Pushing the limits of CAN - scheduling frames with offsets provides a major performance boost

Mathieu Grenier, Lionel Havet, Nicolas Navet

Pushing the limits of CAN - scheduling frames with offsets provides a major performance boost

Mathieu Grenier[1], Lionel Havet[2], Nicolas Navet[2]

1: LORIA - Nancy Université, BP 239, 54506 Vandoeuvre, France
2: INRIA - RealTime-at-Work, 615 Rue du Jardin Botanique, 54506 Vandoeuvre-lès-Nancy, France
Contact author: Nicolas Navet (Nicolas.Navet@loria.fr)

Abstract: With the increasing amount of electronics, making best usage of the bandwidth becomes of primary importance in automotive networks. One solution that is being investigated by car manufacturers is to schedule the messages with offsets, which leads to a desynchronization of the message streams. As it will be shown, this "traffic shaping" strategy is very beneficial in terms of worst-case response times. In this chapter, the problem of choosing the best offsets is addressed in the case of Controller Area Network, which is a de-facto standard in the automotive world. Comprehensive experiments shown in this chapter give insight into the fundamental reasons why offsets are efficient, and demonstrate that offsets actually provide a major performance boost in terms of response times. These experimental results suggest that sound offset strategies may extend the lifespan of CAN further, and may defer the introduction of FlexRay and additional CAN networks.

keywords: Controller Area Network, message scheduling, offsets, response times, resource optimization.

1 Introduction

CAN has been and will most likely remain a prominent network in cars for at least two more car generations. One of the issues CAN will have to face is the growth of traffic with the increasing amount of data exchanged between Electronic Control Units (ECUs). A car manufacturer has to make sure that the set of frames will be schedulable, *i.e.* the response time of the frames is kept small enough to ensure that the freshness of the data is still acceptable when used at the receiver end. Clearly here, for most messages, even periodic ones, we are in the realm of soft real-time constraints: a deadline constraint can be occasionally missed without major consequences. However, the issue on CAN is that worst-case response times increase drastically with the load, which may explain why currently the bus utilization is typically kept at low levels (up to 30 or 40%) and why FlexRay is considered as a must for next generation architectures.

Scheduling theory (see, for instance, [2]) tells us that the Worst-Case Response Time (WCRT) for a frame corresponds to the scenario where all higher priority CAN messages are released synchronously. Avoiding this situation, and thus reducing WCRT, can be achieved by scheduling stream of messages with offsets. Precisely, the first instance of a stream of periodic frames is released with a delay, called the offset, in regards to a reference point which is the first time at which the station is ready to transmit. Subsequent frames of the streams are then sent periodically, with the first transmission as time origin. The choice made for the offset values has an influence on the WCRT, and the challenge is to set the offsets in such a way as to minimize the WCRT, which involves spreading the workload over time as much as possible.

Assigning offsets is a problem that has been addressed in [3] and [4] concerning the preemptive scheduling of tasks. It turns out that these solutions are not efficient when applied to the scheduling of messages because automotive message sets have certain specific characteristics (small number of different periods, etc). We propose here an algorithm tailored for automotive CAN networks, which proved to be efficient in experiments conducted on realistic message sets generated with NETCARBENCH [1]. Then, we study the extent to which offsets can be beneficial in terms of schedulability and how they can help to better cope with higher network loads. In addition, the paper provides some insight into the fundamental reasons why offsets are so efficient, which may lead to further improvements.

Section 2 discusses the algorithm we propose to assign offsets. Section 3 describes the experimental setup. The improvements brought by offsets in terms of response times are studied in Section 4. Finally, Section 5 studies the extent to which offsets enable dealing with higher network loads.

2 Offset assignment algorithm

The problem of best choosing the offsets has been shown in [3] to have a complexity that grows exponentially with the periods of the tasks and there is no known optimal

solution that can be used in practical cases. However, there are heuristics with a low complexity, see [3, 4]. In our experiments, if these algorithms are effective for task scheduling, they are not well suited to message scheduling in the automotive context, which motivates the design of a new offset assignment algorithm.

With no additional protocol, there is no global synchronization among the stations in a CAN network, which means that each station possesses its own local time and that the desynchronizations between the streams of frames are local to each station. This implies that there is always the possibility that frames of any two streams coming from distinct stations are released at the same time, inducing delays for some frames. If one wants to implement a global synchronization among the ECUs, in addition to the complexity and the overhead of the clock synchronization algorithm (see, for instance, [7]), the cases of ECU reboots and local clocks that are drifting apart should be dealt with in order to obtain a robust mechanism. This certainly could be done, for instance by building on the experience gathered with TTCAN, but at the expense of some additional complexity in the communication layers.

In this study, the offset assignment algorithm is executed on each station independently. The underlying idea of the algorithm is to distribute the workload as uniformly as possible over time, in order to avoid synchronous releases leading to traffic peaks and thus to large frame response times. More precisely, we will try to schedule the transmissions as far apart as possible.

2.1 Design hypotheses and notations

The algorithm makes the following hypotheses, which are in our experience most often met in the automotive context:

1. There are only a few distinct values for the periods (e.g., 5 to 10). The algorithm proposed in this study has been conceived to take advantage of this property and its efficiency relies on it. The cases with many different period values can be treated efficiently with the algorithms proposed in [3, 4].

2. The time is discrete with a certain granularity: the offsets of the streams, and their periods, are multiples of g, the period of the communication task in charge of issuing the transmission requests to the communication controller. Typically, g is smaller than 5 ms.

Definition 1 *a time instant that is a multiple of g is called a **possible release time**. By definition, the ith possible release time, with $i \in \mathbb{N}^+$, occurs at time $(i-1) \cdot g$.*

2.2 Notations

On station i, the kth stream of frames, denoted by f_k^i, is characterized by the tuple $(C_k^i, D_k^i, T_k^i, O_k^i,)$: each frame produced by the stream has a worst-case transmission time equal to C_k^i, a relative deadline D_k^i (*i.e.*, the frame must be received 10ms after its release) and T_k^i is the transmission period for stream f_k^i. The number of streams on station i is denoted by n^i. For the sake of clarity, it is assumed that there are no jitters on the release times of the frames but they could be taken into account in the analysis. The first release time of f_k^i on station i occurs at O_k^iwhich is the offset of f_k^i. Said differently, O_k^i is the duration between the first instant at which the station is operational and the transmission of the first frame of stream f_k^i. In the following, to keep the notations as simple as possible, the index of the station will not be indicated because the algorithm is executed on each station independently without considering the streams of the other stations.

2.3 Tool support for worst-case response time analysis

At the time of writing, to our best knowledge, there is no result available in the scientific literature that allows to compute response times with offsets on large sets of messages (*i.e.*, more than 50 messages) in reasonable time. However, some commercial products offer this feature, which is actually needed by car manufacturers. In this study, the WCRT of the frames are computed with the software NETCAR-Analyzer, first developed at our institute, then taken over by the company RealTime-at-Work, which implements exact and very fast WCRT on CAN with offsets.

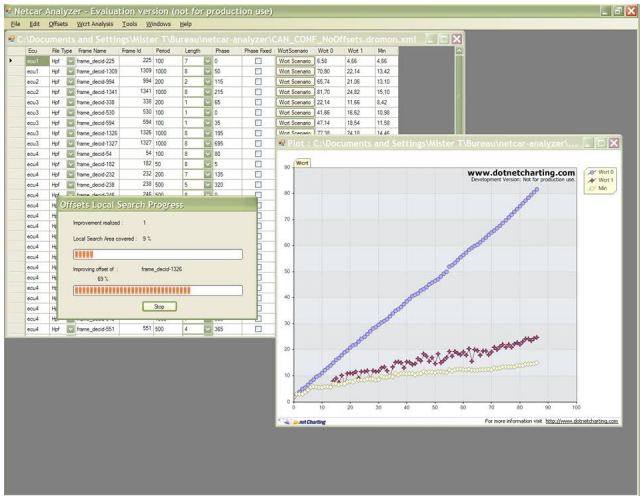


Figure 1: Screenshot of NETCAR-Analyzer during an optimization run. The right-hand graphic shows the response times (by decreasing priority) for different offset configurations. The spreadsheet in the background contains the set of frames, the different offset configurations tested, the corresponding WCRT and certain characteristics of the ECUs, such as the queuing policy at the microcontroller level (e.g., FIFO or prioritized).

NETCAR-Analyzer also includes a set of proprietary offset assignment algorithms, fine-tuned with the experience gained in industrial use, that significantly outperform the

algorithm proposed here, but they cannot be disclosed because of confidentiality. However, as it will be demonstrated, the algorithm shown here is efficient, and it constitutes a sound basis that can be improved and extended according to the user's need. For instance, as permitted by NETCAR-Analyzer, the user may want to optimize the WCRT for only a particular subset of messages, possibly according to a user-defined cost function. Figure 2 shows a screenshot of NETCAR-Analyzer.

2.4 Description of the algorithm

Without loss of generality, the choice of the offset for stream f_k is made in the interval $[0, T_k[$. Indeed, because of the periodic nature of the scheduling (see [3] for more details), an offset $O_k \geq T_k$ is equivalent to $O_k \mod T_k$. Once the initial offset O_k has been decided, all subsequent release times of stream f_k are set: they occur at times $O_k + i \cdot T_k$ with $i \in \mathbb{N}$.
To spread the traffic over time, the offset of each stream f_k is chosen such that the release of its first frame, $f_{k,1}$, is "as far as possible" from other frames already scheduled. This is achieved by 1) identifying the longest interval with the smallest workload and 2) set the offset for f_k in the middle of this interval.

2.4.1 Data structure

Since for each stream f_k the offset is chosen in the interval $[0, T_k[$, we choose to assign the offsets based on an analysis performed over time interval $[0, T_{\max}[$, where $T_{\max} = \max_{1 \leq k \leq n}\{T_k\}$.
The release times of the frames in the interval $[0, T_{\max}[$ are stored in an array R having $T_{\max}/g$ elements where the ith element $R[i]$ is the set of frames released at *possible release time* i *(i.e.*, at time $(i-1) \cdot g$). Table 1 presents the release array R for the frames corresponding to the set of traffic streams $\mathcal{T} = \{f_1, f_2, f_3\}$, where $f_1 = (T_1 = 10, O_1 = 4)$, $f_2 = (20, 8)$ and $f_3 = (20, 18)$ ($T_{\max} = 20$) with a granularity $g = 2$.

time	0	2	4	6	8	10	12	14	16	18
possible release time i	1	2	3	4	5	6	7	8	9	10
$R[i]$ (frames released)			$f_{1,1}$		$f_{2,1}$			$f_{1,2}$		$f_{3,1}$

Table 1: The release array R of the frames corresponding to the set of traffic streams $\mathcal{T} = \{f_1, f_2, f_3\}$ where $f_1 = (T_1 = 10, O_1 = 4)$, $f_2 = (20, 8)$ and $f_3 = (20, 18)$ on the interval $[0, 20[$. The granularity g is equal to 2. The ith element $R[i]$ is the set of frames released at possible release time i. For instance, $R[3] = \{f_{1,1}\}$.

For a given stream f_k, an interval is a set of *adjacent* possible release times.

Definition 2 *For a stream f_k and a time granularity g, the possible release times i and i' are* ***adjacent*** *iff:*

$$\left| \left(i \mod \frac{T_k}{g} \right) - \left(i' \mod \frac{T_k}{g} \right) \right| = 1 .$$

In the above formula, the modulo operators translate the fact that setting the offset of a stream f_k at possible release i is the same as choosing the possible release time $i + u \cdot \frac{T_k}{g}$ with $u \in \mathbb{N}$. Table 2 illustrates this definition with a stream f_1 having a period $T_1 = 10$ where the time granularity g is 2.

time	0	2	4	6	8
possible release time i	1	2	3	4	5
possible release times adjacent to i	{5,2}	{1,3}	{2,4}	{3,5}	{4,1}

Table 2: Possible release times that are adjacent, in the case of stream f_1 having a period equal to 10. For example, possible release times 4 and 1 are adjacent to 5.

This leads to the definition of an interval.

Definition 3 *For a stream f_k, an interval is an ordered set of possible release times where the ith and $(i+1)$th elements are adjacent. The length of this interval is the number of elements in the ordered set.*

For instance, for the stream f_1 (see Table 2), the set $\{4, 5, 1, 2\}$ is an interval of adjacent possible release times. In the algorithm presented here, we consider only the intervals made of possible release times with the same *load.*

Definition 4 *The load of possible release time i is the number of releases scheduled for transmission at i,* i.e., *at clock time $(i-1) \cdot g$.*

For instance, in the example of Table 1, the load of possible release time 3 is 1. We denote by l_k the smallest load in the interval $[0, T_k[$, *the least loaded intervals* only comprise possible release times having a load equal to l_k. For example, in Table 1, l_3 is equal to 0 and interval $\{10, 12\}$ belongs to the set of the least loaded intervals in $[0, 20[$.

2.4.2 Description of the algorithm

We assume that the streams are sorted by increasing value of their period, *i.e.*, $k < h$ implies $T_k \leq T_h$. The algorithm sets iteratively the offsets of streams, from f_1 to f_n. Let us consider that the stream under analysis is f_k.

1. Set offset for f_k such as to maximize the distance between its first release $f_{k,1}$, and the release right before and right after $f_{k,1}$. Concretely:
   (a) Look for l_k in the interval $[0, T_k[$.
   (b) Look for one of the longest least loaded intervals in $[0, T_k[$, where ties are broken arbitrarily. The first (resp. last) possible release time of the interval is noted by B_k (resp. E_k).

(c) Set the offset O_k in the middle of the selected interval, the corresponding possible release time is r_k.

(d) Update the release array R to store the frames of f_k released in the interval $[0, T_{\max}[$:

$$\forall i \in \mathbb{N} \text{ and } r_k + i \cdot \frac{T_k}{g} \leq \frac{T_{\max}}{g}$$

$$\text{do } R\left[r_k + i \cdot \frac{T_k}{g}\right] = R\left[r_k + i \cdot \frac{T_k}{g}\right] \cup f_{k,i+1}$$

A straightforward implementation of the algorithm runs in $O(n \cdot \max_k\{T_k\}/g)$, which, in practice, does not raise any problem even with large sets of messages.

2.4.3 Application of the algorithm

We consider our example where $\mathcal{T} = \{f_1, f_2, f_3\}$ with $f_1 = (T_1 = 10, O_1 = 4)$, $f_2 = (20, 8)$, $f_3 = (20, 18)$ and a time granularity equal to 2. First the algorithm decides the offset for f_1: $l_1 = 0$ (step $1.(a)$), $B_1 = 1$ and $E_1 = 5$ (step $1.(b)$), thus $r_1 = 3$ (step $1.(c)$), which means that the offset of the stream is 4. Then array R is updated: $R[3] = \{f_{1,1}\}$ and $R[8] = \{f_{1,2}\}$ (step $1.(d)$). For stream f_2: $l_2 = 0$, the selected interval is $\{4, 5, 6, 7\}$ thus $B_2 = 4$, $E_2 = 7$ and $r_2 = 5$ with $R[5] = \{f_{2,1}\}$. For stream f_3, $l_3 = 0$, the selected interval is $\{9, 10, 1, 2\}$ thus $B_3 = 9$, $E_3 = 2$ and $r_3 = 10$ with $R[10] = \{f_{3,1}\}$. The results of applying the algorithm are shown in Table 1.

3 Experimental setup

In order to get a precise idea of the real benefits of using offsets, we tried to perform experiments on realistic CAN networks. However, because of confidentiality reasons, very little has been published concerning benchmarks. To the best of our knowledge, the only two publicly available benchmarks are the SAE benchmark [8] and the PSA benchmark [5]. They have been both used numerous times in the literature but they are clearly no more realistic with regard to current in-vehicle networks (see, for instance, [6]).

To overcome the confidentiality issue that prevent us from publishing real sets of messages, we developed NETCARBENCH [1], a software that generates automotive sets of messages according to parameters defined by the user (network load, number of ECUs, distribution of the periods of the frames, etc.). NETCARBENCH is aimed at improving the assessment, the understanding and the comparability of algorithms and tools used in the design of in-vehicle communication networks. To facilitate its diffusion, NETCARBENCH is released under the GPL license and is downloadable at url: *http://www.loria.fr/~nnavet/netcarbench*.

We mostly find three types of CAN networks in a car today: powertrain, body and chassis. In the following, we will consider body and chassis networks which exhibit rather distinct characteristics. In the experiments, except when explicitly stated, the randomly generated networks have an average load equal to 35% (with an interval of variation of 3% around the mean) and the characteristics shown in Table 3. The size of data payload in the frames is uniformly distributed between 1 and 8 bytes. There will be two types of experiments: some will focus on a particular network, while others will involve collecting statistics on a large number of networks (*i.e.*, 1000 in the following). For the former type of experiments, the same body network and the same chassis network have been used throughout all the experiments.

Network	#ECUs	#Messages (stddev)	Bandwidth	Frame periods
Body	15-20	71 (8.5)	125kbps	50ms-2s
Chassis	5-15	58.5 (7.7)	500kbps	10ms-1s

Table 3: Configuration of the networks considered in the experiments. For both body and chassis networks, the average load is 35% and the size of the data payload is drawn at random (uniform law) between 1 and 8. The periods are uniformly chosen in the set $\{50, 100, 200, 500, 1000, 2000\}$ for body networks, and in the set $\{10, 20, 50, 100, 200, 1000\}$ for chassis networks.

In practice, it is often the case that a single station generates a large part of the global network load. For instance, in the body network, there is usually a station that serves as gateway to other networks, and which is responsible for a large fraction of the total load. We model that with a single station that generates about 30% of the total load. In the following, it will be mentioned explicitly when this "load concentration" configuration is used.

4 Benefits of using offsets on worst-case response times

We first evaluate the performance gain with offsets in paragraph 4.1, then, in paragraph 4.2, we provide elements to explain the effectiveness of using offsets.

4.1 WCRT comparison with and without offsets

The main benefit of offsets is the reduction of the WCRT for low priority messages. Figure 2 shows the WCRT of the frames of a typical CAN body network with and without offsets. Two offset strategies are tested: the algorithm presented in Section 2 and a purely random allocation. For this latter strategy, the results in Figure 2 are the average values over 100 random allocations. Also shown in Figure 2 is a lower bound on the WCRT that is provided by NETCAR-Analyzer, this bound can not necessarily be reached in practice but is informative anyway about how

good the offset allocation is. As can be seen, the WCRT is improved for all frames for which a gain is possible. The improvements become more and more pronounced as the priority decreases. For the lowest priority frame of this example, the WCRT with offsets is decreased by 43.2 ms (from 64.8 to 21.6), which represents a reduction of a factor 3, compared to results without offsets. The gain is thus very large.

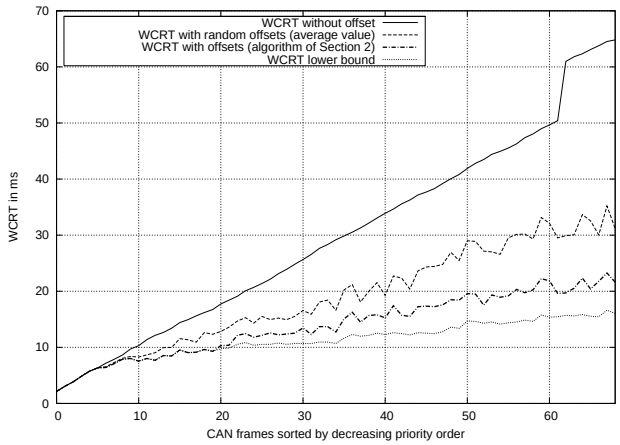
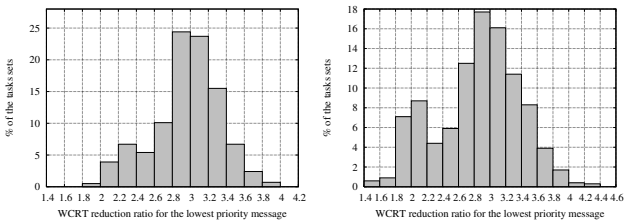


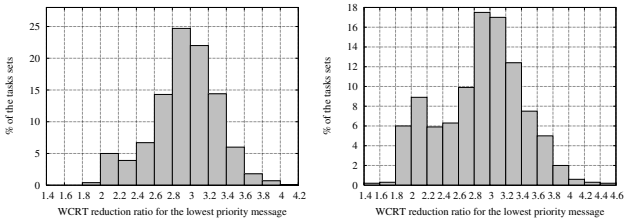
Figure 2: Worst-Case Response Time (WCRT) of the CAN frames with and without offsets for a typical 125kbit/s body network with a network load of 37.6% and 68 messages. The upper curve is the WCRT without offsets, the immediate lower curve is the average value over 100 random offset allocations, the next curve is the WCRT with the algorithm of Section 2. Finally, the lowest curve is a lower bound on the WCRT. The steep increase of the WCRT without offsets at the end can be explained because some high priority frames have a period equal to 50, and two instances of these frames are delaying the lowest priority frames with a WCRT larger than 50ms.

In the next experiments, we evaluate the performance of offset assignments over 1000 random sets of messages. The performance metric is the ratio of WCRT reduction when using offsets with the algorithm of Section 2. We consider body networks and chassis networks, with and without *load concentration*, *i.e.* one station that is more loaded than the others - here this loaded station generates about 30% of the total network load. Figure 3 shows the distribution of the WCRT reduction ratio for the lowest priority frame without load concentration, while Figure 4 presents the case with load concentration.



(a) body network - no load concentration (b) chassis network - no load concentration

Figure 3: Reduction ratio of the WCRT for the lowest priority frames when offsets are used. The distribution is computed over the results obtained on a sample of 1000 random body networks (left-hand graphic) and chassis networks (right-hand graphic). The network load is uniformly distributed over the ECUs (*i.e.*, no concentration). The x-axis is the WCRT reduction ratio (bins of size: 0.2) and the y-axis is the percentage of networks having that level of gain.



(a) body network - with load concentration (b) chassis network - with load concentration

Figure 4: Reduction ratio of the WCRT for the lowest priority frames when offsets are used. Same settings as Figure 3 except that one station alone generates, on average, 30% of the total network load (*i.e.*, load concentration). The x-axis is the WCRT reduction ratio (bins of size: 0.2) and the y-axis is the percentage of networks having that level of gain.

Whatever the experimental condition, the gain is very significant, except for a few outliers out of the 4000 sets of messages that have been considered. This suggests that in practice offsets will most often be very beneficial. It can be observed that the gain is more important for chassis networks. The explanation lies probably in the fact that chassis networks comprise fewer stations than body networks, and thus the desynchronization between streams, which is purely local to the stations, is more efficient. When a single ECU generates a large fraction of the load (*i.e.*, load concentration) the results are very similar to the case where the load is uniformly distributed over the stations, while intuitively they should be better. As suggested by Figure 2, at this level of load, the performance of the shaping algorithm is close to the optimal, which may explain why no difference is observed.

4.2 Explanation of the gain: the network load is better distributed

The evolution of total workload awaiting transmission (or *backlog*) is measured during one second (half of the lcm value here) with and without offsets. More precisely, when there are offsets, we consider the scenario leading to the WCRT for the lowest priority frame. Without offsets, the workload measured is the one corresponding to the synchronous case, *i.e.* the worst-case for all frames in that context. Both workloads are plotted in Figure 5 for a typical body network. It can be immediately noticed that the "peaks" of the workload are much smaller with offsets, which provides a clear-cut explanation about the gains observed in paragraph 4.1. Indeed, the load awaiting transmission directly translates into response times for the lowest priority frames.

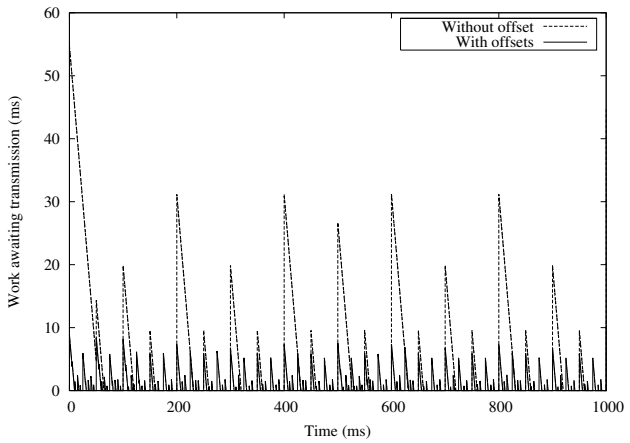


Figure 5: Amount of work awaiting transmission with and without offsets - comparison over 1 second.

The fact that the workload with offsets in Figure 5 is more evenly distributed could lead to us think that there is less workload with offsets, which is actually not the case. Figure 6 corrects this feeling and shows the evolution of the cumulative work arrival function over time with and without offsets for the same network as in Figure 5. The shape of the work arrival function with offsets is much smoother and linear than without offsets, where the "stairs" of the function are larger. This Figure suggests that the algorithm of Section 2 performs well, and also provides us with some insight into the improvements that remain achievable, knowing that the best in terms of load distribution - but not always feasible because of the stream characteristics - would be a straight line.

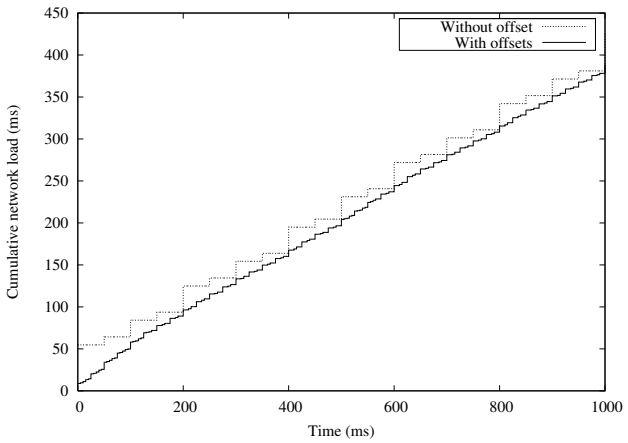


Figure 6: Cumulative network load (expressed in transmission time) with and without offsets - comparison over 1 second.

It is worth mentioning that the better distribution of the load with offsets is also very interesting for reducing peaks of CPU load since ECUs will not have to build, transmit or receive bursts of frames. In practice, this is an major reason why offsets are sometimes already implemented in production vehicles.

4.3 Partial offset usage

So far, we have assumed that offset strategies would be applied to all stations. In practice, the load on a CAN network is generally not evenly distributed between the stations, and it is common to have networks where a single station, or a couple of stations, induce a large fraction of the total load. In this situation, a significant improvement can already be achieved when offsets are used only on the station, or the few stations, that create most of the bus load. This also involves fewer changes for the car-manufacturer.

To obtain some understanding of what to expect from offsets in this case, we generated networks where 30% of the load is concentrated on a single station (*i.e.*, load concentration situation). Figure 7 shows that applying offsets on a few stations is already very advantageous in terms of WCRT of the lowest priority frame. With regard to what would be achieved without offsets, the lowest priority frame has its WCRT reduced by 34.5% with offsets on one station, and by 48% on four stations.

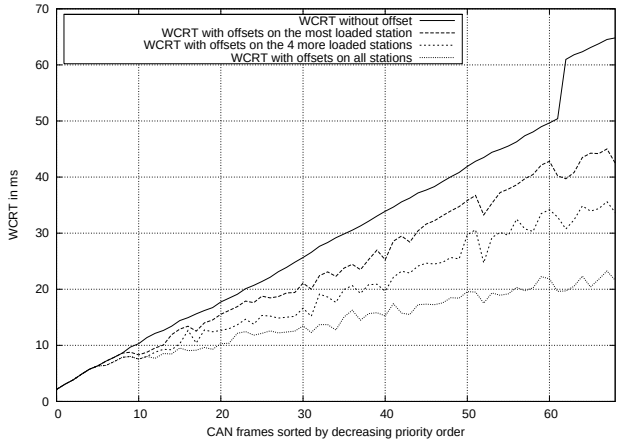


Figure 7: Comparison of the WCRT on a body network with load concentration: 1) without offsets (upper curve), 2) with offsets only on the loaded station (immediate lower curve), 3) with offsets on the 4 more loaded stations (third curve from the top), and 4) with offsets on all stations (lower curve).

5 Offsets allow higher network loads

Up to this point, the experiments have been done on networks with a load corresponding to what is commonly found in today's automotive CAN networks. Now we propose to evaluate the benefits of offsets in the near future situation where network load will increase. We model the load increase in two directions: either by distributing new messages onto existing stations, or by assigning them onto new stations. We proceeded as follows:

- Define a random network net_1 with a given load $load_1$. In this experiment, the body network drawn at random has a load equal to 37.6,
- Define a new load level $load_2$ (e.g., 40%). Define a random set of frames that corresponds to the load difference between $load_1$ and $load_2$. This newly created set of frames is denoted by S_{new},
- Two methods are employed to allocate the frames of S_{new}:
  - dispatch S_{new} on the existing stations of net_1, this new network is called net_2^{frames},
  - dispatch the set of frames S_{new} on new stations (with a limit of 5 frames per station) and add them to net_1. The resulting network is called $net_2^{stations}$,
- Determine offsets using algorithm of Section 2 and compute WCRT for net_2^{frames} and $net_2^{stations}$.

Following this procedure enables us to compare the increase of WCRT for the two scenarios identified. Figure 8 shows the evolution of the WCRT of the lowest priority frame for a network load ranging from 40% to 60%.

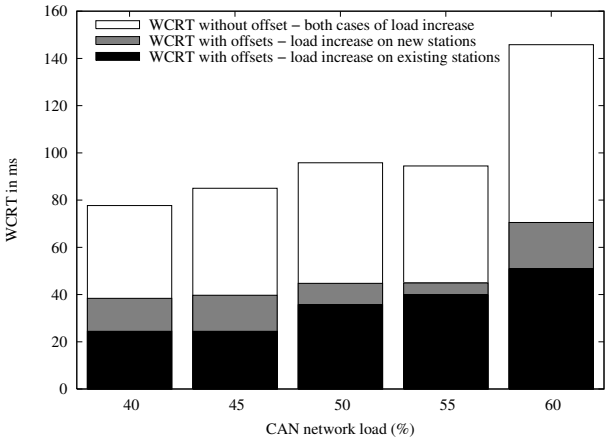


Figure 8: WCRT of the lowest priority frame for a load ranging from 40 to 60%: 1) without offsets (white), 2) with offsets and the additional load assigned to new stations (gray), and 3) with offsets and the additional load assigned to existing stations (black). The additional load is the network load added to a randomly chosen network with an initial load equal to 37.6%. The results presented here are obtained on a single typical body network.

What can be observed is that the gain with offsets remains very significant even when the load increases. For instance, at a load of 60% the gain with offsets is equal to a factor 2.8 if the additional load is distributed on existing stations, or a factor 2.1 if the additional load is allocated to new stations.
Secondly, the experiments show that the WCRT of the lowest priority frame with offsets at 60% is roughly similar to the WCRT at 30% of load without offsets. In other words, the performance at 60% with offsets are equivalent to the performance at 30% without offsets. Although this is not shown in Figure 8, this remark holds true for all frames, whatever their priority level (except at the highest priority levels where there is less gain).
Finally, it is worth noting that there is a difference whether the new load is spread over existing stations or assigned to new stations. In the latter case, offsets are less efficient in general, which is logical because the lack of global time reference implies that the offsets are local to each station.

6 Conclusion

This study provides two contributions. First, we propose a low-complexity algorithm for deciding offsets, which has good performances for typical automotive networks, be they body or chassis networks. This algorithm, the first of its kind in the literature to the best of our knowledge, should constitute a sound basis for further improvements and optimizations. For instance, specific constraints of a particular design process, or even vehicle project, can be taken into account.

The second contribution of the paper is to show that the use of offsets enable very significant performance improvements on a wide range of network configurations. We believe using offsets is a robust technique that might actually provide a solution in the short term to deal with the increasing network load, and thus might allow the use of CAN as the principal network in the next car generations, at least when no safety critical functions are involved.

Offsets, which impose constraints on the frame release dates, can be seen as a trade-off between event-triggered communications and time-triggered communications. Experiments show that it is possible to achieve further gains with synchronization mechanisms between stations, which imposes additional constraints on communication and could constitutes a lightweight time-triggered solution on CAN. The extent to which it can be implemented in a robust way (*i.e.*, resilience to ECU reboots, local clocks that are drifting apart, etc.) is the subject of our ongoing work.

References

[1] C. Braun, L. Havet, and N. Navet. NETCARBENCH: a benchmark for techniques and tools used in the design of automotive communication systems. In *Proc. of the 7th IFAC International Conference on Fieldbuses and Networks in Industrial and Embedded Systems (FeT'2007)*, November 2007. Software and manual available at *http://www.loria.fr/~nnavet/netcarbench/*.

[2] R.I. Davis, A. Burns, R.J. Bril, and J.J. Lukkien. Controller Area Network (CAN) schedulability analysis: Refuted, revisited and revised. Real-Time Systems*, 35(3):239–272, April 2007.*

[3] J. Goossens. Scheduling of offset free systems. Real-Time Systems*, 24(2):239–258, March 2003.*

[4] M. Grenier, J. Goossens, and N. Navet. Near-optimal fixed priority preemptive scheduling of offset free systems. In Proc. of the 14th International Conference on Network and Systems (RTNS'2006), Poitiers, France, May 30-31, 2006.*, 2006.*

[5] N. Navet, Y. Song, and F. Simonot. Worst-case deadline failure probability in real-time applications distributed over CAN (Controller Area Network). Journal of Systems Architecture*, 46(7):407–417, 2000.*

[6] N. Navet, Y. Song, F. Simonot-Lion, and C. Wilwert. Trends in automotive communication systems. Proceedings of the IEEE*, 93(6):1204–1223, 2005.*

[7] L. Rodrigues, M. Guimaraes, and J. Rufino. Fault-tolerant clock synchronization in CAN. In RTSS*, pages 420–429, 1998.*

[8] K. Tindell and A. Burns. Guaranteed message latencies for distributed safety-critical hard real-time control networks. In 1st International CAN Conference Proceedings, Germany, September 1994*, 1994.*