Response-Time Analysis of tasks with offsets

Karim Traore[1], Emmanuel Grolleau[2], Ahmed Rahni[3] and Michaël Richard[4]
LISI / ENSMA - BP 40109 - 1, avenue Clément Ader
86961 Futuroscope Chasseneuil Cedex
Telephone: 00 335 49 49 80 63
Fax : 00 335 49 49 80 64
Email: karim.traore@ensma.fr[1], grolleau@ensma.fr[2], ahmed.rahni@ensma.fr[3] and richardm@ensma.fr[4]

***Abstract*— This article presents some results about schedulability analysis of tasks with offsets also known as transactions, in the particular case of monotonic transactions. The impact of a transaction on the response time of a lower priority task under analysis is computed with the interference implied by the transaction. In the general context of tasks with offsets (general transactions), only exponential methods are known to calculate the exact worst-case response time of a task. However, in this case, Mäki-Turja and Nolin have proposed an efficient approximation method. A monotonic pattern in a transaction (regarding the priority of the task under analysis), occurs when, by rotation of the higher priority tasks in a transaction, it is possible to find a pattern of tasks such that the processor demand of the transaction is monotically decreasing during a period of the transaction. We have shown in our previous work that if a task under analysis is such that all the interfering transactions are monotonic, then it is possible to evaluate its exact response time in a pseudo-polynomial time. This article presents in detail how to apply this method. Then, it compares our results to the multiframe model proposed by Mok and Chen in [5] (AM "Accumulatively Monotonic" pattern). We show that the multiframe model is a particular instance of tasks with offsets but the results presented for AM multiframe cannot be applied on monotonic transactions. Finally, we show that the approximation method proposed by Mäki-Turja and Nolin computes an exact response time in the case of monotonic transactions, even if its complexity is higher than the one of the test that we proposed.**

I. Introduction

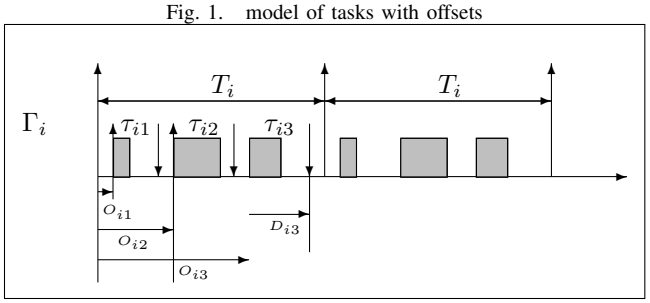
The validation process is an important step in the development of a real-time application. This validation process consists in proving that, whatever happens, the scheduling policy guarantees that all the temporal constraints are met. Usually, the task model is an extension of the model of Liu and Layland [4]. The schedulability conditions obtained with this model are however too pessimistic for certain kinds of pattern of tasks. Thus some authors proposed many other models of tasks: the multiframe model [5] [2], and generalized multiframe model [1], the model of tasks with self-suspension [7] [8] [9], the model of tasks with offsets (transactions) [10] [6] [13] [14]; the models of serial transactions and reverse transactions [11] which is a particular instance of the model of tasks with offset. Tindell [10] suggested the model of tasks with offsets; Palencia and Harbour[6] extended and formalized Tindell's work. Then, Turja and Nolin [13] improved the schedulability conditions by introducing the concept of "imposed interference" different from the "released for execution interference". This model of tasks with offsets is a general model allowing to obtain good results for a broad range of patterns of tasks. In a context of tasks with offsets, all the tasks bounded by relations of offsets form a transaction; and, since a classic task is a transaction containing only one task, a configuration is a set of transactions. For now, the method of determination of the exact worst-case response time of the tasks of a transaction set is exponential. Thus, pseudo-polynomial time approximation methods, giving more or less pessimistic schedulability conditions have been proposed. In any case,The concept of approximation leads to some pessimism. This paper is a complementary contribution for analyzing tasks with offsets. We show that, in certain cases, it is possible to use a method of calculation of the exact worst-case response time, and that this method has lower complexity than the complexity of the approximation methods. Moreover, we show in this article that the multiframe model is a particular case of the model of tasks with offsets, but that the results on the multiframe model cannot be directly applied on transactions., however, the results presented obtained on monotonic transactions can be closely related to the results obtained on the multiframe model. The structure of the article is as follows: in section 2, we present the model of tasks with offsets. Section 3 presents the previous work about monotonic transactions. In section 4, we show that the approximation method proposed in [15] gives an exact worst-case response time for monotonic transactions. In Section 5, we present the relation between the multiframe model and the model of tasks with offsets. Finally, we present the exact method of calculation of the worst-case response time for monotonic transactions on an example.

II. Background

A. Presentation of the model

The model of tasks with offsets was proposed by Tindell [10] in order to reduce existing pessimism of the schedulability analysis where the critical instant for a task occurs when it is released at the same time as all the higher priority tasks. Indeed, certain tasks can have for example the same period and be bound by relations of offsets i.e. they can never be released at the same time. A set of tasks of the

Fig. 1. model of tasks with offsets



same period bound by offset is called a transaction. The release of a transaction is bound to an external event (the transactions themselves are non-concrete), whose worst-case period of occurrence is the period of the transaction. A task system Γ is compound of a set of transactions Γ_i. [6][13]:

$$\Gamma := \{\Gamma_1, \Gamma_2, .., \Gamma_k\}$$

A transaction (see Figure 1) contains $|\Gamma_i|$ tasks of the same period (with $|E|$ is the cardinal of set E) :

$$\Gamma_i :=< \{\tau_{i1}, \tau_{i1}, ..., \tau_{i|\Gamma_i|}\}, T_i >$$

A task is defined by

$$\tau_{ij} :=< C_{ij}, \Phi_{ij}, D_{ij}, J_{ij}, B_{ij}, P_{ij} >$$

where C_{ij} is the worst-case execution time (WCET), Φ_{ij} is the offset (minimal time between the release of the transaction and the release of the task),D_{ij} is the relative deadline, J_{ij} the maximum jitter (giving t_0 the release date of an instance of the transaction Γ_i , then the task τ_{ij} is released between $t_0+\Phi_{ij}$ and $t_0+\Phi_{ij}+J_{ij}$), B_{ij} maximum blocking due to lower priority tasks, and P_{ij} the priority. It has been shown in [6] that it is equivalent , regarding to the worst-case response time analysis to consider $O_{ij} = \Phi_{ij}\%T_i$. Without loss of generality, we consider that the tasks are ordered by increasing offsets O_{ij} ; in our case, we define the response time as being the time between the release of the task and the completion of this task. Let us note also $hp_i(\tau_{ua})$ the set of indices of the tasks of Γ_i with a priority higher than the priority of a task under analysis τ_{ua} i.e. $j \in hp_i(\tau_{ua})$ if and only if $P_{ij} > P_{ua}$. (assuming that the priorities of the tasks are unique). We denote $|\Gamma_i|$ the number of tasks in a transaction.

In order to validate the system, the Response-Time Analysis (RTA) [3] method is to be applied on each task of the transactions. The task under analysis is usually noted τ_{ua}. Tindell showed that the critical instant of τ_{ua} is a particular instant when it is released at the same time as at least one task of higher priority in each transaction Γ_i. The main difficulty is to determine what is the critical instant candidate τ_{ic} of a transaction Γ_i that initiates the critical instant of τ_{ua}. An exact calculation method would require to evaluate the response time obtained by carrying out all the possible combinations of the tasks of priority higher in each transaction and to choose the task in each transaction that leads to the worst-case response time. This exhaustive method has an exponential complexity and is intractable for realistic task systems; several approximation methods giving an upper bound of the worst-case response time have been proposed. The best known approximation method is the upper bound method based on the "imposed interference".

B. Utlisation of the approximation method

The best known approximation method has been proposed in [13]. This method removes the unnecessary overestimation taken into account in the classic computation of the interference imposed by a task τ_{ij} on a lower priority task τ_{ua}. This overestimation does not have any impact in the case of tasks without offset but has a considerable effect in the approximation of the worst-case response time when we are in the presence of tasks with offsets. This method consists in calculating the interference effectively imposed by a task τ_{ij} on a task τ_{ua} with a lower priority during a time interval of length t; the idea is that this interference cannot exceed the interval of time t. In order to calculate this "imposed interference", [13] substracts a parameter x (see Figure 2) from the original interference formula; let us note $W_{ic}(\tau_{ua}, t)$ the interference that Γ_i imposes effectively on the response time of τ_{ua} during a time interval of length t when τ_{ic} is released at the same instant as τ_{ua} [13]. In a first study of transactions, we will focus on cases with no jitter(i.e $J_{ij} = 0$).

$$W_{ic}(\tau_{ua}, t) = \sum_{j \in hp_i(\tau_{ua})} \left(\left(\left\lfloor \frac{t^*}{T_i} \right\rfloor + 1 \right) * C_{ij} - x_{ijc}(t) \right)$$

$$t^* = t - phase(\tau_{ij}, \tau_{ic})$$

$$phase(\tau_{ij}, \tau_{ic}) = (T_i + (O_{ij} - O_{ic}))\ \%\ T_i$$

$$x_{ijc}(t) = \begin{cases} 0 & \text{for } t^* < 0 \\ \max(0, C_{ij} - (t^*\%T_i)) & \text{otherwise} \end{cases}$$

$x_{ijc}(t)$ corresponds to the part of the task τ_{ij} that cannot be executed in the time interval of length t; since this interference is not effectively imposed in this interval, it is not taken into account (see an example on Figure 2).

In order to determine the upper bound of the response-time, [13] uses this function :

$$W_i(\tau_{ua}, t) = \max_{c \in hp_i \tau_{ua}} (W_{ic}(\tau_{ua}, t))$$

With the value of each $W_i(\tau_{ua}, t)$, the upper bound of response-time R_{ua} of τ_{ua} can be calculated: R_{ua} is found by iterative fix-point lookup.

$$R^0_{ua} = C_{ua}$$

$$R^{(n+1)}_{ua} = C_{ua} + \sum_{\Gamma_i \in \Gamma} (W_i(\tau_{ua}, R^n_{ua}))$$

¨We apply this method on the example of figure 3. In this example, we have two transactions Γ_1 and Γ_2. Let us consider a task under analysis τ_{ua} with a WCET equal to 2; let us suppose that the priority of τ_{ua} is lower than

Table 1 : Application of "Imposed Interference" method

Iter	$W_{11}/_{21}$	$W_{12}/_{22}$	$W_{13}/_{23}$	$W_{14}/_{24}$	$W_{15}/_{25}$	$W_{16}/_{26}$	$W_{17}/_{27}$	W_{18}	W_{19}	$W_1/_2$	R_{ua}
0											2
1	2/1	2/1	2/1	2/1	2/1	2/1	2/2	2	2	2/2	6
2	4/2	4/2	4/2	4/2	4/2	4/3	4/3	4	4	4/3	9
3	5/3	5/3	5/3	5/3	5/3	5/4	5/3	6	4	6/4	12
4	6/3	6/3	6/3	6/3	6/5	6/4	8/3	6	4	8/5	15
5	8/4	8/4	8/4	8/6	8/5	9/4	8/3	6	4	9/6	17
6	9/5	9/5	9/5	9/6	9/5	10/4	8/3	6	4	10/6	18
7	10/5	10/5	10/6	10/6	10/5	10/4	8/3	6	4	10/6	18

Fig. 2. "Imposed interference" method on a transaction of 4 tasks

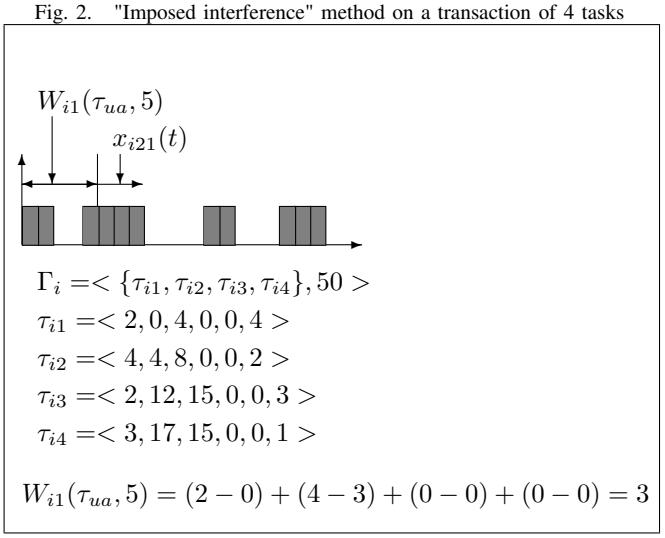
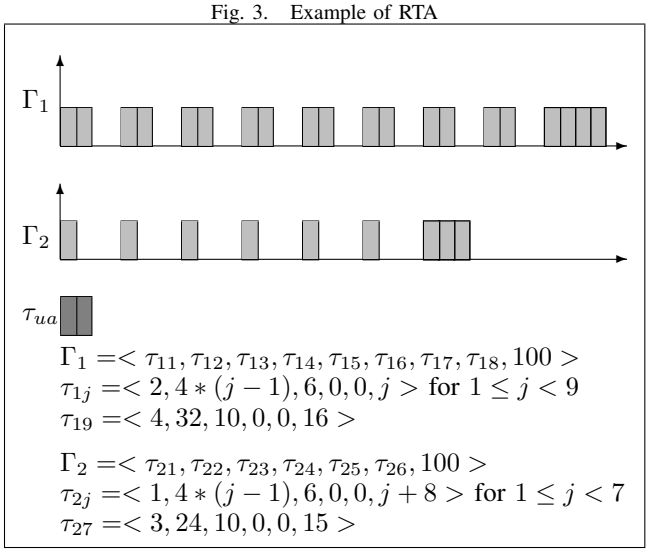


Fig. 3. Example of RTA



the priority of each task of the transactions Γ_1 and Γ_2, for instance ($Priority(\tau_{ua}) = 0$). We present the details of the two first significant iterations in the process of the application of this method.

Iteration 0 : $R^0_{ua} = 2$

Iteration 1 :

Evaluation of $W_2(\tau_{ua}, 2)$:

$W_{2j}(\tau_{ua}, 2) = (1-0) + (0-0) + ... + (0-0) = 1$
for $1 \leq j < 7$

$W_{27}(\tau_{ua}, 2) = (3-1) + (0-0) + ... + (0-0) = 2$

$W_2(\tau_{ua}, 2) = \max_{1 \leq j < 9} (W_{2j}(\tau_{ua}, 2))$
Thus $W_2(\tau_{ua}, 2) = 2$

And $R^{(1)}_{ua} = C_{ua} + W_1(\tau_{ua}, 2) + W_2(\tau_{ua}, 2)$; consequently $R^{(1)}_{ua} = 2 + 2 + 2 = 6$.

Iteration 2 :

Evaluation of $W_1(\tau_{ua}, 6)$:

$W_{1j}(\tau_{ua}, 6) = (2-0) + (2-0) = 4$
for $1 \leq j < 8$

$W_{18}(\tau_{ua}, 6) = (2-0) + (4-2) = 4$

$W_{19}(\tau_{ua}, 2) = (4-0) = 4$

Thus $W_1(\tau_{ua}, 6) = 4$

Evaluation of $W_2(\tau_{ua}, 6)$:

$W_{2j}(\tau_{ua}, 6) = (1-0) + (1-0) = 2$

for $1 \leq j < 6$

$W_{26}(\tau_{ua}, 2) = (1-0) + (3-1) = 3$

$W_{27}(\tau_{ua}, 2) = (3-0) = 3$

Thus $W_2(\tau_{ua}, 6) = 3$

Then $R_{ua}^{(2)} = 2 + 4 + 3 = 9$.

We present in the table 1 the values obtained in the following iterations. The upper bound of the worst-case response time obtained with this method is thus equal to 18.

However, the exact worst-case response time of the task τ_{ua} (obtained in testing every combinations) is equal to 14. This value is obtained when τ_{ua} is released at the same time as τ_{17} and τ_{26}. The "imposed interference" method is less pessimistic than the others methods of approximation but its application needs the evaluation of the value of $x_{ijc}(t)$ for each iteration and for each task. Moreover, the application of these methods of approximations on some concrete real-time application is sometimes unnecessary. Indeed, in certain cases there is a tractable method for determining the real worst-case response time; this method is less complex than all known approximation methods.

III. Monotonic transactions

Let Γ_i be a transaction and τ_{ua} a task under analysis; in order to simplify the notation, we consider that all the tasks of Γ_i have a higher priority than the task under analysis τ_{ua}. Moreover, we assume that the load of the configuration is less than 1.

A. Normalisation of the transaction

A similar normalization process has been used in [15], with a difference because the authors split the tasks when they can end after the period of the transaction, while we don't use the split part. **Definition :** The transaction Γ_i is in normal form if $O_{ij} + C_{ij} < O_{i(j+1)}$ for $1 \leq j < |\Gamma_i|$ and $O_{i|\Gamma_i|} + C_{i|\Gamma_i|} < T_i + O_{i1}$
For example the transactions Γ_1,Γ_2 of Figure 3 and the transaction Γ_i of Figure 4 are in normal form. While the transaction Γ_i of Figure 9 is not in normal form; indeed, we have for example $O_{i3} + C_{i3} > O_{i4}$.
Let us suppose that there is a task τ_{ij} such as $O_{ij} + C_{ij} \geq O_{i(j+1)}$ in a transaction Γ_i; according to theorem 1 of [10], the busy period starting at O_{ij} contains the busy period starting at $O_{i(j+1)}$. Consequently, the task $\tau_{i(j+1)}$ cannot initiate the critical instant for the task τ_{ua}; therefore it is useless to evaluate $W_{i(j+1)}(\tau_{ua}, t)$ in the process of calculation of the worst-case response time. For this reason, if a transaction Γ_i is not in normal form, we group the tasks of Γ_i in order to obtain a normal form before starting the iterative lookup of the fix-point.
Let Γ_i^* be the normal form of Γ_i. Γ_i^* is obtained as follows: Γ_i^* is first initialized with the value of Γ_i :

$$\Gamma_i^* :=< \{\tau_{i1}^*, \tau_{i2}^*, ..., \tau_{i|\Gamma_i|}^*\}, T_i >$$
$$with \quad \tau_{ij}^* = \tau_{ij} \quad for \quad 1 \leq j \leq |\Gamma_i|$$

Process of normalization :

- **Step 1** : for $1 \leq j < |\Gamma_i^*|$, if $O_{ij}^* + C_{ij}^* \geq O_{i(j+1)}^*$ then merge $\tau_{i(j+1)}^*$ into τ_{ij}^*. These two tasks form one task starting at O_{ij}^* with a WCET equal to $C_{ij}^* + C_{i(j+1)}^*$. Renumber the tasks of the transaction in increasing order of O_{ij}^* because $\tau_{i(j+1)}^*$ is deleted
- **Step 2** :
  - if $O_{i|\Gamma_i^*|}^* + C_{i|\Gamma_i^*|}^* \geq T_i + O_{i1}^*$ then merge τ_{i1}^* into $\tau_{i|\Gamma_i^*|}^*$. $C_{i|\Gamma_i^*|}^* = C_{i|\Gamma_i^*|}^* + C_{i1}^*$. Renumber the tasks of the transaction and start again the step 2
  - otherwise it is the end of the process

This process converges, since we cannot merge any task if there is only one task left. The transaction of the figure 10 is the normal form of the transaction of figure 9.

B. Monotonic pattern

Definition:
Let $\Gamma_i =< \{\tau_{i1}, \tau_{i2}, ..., \tau_{i|\Gamma_i|}\}, T_i >$ be a transaction and τ_{ua} a task under analysis. Without loss of generality, we consider that all the tasks of Γ_i have a higher priority than the one of τ_{ua}. Let $\Gamma_i^* =< \{\tau_{i1}^*, \tau_{i2}^*, ..., \tau_{i|\Gamma_i^*|}^*\}, T_i >$ be the normal form of the transaction Γ_i. Let us note:

- $\alpha_{ij} = O_{i(j+1)}^* - (O_{ij}^* + C_{ij}^*) \qquad for\ 1 \leq j < |\Gamma_i^*|$
- $\alpha_{i|\Gamma_i^*|} = (T_i + O_{i1}^*) - (O_{i|\Gamma_i^*|} + C_{i|\Gamma_i^*|})$

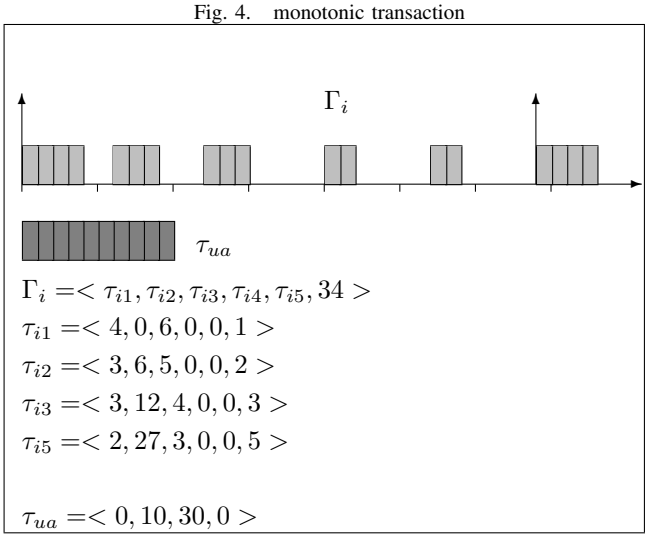
Note that $\alpha_{ij} > 0$ since Γ_i^* is in normal form.
Γ_i is a monotonic transaction for the task τ_{ua} if the WCET of Γ_i^* have decreasing values while the phases α_{ij} have increasing values i.e:

- $C_{i(p+1)}^* \leq C_{ip}^* \qquad for\ \ all\ \ 1 \leq p < |\Gamma_i^*|$
- $\alpha_{ip} \leq \alpha_{i(p+1)} \qquad for\ \ all\ \ 1 \leq p < |\Gamma_i^*|$

Example of monotonic transaction : (see Figure 4) in this example, the task τ_{ua} is a lower priority task for all the tasks of Γ_i; moreover, Γ_i is already in normal form: $\Gamma_i = \Gamma_i^*$. we have $C_{i1} \geq C_{i2} \geq C_{i3} \geq C_{i4} \geq C_{i5}$ and $\alpha_{ip} \leq \alpha_{i(p+1)} \qquad for\ \ all\ \ 1 \leq p < |\Gamma_i^*|$. Therefore, according to the definition of monotonic transaction, Γ_i is monotonic for the task τ_{ua}.

For the transaction $\Gamma_i^* :=< \{\tau_{i1}^*, \tau_{i2}^*, ..., \tau_{i|\Gamma_i|}^*\}, T_i >$, there is no difference, regarding the worst interference pattern, to consider the periodic transaction obtained by rotation

Fig. 4. monotonic transaction



consisting in considering the origin of the transaction being the offset of the task τ_{ik}:

$$\Gamma_i^{*r_k} :=< \{\tau_{ik}^{*r_k}, \tau_{i(k+1)}^{*r_k}, ..., \tau_{i|\Gamma_i|}^{*r_k}, \tau_{i1}^{*r_k}, \tau_{i2}^{*r_k}, ..., \tau_{i(k-1)}^{*r_k}\}, T_i >$$

where the only difference between the original transaction Γ_i^* and its rotation $\Gamma_i^{*r_k}$ is the value of the offsets, chosen such that $r_{ij}^{*r_k} = 0$ with respect to the periodic load pattern of the tasks:

$$r_{ij}^{*r_k} = (T_i + (O_{ij}^* - O_{ik}^*))\%T_i$$

We can rotate the tasks of the transaction Γ_i^* without modifying the interference imposed by Γ_i^* on the tasks having a lower priority. For this reason, we consider that Γ_i is monotonic if we can find a monotonic pattern in Γ_i^* by rotating the tasks of Γ_i^*. Looking for a monotonic pattern is trivial (we know that the first task has the highest WCET), thus in the sequel, we simplify the notation $\Gamma_i^{*r_k}$ for the monotonic pattern of a normalized transaction in writing Γ_i^*.

For example figure 10 shows a monotonic pattern starting from the task τ_{i2}^*; thus, the transaction Γ_i of the figure 9 is monotonic (the transaction of the figure 10 is its normal form).

C. Results for monotonic transaction

In this section, we present a simple RTA method used when a transaction Γ_i is monotonic for a task under analysis τ_{ua}.

Theorem 1: Let $\Gamma_i =< \{\tau_{i1}, \tau_{i1}, ..., \tau_{i|\Gamma_i|}\}, T_i >$ be a transaction and τ_{ua} a task under analysis. Let Γ_i^* be the monotonic normal form of transaction Γ_i in regard to a task under analysis τ_{ua}. The critical instant of τ_{ua} occurs when it is released at the same time as the first task of Γ_i^*.

Proof : The main idea of the proof of the theorem is that the load pattern is higher at the beginning of the transaction than anywhere else. The details of the proof can be found in [12].

Corollary: Let τ_{ua} be a task of a task set. if all the transactions of the task set are monotonic for the task τ_{ua}, then the worst-case response time obtained by supposing that τ_{ua} is released at the same time as the first task of each transaction is exact.

In fact, in the process of fix-point lookup of the approximation method proposed in [13] [14] [15] [6] [10], the task that initiates the critical instant may vary from iteration to iteration, but the interference takes the maximum function of all the interference functions in order to avoid an exponential complexity. This situation can lead to an unrealistic upper bound of the worst-case response time. In the case of monotonic transaction, the task that initiates the critical instant does not change from iteration to iteration; so, there is no pessimism in the fix-point lookup process and the worst-case response time obtained is exact.

IV. "Imposed interference" method on monotonic transactions

Figure 5 shows the interference due to the transaction presented on figure 2 (supposing that all the tasks of the transaction have a greater or same priority as a task under analysis): each line represents the interference imposed, regarding the length of the busy period, on a task under analysis when it is released at the same time as a task of the transaction. The approximation methods use the maximum value of the interference functions. Figure 6 shows the interference functions of the monotonic transaction of figure 10.

We can see clearly that the task giving the maximum value of interference depends on the length of busy period (see Figure 5). For example, from 1 to 6, $W_{i2}(\tau_{ua}, t)$ has the greatest value and from 6 to 15, $W_{i2}(\tau_{ua}, t)$ has the greatest value. For this reason, the worst case candidate in each transaction changes from iteration to iteration during a RTA. In such an analysis, some unrealistic cases can be reached, because the task under analysis could be executed in idle slots between the release of the tasks of the transactions.

In the particular case of monotonic transactions, the task that initiates the critical instant is always the same, thus the maximum interference function corresponds to a single case, which is always realistic (see on Figure 6 that $W_{i2}^*(\tau_{ua}, t)$ has the greatest value). Therefore, if all the transactions of a task set are monotonic for a given task τ_{ua}, the task that initiates the critical instant doesn't change from iteration to iteration; then the worst-case response time calculated with the method of [15] is exact. The difference between this method and the method presented in section III-C is the number of steps in each iteration needed for determining the worst-case response time (there is no need to compute every cases, but only to focus on the first task of the monotonic pattern).

Fig. 5. Representation of interference function

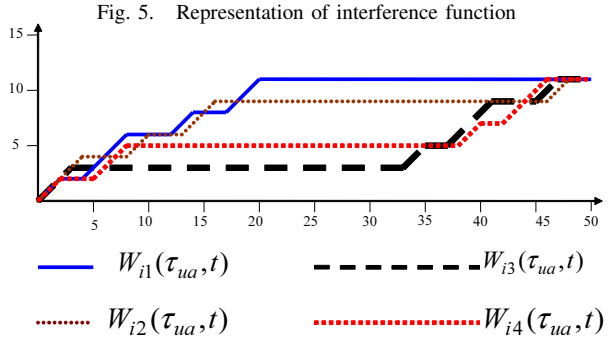
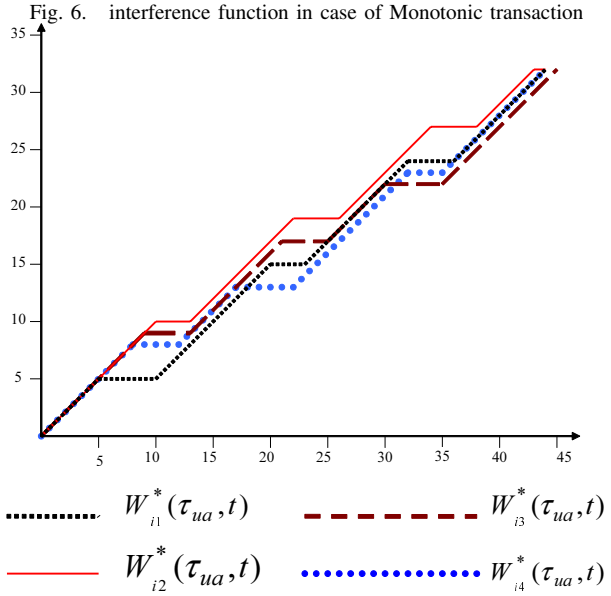


Fig. 6. interference function in case of Monotonic transaction



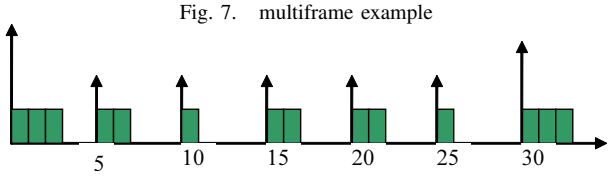
V. Comparison with the results of the multiframe model

The multiframe model has been proposed in [5] in order to reduce pessimism in the schedulability conditions when the WCET is significantly higher than the average-case execution time of a task.

Definition : A multiframe real-time task is a tuple (Γ,P) where Γ is an array of N execution times (C^0,C^1,...,C^{N-1}) for some $N \geq 1$, and P is the minimum separation time,(the ready times of two consecutive frames must be at least P time units apart). The execution time of the i^{th} frame of the task is $C^{((i-1)\%N)}$ where $1 \leq i$. The deadline of each frame is P after its ready time.

A multiframe task Γ is characterized by a finite number of execution time and by the period of the task. A multiframe task with N execution time and a period P is noted $\Gamma = ((C^0, C^1, ..., C^{N-1}), P)$.

Fig. 7. multiframe example



Example : Let $\Gamma = ((3,2,1,2,2,1),26)$ be a multiframe task. We present Γ in the figure 7.

Let us note $C^m = \max_{i=0}^{N-1} C^i$ the peak execution time of task Γ. One of the main result presented by Mok and Chen in [5] is for the multiframe task presenting a "Accumulatively Monotonic" pattern.

Definition : Let C^m be the maximum in an array of execution times $(C^0, C^1, ..., C^{N-1})$. This array is said AM(Accumulatively Monotonic) if

$$\sum_{k=m}^{m+j} C^{(k\%N)} \geq \sum_{k=i}^{i+j} C^{(k\%N)},\ 1 \leq i \leq N-1,\ 1 \leq j \leq N-1$$

A task Γ is said to be AM if its array of execution times is AM.
The multiframe task presented in the figure 7 is a AM multiframe.
If Γ is a multiframe task with a AM pattern, we will rotate the execution time of Γ such as C^0 be the peak execution time.

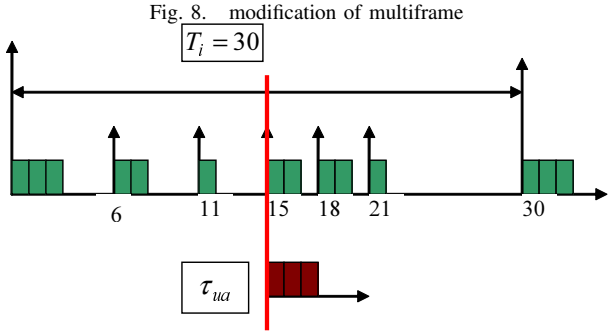
Result for AM pattern [5] :
Let Γ be a AM multiframe task and τ_{ua} a classical task under analysis. Let us suppose that the priority of τ_{ua} is lower than the priority of each instance of Γ; then the critical instant of τ_{ua} occurs when τ_{ua} is released at the same time as the first instance of Γ.

We can see that the multiframe model is a particular instance of the tasks with offsets. One of the main particularity is that

$$O_{i+1} - O_i = P \ for\ 1 \leq i < N \ and\ T_i - O_N = P$$

If this particularity is not satisfied for a task with offsets, we cannot apply the result(for AM pattern) presented in the previous section. For example, figure 8 shows a task with offset Γ_i.
$\Gamma_i =< \{\tau_{i1}, \tau_{i2}, \tau_{i3}, \tau_{i4}, \tau_{i5}, \tau_{i6}\}, 30 >$
$\tau_{i1} =< 3,0,5,0,0,1 >$
$\tau_{i2} =< 2,6,10,0,0,2 >$
$\tau_{i3} =< 1,11,15,0,0,3 >$
$\tau_{i4} =< 2,15,16,0,0,4 >$
$\tau_{i5} =< 2,18,17,0,0,5 >$
$\tau_{i6} =< 1,21,18,0,0,6 >$
The execution times in Γ_i are the same as for Γ in the figure 7.
In the transaction Γ_i, the difference between the releases of

Fig. 8. modification of multiframe



two consecutive tasks is not constant.

$$\exists\ i\ such\ as\ O_{i+1} - O_i \neq O_{i+2} - O_{i+1} \quad 1 \leq i < 5$$

Let τ_{ua} be a task under analysis with a WCET $C_{ua} = 3$ with a lower priority than Γ_i. We can see that the critical instant of τ_{ua} does not coincide with the release of the first task of the transaction Γ_i. The critical instant of τ_{ua} occurs in this case when it is released at the same time as the fourth task of Γ_i.

So, it is clear that we cannot apply directly the results found for "AM multiframe" on the general model of tasks with offsets without taking into account the values of offsets between the tasks of a transaction. Monotonic transaction does this consideration and appears like an analog result to the one presented for multiframe model. Note that the multiframe model does not provide suitable results for serial transactions [11] either.

VI. Applications of the method

In this section we apply the method of monotonic transaction on an example. Let

$$\Gamma_i = \{< \tau_{i1}, \tau_{i2}, \tau_{i3}, \tau_{i4}, \tau_{i5}, \tau_{i6}, \tau_{i7}, \tau_{i8} >, 50\}$$

be a transaction. The tasks of Γ_i are defined as (see Figure 9):
$\tau_{i1} =< 2, 1, 10, 0, 0, 11 >$
$\tau_{i2} =< 5, 9, 10, 0, 0, 12 >$
$\tau_{i3} =< 5, 19, 10, 0, 0, 13 >$
$\tau_{i4} =< 7, 23, 10, 0, 0, 14 >$
$\tau_{i5} =< 1, 34, 10, 0, 0, 15 >$
$\tau_{i6} =< 8, 35, 10, 0, 0, 18 >$
$\tau_{i7} =< 5, 47, 10, 0, 0, 17 >$
$\tau_{i8} =< 1, 48, 10, 0, 0, 18 >$

Let τ_{ua} be a task under analysis with a WCET $C_{ua} = 8$ and a lower priority than all the tasks of Γ_i.

The application of the "imposed interference" method gives the table 2.

Table 2 : "Imposed Interference" method

Iter #	W_{i1}	W_{i2}	W_{i3}	W_{i4}	W_{i5}	W_{i6}	W_{i7}	W_{i8}	W_i	R_{ua}
0										8
1	2	5	9	7	8	8	8	3	9	17
2	7	13	14	13	15	15	13	8	15	23
3	11	17	20	16	17	16	14	14	20	28
4	19	20	21	21	19	19	20	16	21	29
5	19	21	22	23	20	20	21	17	23	31
6	19	23	25	24	22	21	23	19	25	33
7	19	25	28	24	22	21	25	20	28	36
8	22	26	29	24	23	23	25	20	29	37
9	23	26	29	25	24	24	25	21	29	37

Now, we compute the worst-case response using the monotonic transactions characteristics.
Steps of the application of the method:

Step 1: We group the tasks of Γ_i in order to obtain a normal form and we obtain the transaction shown on Figure 10:

$\Gamma_i^* = \{< \tau_{i1}^*, \tau_{i2}^*, \tau_{i3}^*, \tau_{i4}^* > 50\}$
$\tau_{i1}^* =< 5, 9, x, 0, 0, x >$
$\tau_{i2}^* =< 12, 19, x, 0, 0, x >$
$\tau_{i3}^* =< 9, 34, x, 0, 0, x >$
$\tau_{i4}^* =< 8, 47, x, 0, 0, x >$
(see Figure 10)

Step 2: Looking for a monotonic pattern We have :

$$C_{i2}^* \geq C_{i3}^* \geq C_{i4}^* \geq C_{i1}^*$$

$$and \qquad \alpha_{i2}^* \leq \alpha_{i3}^* \leq \alpha_{i4}^* \leq \alpha_{i1}^*$$

A monotonic pattern starts from task τ_{i2}^*. Consequently, the critical instant of the task τ_{ua} coincides with the release of the task τ_{i2}^*. We apply the iterative fix-point lookup with the method presented in this article(see Table 3).

Table 3 : New method

Iter #	I_{12}	R_{ua}
0		8
1	12	20
2	21	29
3	29	37
4	29	37

Let us note that $I_{i2}(\tau_{ua}, t)$ is the value obtained with a classical RTA method.

$$I_{i2}(\tau_{ua}, t) = \sum_{j=1}^{|\Gamma_i^*|} \left(\left\lceil \frac{t^*}{T_i} \right\rceil \cdot C_{ij} \right)$$

With our method, we only have to calculate $I_{i2}(\tau_{ua}, t)$ at each iteration instead of calculating eight values of $W_{ij}(\tau_{ua}, t)$ at each iteration. Moreover, for the calculation of each $W_{ij}(\tau_{ua}, t)$ it is necessary to evaluate $|\Gamma_i|$ times

Fig. 9. Transaction Γ_i

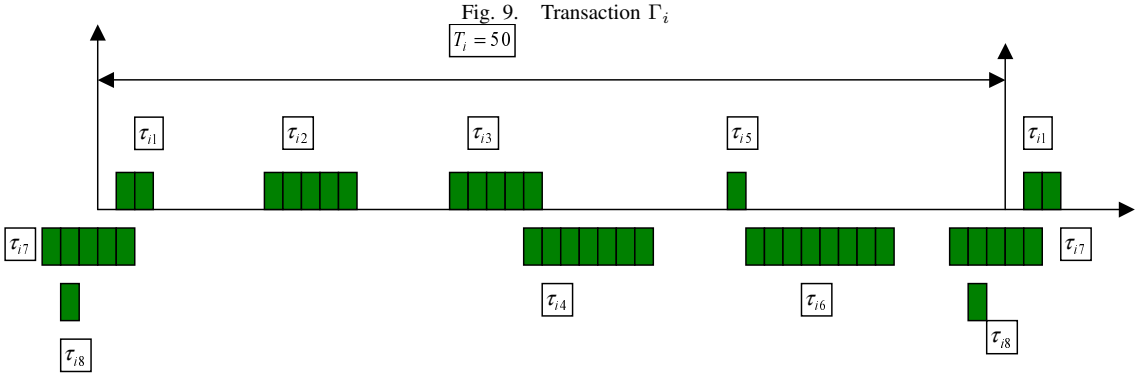
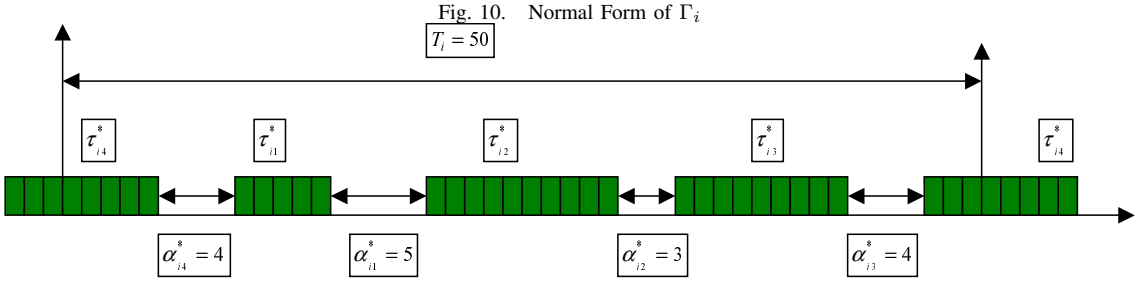


Fig. 10. Normal Form of Γ_i



the value of $x_{ijc}(t)$. This evaluation is no longer necessary with the new method. The number of steps in the fix-point lookup is significantly lower. Finally, let us note that RTA analysis is exact.

A concrete example of application of monotonic transaction can be found in the intermediate priority tasks in a serial transaction in [11].

VII. Conclusion

In a general context of tasks with offsets, the RTA methods are intractable because they are exponential in time. This article focuses on monotonic transactions (which could be compared, in the model of transactions, to the AM class of multiframe tasks). For this class, there is an exact and pseudo-polynomial RTA method which requires less steps than the known approximation methods for the general case. This method consists in grouping at first the tasks of the transaction in a normal form. If the normal form presents a monotonic pattern, the critical instant occurs when the task under analysis is released at the same time as the first task of the pattern. We noted also that the test proposed in [15] gives an exact result too in the case of monotonic transactions.

In our future work on tasks with offsets, we will investigate new classes in order to find less pessimistic schedulability conditions with a lower complexity. Moreover, we will try to extend this method to transactions with jitters.

References

[1] S. Baruah, D. Chen, S. Gorinsky and A. Mok, Generalized Multiframe Tasks, The International Journal of Time-Critical Computing Systems,17,5-22 (1999)

[2] C.C. Han and H.Y. Tyan, A Better Polynomial-Time Schedulability Test for Real-Time Fixed-Priority Scheduling Algorithms, Proc. IEEE Real-Time Systems Symp., pp. 36-45, Dec. 1997.

[3] M. Joseph, and P. Pandya, Finding Response Time in a Real-Time System,Computer journal, Vol. 29, No.5, pp.390-395, Oct.1986

[4] C.L. Liu and J.W. Layland, Scheduling algorithms for mutltiprogramming in real-time environnement, Journal of the ACM 20(1) (1973), 46-61.

[5] A.K. Mok and D. Chen., A Multiframe Model for Real-Time Tasks. In IEEE Real-Time Systems Symposium, pages 22-29, Dec. 1996

[6] J. Palencia Gutierrez and M. Gonzalez Harbour, Schedulability Analysis for Tasks with Static and Dynamic Offsets. In Proc. 19th IEEE Real-Time System Symposium (RTSS), December 1998

[7] F. Ridouard, P. Richard, and F. Cottet, Negative results for scheduling independent hard real-time tasks with self-suspensions. Proceedings of the 25th IEEE International Real-Time Systems Symposium (RTSS 04), 1, December 2004.

[8] F. Ridouard, P. Richard, and F. Cottet, Scheduling independent tasks wit selfsuspension. Proceedings of the 13rd RTS Embedded Systems (RTS 05), 1, April 2005 (in french).

[9] F. Ridouard, P. Richard, F. Cottet and K. Traoré, Some results on scheduling tasks with self-suspensions, Journal of Embedded Computing, 2006

[10] K. Tindell, Addind Time-Offsets to Schedulability Analysis, Technical Report YCS 221, Dept of Computer Science, University of York, England, January 1994

[11] K. Traoré, E. Grolleau and F. Cottet, Schedulability Analysis of Serial Transactions, Real-Time and Network Systems, RTNS'06, Poitiers, France, May 30-31, 2006, pp. 141-149

[12] K. Traoré, E. Grolleau and F. Cottet, Characterization and Analysis of Tasks with Offsets : Monotonic Transactions, proc. 12th International Conference on Embedded and Real-Time Computing Systems and Applications, RTCSA 2006, August 16-18th 2006, Sydney, Australia.

[13] J. Mäki-Turja and M. Nolin, Tighter Response Time Analysis of Tasks with Offsets. In Proc. 10th International conference on Real-Time Computing and Applications (RTCSA'04), August 2004

[14] J. Mäki-Turja and M. Nolin, Faster Response Time Analysis of Tasks with Offsets. In Proc. 10th IEEE Real-Time Technology and Applications Symposium (RTAS), May 2004

[15] J. Mäki-Turja and M. Nolin, Fast and Tight Response-Times for Tasks with Offsets. In 17th EUROMICRO Conference on Real-Time Systems, p 10, IEEE, Palma de Mallorca Spain, July 2005