Approximating WCRT through the aggregation of short simulations with different initial conditions: application to TSN

Patrick Keller
patrick.keller@uni.lu
University of Luxembourg
Esch-sur-Alzette, Luxembourg

Nicolas Navet
nicolas.navet@uni.lu
University of Luxembourg and Cognifyer
Esch-sur-Alzette, Luxembourg

ABSTRACT

Assessing traversal times is the main concern in the verification of embedded real-time networks. Schedulability analysis, as it provides firm guarantees, is the preferred technique for the designers of critical systems. There are however contexts where it is not economically or technically feasible to develop one, typically when the software and hardware components have not been designed with predictability in mind, e.g. as soon as TCP-based traffic is involved in network communication or when the hardware platform is too complex (e.g. heterogeneous System-on-Chips).

In this paper, we study if it is possible to improve the ability of simulation to observe large traversal times, by running many short simulations with appropriately chosen simulation time and varying initial offsets of the stations on the network. The de-facto-standard approach to assess maximal traversal times is to run a single long simulation with synchronized node start offsets and to use randomized node clock drifts inside an acceptable range. This approach is known to yield high traversal times but is not parallelizable. We propose an alternative approach consisting in splitting the simulation time over multiple shorter simulations with, optionally, randomized node start offsets.

We evaluate the optimization potential of this simple approach on several realistic network configurations by comparing long simulations to aggregated short simulations, with and without synchronized node start offsets. We observe, considering all flows, that this allows a median improvement of up to 21.3% in terms of maximum traversal time observed, for the same simulation time budget. Additional randomization of the node start offsets showed further improvements of up to 4.8% in our experiments. Results from this line of work can be used to estimate the pessimism of schedulability analyses and verify systems for which no analysis is available.

CCS CONCEPTS

• **Computer systems organization** → **Embedded systems**; **Real-time systems**; • **Networks** → **Network simulations**; *Network performance analysis*; • **Computing methodologies** → Parallel computing methodologies.

RTNS '22, June 7–8, 2022, Paris, France

ACM ISBN 978-1-4503-9650-9/22/06.
https://doi.org/10.1145/3534879.3534886

KEYWORDS

Worst-Case Traversal Times, Worst-Case Response Times, Ethernet, Time-Sensitive Networking, Network Simulation, Schedulability Analysis Pessimism, Simulation Initial Conditions.

ACM Reference Format:
Patrick Keller and Nicolas Navet. 2022. Approximating WCRT through the aggregation of short simulations with different initial conditions: application to TSN. In *Proceedings of the 30th International Conference on Real-Time Networks and Systems (RTNS '22), June 7–8, 2022, Paris, France.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3534879.3534886

1 INTRODUCTION

Context of the study. Timing analysis is a crucial activity in the design of critical systems as it allows to properly dimension the system's resources and it provides evidence that the timing constraints, such as deadlines, jitters, throughput and synchronization are met. Under-provisioning could lead to critical failures while over-provisioning would lead to poor cost-effectiveness. The two main techniques to assess the timing behavior of a system on models are simulation and worst-case schedulability analysis.

Description of the problem. Schedulability analysis, as it provides firm guarantees on the Worst-Case Response Times (WCRT) of tasks and Worst-Case Traversal Times (WCTT) of packets in a network, is the preferred technique for the designers of critical systems. There are however contexts where the time and complexity needed to develop a tight schedulability analysis is such that it is technically not feasible or not viable considering the time-to-market, skills and cost constraints. This is typically the case as soon as the software and hardware (HW) components have not been designed with predictability in mind. Complex hardware and software platforms running on COTS hypervisors [7] with hierarchical scheduling and complex schedulers fall into that category. In the field of networking, to the best of our knowledge, TCP-based protocols are not amenable to WCTT analysis, which is a problem as their use is increasingly considered in software-defined systems like cars. Similar concerns arise with Ethernet TSN Quality-of-Service (QoS) mechanisms, which can be used in a combined manner. For instance, in TSN it is possible to use priorities, frame preemption, several traffic shaping policies obeying different principles and time-triggered transmissions at the same time, on a system-wide basis or locally only on some egress ports. To make things more complex, network devices may come from different manufacturers (network interfaces versus bridges), and their implementation of TSN might differ in terms of HW capabilities and, possibly, implementation of the QoS mechanisms. Sometimes even mechanisms that are non-standardised yet, like cut-through forwarding, are implemented in different ways in network devices.

Objective of the study. In this work, we investigate how to set simulation parameters, so as to increase the likelihood to observe scenarios leading to large response times without exploiting any expert knowledge about the use case at hand. The approach taken is to aggregate simulation results over many short simulations rather than running a single long one. We assess two important parameters, namely the node start offsets and simulation time, and how they affect the observable traversal times. This paper aims to lay the basis for improved simulation-based approaches tailored to approaching the worst-case trajectories of the system. Concretely, we investigate empirically if the aggregation of short network simulations with well-chosen initial conditions is an effective approach that yields potential for more efficient worst-case verification. In addition, the aggregation of short simulations constitutes a simple approach to parallelizing network simulation, and thus to speed-up the verification time by a factor equal to the number of processors available.

These objectives can be formulated as the following research questions:

RQ1. Does simple aggregation of shorter simulations with synchronized node start offsets yield optimization potential in terms of observing large frame latencies?
RQ2. Does randomization of node start offsets in aggregated short simulations enable optimizations beyond the synchronized case?
RQ3. Is the short simulation duration an important factor in the efficiency of the aggregation approach?

Limitations of existing solutions. The verification of real-time systems is generally performed via two different methods - simulation and formal verification with schedulability analysis. Both these methods have advantages and disadvantages. A schedulability analysis can be difficult and time-consuming to develop, and may not even be available for some systems. In addition, analysis is prone to pessimism (see [3] for an evaluation in the case of AFDX), which possibly leads to overdimensioning of hardware resources. Occasionally, analyses may be flawed [8]. A bigger risk in our view is that the analysis is applied without the system meeting all underlying assumptions (e.g., traffic model, hardware and software behavior more complex than accounted for). Simulation on the other hand does not guarantee that the worst-case scenarios have been observed, which is a severe drawback in the design of critical systems. Another drawback of simulation is that it tends to be more time-consuming than schedulability analysis, especially when the quantities of interest are high quantiles or maximum values, which is typically what is of interest for response times and memory usage.

Contributions. Our experiments give empirical evidence that aggregating multiple short simulations outperforms a single long simulation in most cases in terms of observing large response times. We also show on different realistic test-cases that appropriately selecting the simulation hyperparameters allows, in certain conditions, to significantly increase the efficiency of simulation. Our empirical findings open new paths to further exploit the potential of short simulations with optimized hyperparameters. Finally, our approach allows for significant speedup of usually long-running simulations through simple and scalable parallelization.

Organisation of the paper. The remainder of this paper is organized as follows. In Section 2, we introduce core concepts that are relevant for the positioning and understanding of this work. In Section 3 we define the system model, formalize the problem and describe the best known solution. Section 4 explains the experiment design and evaluation approach, followed by a description of the experimental setup and a presentation of the experiments' results. Section 5 summarizes the related work. Finally, Section 6 concludes with a summary and Section 7 discusses the limitations of this work and the future research directions.

2 BACKGROUND

Network simulation. Discrete Event Simulation is widely used for networking simulation. Conceptually, at each step n of the simulation, the system is fully characterized by a state Sn and the set of rules to change from state n to $n+1$: $S_{n+1} = F(S_n)$ is defined by the simulation model. The evolution of the system depends on this set of rules and the sequence of values provided by the random generator. During the simulation, various statistics on the underlying system can be collected, such as packet traversal times, memory usage and link loads. Discrete event simulation by its nature is not parallelizable as each state depends on the previous iteration. To obtain representative and statistically significant results, it is necessary to run very long simulations if higher quantiles should be estimated. For instance, evaluating the $(1 - 10^{-5})$ quantile if the frame latencies with 10 observations, which will be too little for satisfactory confidence intervals in most cases, requires 10^6 observations, the equivalent to 277 hours of functioning time for a 100ms frame. This might require days of computation for complex communication architectures made up of several networks transmitting hundreds of streams like in cars or airplanes.

Since real communication systems are subject to non-deterministic phenomena, such as jitters, clock drifts and variable switching delays, network simulation includes randomness to account for them. The simulation results by consequence are of stochastic nature. As a result, running the same simulation model with different random seeds can yield significantly different observed maximum traversal times. Section 4 outlines how experiments were conducted and evaluated to consider this factor.

The modeling of the networks and the simulations are conducted using the network performance evaluation software RTaW-Pegase [1, 4], developed by RealTime-at-Work, which has been used by OEMs and tier1s in the automotive and aerospace domains for more than 10 years. The tool is representative of the state-of-the-art in the industry in terms of timing-accurate simulation, offering control over the simulation parameters and a high-performance simulation engine that runs in constant memory and can execute batches of simulations in parallel. To perform the simulation, the tool generates packet flows that are propagated through the network according to the routing, the selected protocols and their parameters. Over the chosen simulation time, samples comprising the observed communication latencies (along with other statistics such as memory usage, link load, etc.) are collected. The traversal

times depend on many factors such as switching and transmission delays, waiting times at egress ports due to interfering traffic, ...

Targeted Applications. With increasingly complex systems and shorter development cycles in industrial applications, it is important to optimize the resources spent on simulation for faster and more agile validation of the design choices. In that context, in this work, we focus on optimizing the approximation of maximum communication latencies. The results of this research can serve two important use-cases in the field of critical systems:

- Approximation of worst-case timing behavior when formal analysis is not available or excessively pessimistic.
- Empirical determination of upper bounds on the pessimism of existing schedulability analyses.

Competitive nature of network traffic. In TSN networks, flows have to compete for resources with other flows. The competition defines how large traversal times will be, and it is influenced by many parameters. Particularly, it can be observed that when two flows at the same priority level share parts of their path, they cannot experience their respective worst-case traversal times at the same time. This can be illustrated as follows: when reordering these two flows in the queue without changing other factors, the delay of the first packet in the queue would increase while the second packet's delay would decrease. As a consequence, we can derive that it is not possible to find a single simulation trajectory that maximizes traversal times for all flows simultaneously. So the worst-case scenario across all flows usually consists of a set of multiple simulation trajectories.

3 NETWORK MODEL AND PROBLEM DESCRIPTION

This work considers switched Ethernet networks with timing QoS extensions defined in the IEEE Time-Sensitive Networking (TSN) standards (see [16] for a survey and [13] for an example application in helicopters). The topologies of such networks consist of a set of communication switches (or "bridges"), full-duplex links and end-nodes (or "end-systems"), each with a network interface. The network supports unicast and multicast communications between a set of software components distributed over a number of end-nodes. In the following, both "traffic flow" or "traffic stream" refer to a sequence of frames sent to one or several receivers (*i.e.* a multicast connection with n receivers generates n distinct traffic flows).

3.1 Assumptions

In this study, a number of assumptions about the networks considered are made:

- The network topology and the routing of the traffic flows is static, as most often in time-critical systems.
- The traffic flows obey one of the following transmission patterns: periodic, periodic burst (*i.e.* packets making up a segmented message, like a camera frame, are queued at once), sporadic and TFTP [21].
- The size of the packets is fixed or upper bounded.
- The QoS mechanism is either FIFO (a single priority level), static priorities (up to 8 priority levels in TSN) and priorities in combination with Credit Based Shaping (CBS) [11]. The latter mechanism shapes the bursty flows by inserting pauses between successive packets, based on the current "credit" of the traffic class the flow belongs to.
- Node clock drifts [15], that is the departure of the sending node clocks with respect to a perfect clock, remain fixed over time and are set to a realistic randomly generated value that is unique per node (see Section 4.2).

3.2 Problem description

The essential problem we want to tackle in this paper is to find the simulation states leading to the maximal traversal time that can possibly occur for each flow. As flows compete for resources along their path on the network, large traversal times will be observed at times when there is the most competition. Further, due to this competitive nature, there can not be a single simulation state where all flows experience their worst-case traversal time simultaneously. A solution to the problem will thus consist of a set of simulation states as follows:

$$S_{wctt} = \{\forall f \in \mathcal{F}, t_s, t_x \in \mathcal{T} : t_s \mid TT(t_s, f) >= TT(t_x, f)\}$$

where:

- $\mathcal{T}$ is the set of all possible simulation states
- $\mathcal{F}$ is the set of all packet flows in our network
- $TT(t, f)$ is the Traversal Time for flow f for state t

We will explore in this paper a technique, concretely the aggregation of short simulations with randomized node start offsets, to approximate the solution. Our hypothesis is that this approach will allow to more efficiently approximate the worst-case in comparison to the baseline solution presented below.

3.3 Baseline solution

To the best of knowledge, the most effective available approach to approximating worst-case traversal times via simulation is to run a single long simulation, starting from synchronized node start offsets while using randomized node clock drifts. This approach has shown to yield significantly higher traversal times in practical applications in comparison to long simulations with homogeneous clock drifts or non-synchronized node start offsets [17].

The intuitive reasoning behind this approach is that synchronized node start offsets will result in more packets arriving in close succession at the switches, creating a "critical instant". This leads to more congested queues, increasing thus the end-to-end traversal times. Additionally, the randomized clock drifts are necessary to observe different packet orders in the queues in later simulation states, and thus explore more of the simulation state space, possibly yielding higher traversal times. If homogeneous clock drifts for all nodes were used, it would result in smaller hyper periods and thus significantly reduce the simulation state space that can be explored.

4 EXPERIMENTS AND EVALUATION

The goal of our experiments is to investigate statistically how our approach performs in comparison to the baseline solution, namely single long simulations. The experiments are designed such that they yield a statistically robust evaluation of our collected simulation data, which is achieved by resampling our set of short

simulations to reduce the influence of single outliers. The experimental design, which describes the general steps used to carry out a single experiment and evaluation of its results, is explained in the next subsection. Thereafter, we describe the experimental setup which presents the details about the network architectures and configurations used in the experiments, and explain the parameters we control therein.

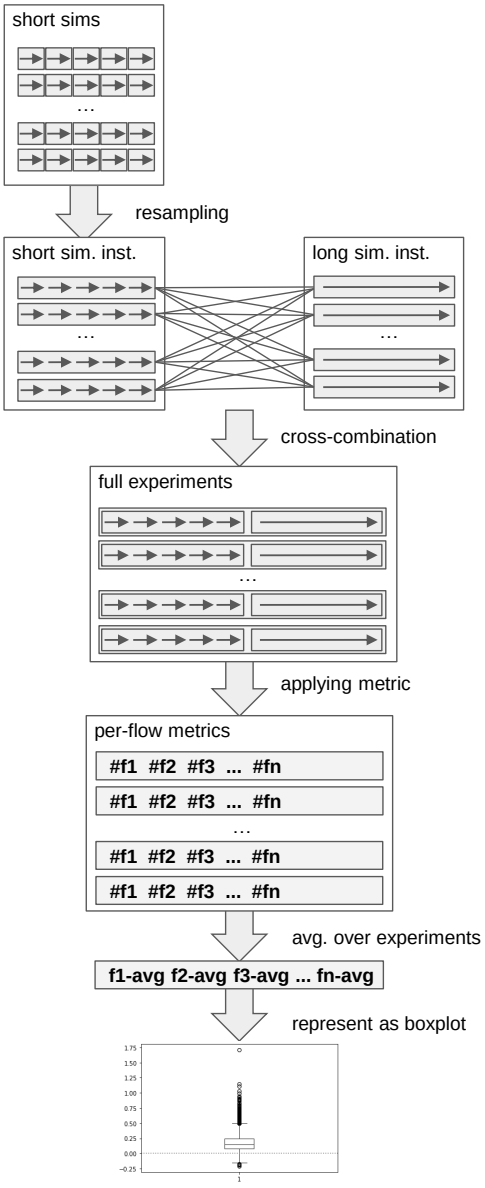

Figure 1: Illustration of the multi-layer experimental design: execution of simulation, bootstrapping of short simulations, formation of experiment instances, application of metric in equation 1, averaging over all experiment instances on a per-flow basis and representation as boxplots.

4.1 Experiments design

Terminology: We call a *full experiment* an experiment that involves all steps based on a number of experiment instances as described below and depicted in Figure 1. Each full experiment consists of a large number of *experiment instances* that are evaluated according to that process. An *experiment instance* is a pair made up of a *short simulation experiment component* and a *long simulation experiment component*. A short simulation experiment component consists of a set of short simulations. A long simulation experiment component consists of a single long simulation. Importantly, for the sake of comparability, each experiment component of an experiment instance, be it short or long, amounts to the same total simulation length d_t. In this paper $d_t = 100h$ of functioning time for all experiment instances. The number of short simulations to perform, denoted by n_s, is dependent on the short simulation duration d_s chosen per experiment instance. We have $n_s = d_t/d_s$ where d_t is, by design, always an integer multiple of d_s.

Experiment flow: We conduct two (or more) experiments per network configuration to answer RQ1 and RQ2, respectively with synchronized and randomized node start offsets. In one of the experiments, we additionally investigate the influence of the simulation duration on the performance. Each experiment is performed by carrying out the following steps:

(1) Dataset collection by running short and long simulations.
(2) Application of bootstrapping (resampling) to generate a number of short simulation experiment components.
(3) Construction of experiment instances by the pairwise combination of long and short simulation experiment components.
(4) Evaluation of the individual experiment instances by applying the metric defined in equation 1.
(5) Per-flow averaging over all evaluated experiment instances.
(6) Representation of flow statistics using box plots.

The first step is *data collection*, where we register the maximal observed traversal times for each flow per simulation.

After data collection, a bootstrapping [9] approach is used to resample the dataset of short simulations in order to create valid short simulation experiment components of functioning time d_t. As each short simulation is independent of the others and each resampling is thus valid. The resampling is performed by randomly selecting short simulations following a uniform distribution. This technique allows us to greatly increase the number of short simulation experiment components, by re-using short simulations several times, and reduce the overall influence of single over- or under-performing short simulations on the final results.

Construction of pairwise combinations between all short and long simulation components is applied for the same purpose of reducing the influence of single outlier experiment instances on the overall result.

A *per-instance evaluation*, using the metric defined in equation 1, is done in order to get a relative measure of the per-flow differences between short and long simulation components as discussed earlier. *Averaging over these metric values* is then performed over all experiment instances as to receive a statistically robust relative per-flow performance value, that is not overly skewed by single outliers.

The final step of *plotting the per-flow averages* as a box plot serves the purpose of enabling human understanding of the general performance of both approaches and allows for easier comparison between experiments on the same network configuration. This would not be possible by considering hundreds or thousands of individual flow values simultaneously.

Evaluation problem properties: Due to the stochastic nature of network simulation, each simulation run will yield varying traversal time amplitudes for a certain flow. In combination with the high number of flows, it is thus not useful to investigate each flow individually. Further, amplitudes of traversal time can vary significantly across different flows, which is a direct consequence of the scheduling mechanisms used to meet the timing constraints. As an example, the absolute observed differences in traversal time of a flow with a 5ms deadline is likely to be insignificant in comparison to the variance of observed traversal times of a flow with a 200ms deadline.

Even though our aim is to maximize the traversal times, comparing long and short simulation experiment components directly on a one-to-one basis is not a statistically robust approach as values observed per flow can vary significantly across simulations, particularly on shorter ones. Hence, we must evaluate the relative performance across a large number of experiment instances based on the distribution of their relative per-flow differences.

Evaluation approach: To address these factors, we adopt the metric defined by equation 1, which allows to compare the results on a statistical and relative basis. To compare the results of a long and a short simulation experiment component pair, we can compute the relative difference between the traversal times of the same flow for each component, while using the long simulation component as reference value, as follows:

$$\forall f \in \mathcal{F} : \delta_{TT}(f) = (R_s(f) - R_l(f))/R_l(f) \tag{1}$$

Where $R_s(f)$ and $R_l(f)$ represent the maximal observed traversal time for flow f during the short respectively long simulation experiment component. This metric can be interpreted as follows: e.g. a value of 0.1 means the aggregated short simulations yielded a 10% larger maximum traversal time for that flow than the long simulation, and vice-versa for a value of -0.1.

As for the presentation of the evaluation results, each boxplot represents a single experiment. We choose to plot experiments based on the same network configuration on the same graph to allow an easier comparison of the different experiment variants (*i.e.*, synchronized/randomized node start offsets and different short simulation durations) to each other. Furthermore, experiments on a specific network configuration - and thus the box plots of the same graph - are based on the exact same data for the long simulation to allow a more direct comparison across experiments. To give an example, the long simulation data is reused for experiments EX1 (synchronous node start offsets) and EX2 (randomized node start offsets), as the long simulation component node start offsets are always synchronized. This choice was taken as our baseline solution is known to perform best with synchronized node start offsets as explained in Section 3.

4.2 Experimental setup

The previous section explained how a single full experiment is performed. In this section, the set of concrete experiments conducted is presented. Three different network architectures are used in combination with different traffic configurations. The topologies used, include a redundant topology representative of a space launcher [19] (see Figure 2), the topology of an automotive network used in [14] (see Figure 4) and a topology inspired from medium-size AFDX avionics networks [3] (see Figure 3)). These three architectures have vastly different properties and are representative of the mission-critical systems targeted in this paper. A more detailed overview on the topology properties is provided in Table 1.

A total of 12 full experiments are conducted on the networks introduced above. Table 2 gives an overview on the experiments and their specific parameter settings. For each full experiment an identifier in the format **EX*k*** is assigned for easy reference.

In all of the experiments the total simulation time for each experiment instance amounts to 100 hours. Each full experiment is made up of 16 long simulation experiment components cross-combined with 100 short simulation experiment components after bootstrapping, amounting to a total of 1600 experiment instances per full experiment. The resampling is performed in order to increase the short simulation experiment component dataset and more accurately approximate the traversal time distribution produced by our short simulations.

Across full experiments, synchronized and randomized node start offset variants of the same network configuration have been evaluated using the same long simulation data. This allows for a more direct comparison in the context of RQ2, namely evaluating the improvements with randomized node start offsets. We use the metric defined in equation 1 to convert our absolute traversal times into relative values, using the long simulations as a reference point. Finally, we average these values per flow and construct the boxplots that allow to argue about the general statistical performance of the approach.

While the general system parameters remain constant over a full experiment, there are three variables that are adjusted and their effects evaluated: simulation time, node start offsets and simulation random seeds. The anticipated effects of these parameters are as follows:

- **Simulation time:** As the experiments show, simulation time plays a crucial role in the performance of aggregated short simulations. Importantly, the short simulation time also defines the limit to the parallelism that can be achieved compared to a long simulation.
- **Node start offsets:** They are a natural variable to explore as they play a key role in the baseline solution. In the following, their values are randomized to explore their optimization potential.
- **Simulation random seeds:** Changing random seeds allows covering more of the simulation result space, as running the simulation repeatedly with equal random seeds would yield the exact same results.

It should be mentioned that these three parameters do not alter the properties of the network configurations in any way. This is not the case for the node clock drifts, an important parameter, that changes

Topology	# nodes	# switches	# links	# flows	# flow receivers	QoS mecha.	Flow types
Space	18	18	24	100	985	4 Priorities	21 Command and Control (periodic) 78 Telemetry (periodic) 1 Video (periodic burst)
AFDX	52	4	57	453	3214	5 Priorities FIFO	453 Uncategorized (sporadic)
Automotive	14	5	18	58	70	4 Priorities + CBS 4 Priorities	19 Command and Control (periodic) 10 Audio (periodic) 11 Video (periodic burst) 6 Best Effort (periodic) 4 TFTP (each ACK+DAT+RRQ)

Table 1: Characteristics of the network configurations.

ID	Topology	QoS mecha.	Sim duration	NSO range
EX1	Space	Priorities	30s	0.0ms
EX2			30s	[0.0, 0.1]ms
EX3	AFDX	FIFO	30s	0.0ms
EX4			30s	[0.0, 0.1]ms
EX5		Priorities	30s	0.0ms
EX6			30s	[0.0, 0.1]ms
EX7	Automotive	Prio. + CBS	30s	0.0ms
EX8			30s	[0.0, 0.1]ms
EX9		Priorities	30s	0.0ms
EX10			30s	[0.0, 0.1]ms
EX11			120s	0.0ms
EX12			120s	[0.0, 0.1]ms

Table 2: Full experiment parameter settings. QoS stands for Quality of Service. FIFO means all frames are scheduled at the same priority level, while priorities means that several priority levels are used (see Table 1). CBS stands for the Credit Based Shaping. NSO stands for Node Start Offsets.

the traffic characteristics by speeding up or slowing down the clock that drives frame transmissions. While in the real world we cannot generally control the node clock drifts, we select randomized but constant clock drifts for each node across all simulations, as dictated by the baseline solution. We select the randomization interval as $[0.0, 0.02]\%$, *i.e.*, less than 200PPM, which is sometimes used as an accepted limit in the automotive industry [15] and is small enough to not drastically change the traffic characteristics. The simulation random seeds were randomized in no specific way as the effects of a seed are not predictable. The applied randomization interval for the node start offsets for the randomized full experiments is [0.0, 0.1]ms, using a uniform distribution. This start offset interval was chosen to explore the close proximity of the synchronous case. The exact effects of the node start offset range are not explored in this work and may yield potential for further optimizations beyond this range. We leave the exploration of this parameter as a future work.

4.3 Results

Applying the methodology introduced in Section 4.1, a total of 5 boxplot charts, one per QoS mechanism per architecture, are produced to summarize the experiments' results.

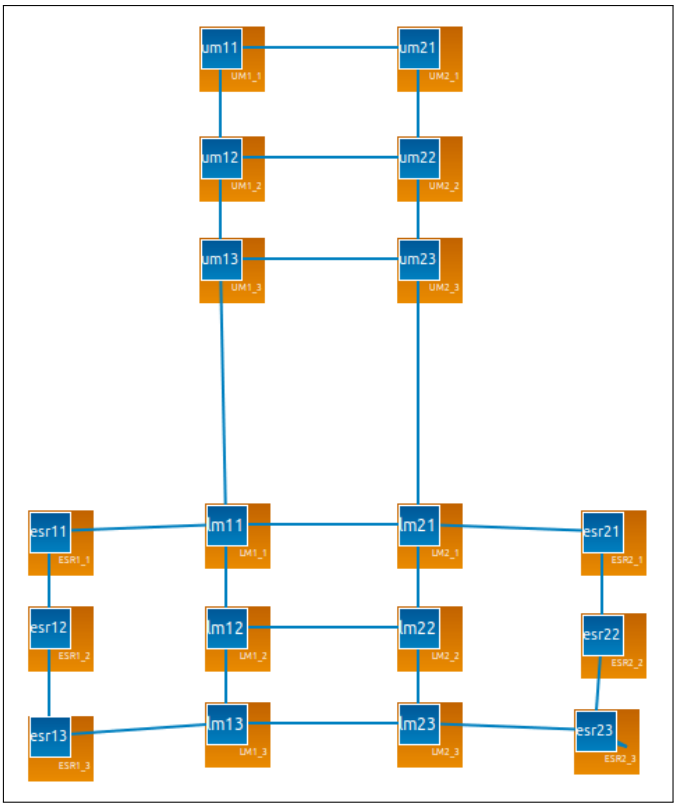

Figure 2: Space launcher topology.

Results - Space launcher with priorities. Results of experiments EX1 and EX2 are shown in Figure 5. The two boxplots respectively compare the performance of short simulations with long simulations in the synchronized case (*i.e.*, all node start offsets set to 0) and in the non-synchronized case (*i.e.*, node start offsets randomly assigned in $[0.0, 0.1]ms$). Indeed, the bulk of the two boxes is located in the positive range. More precisely, over the average of all experiment instances, short simulations outperform long simulations: e.g. the median value of the observed WCTTs is 21.3% larger with short simulations than with long simulations in the synchronized case and 25.8% in the non-synchronized case. In rare instances (see the

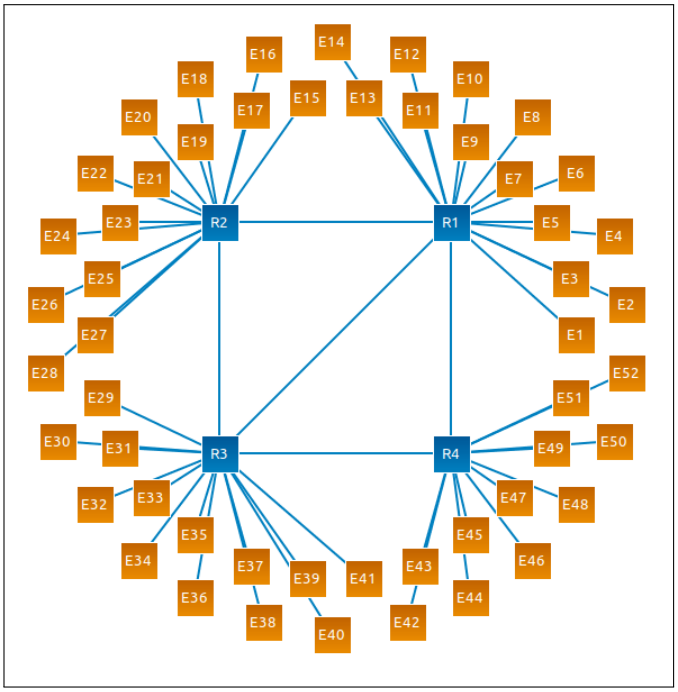

Figure 3: Avionics system topology.

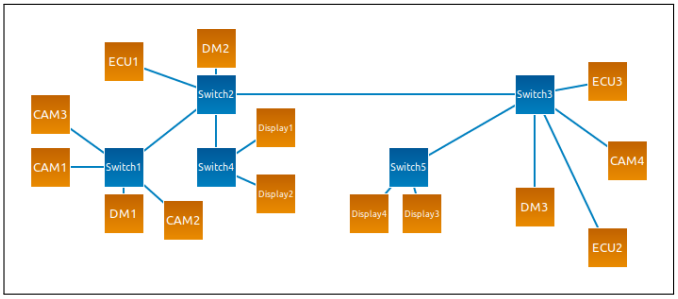

Figure 4: Automotive system topology.

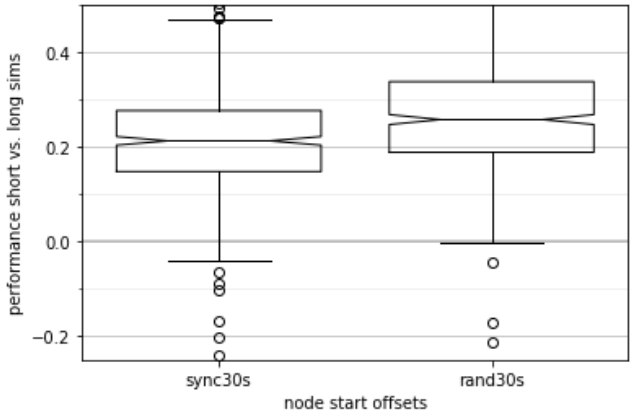

Figure 5: Space launcher topology results with priorities (*EX1* and *EX2*). The y-axis shows the relative difference (as decimal fraction) between the max. WCTT observed with short and long simulations over all flows (averaged over all experiment instances). Short simulations here clearly outperform long simulation in both, synchronized (median 21.3%) and non-synchronized (median 25.8%), cases.

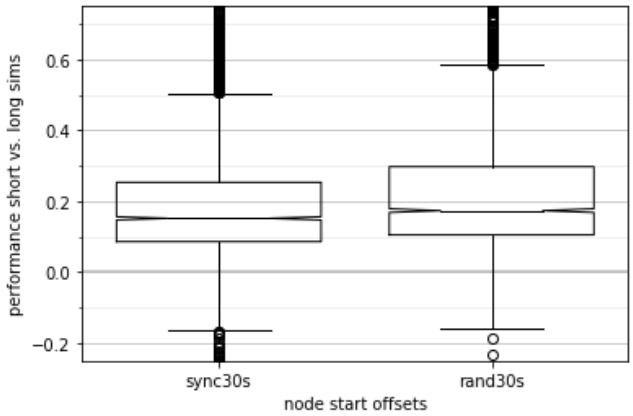

Figure 6: Avionics topology results with FIFO (*EX3* and *EX4*. As seen by the position of the boxes (*i.e.*, interquartile range), short simulations perform better than long simulations.

outliers at the bottom), it happens however that long simulations perform better, which is expressed as the negative portion on the graphs. This is the case for 3.0% of the flows in the synchronized case and 1.2% of the flows in the non-synchronized case.

Results - Avionics network with FIFO. The results of experiments EX3 and EX4, visualized in Figure 6, show similar behavior as for the space topology. Here the median improvement for the synchronous case is 15.2%, and the additional median improvement with non-synchronized node start offsets is about 2.2%. The general skew of the data in favor of the randomized case has again a similar amplitude of up to 5%. If, overall, short simulations are beneficial, for 6.3% of the flows long simulations performed better on average in the synchronous case and 2.1% in the non-synchronous case.

Results - Avionics network with priorities. The results of experiments EX5 and EX6 in Figure 7 show again a general similar behavior although the gain is reduced here. Indeed, the median value of the observed WCTTs is 10.4% larger with short simulations than with long simulations in the synchronized case and 14.9% in the non-synchronized case. Also, for the synchronized case, 7.7% of the flows performed better with long simulations and 2.9% in the non-synchronized case.

Results - Automotive network with priorities and CBS. Figure 8 shows the results of experiments EX7 and EX8. Here the median performance increases by 4.0% for the synchronous case and 8.8% for the non-synchronized case. It should be noted that, for that specific network configuration, about 42.9% of the flows performed better with long simulations in the synchronized case, against 27.1% in the non-synchronized case.

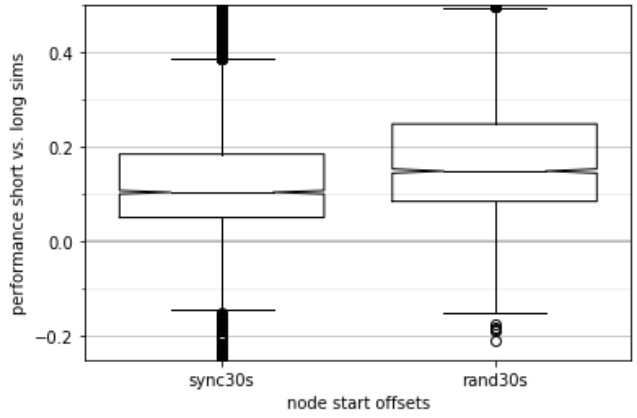

Figure 7: Avionics topology results with priorities (*EX5* and *EX6*). Short simulations here again outperform long simulations as the two medians are above a 10% gain.

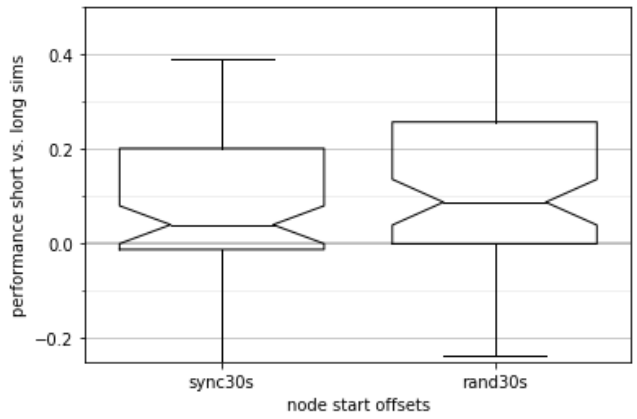

Figure 8: Automotive topology results with priorities and CBS (*EX7* and *EX8*). Aggregation of short simulations performs better for about 57.1% (synchronized case) and 72.9% (non-synchronized case) of the flows.

Results - Automotive network with only priorities. Figure 9 shows the results of experiments EX9, EX11, EX10 and EX12. EX9 is the first experiment to yield overall negative results with the median value with a loss of about -1.6% and a bulk of the distribution in the negative range. Increasing the simulation time (boxplots #2 and #4) and randomizing the node start offsets (boxplots #3 and #4) allows to change this trend into the positive direction. However, the gain remains modest whatever the experimental settings explored: e.g. a 0.4% improvement of the median with 120 seconds simulations in the non-synchronized case (boxplot #4). It should be noted that the number of flows that perform better with long simulation on average decreases from 58.6% to 45.7% for EX9 (synchronized, 30 seconds) and EX10 (non-synchronized, 30 seconds), respectively. For the 120 second variant of the experiment, these values decrease from 50.0% to 44.2% for EX11 (synchronized, 120s) and EX12 (non-synchronized, 120s), respectively.

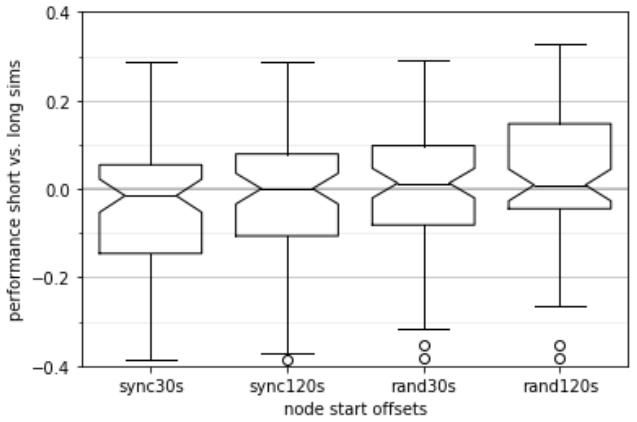

Figure 9: Automotive topology results with only priorities (*EX9, EX11, EX10* and *EX12*). The difference between short and long simulations is small here, as all medians are around zero. A small increase in the relative performance of short simulations can be observed across the four experiments.

5 RELATED WORK

In this section, we discuss previous works that belong to two main research areas related to the objectives of this paper. These areas are network simulation parallelization and assessment of the pessimism of WCRT analysis. To the best of knowledge, no existing work covers directly the optimization of simulation for WCRT approximation in TSN Ethernet networks.

Approximation of WCRT via simulation. A closely related work is [20] by Samii et al.. The objectives of their work is to approximate the WCRT of tasks as efficiently as possible. They apply their proposals to applications distributed over CAN and FlexRay networks. To that end, they propose a method to reduce the search space combined with a genetic-algorithm-based exploration strategy. This approach is different from ours as it relies on extensive domain-specific knowledge, and the system model is different from the one in this paper. For instance, they do not account for clock drifts and they target different networking technologies.

Parallelization of network simulation. There are two main types of parallelization for networking simulation that are discussed in the literature: spatial and temporal parallelization. Also, hybrid approaches, that combine both approaches, exist.

Spatial parallelization aims at subdividing the network and running simulation of the sub-networks in parallel before aggregating the results. An early notable work in this area by Riley et al. [18] proposes a generic framework for spatial parallelization, including its integration into the *ns* simulation software package [5].

Temporal parallelization. The approach proposed in our paper belongs to this latter category. In temporal parallelization, the aim is to subdivide the simulation in the time domain, running the simulation splits from certain temporally-spaced simulation states. A precise temporal parallelization is hard to achieve because, in discrete event simulation, each simulation step depends on the previous. Thus, as Wang et al. [23] describe, the starting conditions

for the later simulation states have to be predicted or approximated, which may produce results that are different from those obtained with a single long simulation.

Hybrid parallelization. One of the more closely related works belongs into the hybrid, or temporal-spatial, category. In their work, Gupta et al. [10] propose a hybrid approach based on the combination of spatial and temporal parallelization. The objective of their work is to maximize the parallelism, with the objective of processing very large networks on heavily parallel infrastructures. A main difference with our work is that our focus is on optimizing the observed WCTT using constant computational resources, not on maximizing parallelism.

Pessimism of schedulability analysis. While by construction a WCRT analysis is generally pessimistic, the extent of the pessimism is unknown in most cases. Additionally, the pessimism may differ significantly depending on the input data and the analysis, as some have been more optimized in terms of accuracy than others.

Important and useful results on that topic were proposed in [2] where the authors develop an analysis to determine a lower bound on the WCTT. This analysis was then used in [3] to estimate the accuracy of the network-calculus-based analysis for AFDX networks that is implemented in RTaW-Pegase. Unfortunately, deriving lower-bound on the WCTT requires extensive domain expertise, and, to the best of our knowledge, similar results do not exist for TSN QoS mechanisms and for traffic models beyond the sporadic streams used in AFDX.

Navet et al. [17] explain the limitations of WCRT analysis in the context of complex industrial networking architectures, including pessimism. They also present simulation experiments that suggest the efficiency of clock drifts and node offsets.

Charara et al. [6] conduct an experiment to evaluate the pessimism of their network calculus analysis approach using simulation in the context of an AFDX network. They point out the shortcomings of simulation for precise evaluation of the pessimism, which this work aims to improve.

6 CONCLUSIONS

We here summarize the outcomes of the experiments that are relevant to the research questions stated in Section 1.

RQ1: The hypothesis behind RQ1 is that aggregating short simulations can yield improvements over a single long simulation in terms of the maximum latencies observed. Our experiments give empirical evidence that supports this hypothesis. Indeed the median of the maximum latencies[1], in the experiments without node offsets, increases from 1.6% up to 21.3% (with an average of 10.5%).

We conclude that aggregation of short simulations can be a valuable alternative to long single simulations, as it shows significant increases in efficiency. Furthermore, it offers a relevant and simple approach to parallelization as it transforms the usually non-parallelizable discrete event simulation into a "embarrassingly parallel" problem, allowing to make use of highly parallel computing facilities such as HPC platforms.

[1]We disregard EX9 whose simulation time was insufficient for the complexity of the systems, which led us to perform EX11 (see also the conclusions on RQ3 in Section 6 and Section 7).

RQ2: The hypothesis is that aggregation of short simulations combined with randomization of node start offsets can yield performance improvements beyond aggregation with synchronized node start offsets. Our experiments conclusively support this hypothesis, as the medians increase from 0.82% up to 4.8% (with an average of 3.27%) in comparison to the experiments without randomized node start offsets.

We conclude that modifying node start offsets yields substantial additional optimization potential beyond pure aggregation of short simulations with synchronous node start offsets. This approach may yield even higher benefits when combined with advanced optimization techniques instead of random sampling.

RQ3: The last research question is based on the hypothesis that adjusting the short simulation time is an important factor to enable improvements in efficiency with the aggregation of short simulations. As the experiment results show, a short simulation duration of 30 seconds can already yield significant improvements in most cases. Experiments EX9 and EX11 further show that there is no single optimal simulation time for all systems, and that choosing an adequate simulation time is important. Choosing the optimal simulation time is a question that should be addressed in further research.

7 DISCUSSION AND FUTURE WORK

Short simulations have shown in EX9 mediocre performance on the automotive architecture using only priorities. Upon closer inspection, this configuration possesses many low-priority flows with very large traversal times. It turns out that the observed maximum traversal times of low-priority flows are usually less with shorter simulations. Indeed, as can be seen in Figure 10, for the automotive network, the flows that lose the most with short simulations (in terms of maximum observed latencies) all belong to the two lowest priority classes. On the other hand, as seen in Figure 11, the flows that benefit the most from short simulations are the higher priority flows with much shorter traversal times.

This phenomenon suggests that there might not be a single optimal simulation duration for short simulations that maximize the performance gain across all flows, but rather a per-priority or per-flow based optimal simulation duration. Additional research on using a mix of different simulation times, and how to set them, might enable further improvements.

The experiments in this work suggest that the combination and careful adjustment of the explored parameters - node start offsets, offset range and simulation duration - offer further significant optimization potential and merit further investigation. In that regard, optimization techniques, such as simulated annealing or genetic algorithms, could be used to set the parameter values and increase the performance.

This work bears a number of limitations:

- **Generalizability**: Our findings might be limited to the specific types of architectures and applications considered in the paper. Pre-tests run on each unseen network configuration could help determine the suitability of this approach and, possibly, adequate parameter values.
- **Applicability:** Simulation engines have a certain initialization overhead, which may render the use of very short

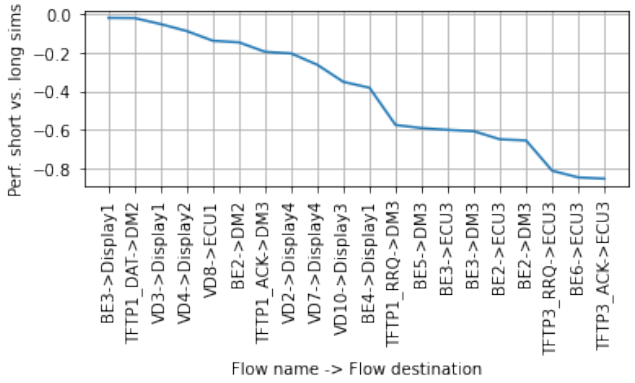

Figure 10: Relative performance details for bottom 20 flows on automotive topology with only priorities in the non-synchronized case and 120s simulation time (EX12). It can be observed from the flow names that Video (VD, second-lowest priority), Best Effort (BE, lowest priority) and TFTP (lowest priority) flows performed the worst with aggregated short simulations.

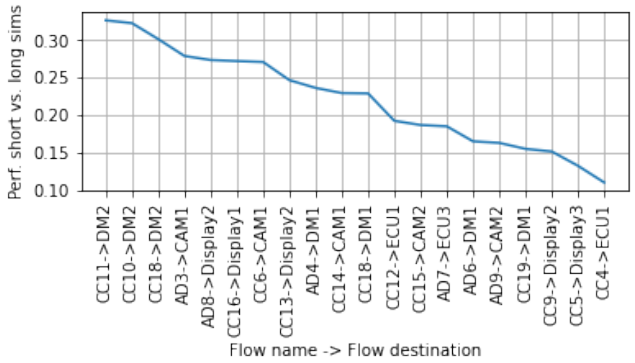

Figure 11: Relative performance details for top 20 flows on automotive topology with only priorities in the non-synchronized case and 120s simulation time (EX12). It can be observed from the flow names that Command and Control (CC, highest priority) and Audio (AD, second-highest priority) flows gained the most performance with aggregated short simulations.

simulations unpractical. It should be noted that this overhead depends on the size of the system under study. As this initialization overhead depends on the specific implementation of the simulator, it was not accounted for in this study as the aim was to compare the effects on equal amounts of simulation time. However, a practitioner needs to set the short simulation duration in such a way that the initialization cost does not outweigh the performance gains.

7.1 Future work

As shown in this work, splitting simulations into many shorter simulations combined with randomized node start offsets can be a viable optimization approach without even tapping into the potential of more advanced optimization techniques.

A potential avenue for future research would thus be the application of advanced optimization techniques to adjust node start offsets in a controlled manner that might further increase the observed traversal times, or optimizing the start offset range and other configuration parameters.

As generalizability is a concern, an important area to explore could be to investigate how specific configuration parameters affect the effectiveness of the approach. These parameters include simulation duration, network topology, flow number, flow offsets, network load, frame size, routing, additional QoS mechanisms, and many more.

Furthermore, as we have discussed, there seem to be complex relationships between several key parameters, such as node start offsets, offset range and short simulation duration, that are not understood at this time. This suggests the application of machine learning, which has proven its ability to capture relations between variables in an efficient and effective manner beyond human understanding. As shown in [12], deep learning techniques are able to learn network structures and predict certain properties, such as feasibility, with high accuracy and reduced computation cost. In this context, a potentially interesting avenue to pursue could be training a reinforcement learning algorithm to optimize the simulation parameters in order to achieve higher traversal times. A possible addition could be the use of transfer learning ([22],[24]) to apply the benefits to other network configurations not seen during training.

If such algorithms prove to be successful to significantly increase the efficiency of WCTT approximations with simulation, it would open the door to a new class of "push-button" verification techniques, relying on simulation and machine learning, and little domain expertise, offering a new tradeoff between simulation and worst-case schedulability analysis.

REFERENCES

[1] RealTime at Work. 2022. *RTaW-Pegase Homepage.* https://www.realtimeatwork.com/ Retrieved 2022/03/09.

[2] Henri Bauer, Jean-Luc Scharbarg, and Christian Fraboul. 2010. Improving the worst-case delay analysis of an AFDX network using an optimized trajectory approach. *IEEE Transactions on Industrial informatics* 6, 4 (2010), 521–533.

[3] Marc Boyer, Nicolas Navet, and Marc Fumey. 2012. Experimental assessment of timing verification techniques for AFDX. In *ERTS 2012 - 6th European Congress on Embedded Real Time Software and Systems.* Toulouse, France. https://hal.archives-ouvertes.fr/hal-01345472

[4] Marc Boyer, Nicolas Navet, Xavier Olive, and Eric Thierry. 2010. The PEGASE project: precise and scalable temporal analysis for aerospace communication systems with network calculus. In *International Symposium On Leveraging Applications of Formal Methods, Verification and Validation.* Springer, 122–136.

[5] Gustavo Carneiro. 2010. NS-3: Network simulator 3. In *UTM Lab Meeting April*, Vol. 20. 4–5.

[6] H. Charara, J.-L. Scharbarg, J. Ermont, and C. Fraboul. 2006. Methods for bounding end-to-end delays on an AFDX network. In *18th Euromicro Conference on Real-Time Systems (ECRTS'06).* 10 pp.–202. https://doi.org/10.1109/ECRTS.2006.15

[7] Dakshina Dasari, Arne Hamann, Holger Broede, Michael Pressler, and Dirk Ziegenbein. 2021. Brief Industry Paper: Dissecting the QNX Adaptive Partitioning Scheduler. 477–480. https://doi.org/10.1109/RTAS52030.2021.00056

[8] Robert I Davis, Alan Burns, Reinder J Bril, and Johan J Lukkien. 2007. Controller Area Network (CAN) schedulability analysis: Refuted, revisited and revised. *Real-Time Systems* 35, 3 (2007), 239–272.

[9] Linda W Friedman and Hershey H Friedman. 1995. Analyzing simulation output using the bootstrap method. *Simulation* 64, 2 (1995), 95–100.

[10] Mukta Gupta, Ramakrishnan Durairajan, Meenakshi Syamkumar, Paul Barford, and Joel Sommers. 2015. pfs: Parallelized, flow-based network simulation. In *2015 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*. IEEE, 1–8.

[11] IEEE. 2009. *IEEE Standard for Local and Metropolitan Area Networks – Virtual Bridged Local Area Networks Amendment 12 Forwarding and Queuing Enhancements for Time-Sensitive Streams* (Std 802.1Qav-2009 ed.).

[12] Tieu Long Mai and Nicolas Navet. 2021. Deep learning to predict the feasibility of priority-based Ethernet network configurations. *ACM Transactions on Cyber-Physical Systems (TCPS)* 5, 4 (2021), 1–26.

[13] Cédric Mauclair, Marina Gutiérrez, Jörn Migge, and Nicolas Navet. 2021. Do We Really Need TSN in Next-Generation Helicopters? Insights From a Case-Study. In *2021 IEEE/AIAA 40th Digital Avionics Systems Conference (DASC)*. 1–7. https://doi.org/10.1109/DASC52595.2021.9594349

[14] Jörn Migge, Josetxo Villanueva, Nicolas Navet, and Marc Boyer. 2018. Insights on the Performance and Configuration of AVB and TSN in Automotive Ethernet Networks. In *9th European Congress on Embedded Real Time Software and Systems (ERTS 2018)*. Toulouse, France. https://hal.archives-ouvertes.fr/hal-01746132

[15] Aurélien Monot, Nicolas Navet, Bernard Bavoux, and Cristian Maxim. 2012. Fine-grained Simulation in the Design of Automotive Communication Systems. In *ERTS 2012 - European Congress on Embedded Real Time Software and Systems*. Toulouse, France. https://hal.inria.fr/hal-00767046

[16] A. Nasrallah, A.S. Thyagaturu, Z. Alharbi, C. Wang, X. Shao, M. Reisslein, and H. ElBakoury. 2018. Ultra-low latency (ULL) networks: The IEEE TSN and IETF DetNet standards and related 5G ULL research. *IEEE Communications Surveys & Tutorials* 21, 1 (2018), 88–145.

[17] Nicolas Navet, Jan Seyler, and Jörn Migge. 2016. Timing verification of real-time automotive Ethernet networks: what can we expect from simulation?. In *8th European Congress on Embedded Real Time Software and Systems (ERTS 2016)*. TOULOUSE, France. https://hal.archives-ouvertes.fr/hal-01292312

[18] G.F. Riley, R.M. Fujimoto, and M.H. Ammar. 1999. A generic framework for parallelization of network simulations. In *MASCOTS '99. Proceedings of the Seventh International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*. 128–135. https://doi.org/10.1109/MASCOT.1999.805048

[19] Jérémy Robert, Jean-Philippe Georges, Thierry Divoux, Philippe Miramont, and Badr Rmili. 2015. On the observability in switched Ethernet networks in the next generation of space launchers: problem, challenges and recommendations. In *The Seventh International Conference on Advances in Satellite and Space Communications, SPACOMM 2015*. Barcelone, Spain, 13–18. https://hal.archives-ouvertes.fr/hal-01147643

[20] Soheil Samii, Sergiu Rafiliu, Petru Eles, and Zebo Peng. 2008. A Simulation Methodology for Worst-Case Response Time Estimation of Distributed Real-Time Systems. In *2008 Design, Automation and Test in Europe*. 556–561. https://doi.org/10.1109/DATE.2008.4484735

[21] Dr. Karen R. Sollins. 1992. The TFTP Protocol (Revision 2). RFC 1350. https://doi.org/10.17487/RFC1350

[22] Matthew E Taylor and Peter Stone. 2009. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research* 10, 7 (2009).

[23] Jain J Wang and Marc Abrams. 1993. Determining initial states for time-parallel simulations. In *Proceedings of the seventh workshop on Parallel and distributed simulation*. 19–26.

[24] Zhuangdi Zhu, Kaixiang Lin, and Jiayu Zhou. 2020. Transfer Learning in Deep Reinforcement Learning: A Survey. *CoRR* abs/2009.07888 (2020). arXiv:2009.07888 https://arxiv.org/abs/2009.07888