Exploiting Inter-Event Stream Correlations Between Output Event Streams of non-Preemptively Scheduled Tasks

Jonas Rox, Rolf Ernst
Institute of Computer and Communication Network Engineering
Technical University of Braunschweig
D-38106 Braunschweig / Germany
{rox|ernst}@ida.ing.tu-bs.de

***Abstract*—In this paper we present a new technique which exploits timing-correlation between tasks for scheduling analysis in multiprocessor and distributed systems with non-preemptive scheduled resources. Previously developed techniques also allow capturing and exploiting timing-correlation in distributed systems. However, they focus on timing correlations resulting from data dependencies between tasks. The new technique presented in this paper is orthogonal to the existing ones and allows capturing timing-correlations between the output event streams of tasks resulting from the use of a non-preemptive scheduling policy on a resource. We also show how these timing-correlations can be exploited to calculate tighter bounds for the worst-case response time analysis for tasks activated by such correlated event streams.**

I. Introduction

System and communication platform integration is a major challenge and systematic analysis of the complex dynamic timing effects (scheduling, arbitration, blocking, buffering) becomes key to building safe and reliable systems.

To give worst case guarantees for the timing behaviour of a distributed system, different compositional approaches for system level performance analysis have been developed, e.g. [1][2]. For simplicity, these formal scheduling analysis techniques often ignore correlations between task execution times or communication timing. However, such correlations can have a large influence on system timing as has been shown for special system topologies [3][4][5]. The existing approaches only consider event stream correlations that arise from application specific properties, e.g. tree-shaped task dependencies [4].

In this paper, we will consider another source of event stream correlations, namely, non-preemptively scheduled tasks. This is especially important for the analysis of distributed systems from the automotive domain, where it is very common to have static priority preemptively scheduled ECUs, interconnected by non-preemptively scheduled communication resources, e.g. a CAN-Bus. Non-preemptive scheduling is also typically used in most real-time Ethernet solutions like, e.g. AFDX.

As an example take the system in Figure 1. The system consists of several tasks mapped on three different CPUs that are interconnected by a CAN-Bus. Some of the tasks executing on CPU1 and CPU2 send data to the tasks executing in CPU3. The tasks mapped on the CPU3 are event triggered, i.e. they are activated as soon as the corresponding communication

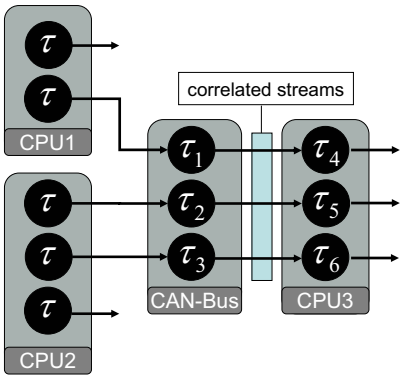

Fig. 1. Distributed System with timing correlations at the outputs of non-preemptively scheduled tasks

task, running on the CAN-Bus, completes its execution. First, assume that the bus would be arbitrated by a static priority **preemptive** scheduler.

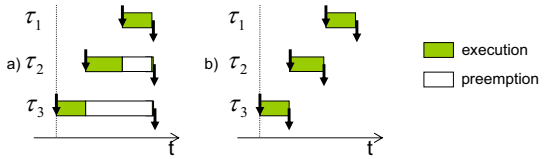

Fig. 2. a) possible completion scenario under a preemptive scheduling policy and b) a possible completion scenario under a non-preemptive scheduling policy

Figure 2a illustrates a possible execution scenario for the three tasks τ_1,τ_2 and τ_3. As can be seen, τ_2 is activated and interrupts τ_3 just before it finishes its execution. And τ_2 is interrupted by τ_1 just before it finishes its execution. Then, immediately after τ_1 completes, also the tasks τ_2 and τ_3 will complete their execution. In this scenario, the three tasks can complete quasi-simultaneously. And in this case, the tasks running on CPU3 would be activated simultaneously. If we now assume that the three tasks τ_1,τ_2 and τ_3 are scheduled according to a static priority **non-preemptive** scheduling policy, they cannot complete simultaneously anymore. A possible execution scenario is depicted in Figure 2b. After the completion of τ_3, τ_2 has to execute at least for its minimum execution time before it can complete. The

978-3-9810801-6-2/DATE10 © 2010 EDAA

same holds for τ_1. Obviously, in the latter case, the output streams of the three communication tasks are correlated and the tasks τ_4, τ_5 and τ_6 cannot be activated simultaneously. The existing compositional performance analysis approaches do not consider the correlations described above. Hence, for the local scheduling analysis of CPU3 they would assume the simultaneous activation the tasks τ_4, τ_5 and τ_6, leading to unnecessary conservative results.

The contributions of this work can be summarized as follows:

- We define *minimum distance correlations* between event streams resulting from non-preemptive scheduled tasks and show how they can be captured.
- We present how the worst case response time (WCRT) analysis for static priority preemptive scheduled tasks can be adopted to consider these correlations to obtain tighter WCRT bounds.

The rest of the paper is organized as follows. In the following section, we will review related work. In Section III we introduce our application model. In Section IV we formally define minimum distant correlated event streams and show how such correlations of input event streams can be exploited to obtain tighter worst case response times. Experiments are carried out in Section V. We interpret the experimental results, before we draw our conclusions

II. Related Work

Different methods to capture timing correlation between events of different event streams in a way that can be exploited by scheduling analysis have been proposed.

Tindell [6] groups a set of time correlated tasks into so called transactions. Each task of such a transaction is activated when a relative time, called offset, elapses after the start of the transaction. In [3] Palencia and Harbour presented the WCDO (Worst Case Dynamic Offsets) algorithm which extends the analysis presented by Tindell, by allowing the task offsets to be larger than the transaction period and extending the technique for distributed systems to dynamic offsets, which vary from one job to another. Palencia and Harbour further refined their method in [7] by presenting a new analysis technique for tasks with precedence relations in distributed systems.

In [4], Henia introduced a new technique to capture inter event stream correlations in distributed systems with tree-shaped task-dependencies. Compared to the previous approaches, this enabled capturing timing-correlation between tasks in parallel paths in a more accurate way, which in turn leads to tighter analysis results. The improvement presented in [8], considers multiple timing-references which allows to capture more information about the timing correlation between tasks.

In [5], Huang et. al. consider data streams that are partitioned into separate sub-streams which are processed on parallel hardware components. The resultant correlations between the sub-streams can be exploited to obtain more accurate analysis results. Here, the correlations between two event streams result from the fact that they both model parts of the same data stream that was partitioned.

All the above approaches have in common that they consider timing correlations between tasks that share some kind of data dependency. And the resultant restrictions on the activation timing of tasks is defined by an offset relative to an (or several) external events. This offset (which may also be dynamic) specifies, when an activation occurs. In contrast, the timing correlations we will present results from a property of the execution/communication platform, namely non-preemptive scheduling. Also the implications on the activation timing of tasks is different, since the correlations between event streams we present in this paper lead to time intervals in which task activations can **not** occur, i.e. a certain time after the activation of another task.

III. Application Model

We adopt the application model of the compositional system level analysis approach described in [9] which uses traditional scheduling analysis techniques to analyse the local components.

In particular, we consider a set of n tasks $\tau = \{\tau_1, .., \tau_n\}$ mapped on one resource which uses a static priority preemptive (SPP) scheduling policy. The tasks are ordered according to their priority, i.e. τ_1 has the highest priority and τ_n has the lowest priority. Each task is activated by events which are modeled as arbitrary event streams using solely the functions $\delta^-(n)$ and $\delta^+(n)$ which specify the minimum, respectively the maximum distance between n consecutive events. Each activation of a task τ_i executes no longer than a worst case execution time C_i.

In the following we will also use the function $\eta^+(\Delta)$, which returns the maximum number of events that can occur in any time interval of size Δ. This function can be derived based on $\delta^-(n)$ as follows:

$$\eta^+(\Delta) = \max_{2 \leq n \in \mathbb{N}} [\{n \mid \delta^-(n) < \Delta\} \cup \{1\}] \tag{1}$$

Another important concept we use in the following section is the level-i busy period as defined by Lehoczky [10].

Definition 1. *Level-i Busy Period:*
A level-i busy period is a time interval $[a, b]$ within which jobs of priority i or higher are processed throughout $[a, b]$ but no jobs of priority i or higher are processed in $(a - \epsilon, a)$ or $(b, b + \epsilon)$ for sufficiently small $\epsilon > 0$.

IV. Minimum Distance Correlated Event Streams

As explained in Section I, whenever a non-preemptive scheduling policy is used to schedule different tasks, all output streams of these tasks are correlated with all output streams of all other tasks mapped on the same resource. In this case, the minimum response time of each task not only bounds the minimum distance between events within a single stream, but it also restricts the minimum distance to events of the other streams.

Definition 2. *Minimum Distance Correlated Event Streams:*
A set of minimum distance correlated event streams $ES = \{ES_1, \ldots, ES_n\}$ is a set of event streams, where the timing of events of each event stream is constrained as follows:
Each event stream $ES_i \in ES$ has an additional parameter $d_i \in \mathbb{R}$ that specifies a time interval which has to elapse after any event occurrence of one of the streams, before an event of the stream ES_i can occur.

As an Example, consider two minimum distance correlated (MDC) event streams ES_i and ES_k. If an event $e_i \in ES_i$ occurs at the time instant T_i, in the interval $(T_i - d_i, T_i + d_k)$ no event of ES_k can occur.

The additional parameter d_i of the output event stream ES_i of the task τ_i is given by the best case response time (BCRT) of τ_i. The BCRT of a task can be obtained by best case response time analyses or, more easily, it can be bounded by the best case execution time of the task. Obviously, for both methods, the best case execution time of the task is needed. While it may be difficult to obtain reliably best case execution times for computational tasks, it is usually not a problem to obtain best case execution times (or transmission times) for communication tasks. These can directly be derived from the minimum amount of transmitted data and the bandwidth of the used communication medium.

A. Analysis of Tasks with MDC Input Event Streams

In the classical static priority preemptive WCRT analysis, the critical instant is assumed, where all tasks are activated simultaneously. It is further assumed that all following activations occur as early as possible and all activations execute for the maximum execution time of the task. So, for the analysis, only one particular sequence of each event stream is considered. We call this sequence the critical event sequence.

Definition 3. *Critical Event Sequence:*
The critical event sequence $es_i^{\delta^-} \in ES_i$ is the event sequence $[e_1, e_2, e_3, \ldots]$, where the distance between the event e_1 and e_k is given by $\delta_i^-(k)$, $\forall\ k > 1$.

Obviously, the critical instant is obtained if

1) all tasks are activated by their critical event sequence as defined by their input event models and
2) the critical sequences of all input event streams of the considered tasks start at the same instant.

To calculate the worst case response time R_i^{max} for task τ_i in this scenario, the following formula (derivation of the WCRT Equations by Tindell et al.[11]) can be used:

$$R_i^{max} = \max_{q=0,1,2,\ldots} (w_i(q) - \delta_i^-(q+1)) \tag{2}$$

where

$$w_i(q) = (q+1) \cdot C_i + I_{hp(i)}(w_i(q)) \tag{3}$$

with the interference of higher priority tasks $I_{hp(i)}$ given by

$$I_{hp(i))}(\Delta) = \sum_{\forall j \in hp(i)} \eta_j^+(\Delta) \cdot C_j \tag{4}$$

If we now consider tasks, activated by MDC input event streams, the individual critical sequences are still the same, but they can no longer start at the same instant. Furthermore, not only the first events cannot occur simultaneously anymore, but also all following events of the different streams must be separated by the corresponding minimum distances. Thus, it is not obvious anymore which activation scenario, i.e. combination of event sequences, will lead to the worst case response time of a certain task. The number of different scenarios that need to be checked not only depends on the number of involved tasks, but also on their input event models. The more activations of tasks are included in the examined busy window, the more possible scenarios exist. For the calculation of an exact worst case response time, all possible scenarios must be checked.

As an example, assume that only two tasks, τ_1 and τ_2, execute on the same resource. If they have MDC input event streams, to find the WCRT of the task τ_2, we need to consider two different scenarios: Either τ_1 is activated first at T_1 and τ_2 is activated at $T_1 + d_2$, or τ_2 is activated first at T_2 and τ_1 is activated at $T_2 + d_1$. For three tasks, we would already have to explore 6 different scenarios and so on. So, only considering the first activation of a set of n tasks, already gives $n!$ different activation scenarios. It becomes even worse, if more than one activation of some of the task can occur in the examined busy window and can conflict with activation of other tasks. Again, all possible combinations would have to be considered. Obviously, such an exact analysis can become computationally expensive, already for relatively small systems. Hence, we propose a method to calculate an approximated conservative upper bound of the worst case response time of a task τ_i in the presence of MDC input event streams.

In our method, we simplify the determination of the WCRT of a task τ_i by making the following assumptions:

1) The task τ_i is activated by its critical event sequence $es_i^{\delta^-} \in ES_i$.
2) We ignore the minimum distance correlations between the input event streams of the higher priority tasks $\tau_j \in J = \{\tau_1, .., \tau_{i-1}\}$.
3) We only consider the minimum distance correlations between the input event stream of τ_i and the input event streams of other tasks $\tau_j \in J$. We further only consider the correlations between the first event of $es_i^{\delta^-}$ and events of other event streams.

Obviously, ignoring the minimum distance correlations between events of different event streams can only lead to more event arrivals in a given time interval. Hence, by ignoring the minimum distance correlations between events (assumption 2 and 3) the interference of tasks with higher priority than τ_i, cannot be smaller than when considering the correlations.

It is clear, that the WCRT of R_i^{max} of τ_i must occur in one of the following two cases:

Case 1: R_i^{max} occurs in a level-i busy period initiated by the task τ_i or
Case 2: R_i^{max} occurs in a level-i busy period initiated by the activation of one of the higher priority tasks $\tau_j \in J$.

In **Case 1**, if the task τ_i initiates the busy period at some instant $T_i = T^0$, a higher priority task $\tau_j \in J$ cannot be activated earlier than $T^0 + d_j$ and it cannot be activated more often than expressed by its critical event sequence $es_j^{\delta^-}$.

As an example consider the gantt-chart depicted in Figure 3. It shows a set of three tasks τ_1, τ_2 and τ_3 and time intervals the different tasks execute. The illustrated scenario corresponds to **Case 1** for the task τ_3. The tasks τ_3 initiates the busy period and both higher priority tasks τ_1 and τ_2 are activated as early as possible after T^0 (at $T^0 + d_1$, respectively at $T^0 + d_2$). Since we ignore the event stream correlations between τ_1 and τ_2 they are assumed to be activated by their critical event sequence starting at $T^0 + d_1$, respectively at $T^0 + d_2$. In this scenario, τ_3 can only be interrupted once by τ_1 and it cannot be interrupted at all by τ_2.

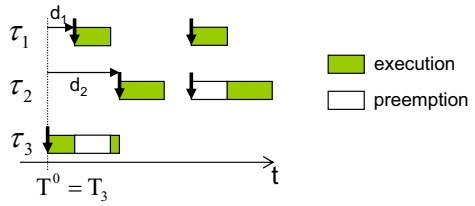

Fig. 3. The resulting scenario if τ_3 initiates the busy period

In **Case 2**, the time instant T_i falls into a level (i-1) busy period that started before T_i. Since we ignore the correlations between input event streams of the higher priority tasks, we obtain the longest possible level (i-1) busy period by assuming that all higher priority tasks $\tau_j \in J$ are activated by their critical event sequence $es_j^{\delta^-}$ starting simultaneously at the instant $T^0 < T_i$. In this case, the first possible activation instant of the task τ_i would be $T_i = T^0 + d_i$. A later activation of τ_i at some instant $T_i^{'}$ could only be worse (i.e. lead to a higher WCRT) if it would allow more activations of one of the higher priority tasks to occur in between $T_i^{'} - T^0$. Let the instant $T^1 > T^0$ be the earliest instant where the second activation of a higher priority task can occur after T^0. Then, the instant for the first activation of τ_i to consider would be $T_i^{'} = T^1 + d_i$.

As an example, consider again the set of the three tasks τ_1, τ_2 and τ_3. Figure 4 shows the activation scenario which leads to the longest level-2 buy period. After each activating event of one of the higher priority tasks (τ_1 and τ_2) within this busy period, at least d_3 time units have to pass before τ_3 can be activated. Hence, for τ_3 we obtain the possible activation instants T_3, $T_3^{'}$ and $T_3^{''}$ as shown in Figure 4. As can be seen,

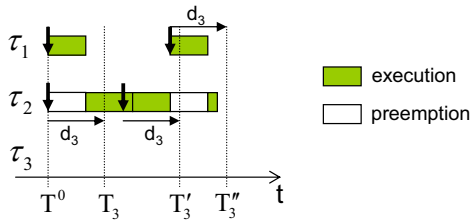

Fig. 4. Possible activation instants of τ_3 if it is activated in a busy period started by higher priority tasks.

the instant $T_3^{''}$ doesn't fall into the busy period and hence this activation instant starts a new busy period. Therefore, it is already covered by **Case 1**.

So, only the two instants T_3 and $T_3^{'}$ have to be considered as possible activation times for the task τ_3. If an activation of τ_3 occurs in either of these two instants, later activations of the higher priority tasks may be deferred according to their minimum distance. This is depicted in Figure 5a and Figure 5b.

If τ_3 is activated at T_3 (Figure 5a), the second activation of the task τ_2 cannot occur at $T^0 + \delta_2^-(2)$ (as it does in the case illustrated in Figure 4), since this would fall into the interval $(T_3 - d_3, T_3 + d_2)$ in which no event of τ_2 can occur. The earliest time it can occur in this scenario is at $T_3 + d_2$. Hence, the only interference for τ_3 is due to the unfinished execution of the first activation of τ_2.

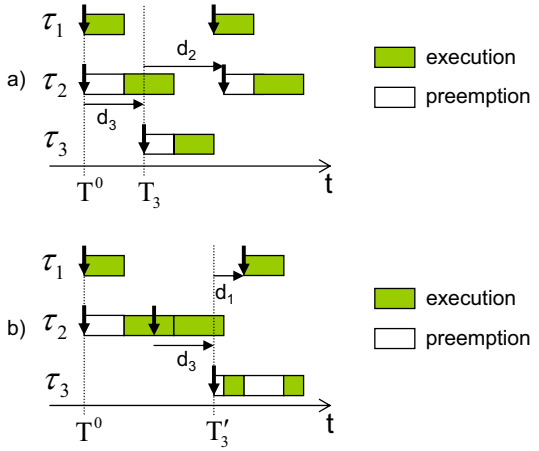

Fig. 5. a) The resulting scenario if τ_3 is activated for the first time at T_3 and b) the resulting scenario if τ_3 is activated for the first time at $T_3^{'}$

If τ_3 is activated at $T_3^{'}$ (Figure 5b), the second activation of τ_1 would fall into the interval $(T_3^{'} - d_3, T_3^{'} + d_1)$ if it would occur at $T^0 + \delta_1^-(2)$. Hence, the earliest time it can occur is $T_3^{'} + d_1$. In this scenario the task τ_3 suffers an interference of one complete execution of the task τ_1 plus the unfinished second execution of the task τ_2. So, this leads to a larger WCRT of τ_3 than activating it at T_3.

Since in **Case 1**, where τ_3 started the busy window, the interference of the higher priority tasks was only one complete execution of τ_1 the WCRT of τ_3 is found when it is activated at $T_3^{'}$.

In the following we will formally derive the response times for the different activation scenarios. We start with **Case 1**.

Lemma 1. *Assume that the task τ_i is activated by its critical event sequence $es_i^{\delta^-}$ starting at time T^0 and all higher priority tasks $\tau_j \in J$ are activated by their critical event sequence starting at the instant $T^0 + d_j$. The maximum interference of the higher priority tasks $I_{hp(i)}$ in the time interval $[T^0, T^0 + \Delta)$ can be bounded by:*

$$I_{hp(i)}(\Delta) = \sum_{j \in J} \eta_j^+([\Delta - d_j]^0) \cdot C_j$$

with

$$[x]^0 \equiv \max\{x, 0\}$$

Proof: The lemma directly follows from the definition of $\eta^+(\Delta)$. ■

Using the above lemma to calculate the higher priority interference $I_{hp(i)}(\Delta)$, we can use Equation 2 and Equation 3 to calculate the maximum response time for the **Case 1**, where the task τ_i initiates the busy period.

To determine the response time in **Case 2**, we must calculate the response time for each possible activation instant of τ_i. In each of these different activation scenarios, the higher priority tasks are activated simultaneously at T^0 and $m_j \geq 1$ activations of each higher priority task $\tau_j \in J$ occur before T_i. While these activations before T_i occur according to the critical input event sequences of the higher priority tasks, the

activations after T_i are further restricted as follows. The first activation of a task τ_j after T_i (the (m_j+1)-th activation of τ_j after T^0) cannot occur in the interval $[T_i, T_i + d_j)$. Hence, its earliest occurrence time is additionally bounded by $T^-_{i,j} = T_i + d_j$. All further activations must also at least be separated from the (m_j+1)-th event by the minimum distance as specified by $\delta^-_j(n)$. This means that, the (m_j+1+k)-th activation of τ_j cannot occur earlier than $T^-_{i,j} + \delta^-_j(k+1)$. This leads to the following lemma.

Lemma 2. *Assume that all higher priority tasks $\tau_j \in J$ are activated simultaneously at T^0. If the task τ_i is activated at T_i with $T_i > T^0$, the earliest possible activation instants of a task $\tau_j \in J$ can be expressed by the event sequence $[e_1, e_2, e_3, \ldots]$, where the event e_1 arrives at T^0 and the event e_n arrives at $T^0 + \delta^{'-}_j(n)$ which is defined as follows:*

$$\delta^{'-}_j(n) = \begin{cases} \delta^-_j(n) & : 1 < n \leq m \\ \max\{\delta^-_j(n), T^-_{i,j}\} & : n = m+1 \\ \max\{\delta^-_j(n), T^-_{i,j} + \delta^-_j(n-m)\} & : n > m+1 \end{cases}$$

where
$T^-_{i,j} = T_i + d_j$ which is the earliest possible instant the first activation of τ_j can occur after T_i and
$m_j = \eta^+_j(T_i - d_i - T^0)$ which is the maximum number of activations of task τ_j that occur in the interval $[T^0, T_i - d_i)$.

Proof: Follows from the discussion above. ∎

From Lemma 2 we can directly derive the number of event arrivals of a task τ_j in the interval $[T^0, T^0 + \Delta)$ in the considered activation scenario with:

$$\eta^{'+}_j(\Delta) = \max_{2 \leq n}[\{n \mid \delta^{'-}_j(n) < \Delta\} \cup \{1\}] \tag{5}$$

Therewith, we can calculate the interference of higher priority tasks for a given activation scenario as follows.

Lemma 3. *Assume that the task τ_i is activated by its critical event sequence $es^{\delta^-}_i$ starting at time T_i and all higher priority tasks $\tau_j \in J$ are activated simultaneously by their critical event sequence starting at the instant T^0, with $T^0 < T_i$. The maximum interference of the higher priority tasks $I_{hp(i)}$ in the time interval $[T^0, T^0 + \Delta)$ can be bounded by:*

$$I_{hp(i)}(\Delta) = \sum_{j \in J} \eta^{'+}_j(\Delta) \cdot C_j$$

Proof: The lemma follows from the definition of $\eta^{'+}(\Delta)$. ∎

Compared to **Case 1**, in **Case 2** we also have to consider, that the first activation of the task τ_i does not occur at the start of the considered busy period, but $T_i - T^0$ later. Also, all further activations of τ_i that fall into this busy period also occur $T_i - T^0$ later than compared to the typical critical instant scenario. Hence, when calculating the response times by subtracting the arrival time of an event from its completion time, we have to consider the later arrival. We do this by subtracting $T_i - T^0$ from each response time calculated with Equation 2.

W.l.o.g. we set $T^0 = 0$ and get:

$$R^{max}_i = \max_{q=0,1,2,\ldots}(w_i(q) - \delta^-(q+1) - T_i) \tag{6}$$

where

$$w_i(q) = (q+1) \cdot C_i + I_{hp(i)}(w_i(q)) \tag{7}$$

with

$$I_{hp(i))}(\Delta) = \sum_{\forall j \in hp(i)} \eta^{'+}_j(\Delta) \cdot C_j \tag{8}$$

The WCRT R^{max}_i-impr. of the task τ_i is the maximum of the response time obtained in **Case 1** and the response times of each possible activation instant in **Case 2**.

V. Experiments

For our experiments, we consider the hypothetical example system depicted in Figure 6. The shown example system could very well be part of a larger system, e.g. like the introductional example depicted in figure 1. But since we want to focus on the analysis of tasks with MDC input streams we only consider the relevant parts, which are a CAN-Bus (using a static priority **non-preemptive** scheduling policy) and a CPU where tasks are activating upon the reception of a CAN message, and are executing according to a static priority **preemptive** scheduling policy.

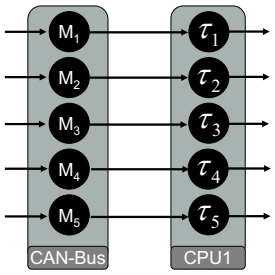

Fig. 6. A hypothetical example system

The computational tasks τ_1,τ_2,τ_3,τ_4 and τ_5 executing on CPU1, as well as the communication tasks M_1, M_2, M_3, M_4 and M_5 mapped on the CAN-Bus, modeling the message transmissions, are ordered according to their priority, i.e. τ_1 (M_1) has the highest priority and τ_5 (M_5) has the lowest priority. In the first experiment, we assume a 125 Kbit/s CAN Bus. This gives us the transmission times of the different messages listed in table I. Although, we assume the same number of bytes as payload in the best case as in the worst case, the transmission times are specified as [best case, worst case] interval, because the used analysis assumes zero stuff bits in the best case and the maximum number of possible stuff bits in the worst case. We further assume that the messages are triggered periodically with the periods given in the table I. The core execution times of the tasks are summarized in table II

For each task executing on CPU1 we calculate its WCRT R^{max}-blind without considering the minimum distance correlations between input event streams and its WCRT R^{max}-impr. using the approximation presented in section IV. We also calculate the exact WCRT R^{max}-exact, by exploring all possible activation scenarios. The results are listed in table III.

TABLE I
THE TRANSMISSION TIMES AND ACTIVATION PERIODS OF THE CAN MESSAGES

message	Tx time (in ms)	activation period (in ms)
M_1	[0.8, 0.976]	15
M_2	[0.608, 0.736]	30
M_3	[0.864, 1.056]	75
M_4	[0.864, 1.056]	40
M_5	[0.608, 0.736]	15

TABLE II
THE CORE EXECUTION TIMES OF THE TASKS MAPPED ON CPU1

Task	execution time
τ_1	0.8 ms
τ_2	0.35 ms
τ_3	0.15 ms
τ_4	0.4 ms
τ_5	1 ms

TABLE III
RESULTING ACTIVATION AND WCRTS OF THE TASKS MAPPED ON CPU1 FOR A CAN BUS SPEED OF 125KBIT/S. ALL VALUES ARE IN ms

Task	activation(P/J/d)	R^{max}-blind	R^{max}-impr.	R^{max}-exact
τ_1	(15/1.256/0.8)	0.8	0.8	0.8
τ_2	(30/2.208/0.608)	1.15	0.542	0.542
τ_3	(75/4.422/0.864)	1.3	0.436	0.15
τ_4	(40/4.032/0.864)	1.7	0.836	0.4
τ_5	(15/5.748/0.608)	2.7	2.7	2.2

TABLE IV
RESULTING ACTIVATION AND WCRTS OF THE TASKS MAPPED ON CPU1 FOR A CAN-BUS SPEED OF 250 KBIT/S. ALL VALUES ARE IN ms

Task	activation(P/J/d)	R^{max}-blind	R^{max}-impr.	R^{max}-exact
τ_1	(15/0.628/0.4)	0.8	0.8	0.8
τ_2	(30/1.104/0.304)	1.15	0.846	0.846
τ_3	(75/1.516/0.432)	1.3	0.868	0.868
τ_4	(40/1.896/0.432	1.7	1.7	1.7
τ_5	(15/2.024/0.304)	2.7	2.7	2.7

All values are in ms. The table also lists the activating event models of the tasks in form of period P, jitter J and minimum distance d as they result from the output model calculations after the local analysis of the CAN-Bus.

As can be seen, for the task τ_2, τ_3 and τ_4 we obtain a reduction of more than 50% in the WCRT bound using the approximation approach. Exploring all possible scenarios, we see that, especially for τ_3 and τ_4 the obtained WCRT becomes even much tighter. Obviously, there is no improvement for τ_1, because it is the highest priority task, and therefore it isn't interrupted in any case. For τ_5, there is no improvement with the approximation method, because of its relative long execution time of $1ms$. This is long enough, that if it self starts a busy period (**Case1**), all other tasks can interrupt it, since their minimum distances are all smaller than $1ms$ and thus, they are all assumed to be activated before τ_5 completes. But calculating the exact response time reveals that it cannot be interrupted by all higher priority tasks, since their activations also have to be separated by their corresponding minimum distances.

To obtain R^{max}-blind, for each task, we calculate the response times considering only one activation scenario, which is the critical instant. Since the jitter of the activating event streams of the tasks running on CPU1 is relative small compared to the period, here we have no recurrent activations of a task within the busy period. This means, to determine R^{max}-impr. we only had to examine two different activations scenarios for each task. And for R^{max}-exact we had to consider $i!$ different scenarios for τ_i, i.e. for τ_5 we checked $5! = 120$ different scenarios to obtain its exact WCRT.

In a second experiment, we doubled the speed of the CAN-Bus to 250 Kbit/s, which results in the messages only having half the minimum transmission times as before. Correspondingly, the minimum distances between events of the input streams of the tasks mapped on CPU1 change. As expected, also the improvement obtained from considering the minimum distance correlation between the input streams decreases, as can be seen in Table IV.

We still have a significant improvement for the WCRT of τ_2 and τ_3, using either the approximation or the exact determination for the WCRTs. But besides the decrease in the improvement in the WCRTs, we can also observe, that compared to the approximation, we don't obtain tighter WCRTs by exploring all possible scenarios anymore.

VI. CONCLUSION

In this paper we have considered correlations between event streams resulting from non-preemptive scheduling and we have shown how these correlations can be captured.

We have shown, how these event stream correlations can be used to obtain tighter worst case response times (WCRT) for static priority preemptive scheduled tasks. While an exact WCRTs can be calculated by exploring all possible activation scenarios, this can easily become computational expensive for larger systems, or if the correlated event streams are bursty. Therefore, we presented an method to obtain a conservative approximation for which only a few different activation scenarios have to be considered.

Through experiments, we showed that significant tighter WCRTs can be obtained if the introduced correlations are considered.

REFERENCES

[1] K. Richter, D. Ziegenbein, M. Jersak, and R. Ernst, "Model composition for scheduling analysis in platform design," in *Proceedings 39th Design Automation Conference (DAC 2002)*, June 2002.

[2] L. Thiele, S. Chakraborty, and M. Naedele, "Real-time calculus for scheduling hard real-time systems," *Circuits and Systems, 2000. Proceedings. ISCAS 2000 Geneva. The 2000 IEEE International Symposium on*, vol. 4, 2000.

[3] J. Palencia and M. Harbour, "Schedulability analysis for tasks with static and dynamic offsets," in *Proc. 19th IEEE Real-Time Systems Symposium (RTSS98)*, 1998.

[4] R. Henia and R. Ernst, "Context-aware scheduling analysis of distributed systems with tree-shaped task-dependencies," in *Proceeding Design Automation and Test in Europe*, March 2005.

[5] K. Huang, L. Thiele, T. Stefanov, and E. Deprettere, "Performance analysis of multimedia applications using correlated streams," in *Design, Automation and Test in Europe (DATE 07)*, Nice, France, Apr. 2007, pp. 912–917.

[6] K. Tindell, "Adding time-offsets to schedulability analysis," University of York, Tech. Rep., 1994.

[7] J. Palencia and M. Harbour, "Exploiting precedence relations in the schedulablilty analysis of distributed real-time systems," in *Proc. 20th IEEE Real-Time Systems Symposium (RTSS99)*, 1999.

[8] R. Henia and R. Ernst, "Improved offset-analysis using multiple timing-references," *Proceedings of the conference on Design, automation and test in Europe: Proceedings*, pp. 450–455, 2006.

[9] R. Henia, A. Hamann, M. Jersak, R. Racu, K. Richter, and R. Ernst, "System level performance analysis - the symta/s approach," in *IEE Proceedings Computers and Digital Techniques*, 2005.

[10] J. Lehoczky, "Fixed Priority Scheduling of Periodic Task Sets with Arbitrary Deadlines," *Real-Time Systems Symposium, 1990. Proceedings., 11th*, pp. 201–209, Dec 1990.

[11] K. Tindell, A. Burns, and A. Wellings, "An extendible approach for analyzing fixed priority hard real-time tasks," *Real-Time Systems*, vol. 6, no. 2, pp. 133–151, 1994.