Scharbarg, J. Ermont, and C. Fraboul. Methods for bounding end-to-end delays on an AFDX network. In *Proc. Of 18th Euromicro Conference on Real-time Systems*, pp. 193-202, July 2006.

[8] H. Bauer, J. Scharbarg, and C. Fraboul. Worst-case end-to-end delay analysis of an avionics AFDX network. In *Proc. Of Design, Automation & Test in Europe Conference & Exhibition*, pp. 1220-1224, March 2010.

[9] J. J. Gutierrez, J. C. Palencia, and M. G. Harbour. Response time analysis in AFDX networks with sub-virtual links and prioritized switches. XV Jornadas de Tiempo Real, Santander, January-February 2012.

[10] S. Dong, Z. Xingxing, D. Lina, and H. Qiong. The design and implementation of the AFDX network simulation system. In *Proc. Of International Conference on Multimedia Technology*, pp. 1-4, October 2010.

[11] M. Tawk, X. Liu, L. Jian, G. Zhu, Y. Savaria, and F. Hu. Optimal scheduling and delay analysis for AFDX end-systems. *SAE Technical Paper*, 2011-01-2751, 2011.

[12] 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, September 2012.