Progress in Flight Dynamics, GNC, and Avionics **6** (2013) 501-514
DOI: 10.1051/eucass/201306501

ANALYSIS OF REAL-TIME NETWORKS WITH MONTE CARLO METHODS

C. Mauclair and G. Durrieu

Office National de la Recherche Aérospatiale
BP 74055 — 2 avenue Édouard Belin
Toulouse 31055, France

Communication networks in embedded systems are ever more large and complex. A better understanding of the dynamics of these networks is necessary to use them at best and lower costs. Todays tools are able to compute upper bounds of end-to-end delays that a packet being sent through the network could suffer. However, in the case of asynchronous networks, those worst end-to-end delay (WEED) cases are rarely observed in practice or through simulations due to the scarce situations that lead to worst case scenarios. A novel approach based on Monte Carlo methods is suggested to study the effects of the asynchrony on the performances.

1 INTRODUCTION

Since the next-generation networks arrived, such as Avionics Full DupleX switched Ethernet (AFDX) in avionics, embedded communication networks tend to grow ever more and become more complex. So far, AFDX networks have been studied from the what-happens-in-the-worst-situations point of view. Of major interest, there are the network calculus [1, 2] and, more recently, the trajectories method [3, 4].

While the WEED obtained are necessary for certification needs, they are not representative of what happens "in the general case." As a consequence, manufacturers observed WEED in simulations that are only 10% of the computed ones for the A380. A better understanding of the dynamics of these networks is necessary in order to use them at their maximum capacity. Some extensions have been suggested to compute the probability for a packet to experience an end-to-end delay larger than a given bound [5–8]. However, neither offer a global overview of the distribution of the end-to-end delays.

The third approach based on simulation and called guided simulation is used to estimate the pessimism of the bounds computed by network calculus. It

derives a simplified network for each route of the original network where low-interacting routes are taken away [9,10]. The complexity is greatly reduced and such a method could be used to compute an estimation of the average end-to-end delays. However, one must be confident in the representativeness of the networks after simplification.

The worst case scenarios that lead to the WEED for packets going through the network are roughly those where a packet from every Virtual Link (VL) concerned arrives in the switches at the same time. This is not true in the average case where packets are more or less evenly distributed along their path and across the VLs and where there are few bottlenecks in the network. In this paper, a novel approach to compute the probability mass function (pmf), mean and standard deviation of the maximum end-to-end delay (MEED) of the packets going through an AFDX network by using Monte Carlo methods from the statistics field is described. These methods, described in subsection 4.3, approximate the real pmf of a phenomenon by averaging sampled experiments.

Section 2 briefly describes AFDX networks; section 3 gives a brief overview of existing works made on these networks; section 4 presents the framework used to compute an approximation of the probability distribution function of the MEED of a packet (as well as a small remainder on statistics); a small example is given in section 5; section 6 concludes this contribution.

2 AVIONICS FULL DUPLEX SWITCHED ETHERNET NETWORKS

Avionics Full DupleX networks are the data networks for safety critical applications that utilize dedicated bandwidth while providing deterministic Quality of Service (QoS). They are based on Ethernet in a star-like topology and equip aircrafts such as the Airbus A380 and Boeing 787. They were primarily intended to reduce wires aboard planes and bring high-speed transfers between equipments (called end-systems (ES)). Of most significance (see section 4), AFDX networks are asynchronous: each end-system and switch (SW) has it own clock.

Data transmissions between two applications executing on two different end-systems use the notion of VL which is virtually equivalent to the former unidirectional, multicast ARINC 429 wires. The VLs are multiplexed over the Ethernet wires that connect the end-systems and the switches. In Fig. 1, some application executing on end-system 4 sends its data to another application executing on end-system 2 via VL2 (ES4–SW1–ES2). Similarly, VL 1 links three applications: the source executes on end-system 1, the destinations execute on end-systems 4 and 5 (ES1–SW1–ES4 or ES1–SW1–SW2–ES5). Virtual links 1 and 3 are multiplexed between SW2 and ES5.

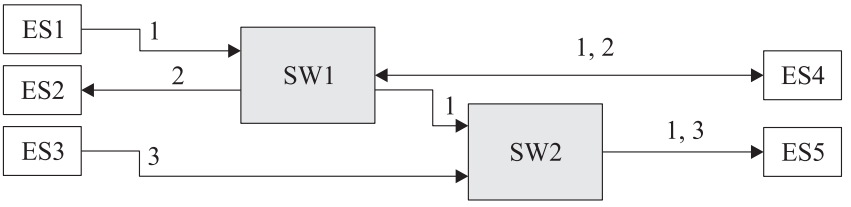

Figure 1 Basic AFDX network

The QoS offered by AFDX networks is achieved by allocating a fraction of the total bandwidth of each Ethernet wire to each VL equal to $S_{\max}/\mathrm{BAG}$, where $S_{\max}$ is the maximum size of each packet of the VL and BAG is the minimum interval between two packet emissions called Bandwidth Allocation Gap.

Packets arriving in the switches are routed according to a static configuration of the network and are treated according to the priority of the source VL (the priority is static over time). If multiple packets with equal priorities arrive at the same time, they are placed in arbitrary order in a queue, waiting to be processed. The emissions are not preemptive and the packet with the highest priority leaves the switch before any other. In practice, the AFDX network of the A380 uses only one priority. Thus, each VL, and thus packet, has the same priority and packets are treated according to a FIFO (First In First Out) policy.

3 BRIEF OVERVIEW OF EXISTING WORKS ON AVIONICS FULL DUPLEX SWITCHED ETHERNET

3.1 Conservative Worst End-to-End Delay Computations

By conservative, larger or equal to the real values are meant. These approaches are perfectly suited for the certification process where it must be mathematically proved that the network can handle all the packets it has to convey in a bounded time, whatever the traffic.

The Network Calculus was specifically developed to study networks components (as the name suggests) and is heavily used [1,11,12]. Applied to AFDX, it can compute bounds on end-to-end delays for packets transiting over a network. These bounds can be absolute [2] or accompanied by a probability to be overrun [5,8,13]. The WEED is computed as the maximum horizontal deviation between the traffic going in and out, upper- and lower-bounded by an arrival

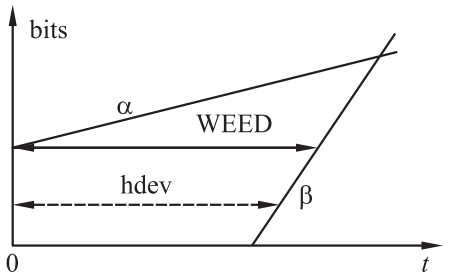

Figure 2 Example arrival and service curves

curve α and a service curve β, respectively. If $R(t)$ is the cumulated number of bits arrived in a switch and $R'(t)$ is the cumulated number of bits departed from the switch of a VL at time t, then these curves have the following properties for all $0 \leq s \leq t$:

$$\alpha(s) \geq R(t+s) - R(t)\,; \qquad \beta(s) \leq R'(t+s) - R'(t)\,.$$

At most, $\alpha(s)$ bits arrive in the switch during any interval of length s and at least $\beta(s)$ bits leave it. The WEED for the VL is then

$$\mathrm{WEED}\,(\mathrm{VL}) = \max_{0 \leq t} \{\mathrm{hdev}_{\mathrm{VL}}(t)\}$$

where

$$\mathrm{hdev}_{\mathrm{VL}}(t) = \min_{0 \leq s} \{\beta_{\mathrm{VL}}(t+s) - \alpha_{\mathrm{VL}}(t) \geq 0\}\ .$$

Figure 2 shows the WEED and horizontal deviation graphically on an example.

The trajectories method recently received an increasing attention from aircraft manufacturers [14]. Like the network calculus, it can compute absolute [3, 4] and probabilistic bounds [6, 7]. This is an extension of the holistic schedulability analysis [15] that focused on the propagation times between communicating tasks exchanging messages. If C is the time needed to send a packet of a VL and T is its period, then the WEED is defined by

$$\mathrm{WEED}\,(\mathrm{VL}) = \max_{r_i \leq t} \{W_{\mathrm{VL}}(t) - t\} + C_i\,.$$

Here,

$$W_{\mathrm{VL}}(t) = I(\mathrm{hp}_i, W_{\mathrm{VL}}(t)) + I(\mathrm{sp}_i, t) + \left\lfloor \frac{t}{T_i} \right\rfloor \times C_i + \max_{j \in \mathrm{lp}_i} \{C_j - 1\}$$

where

$$I(a,b) = \sum_{j \in S} \left(1 + \left\lfloor \frac{t}{T_j} \right\rfloor\right) \times C_j\,;$$

and W_{VL} is the time at which packet p from VL, arrived at time t, will be emitted; after another C_i, it will have quit the switch permanently. Thus, packet p will spend $(W - t + C_i)$ in a queue of the switch and WEED is just the maximum of these values. The second, third, and fourth terms of W are, respectively, the longest durations of lower (lp_i), equal (sp_i), and higher priority (hp) packets can interfere with the emission of p. The second term is the duration of other packets from the same VL take to be sent.

While an asset for these methods is to offer results *not* optimistic for certification purposes, it, clearly, is a drawback when it comes to performance evaluation as the results obtained are sometimes too conservative (pessimistic) and (quite) rare in practice.

3.2 Approximative Worst End-to-End Delay Computations

Approximative approaches perform WEED computations on models that are a subset of or simpler than the original system. The values computed can be greater or less than the actual value. Their interest is their capacity to take into account quite large systems at a reasonable cost. They cannot be used for certification, but are valuable for general performance of a network.

Guided simulation is based on simulation but uses an affinity function between VL in order to only simulate a subset [9, 10]. It tries to estimate the pessimism of the bounds computed with the network calculus by estimating the pmf of the end-to-end delays. As with all simulation, one must be confident in the fact that the scenarios simulated are effectively representative of the use of the network and can lead to a valid distribution of the end-to-end delays.

Queuing systems have also been used to study the networks, though less recently [16]. They model switches activity as Markov chains: any packet going in/out contributes positively/negatively to the queue (see Fig. 3 for an example of an $M/M/1/K$ queue, decreasing exponential probability function of arrivals and departures, one departure at most at the same time, K slots in the queue). The probability to find the queue in a specific state (number of packets) is easily computed if arrival and departure rates are exponential, which is not true in

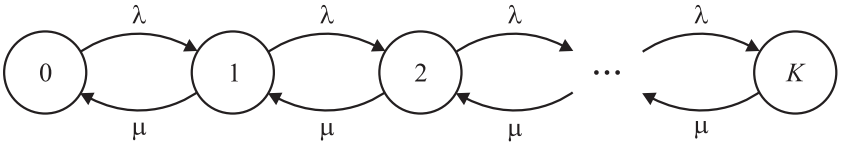

Figure 3 $M/M/1/K$ queue

the case of AFDX networks. The general case ($G/G/n/m$) can be studied using automatas, but combinatorial explosions arise quickly.

The small complexity of these methods makes them attractive to solve big (even huge) systems. However, approximations that reduce the complexity also have a significant influence on the precision. For (not so) small probabilities of exceeding the bound computed, they give nonnull probabilities to events that cannot happen (i. e., rejected by conservative approaches). On the other hand, for "reasonable" probabilities, the gain is substantial.

3.3 Exact Worst End-to-End Delay Computations

Exact means the values computed are the actual values. The obvious interest is the precision of such methods. They are extensions of (timed) automata and use model-checking techniques to explore them [17–19]. A model-checker (like UPPAAL, for example) skims through the various states to find a property or give a counterexample [20]. The major drawback is the combinatoric explosion (of the number of states) which prevents the use of such methods with medium to large systems.

4 FRAMEWORK

The objective is to obtain the pmf of the MEED of packets transiting over an AFDX network, not just the worst delay, which is only the maximum of the MEEDs. To do that, the fact that AFDX networks are asynchronous was taken into account: each time the plane is powered up, the phases between the VLs are different (i. e., their relative dates of emissions are not the same). The Monte Carlo methods have been used to approximate the pmf from a set of randomly generated phases between the VLs for whom the MEED of the packets have been computed for each VL. Figure 4 explains the different steps. The following subsections describe the model (4.1) and computation of the MEED (4.2).

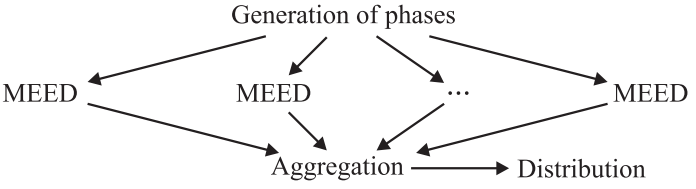

Figure 4 General work flow

Then, the paper reminds some key points on Monte Carlo methods and statistics (4.3).

Though the state space of phases can be reduced to a finite number of the distinct ones [21], there are too many to just compute MEED for all of them. As an illustration, there are $576 \cdot 10^9$ distinct phases for the example shown in subsection 5.1. Treating each of them for only 1 ms would take more than 18 years of nonstop calculations.

4.1 Model Used for the Computation

The common notations among the scheduling community have been used to describe the present model. Virtual links and packets are described by tasks and jobs, tasks instances. When a packet arrives in a switch, the corresponding task is activated; the task instance created is then ready to be selected for execution according to the policy HP/FIFO. When a job is executing, this means the corresponding packet (the one that triggered the task activation) is being sent. Tasks are described by a 4-uplet: (P_i, T_i, C_i, r_i), priority, period (i. e., BAG), worst case execution time (i. e., time needed to send the largest packet of VL), and date of first instance. There is no need in a deadline as there is no interest in the schedulability. With these notations, a phase is a valued vector of r_i.

Figure 5 illustrates different notations.

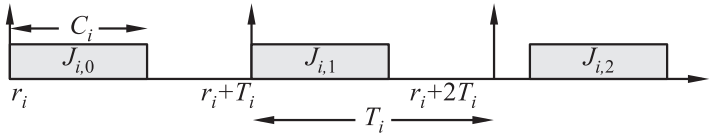

Figure 5 Notations for task τ_i

In the rest of this paper, it is assumed that each job has an execution time equal to the worst case: that is, each packet of a VL has the same size as the others. This is not a very restrictive assumption in the context of networks. Also, as previously stated, a single priority is used aboard the A380: the example presented in subsection 5.1 will do the same. Each emission of a packet is made as soon as possible, each T_i, that is. Finally, each VL is independent from the others.

This last assumption is pessimistic and supposes two VLs following the same route will interfere in each switch which is not true as one will get in front of the other once and stay in that place all along. The network calculus takes advantage of that and calls it "pay burst only once."

4.2 Computation of the Maximum End-to-End Delay

A specialized tool to compute the MEED of each task, i. e., VL, for each generated phase has been developed. The obtained values of the different MEEDs have been aggregated in order to obtain a pmf as close as possible as the actual pmf. The trajectories method that gives rather precise results too could be used but the MEED would be slightly pessimistic as the method generally overestimates the influence of the jobs on each other.

More generally, anything that takes the phases as input could be used to do the computation: that excludes the network calculus that cannot take them into account.

For example, one can use exhaustive methods based on model checking: instead of verifying properties, it can be tweaked to compute the different end-to-end delays and only keep the maximum.

Each task being periodic, it is enough to study what happens in the range $[\max\{r_i\} + \mathrm{lcm}\{T_i\}; \max\{r_i\} + 2\mathrm{lcm}\{T_i\})$ (see [22] for further details). The interest is only in the stationary state of the network; thus, the interval between $\min\{r_i\}$ and the beginning of the study interval is omitted though a higher value than the MEED could happen there.

First, let compute each date where some task is activated: these are of the form $r_i + kT_i$ for some task i and some integer k. For each of them, let compute the backlog since the last activations as the difference between the amount of work, the processor had to process at that previous date and the time that elapsed since, if the result is negative, let force it to 0. The backlog at the first activation in $\min\{r_i\}$ is null:

$$B_0 = 0\,; \qquad B_{n+1} = [B_n + A_n - (t_{n+1} - t_n)]^+$$

with

$$A_n = \sum_{\substack{\text{activated} \\ \text{tasks at } t_n}} C_i\,.$$

Then, for each date computed and for each job, let assume the job is put last in the queue. Therefore, as the scheduling is FIFO, its MEED is equal to the backlog plus A_{n+1}. The interesting thing is that each job will stay the same time as the others in the system if put last in the queue.

As of today, the simulator only works for a network with a single switch — more precisely, each VL must go through one and only one switch. Besides, it takes into account only one priority as is the case in AFDX networks, a new prototype is on its way that will be able to handle several priority levels.

4.3 Brief Reminder of Monte Carlo Methods

Let $(Y_i)_{i\leq n}$ be a series of independent and identically distributed random variables with the same pmf as a random variable Y (Fig. 6). Let also $y_n = (y_i)_{i\leq n}$ be a sample of the Y_i. A useful result from the probability theory is that the sample mean $\overline{y_n}$ and sample variance s_n^2 of y_n are unbiased estimators of the mean and variance of the random variable Y, noted $\mathsf{E}[Y]$ and σ^2, respectively.* Basically, this means the following

$$\overline{y_n} = \frac{\sum_{i=1}^{n} y_i}{n} \xrightarrow[n\to\infty]{} \mathsf{E}[Y]\,; \qquad s_n^2 = \frac{1}{n-1}\sum_{i=1}^{n}\left(y_i - \overline{y_n}\right)^2 \xrightarrow[n\to\infty]{} \sigma^2\,. \tag{1}$$

Both $\overline{y_n}$ and s_n^2 are the random variables themselves and Monte Carlo methods states that their expectations $\mathsf{E}[\overline{y_n}]$ and $\mathsf{E}[s_n^2]$ are equal to $\mathsf{E}[Y]$ and σ^2. One can express their variances too:

$$\mathrm{Var}\left(\overline{y_n}\right) = \frac{\sigma^2}{n} \approx \frac{s_n^2}{n}\,;\ \mathrm{Var}\left(s_n^2\right) = \sigma^4\left(\frac{2}{n-1} + \frac{\kappa}{n}\right) \approx s_n^4\left(\frac{2}{n-1} + \frac{\kappa}{n}\right)\,. \tag{2}$$

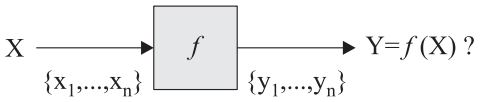

Figure 6 Basics of Monte Carlo methods

The kurtosis of Y, the term κ, is a measure of the "peakedness" of the pmf of a real-valued random variable. Higher kurtosis means more of the variance is the result of infrequent extreme deviations, as opposed to frequent modestly sized deviations. The skewness may also be of interest and is a measure of the asymmetry of the pmf of a real-valued random variable. Algorithms exist to compute these values efficiently. The formula of the sample kurtosis is given below, it is a biased estimator:

$$\kappa = n\frac{\sum_{i=1}^{n}\left(y_i - \overline{y_n}\right)^4}{\left(\sum_{i=1}^{n}\left(y_i - \overline{y_n}\right)^2\right)^2} - 3\,.$$

*The sample mean is a more robust unbiased estimator of $\mathsf{E}[Y]$, but the sample mean is easier to compute.

From there, if f is the "function" that takes a network and phases between VLs as input and returns the MEED of the VLs, the pmf of $(f(y_i))_{i\leq n}$ is an approximation* of the searched pmf. One can then deduce the mean and variance (or standard deviation) of the (real) MEED and compare the values with the WEED computed by conservative methods for example.

5 EXAMPLE AND RESULTS

5.1 Description of a Small Example

Let illustrate the approach with a (rather) small and (rather) useless example. As previously stated in the paper, the example plays with one priority only. Let consider the following set of tasks (i. e., VLs) $\tau_1 : (C_1 = 10,\ T_1 = 100)$, $\tau_{2\ldots4} : (10, 200)$, $\tau_5 : (4, 60)$, $\tau_6 : (4, 30)$, $\tau_7 : (5, 30)$, and $\tau_8 : (1, 8)$ independent of each other. As already mentioned, there are $576 \cdot 10^9$ phases for the tasks (combinations of $[r_i]$) [21]. The MEED cannot be computed for all those phases and let proceed with the present method. Let randomly generate a set of phases (10 000 and 100 000), compute the MEED for all of them, and aggregate and normalize the results.

5.2 Results

Figures 7*a* and 7*b* show pmf for tasks τ_1 and τ_8, respectively. It is a normalization of the number of times, each particular value has been computed for the MEED.

It can be deduced from these results that the probability to see the WEED is less than 10^{-5} **if the VLs are independent**. In this particular case, the worst case is pretty rare. In real life examples, however, the proportions will be more likely less "extreme." However, one has to be careful as the sample is really only a small fraction of all the phases. Virtual links are assumed to be independent, which may not be true; as a consequence, the assertion that the probability to see the WEED is less than 10^{-5} may be not true either.

Let also remark that there is a nonnegligible probability, about 2.3%, that all of the packets will not be retarded at all for VL1. This corresponds to a MEED equal to C_1. For τ_8, whatever the phase — with a probability higher than $99,99999\%$, actually — one job at least is retarded. This is because there is no phases where the MEED is zero. Beware, this does not mean that each packet of VL8 will be retarded.

*Convergence in law.

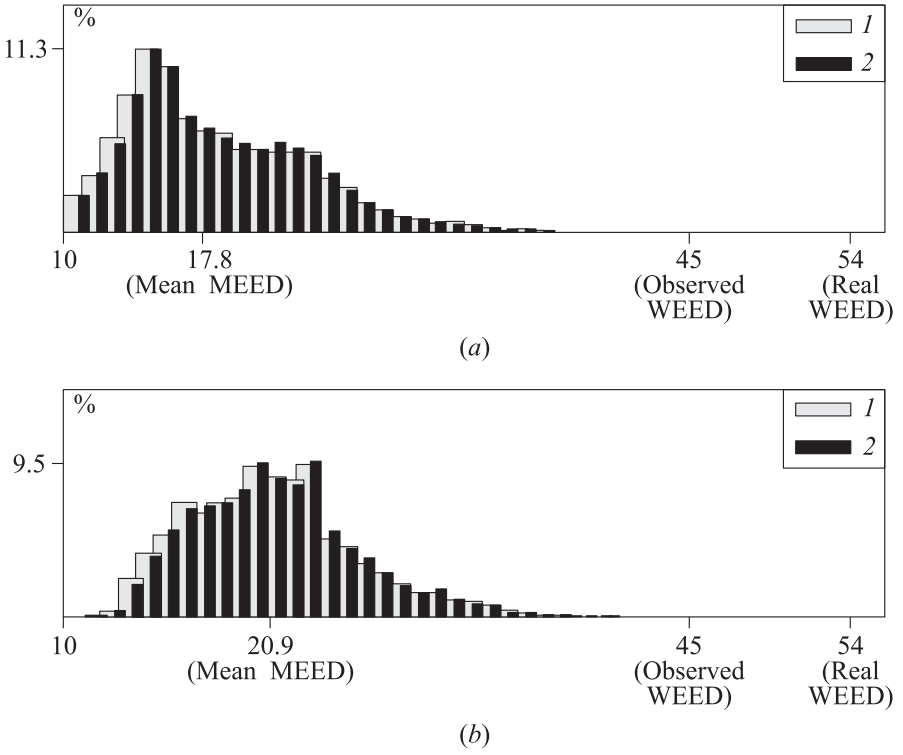

Figure 7 Probability mass function of the MEED for τ_1 (a) and τ_8 (b): *1* — 10^4 phases and *2* — 10^5 phases)

Also of interest are the small variations between the two curves. Sometimes, the black pmf is over the other one and sometimes, it is under, but overall, they are very similar (which they should be).

Tables 1 and 2 summarize the results of Eqs. (1) and (2) for the two samples. From a statistical point of view, it is equivalent to consider 10 samples of size 10^4 and a sample of size 10^5. This can be verified with the two pmfs obtained and

Table 1 Mean, variance, and standard deviation (10^4 phases)

Deviation	τ_1		τ_8	
	$\bullet$	Var($\bullet$)	$\bullet$	Var($\bullet$)
$\overline{y_n}$	17.8	$2.5 \cdot 10^{-3}$	20.9	$2.1 \cdot 10^{-3}$
s_n^2	25.6	$1.8 \cdot 10^{-1}$	22.3	$1.3 \cdot 10^{-1}$
s_n	5.1	$4.2 \cdot 10^{-1}$	4.7	$3.6 \cdot 10^{-1}$

Table 2 Mean, variance, and standard deviation (10^5 phases)

Deviation	τ_1		τ_8	
	$\bullet$	Var($\bullet$)	$\bullet$	Var($\bullet$)
$\overline{y_n}$	17.8	$2.5 \cdot 10^{-4}$	20.9	$2.1 \cdot 10^{-4}$
s_n^2	24.9	$1.6 \cdot 10^{-2}$	21.4	$1.1 \cdot 10^{-2}$
s_n	4.9	$1.3 \cdot 10^{-1}$	4.6	$1.0 \cdot 10^{-1}$

the results of the mean, variance (and standard deviation). There is an order of magnitude between the two samples. This verifies Eqs. (1) and (2) which state that the variance of the mean and variance should vary as $1/n$; in this case, $n = 10$.

The variance of the sample variance does not take the kurtosis into account. Its value for τ_1 and the sample with 10,000 different phases was negative around -3. This means that the real variance Var $\left(s_n^2\right)$ should be less than what is in the tables, which means that the results are actually better than what is shown here.

6 CONCLUDING REMARKS

In this paper, a novel approach for the computation of the "average" MEED for the VL in an AFDX network was suggested. The method can also be used in the context of a set of real-time tasks as it serves as the the underlying model. The benefit of the present method is the ability to have access not only to the mean, but also to the variations around this value and the variations of the variations. This gives confidence in the MEEDs computed if the variations are small, at least compared to the MEED.

A specialized tool to compute the full pmf of the time spent by a task in a real-time system was developed. However, since this work is being patented, it is only presented the special case of the maximum, the minimum works the same. In the best case scenario, each task has the same execution time, hence the complexity is linear with the number of tasks, and in the worst case scenario, each task has a different execution time, hence the complexity is exponential with the number of tasks. We do not know any other method that can treat the problem in less than exponential time and space.

These results are very encouraging and the authors will continue to develop their evaluation tool to take into account several priority levels and hope to confirm these results soon in future contributions.

REFERENCES

1. Le Boudec, J.-Y., and P. Thiran. 2004. *Network calculus: A theory of deterministic queing system for the Internet*. Springer Verlag.
2. Grieu, J. 2004. Analyse et évaluation de techniques de commutation Ethernet pour l'interconnexion des systèmes avioniques. Ph.D. Thesis. Toulouse, France: Institut National Polytechnique de Toulouse.
3. Bauer, H., J.-L. Scharbarg, and C. Fraboul. 2009. Applying and optimizing trajectory approach for performance evaluation of AFDX avionics network. *Emerging Technol. Factory Automation (ETFA)*. Palma de Mallorca: IEEE.
4. Bauer, H., J.-L. Scharbarg, and C. Fraboul. 2009. Applying trajectory approach to AFDX avionics network. *ECRTS (session WiP)*. Dublin. 01/07/2009-03/07/2009.
5. Ridouard, F., J.-L. Scharbarg, and C. Fraboul. 2007. Stochastic network calculus for end-to-end delays distribution evaluation on an avionics switched Ethernet. *5th IEEE International Conference on Industrial Informatics*. 1:559–64.
6. Azouz Saidane, L., S. Azzaz, S. Martin, and P. Minet. 2007. FP/FIFO scheduling: deterministic versus probabilistic QoS guarantees and p-schedulability. *ICC*.
7. Minet, P., S. Martin, L. Azouz Saidane, and S. Azzaz. 2007. FP/FIFO scheduling: Coexistence of deterministic and probabilistic QoS guarantees. *DMTCS*.
8. Ridouard, F., J.-L. Scharbarg, and C. Fraboul. 2008. Stochastic upper bounds for heterogeneous flows using a static priority queueing on an AFDX network. *J. FAC'2008, SVF/FéRIA*.
9. Charara, H., J.-L. Scharbarg, J. Ermont, and C. Fraboul. 2006. Methods for bounding end-to-end delays on an AFDX network. *18th Euromicro Conference on Real-Time Systems*.
10. Scharbarg, J.-L., and C. Fraboul. 2007. Simulation for end-to-end delays distribution on a switched Ethernet. *IEEE Conference on Emerging Technologies Factory Automation, ETFA*. 1092–99.
11. Cruz, R. 1991. A calculus for network delay. I. Network elements in isolation. *IEEE Trans. Information Theory* 37(1):114–31.
12. Cruz, R. 1991. A calculus for network delay. II. Network analysis. *IEEE Trans. Information Theory* 37(1):132–41.
13. Burchard, A., J. Liebeherr, and S. D. Patek. 2006. A min-plus calculus for end-to-end statistical service guarantees. *IEEE Trans. Information Theory* 52(9):4105–14.
14. Martin, S. 2004. Maîtrise de la dimension temporelle de la qualité de service dans les réseaux. Ph.D. Thesis. Université Paris XII.
15. Tindell, K., and J. Clark. 1994. Holistic schedulability analysis for distributed hard real-time systems. *Microprocess. Microprogram.* 40(2-3):117–34.
16. Kvols, K., and S. Blaabjerg. 1992. Bounds and approximations for the periodic on/off queue withapplications to ATM traffic control. *11th Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM'92*. Florence, Italy: IEEE.
17. Carcenac, F., F. Boniol, and Z. Mammeri. 2004. Verification of an avionic system using timed model checking. *1st Symposium (International) on Leveraging Applications of Formal Methods, ISoLA*. Paphos, Cyprus.

18. Carcenac, F. 2005. Une méthode d'abstraction pour la vérification des systèmes embarqués distribués: Application p l'avionique. Ph.D. Thesis. École Nationale Supérieure de l'Aéronautique et de l'Espace.
19. Carcenac, F., and F. Boniol. 2006. A formal framework for verifying distributed embedded systems based on abstraction methods. *j-INT-J-SOFTW-TOOLS-TECHNOL-TRANSFER* 8(6):471–84.
20. Amnell, T., G. Behrmann, J. Bengtsson, P. R. D'Argenio, A. David, A. Fehnker, T. Hune, B. Jeannet, K. G. Larsen, M. O. Müller, P. Pettersson, C. Weise, and W. Yi. 2001. UPPAAL: Now, next, and future. In: *Modeling and verification of parallel processes*. Eds. F. Cassez, C. Jard, B. Rozoy, and M. D. Ryan. Lecture notes in computer science ser. Springer-Verlag New York, Inc. 2067:99–124.
21. Goossens, J. 2003. Scheduling of offset free systems. *Real-Time Syst.* 24(2):239–58.
22. Choquet-Geniet, A., E. Grolleau. 2004. Minimal schedulability interval for real-time systemes of periodic tasks with offsets. In: *Theor. Comput. Sci.* Elsevier Science Publs. Ltd.