Methods for bounding end-to-end delays on an AFDX network

Hussein Charara, Jean-Luc Scharbarg, Jérôme Ermont, Christian Fraboul
IRIT - ENSEEIHT
2, rue Camichel
31000 Toulouse - France
Hussein.Charara@enseeiht.fr

Abstract

Architectures of avionics networks, such as that of the Airbus A380, currently know important evolutions. This is principally due to the increase in the complexity of the embedded systems, in term of rise in number of integrated functions and their connectivity. The evolution of Switched Ethernet technologies allows their implementation as an avionics architecture (AFDX: Avionics Full Duplex Switched Ethernet).

The problem is then to prove that no frame will be lost by the network (no switch queue will overflow) and to evaluate the end-to-end transfer delay through the network.

The objective of this paper is to present and shortly compare three methods for the evaluation of end-to-end delays: network calculus, queuing networks simulation and model checking.

1. Introduction

The evolution of avionics embedded systems and the amplification of the integrated functions number in the current aircraft imply a huge increase in the exchanged data quantity and thus in the number of connections between functions.

To control this complexity, we can benefit from the technological developments based on the concept of modular architecture [2, 3]. But the growth of the number of multi point communication, such as the setting in motion of embedded networks, constitutes one of the major stakes of new generation architectures. Several avionics architectures were developed but the most of them rest on rather old means of communication, like the ARINC 429 data busses which are mono transmitter buses with limited performances (100 Kbits/s) [6].

The solution adopted by Airbus for the new A 380 generation consists on the utilization of a recognized standard which allows a re-use of development tools as well as of existing material components while achieving better performance. It consist of the Switched Ethernet technology which benefits from a long industrial experiment [4], that allows to have confidence in the reliability of the material and on the facility of its maintenance. Hence aeronautical system can profit of a much more powerful technology than the traditional avionics bus (Switched Ethernet / 100 Mbps).

The disadvantage of Ethernet, opposite to an avionics application, is the non intrinsic determinism of its access method to the physical support, CSMA/CD (which induces possible collisions on the point-to-point links level) [5]. The solution is thus to use bi-directional links and so Full Duplex Switched Ethernet (AFDX) [7], where each equipment, in this architecture is only connected to a switch by the means of a Full Duplex link [8]. This way, there cannot be any more collisions on the physical support, and the CSMA/CD is no more necessary [24].

This standardized solution via the ARINC 664, eliminates the inherent indeterminism of the traditional Ethernet and the collision frame losses. But, the ARINC 664 shifts in fact the problem to the switch level where various flows will enter in competition for the use of the resources of the switch. This problem results in:

- Congestion on the output ports of the switch, if at a given time too much traffic moves towards one port only, there will be frame losses by overflow of it queues.
- Moreover, a burst of traffic, due to a passenger obstruction on a switch port, can likely encumber the neighbor switch. Thus, congested switches can lead to loss of frames.
- In the same way, the storage of frames in the switches queues can involve latency and important jitter.

In order to understand if such congestions can occur in the network, it is necessary to study the performance of the network. The objective is to measure:

- The required crossing delay of the network in order to allow the applications to preserve their response times. Network latency is a key performance parameter since flight-critical data must be delivered on time. Network latency is defined as the duration of time it takes for a frame to pass through a network.

- The output queues sizes which allow us to dimension the frame loss caused by the congestions.

The objective of this paper is to present and shortly compare three methods for the evaluation of end-to-end delays: network calculus, queuing networks simulation and model checking.

In a first step, we present main characteristics of an AFDX network and end-to-end traffic. In a second step, we compare the network calculus approach on a realistic example. In a third step, we compare on a simpler example the two previous approaches with a model checking approach.

2. The AFDX network main characteristics

In this section, we present main characteristics of the network architecture and the traffic that flows on the network.

2.1. AFDX network architecture

Avionics Full Duplex Switched Ethernet is a static network (802.1D tables are statically set up and no spanning tree mechanism is implemented). Flows are statically identified in order to obtain a predictable deterministic behavior of the application on the network architecture.

An example network architecture is depicted on figure 1. It corresponds to a test configuration provided by Airbus for a previous study [22]. It is composed of several interconnected switches. There is at most 24 ports per switch (8 on this example). There are no buffers on input ports and one FIFO buffer for each output port. The inputs and outputs of the networks are called *End Systems* (the little circles on figure 1). Each End System is connected to exactly one switch port and each switch port is connected to at most one End System. Links between switches are all full duplex. On figure 1, the values on End Systems indicates number of flows that are dispatched between End Systems. Number of input and output End Systems per switch are not specified on figure 1.

2.2. End-to-end traffic characterization

The Virtual Link is the basis of the Avionics Switched Ethernet protocol. As defined by ARINC-664, Virtual Link (VL) is a concept of virtual communication channels; It has the advantage of statically defining the flows which enters the network [9].

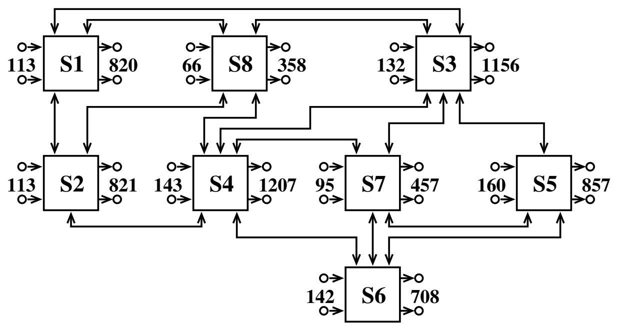

Figure 1. AFDX network architecture

End-Systems exchange Ethernet frames through VL. Switching a frame from a transmitting to a receiving End System is based on a VL (deterministic routing). The Virtual Link defines a logical unidirectional connection from one source End-system to one or more destination End systems. It is a path with multicast characteristic. Figure 2 shows an example of a multicast Virtual Link, considering the network architecture of figure 1. Its source End System is an input of switch **S1** and its destination End Systems are outputs of switches **S8**, **S3**, **S4** and **S7**. This VL includes the four paths **S1-S8**, **S1-S3**, **S1-S8-S4** and **S1-S8-S4-S7** (they are depicted as plain lines on figure 2).

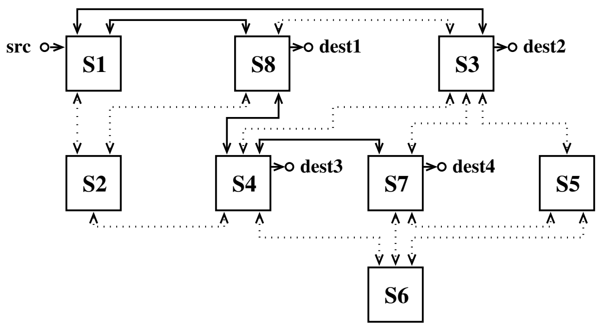

Figure 2. A multicast Virtual Link

The routing of each VL is statically defined. Only one End System within the Avionics network can be the source of one Virtual Link, (i.e., Mono Transmitter assumption).

The objective is to provide a logical isolation of VL: a given maximum bandwidth is allocated to each VL. Regardless of the attempted utilization of a VL by one application, the available Bandwidth on any other VL is unaffected.

A virtual Link is defined by the following parameters :

- the name of the VL,

- the Bandwidth Allocation Gap (BAG) of the VL, which corresponds to the minimum delay between the emission of two consecutive frames of the VL by its source End System,

- the minimum (S_{min}) and the maximum (S_{max}) lengths of the VL frames.

3. Simulation vs network calculus for obtaining end-to-end delays

We present a comparative evaluation of end-to-end upper bound delays using two fundamentally different approaches:

- the network calculus that analytically derives sure upper bounds on delays,
- simulation that gives experimental upper bounds on a set of scenarios.

The goal is to try to estimate how pessimistic the network calculus is.

3.1 The network calculus approach

Network Calculus results [16, 17] have been used to describe VLs by arrival curves [20], and further Network Calculus studies to describe the minimal service offered by network elements [10, 14, 18, 26, 27, 28] The Calculus gives the latency bound of any elementary network entity and for those elements that have a queuing capability, a queue-size bound expressed either in a number of bits or in a number of frames (with a simple majorization using S_{min}). Given an elementary entity that offers a service curve β to an input flow constrained by an arrival curve α , the calculus also brings the arrival curve $\alpha^{\star}$ of the output flow: $\alpha^{\star} = \alpha \phi \beta$ where $\alpha \phi \beta$ is defined by:

$$\alpha \phi \beta(t) = \sup u \geq 0 \{(\alpha(t+u) - \beta(u)\}$$

A Network Calculus tool that propagates these results on a complete network in a dataflow way is able to compute the latency and queue-size bounds in every element of the network [20, 23].

The group concept [21, 23] could be used to improve the Calculus of the AFDX bounds, defining groups of VLs that exit from the same multiplexer and enter another multiplexer together, i.e. Virtual Links that share two segments of path at least. The key issue is that the frames of those VLs are serialized once exiting the first multiplexer and thus they dont have to be serialized again in the following multiplexers.

Results presented in this paper don't consider this improvement.

3.2 The queuing network simulation approach

The simulation approach aims at giving experimental upper bounds on a set of scenarios, while the network calculus analytically derives sure upper bounds on delays [15]. The simulation is based on a queueing network modelling using QNAP2. The elements of the network will be built as a queuing station or object structure. They represent the simple network elements: One-way Links, Buffers, Demultiplexer, Scheduler Multiplexes. The selected policy of service is FIFO.

The global simulation model is depicted in figure 3. It is composed of the following elements:

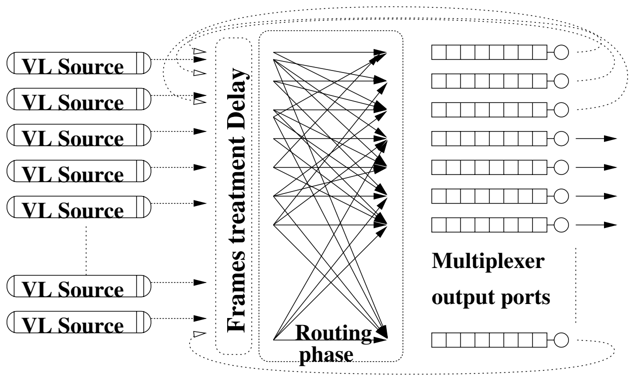

Figure 3. Network simulation model

VLs flows emission A VL application flow has a single source. The emitted flows must be multiplexed by the End Systems which controls the flow with the concept of Virtual Links. In this study we use a direct traffic generation at the switch level (abstracting End Systems since it is assumed they have no influence on the evaluated end-to-end delay). An ensured minimum time (BAG) between two consecutive frames is guaranteed for each VL. The following strategies can be used for frame generation:

- frames can be generated periodically, using the BAG as a period,
- frames can be generated sporadically, using the BAG as a minimum inter emission time,

In this study, we have considered, on the one hand the first strategy, on the other hand a specific case of the second strategy where the time between two consecutive frames of a given VL is always a multiple of the BAG. The goal is to evaluate the influence of BAG occupation on end-to-end delays.

Another parameter of the system is the phasing between VLs. In this study, we experiment two scenarios:

- the synchronized one, in which the first frame of every VL is transmitted at the same time,
- the desynchronized one, in which we associate to each VL a phase randomly distributed between 0 and its BAG.

The last parameter is the frame size associated to each VL. It has been mentioned earlier that it is comprised between a minimum and a maximum value. In this study, we always consider the following cases:

- the minimum length for every frame of every VL,
- the maximum length for every frame of every VL,
- the average length between the minimum and the maximum ones for every frame of every VL,
- a random length between the minimum and the maximum ones for every frames of every VL

The goal is to evaluate the influence of frame lengths on end-to-end delays.

VLs routing The objective is to model the switching and supported functions. The switches used in Avionics Switched Ethernet performs 802.1D bridging based on Virtual Link ID. Each switch is configured with a static bridging table. This table defines the Virtual Link assignment to physical network ports. As a frame is received, its Virtual Link ID is checked against the configuration table to route the frame to its destination port(s). A switch can be considered as a matrix of 100 Mbps connections.

For this study we consider that the switch only performs filtering and forwarding operations, modelled by a bounded delay (0.016ms for each frame routing).

The servicing policy on each output port file is FIFO, as described earlier. The service time depends on the capacity of the physical layer (100 Mbps) and the frame size. The delay depends on the number of frames in the output port file.

3.3. A realistic network configuration

In this section, we present a realistic network configuration on which we will evaluate worst-case end-to-end delays on VL paths. This configuration is implemented on the network architecture of figure 1. As already mentioned in section 2.1, it corresponds to a test configuration provided by Airbus for a previous study [22]. Values on input and output End Systems of each switch gives respectively the number of VL generated and received by the corresponding End Systems. For each switch, we have considered as a whole, on the one hand the input End Systems, on the other hand the output End Systems. The configuration includes a total of 964 VL and 6384 paths (due to multicasting). Table 1 gives the number of VL from each switch S to each switch D. The left part of table 2 gives the dispatching of VL among BAGs. It can be seen that BAGs are harmonic between 2 and 128 and that the larger a BAG the more it is used. The right part of table 2 gives the dispatching of VL among frame lengths, considering the minimum value associated with each value. The majority of VL consider short frames. Table 3 gives the dispatching of VL paths among the number of crossed switches. In our application, a VL path crosses at most four switches.

S \ D	1	2	3	4	5	6	7	8
1		71	78					34
2	72			77				34
3	90			212	35		42	52
4		97	134			37	35	48
5			80			72	64	
6				82	61		52	
7			52	47	59	67		
8	51	45	43	52				

Table 1. Total of VL from switch S to switch D

Bag (ms)	Number of VL
2	20
4	40
8	78
16	142
32	229
64	220
128	255

Frame length (bytes)	Number of VL
0-150	561
151-300	202
301-600	114
601-900	57
901-1200	12
1201-1500	35
> 1500	3

Table 2. Bags and frame lengths

Figure 4 shows an overview of the load of the network. More precisely, it gives the number of physical link for each possible load. The load of a physical link is defined as the portion of time a link is busy. Physical links are lightly loaded for our application.

Nb of crossed switches	Number of paths
1	1797
2	2787
3	1537
4	291

Table 3. VL paths lengths

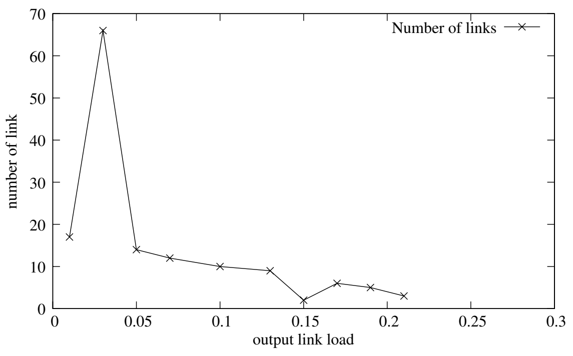

Figure 4. Overall network load

3.4 Comparison of the two approaches

We have conducted an evaluation of end-to-end delays on the network configuration presented in section 3.3, using on the one hand the network calculus approach explained in section 3.1, on the other hand the simulation model presented in section 3.2. The first goal is to evaluate how pessimistic network calculus results are. Figure 5 compares end-to-end-delays obtained by both approaches under the following hypothesis for the simulation:

- frames are generated periodically, using the BAG as a period,
- the minimum length is considered for every frame of every VL.

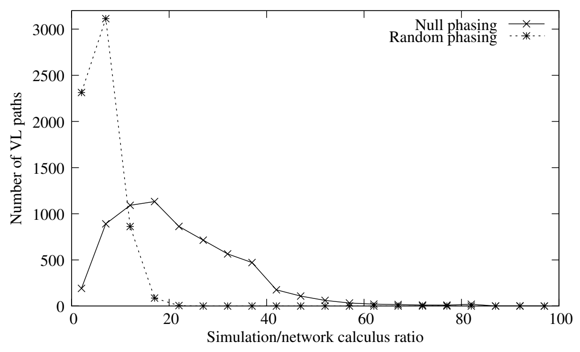

Figure 5. Simulation versus network calculus

More precisely, we have computed for each Virtual Link path the ratio the end-to-end delay obtained by simulation and the one calculated with the network calculus approach. Figure 5 shows the distribution of this ratio for two simulation configuration.

In the first one (null phasing), all VL are synchronous. It means that they all initiate their first frame at the same time. Results show that, for most of the VL paths, the ratio is between 5 % and 40 %. Moreover, all VL paths with a ratio of at least 70 % have a length of 1 (they cross a single switch). This corroborate the property that the deterministic upper bound obtained by the network calculus approach is reachable mostly in case of single switching communication level.

In the second simulation configuration (random phasing), VL are desynchronized by applying to each of them a specific random delay before the emission of its first frame. For each VL, the delay is chosen between 0 and its BAG. Results show that, for all VL paths, the ratio is under 20 %.

Then, we have studied the influence of BAG occupation on end to end delays. Figure 6 takes the all BAG occupation case as a reference and shows the dispatching of end-to-end delays variations for the randomized BAG occupation cases of 80 %, 50 %, 30 % and 10 %. Values under 100 on the abscissa correspond to smaller end-to-end delays than for the all BAG occupation case, while values other 100 correspond to bigger ones. Results show that end-to-end delays globally decrease with BAG occupation. However, for some VL paths, the end-to-end delay increases with the decreasing of other VL BAG occupation. That means that a simulation of the case where all BAG are occupied does not always lead to a worst-case end-to-end delay.

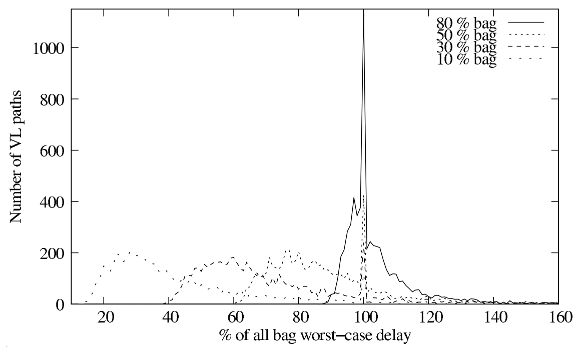

Figure 6. BAG occupation influence

Next, we have studied the influence of frame lengths on end-to-end delays. Figure 7 takes the minimum length case as a reference and shows the dispatching of end-to-end delays variations for the mean length case (average between minimum and maximum length), the maximum length case and the random length case (between minimum and maximum length). Values under 100 on the abscissa correspond to smaller end-to-end delays than for the all BAG occupation case, while values other 100 correspond to bigger ones. Results show that end-to-end delays globally increase with frame lengths. More precisely, the average variations for the mean length and maximum length cases are respectively 129 and 157. However, as previously stated concerning BAG occupation, the end-to-end delay of some VL paths decreases with the increasing of other VL frame lengths.

Concerning the random length case, the average variation is 144, that is more than for the mean length case. This is due to the fact that we measure a worst-case end-to-end delay. It is globally encountered when frame lengths are closer to the maximal values.

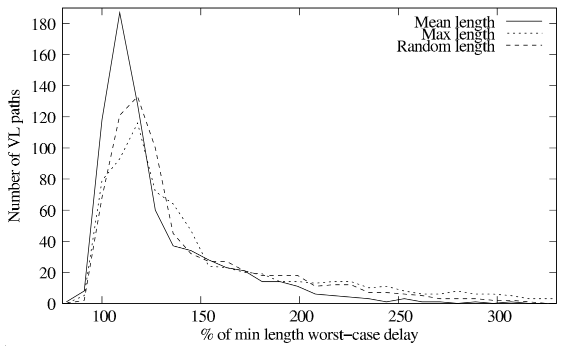

Figure 7. Frame lengths influence

The simulations that have been conducted are clearly insufficient to evaluate precisely the pessimistic degree of the network calculus approach. Obviously, the null phasing configuration is often quite close from the worst-case configuration. However, we will study in section 4 a simple example with a classical VL configuration, where null phasing is not the worst-case configuration. Moreover, one goal of simulations is to obtain the repartition of worst-case end-to-end delays on a representative set of scenarios. The simulations we have conducted are a first step to reach this goal. Much more has to be done, especially concerning the phasing of VL.

However, we are convinced that results we have obtained with the network calculus approach are too pessimistic and could be improved. In order to do that, we can use the group concept [21]. A different solution will be presented in section 4. It is based on timed automata and model checking.

Coming back to the experiments we have conducted, we have studied the relationship between the end-to-end delay of a VL path and its length (number of switches it crosses), path congruence and switch congruence. The congruence of an output port is defined as the total number of VL that crosses this port. The path congruence of a VL path is the sum of the congruences of the output ports it crosses. The switch congruence of a VL path is its path congruence divided by its length. As an example, consider the VL path **S1-S8-S4** on the network configuration of figure 1. Its length is 3, its path congruence is 207 (34 + 52 + 121), 121 being the congruence of the port on which it exits from switch **S4**. The switch congruence of this VL path is then $\frac{207}{3} = 69$.

Figure 8 shows the path and switch congruences as functions of the end-to-end delay for both simulation and network calculus approaches.

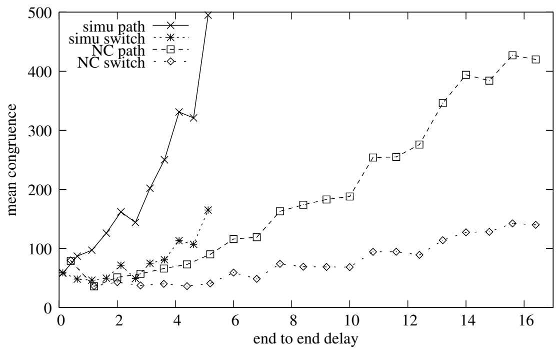

Figure 8. Delay as a function of congruence

Concerning the simulation approach, the *simu path* curve shows that the path congruence grows rapidly with the end-to-end delay. Meanwhile, the switch congruence (curve *simu switch*) grows slowly with the end-to-end delay. Similar observations can be made with the network calculus approach.

Figure 9 shows the VL path length as a function of the end-to-end delay for both simulation and network calculus approaches.

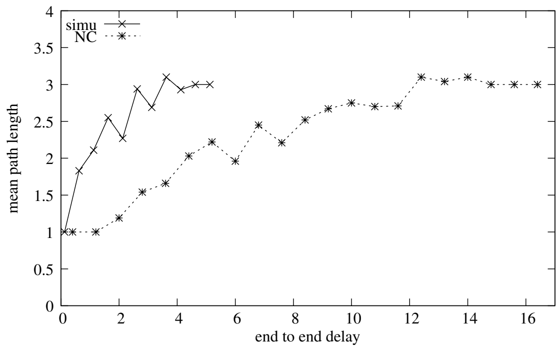

Figure 9. Delay as a function of path length

For both approaches, the length first grows with the end-to-end delay, and then stays quite constant.

Results of figures 8 and 9 shows that path and even switch congruences are at least as important as path length to determine the end-to-end delay of a VL path. This is not surprising, since, as soon as a VL path shares output ports with many other VL, the transmission time over Ethernet links becomes small compared with the waiting time in output port buffers. An interesting study should be to try to express the end-to-end delay of a VL path, knowing the path length and the path and switch congruence. This implies especially a more global study on a set of representative network configurations.

4. Performance evaluation by model checking

Two methods for evaluating the worst-case end-to-end delay of the Switched Ethernet Full Duplex architecture have been described in previous sections. The Network Calculus and simulation approaches have been applied in this context. However, it is often difficult to evaluate the quality of the obtained worst-case end-to-end delay: it is possible to approach or reach this delay ? As presented in previous sections, network calculus gives a mathematical upper bound which is a guaranteed upper bound, but cannot always be reached. Conversely, simulation gives an experimental bound that can awfully be exceeded (simulation can miss rare events).

Here, we propose to use a model-checking approach based on timed automata. This method consists in exploring all the possible states of the system and will thus determine an exact worst-case end-to-end delay. It implies to compute if a propriety, expressed by a timed logic, is verified or not. Such an approach has already been explored on a slightly different CAN / Switched Ethernet architecture [19].

To compare the different methods, we consider a simpler example as depicted in Figure 10. It is composed of 5 end-systems. Each one is the transmitter of one VL (named $send_i$ in the considered system, see figure 10) and 3 Ethernet switches. For all VLs, the BAG is 4ms and the length is 500 bytes. So, the transmission delay over one physical link is 0.040ms. Here, we consider that the routing delay in the switch is null, as it has no influence on the comparison of the three methods.

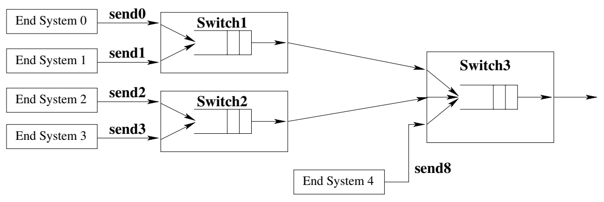

Figure 10. The considered system

In the following section, we present the timed automata model and we compare the worst-case end-to-end delay obtained using model-checking, Network Calculus and simulation.

4.1 Modelling with timed automata

Timed automata have been first proposed by Alur and Dill [11] to describe systems behavior with time. A timed automaton is a finite automaton with a set of clocks, *i.e.* real and positive variables increasing uniformly with time. Transitions labels are:

- a guard, *i.e.* a condition on clock values,
- actions,
- updates, which assign new value to clocks.

Composition of timed automata is obtained by synchronous product. Each action *a* executed by a first timed automaton corresponds to an action with the same name *a* executed in parallel by a second timed automaton. In other words, a transition which executes action *a* can only be done if another transition labeled *a* is possible. The two transitions are performed simultaneously. So communication use rendezvous mechanism.

Performing transitions requires no times. Conversely, time can run in nodes. Each node is labeled by an invariant, that is a boolean condition on clocks. Node occupation is dependent of the invariant. The node is occupied if the invariant is true.

Timed automata have been extended. One extension is committed nodes. The goal of these nodes is to ensure atomicity between consecutive execution of discrete actions [25]. As an example, consider the three automata of the figure 11.

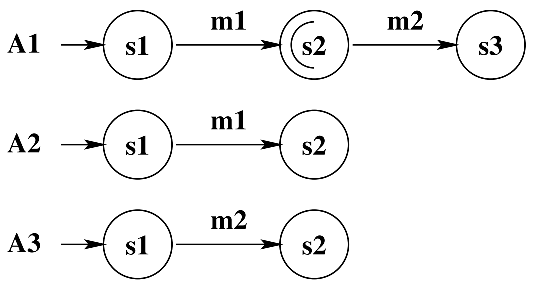

Figure 11. Example of committed nodes

A1 performs m1 and simultaneously A2 performs m1. Then A1 performs m2 and simultaneously A3 performs m2. As s2 of A1 is committed, the two transitions m1 and m2 are performed simultaneously without time evolution. So, this extension allows to model broadcast communication mechanism through timed automata.

Another extension is timed automata with shared integer variables. In timed automata with shared integer variables, a set of variables is shared by timed automata. In such a way, these values can be consulted and updated by any timed automata [25, 13].

A system modelled with timed automata can be verified using model-checking. The reachability analysis is performed by model-checking. It consists in encoding the property in terms of reachability of a given node of one of the automata. So, the property is verified by the reachability of node if and only if the node is reachable from an initial configuration. Reachability is decidable and algorithms exist [25]. Unfortunately, reachability analysis is undecidable on timed automata with shared integer variables, but some semi-algorithms exist.

In the following subsections, we model the AFDX configuration example depicted on figure 10 using timed automata with shared integer variables. Properties will be verified using UPPAAL model-checker [1].

4.1.1 Modelling sending functions

In respect to the BAG delay, we consider that each function sends a message periodically. This message is represented by the $send_i$ signal. An example of function in UPPAAL is depicted in Figure 12.

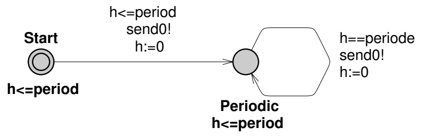

Figure 12. Automaton of the function 0

Functions send the signal $send_i$ periodically at exactly period time (transition from the periodic node in the Figure). This time period can be delayed initially by letting a duration time between 0 and period ms to pass (transition from the start node).

4.1.2 The switch automaton

The switch uses FIFO policy on each queued transmit port. Each node of the automaton models a location in the queue. Consequently, the number of nodes of the automaton equals the size of the queue. Figure 13 shows an example of a queue model transmit port with queue size 3. Each transition from a node $Position_i$ to a node $Position_{i+1}$ of the automaton models the arrival of one frame at the transmit port (represented by signals $send0$ and $send1$). $delay$ is the transmission time of the frame. In the considered example application, it is the same for every frame, since frame length is 500 bytes for all VLs. The frame is then transmitted using $send4$ and $send5$ corresponding respectively to $send0$ and $send1$ signals.

4.1.3 The global system

The global system is composed of 5 functions and 3 switches as represented in figure 14, where $send9$ corresponds to $send0$, $send10$ to $send1$, $send11$ to $send2$, $send12$ to $send3$, and $send13$ to $send8$. Each VL is produced by a separate End System. These components are modelled by automata as described in previous sections. The global model is obtained by composition of all au-

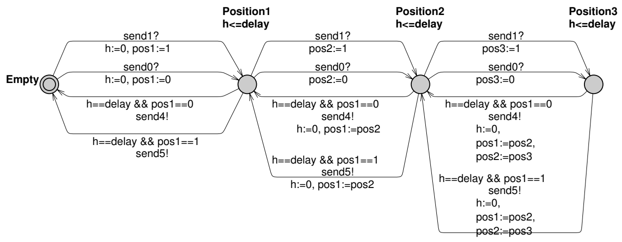

Figure 13. Automaton of a switch queue receiving two messages from two different ports

tomata:

$$
\begin{aligned}
System &= Switch_1||Switch_2||Switch_3|| \\
&\quad F_0||F_1||F_2||F_3||F_4
\end{aligned}
$$

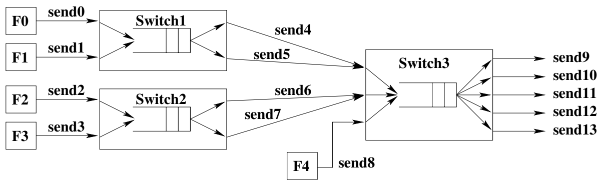

Figure 14. The global modelled system

4.1.4 Worst-case delay calculation

In this section, we show how worst-case end-to-end delay can be obtained from the models previously described. Model-checking is used to determine the global transmission delay of each frame in the system. The method consists in verifying that a frame is received before a global transmission delay. In other words, the property to verify is "given a message m_i, the global transmission delay of the frame $frame_i$, noted $d(frame_i)$ must be lower than a bounded delay $d_i : d(frame_i) \leq d_i$". The test automata method can be used to help the verification process. This method is described in [13, 12] and consists in constructing a test automaton which encodes the considered property. Then, the model-checking consists in calculating if a reject node is reachable or not. The test automaton of our property for frame 0 is depicted in Figure 15.

When a signal $send0$ is received, the test automaton waits for $bound$ time units. If $h > bound$, i.e. no signal $send9$ is detected before $bound$ time units, the rejected node, named $unhappy$, is reached. The property is false. So, we compute for each frame the lower value of the global transmission delay using the model-checker UPPAAL.

The computed worst-case delay for any VL is 0.240ms in our system. It corresponds to the case where VL $send0$,

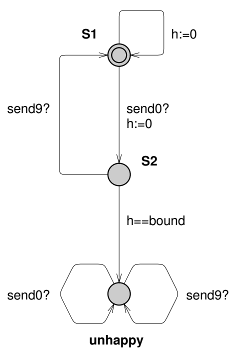

Figure 15. The test automaton to calculate the worst-case delay of a frame sent by function 0

$send1$, $send2$ and $send3$ are synchronous, while VL $send8$ has a delay phase of 40 μs with the four other ones. This worst-case delay can be experimented by VL $send0$, $send1$, $send2$ and $send3$.

An evaluation by simulation has been undertaken on the same application. If we consider that all VL are synchronous, the worst-case delay for any VL is 0.2ms and it is experimented by $send0$, $send1$, $send2$ and $send3$. A random phasing is very unlikely to give the worst-case delay computed by model checking, since there is a huge number of possible phasings.

The network calculus approach considering no grouping has also been processed. It gives a worst-case delay for any VL of 0.287ms. Obviously, grouping should reduce this delay, but it will certainly remain over 0.240ms, since it will consider a pessimistic upper bound on traffic.

The system of figure 10, considered for the comparison, corresponds to a classical pattern that is depicted on figure 16.

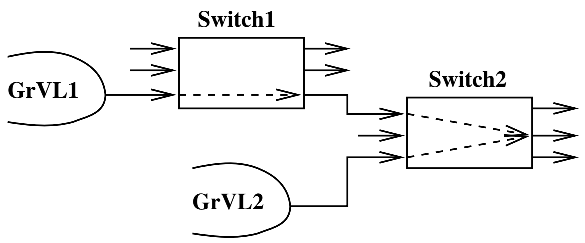

Figure 16. Generalization of the considered system

GrVL1 is a group of VL that all merge in *Switch1* by the same input port, cross *Switch2* and go out by the same output port. Similarly, *GrVL2* is a group of VL that all merge in *Switch2* by the same input port and go out by the same output port. On the system of figure 10, {*send0*}, {*send1*}, {*send2*} and {*send3*} are *GrVL1* groups, while {*send8*} is a *GrVL2* one.

What says model-checking is that the worst-case end-to-end delay is reached when *GrVL1* and *GrVL2* groups are synchronous. It can help the simulation approach choosing a phasing between VL that gives higher end-to-end delays that the null phasing.

5. Conclusion

In this paper, we presented three different methods for evaluating end-to-end delays on an avionics full duplex Ethernet. Obtained results strongly depend on ARINC 664 assumptions (Virtual Link static definition).

The network calculus approach gives guaranteed upper bounds on end-to-end delays that usually cannot be reached, as it is based on strong pessimistic assumptions. In this paper, we didn't exploit all the possible improvements of network calculus, especially the grouping concept (frames of VLs that have been serialized once exiting a multiplexer don't have to be serialized again on the following multiplexer). Moreover, more tight upper bounds of arrival curves and inter switches traffic could lead to more accurate upper bounds on end-to-end delays. Such an approach can be very useful for certification reasons.

The simulation approach gives experimental bounds that can awfully be exceeded, as simulation can miss rare events. Results presented in this paper only take into account a very limited set of scenarios that have to be extended. Trying to drive simulation towards rare events could be a promising track. Such an approach is valuable for obtaining a global estimation of the network load.

The model checking approach determines an exact worst-case end-to-end delay and the corresponding scenario, since it explores all the possible states of the system. An open question is: can model checking be used on a realistic network configuration ? Clearly, the model presented in this paper will lead to combinatorial explosion. Abstraction and aggregation techniques are being studied in order to overcome this problem.

References

[1] http://www.uppaal.com.

[2] *ARINC 651, Aeronautical Radio Inc. ARINC specification 651. Design Guidance for Integrated Modular Avionics.*, 1991.

[3] *ARINC 653, Aeronautical Radio Inc. ARINC specification 653. Avionics application Software Standard Interface.*, 1997.

[4] *IEEE 802.1D, Local and Metropolitan Area Network: Media Access Control Level Bridging.*, 1998.

[5] *IEEE 802.3, Information technology.*, 1998.

[6] *ARINC 429, Aeronautical Radio Inc. ARINC specification 429. Digital Information Transfer System (DITS) parts 1,2,3.*, 2001.

[7] *ARINC 664, Aircraft Data Network, Part 1: Systems Concepts and Overview.*, 2002.

[8] *ARINC 664, Aircraft Data Network, Part 2: Ethernet Physical and Data Link Layer Specification.*, 2002.

[9] *ARINC 664, Aircraft Data Network, Part 7: Deterministic Networks.*, 2003.

[10] R. Agrawal, R. Cruz, C. Okino, and R. Rajan. Performance bounds for flow control protocols. *IEEE Transaction on Networking*, 7(3), June 1999.

[11] R. Alur and D. L. Dill. Theory of Timed Automata. *Theoritical Computer Science*, 126(2):183–235, 1994.

[12] B. Bérard, M. Bidoit, A. Finkel, F. Laroussinie, A. Petit, L. Petrucci, and Ph. Schnoebelen. *Systems and Software Verification. Model-Checking Techniques and Tools*. Springer, 2001.

[13] A. Burgueño Arjona. *Vérification et synthèse de systèmes temporisés par des méthodes d'obervation et d'analyse paramétrique*. PhD thesis, Ecole Nationale Supérieur de l'Aéronautique et de l'Espace, Toulouse, France, 1998.

[14] C. Chang. On deterministic traffic regulation and service guarantee: a systematic approach by filtering. *IEEE Transactions on Information Theory*, 44, May 1998.

[15] H. Charara and C. Fraboul. Modelling and simulation of an avionics full duplex switched ethernet. In *Proceedings of AICT*, Lisboa, Portugal, 2005.

[16] R. Cruz. A calculus for network delay, part I. *IEEE Transactions on Information Theory*, 37(1):114–131, January 1991.

[17] R. Cruz. A calculus for network delay, part II. *IEEE Transactions on Information Theory*, 37(1):132–141, January 1991.

[18] R. Cruz. Quality of service guarantees in virtual circuit switched networks. *IEEE Journal of selected areas in communication*, 13(6), August 1995.

[19] J. Ermont, J.-L. Scharbarg, and C. Fraboul. Worst-case analysis of a mixed can/switched ethernet architecture. In *Proceeding of the Real-Time Networked Systems Conference*, Poitiers, France, 2006.

[20] C. Fraboul and F. Frances. Applicability of Network Calculus to the AFDX. Technical Report PBAR-JD-728.0821/2002, 2002.

[21] F. Frances, C. Fraboul, and J. Grieu. Using network calculus to optimize the AFDX network. In *Proceedings of ERTS*, Toulouse, France, 2006.

[22] J. Grieu. *Analyse et évaluation de techniques de commutation Ethernet pour l'interconnexion des systèmes avioniques*. PhD thesis, INP-ENSEEIHT, France, 24 Sept. 2004.

[23] J. Grieu, F. Frances, and C. Fraboul. Preuve de déterminisme d'un réseau embarqué avionique. In *Actes du 10ème Colloque Francophone sur l'Ingenierie des Protocoles*, Paris, 7-10 Octobre 2003.

[24] J. Jasperneite, P. Neumann, M. Theis, and K. Watson. Deterministic Real-Time Communication with Switched Ethernet. In *Proceedings of the 4th IEEE International Workshop on Factory Communication Systems*, pages 11–18, Västeras, Sweden, Aug. 2002. IEEE Press.

[25] K. G. Larsen, P. Pettersson, and W. Yi. UPPAAL in a Nutshell. *International Journal on Software Tools for Technology Transfer*, 1(1–2):134–152, 1997.

[26] J. Le Boudec. Application of network calculus to guaranteed service networks. *IEEE Transactions on Information Theory*, 44, May 1998.

[27] J.-Y. Le Boudec and P. Thiran. *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet*, volume 2050 of *Lecture Notes in Computer Science*. Springer-Verlag, 2001. ISBN: 3-540-42184-X.

[28] H. Sariowan, R. Cruz, and G. Polyzos. Scheduliing for quality of service guarantees via service curves. In *Proceedings of the International Conference on Computer Communications and Networks*, Las Vegas, 1995.