A Task Model for TDMA Communications

Shuai Li†, Stéphane Rubini†, Frank Singhoff†, Michel Bourdellès*
*Thales Communications & Security, 4 av. des Louvresses, 92622 Gennevilliers, France
†Lab-STICC/UMR 6285, UBO, UEB, 20 av. Le Gorgeu, 29200 Brest, France
Email: {first-name}.{last-name}@fr.thalesgroup.com
Email: {last-name}@univ-brest.fr

Abstract—In this article a new task model, called DGMF, is proposed to improve scheduling analysis of dependent tasks in radio stations that embed a TDMA communication protocol. TDMA is a channel access protocol that allows several stations to communicate in a same network, by dividing time into several time slots. This protocol has an impact on task release times, execution times, and deadlines, which needs to be considered for less pessimistic scheduling analysis results. We experiment on software radio protocols from Thales, which are representative of the system we want to analyze.

I. INTRODUCTION

This article proposes to improve scheduling analysis of systems with Time Division Multiple Access (TDMA) communications. In this kind of system, task release times, execution times, deadlines depend on the TDMA communication protocol. Furthermore, we focus on TDMA Software Radio Protocols (SRP) [7] where tasks have dependencies (shared resource and precedence).

A TDMA SRP is both a time-triggered [3] (TDMA) and event-triggered [3] (task precedence) system, with variable task parameters. Common scheduling analysis techniques are either limited to the time-triggered aspect [10], [17], event-triggered aspect [11], [13], [9], or do not allow task dependencies or variable task parameters [8], [4].

To specify more accurately tasks in such systems, a new task model is proposed: the Dependent General Multiframe (DGMF) model, which extends the General Multiframe (GMF) model [1] with task dependencies. This task model is applied to TDMA SRPs from Thales.

The rest of the article is organized as follows: in Section II our system architecture and assumptions are exposed. In Section III the DGMF task model is proposed. Section IV shows how to perform scheduling analysis of DGMF tasks. Evaluation of the proposed model is done in Section V. We conclude with future works in Section VI.

II. SYSTEM ARCHITECTURE AND ASSUMPTIONS

In this article, we consider SRPs embedded in radio stations. These radio stations communicate in a mobile ad-hoc wireless network. We assume that the effects of non-determinism in wireless networks, on scheduling analysis of a single station, are negligible. A SRP is a software that implements a communication protocol. TDMA is a common communication protocol in SRPs.

In this protocol, a TDMA frame is divided into several time slots of different types and durations. When several stations want to communicate, control and data flows are handled by different tasks. The tasks are constrained by the TDMA frame, e.g. a release depends on a slot start time, an execution time depends on a slot type, a deadline depends on a slot duration. Tasks may communicate (i.e. precedence dependency) and use shared resources protected by a mechanism [14].

Tasks run on an execution platform for which we make some assumptions. A task is allocated on a processor that is scheduled by a preemptive fixed priority policy. There are several processors, i.e. the system is partitioned.

III. DEPENDENT GENERAL MULTIFRAME

The task model we propose to specify and analyze a TDMA SRP is called DGMF, an extension of the GMF model.

A DGMF task $G_i$ is a vector composed of $N_i$ frames $F_j^i$, with $1 \leq j \leq N_i$. Each frame has some parameters.

$E_i^j$ is the Worst Case Execution Time (WCET) of $F_j^i$,$D_i^j$ the deadline, and $P_i^j$ the minimum separation time between releases of $F_j^i$ and $F_j^{i-1}$ [1].

$\mathcal{U}_p^i$ is a set of $(R, S, B)$ tuples denoting shared resource critical sections. $F_j^i$ allocates resource $R$ after it has run $S$ time units of its execution time, and then locks the resource during the next $B$ time units of its execution time.

$prio(F_j^i)$ is the priority of $F_j^i$ and $proc(F_j^i)$ is the processor on which $F_j^i$ is allocated on.

The first frame to be released by a DGMF task $G_i$ is denoted $F_1^i$. We call $r_1^i$, the release time of first frame $F_1^i$. We call $r_i$, the release time of $G_i$ and thus $r_i = r_1^i$. For any $F_j^i$, with $j > 1$, its release time is: $r_j^i = r_1^i + \sum_{h=1}^{j-1} P_h^i$. For any $F_j^i$, we call value $r_j^i + D_j^i$ the global deadline of $F_j^i$.

A DGMF task set may have the following property:

Property 1 (Unique Predecessor): Let $F_2^i$ be a frame of a task $G_i$, in a DGMF task set. Let $\mathcal{F}_p^{i,i}$ be the set of predecessor frames of $F_2^i$. $F_{j-1}^i$ is the previous frame of $F_j^i$ in the vector of $G_i$, if $j > 1$. The set of frames that precede $F_2^i$ is the set $\mathcal{F}_p^{i,j}$ and $F_{j-1}^i$ (if $j > 1$). A DGMF task set is said to respect the Unique Predecessor property if, for all frames $F_j^i$, there is at most one frame $F_{k}^i$, among frames that precede $F_j^i$, with a global deadline (i.e. $r_2^i + D_2^i$) greater than or equal
to the release time of $F_1^i$. Formally the Unique Predecessor property is defined as:

$$\exists t^i_1 (F_1^j) \in \text{pred}(F_1^i), r^i_1 + d^i_1 \geq \max \{ \max_{F_1^j \in \text{pred}(F_1^i)} (r^j_1 + E^j_1), r^i_1 \}$$

(1)

Where $\exists t^i_1$ means "there exists at most one", and the set $\text{pred}(F_1^i)$ is defined as:

$$\text{pred}(F_1^i) = \begin{cases} [F_1^{j,i}] \cup \{F_1^{j-1}\} & \text{if } j > 1 \\ [F_1^{j,i}] & \text{otherwise.} \end{cases}$$

(2)

We assume that TDMA SRPs are modeled with DGMF task sets having the Unique Predecessor property.

Consider the DGMF task set in Table I, modeling tasks constrained by a TDMA frame.

**TABLE I**

DGMF TASK SET

<table>
<thead>
<tr>
<th>$E_1^i$</th>
<th>$D_1^i$</th>
<th>$P_1^i$</th>
<th>$[U_1^i]$</th>
<th>$[F_1^i]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F_{G1}; r_1 = 0$</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>$F_2^i$</td>
</tr>
<tr>
<td>$F_{G2}; r_2 = 0$</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>$F_2^i$</td>
</tr>
<tr>
<td>$F_{G3}; r_3 = 4$</td>
<td>1</td>
<td>2</td>
<td>6</td>
<td>$F_2^i$</td>
</tr>
<tr>
<td>$F_{G4}; r_4 = 4$</td>
<td>4</td>
<td>8</td>
<td>8</td>
<td>$(R, 1, 3)$ $F_2^i$</td>
</tr>
<tr>
<td>$F_{G5}; r_5 = 4$</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>$F_2^i$</td>
</tr>
<tr>
<td>$F_{G6}; r_6 = 4$</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>$F_2^i$</td>
</tr>
<tr>
<td>$F_{G7}; r_7 = 4$</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>$(R, 0, 1)$ $F_2^i$</td>
</tr>
</tbody>
</table>

All frames are allocated on CPU1 except $F_7^i$, which is allocated on CPU2. Fig. 1 shows an example of a schedule produced by the task set, over 20 time units. In Fig. 1, there is a TDMA frame of 1 S slot, 2 B slots, and 3 T slots. $G_2$ is released at S and T slots. $G_2$ releases $G_1$ upon completion. $G_4$ is released at B slots. $G_4$ releases $G_3$ upon completion.

**IV. DGMF SCHEDULING ANALYSIS**

Original analysis techniques for GMF tasks [1], [16] cannot be directly applied to DGMF tasks due to task dependencies.

In [12] a GFM to transaction transformation algorithm is proposed. The author argues that scheduling analysis techniques for transactions can be applied to GMF tasks. We propose to use transactions, and their associated scheduling analysis techniques, to assess schedulability of DGMF tasks. The transformation in [12] is thus extended for DGMF.

A. Transaction Definitions

A transaction [17] (denoted $\Gamma_i$) is a set of periodic tasks (denoted $\tau_{ij}$). A transaction is released by a periodic event that occurs every $T_i$. A particular instance of a transaction is called a job.

An instance of a task in a transaction is released after the event that releases the job of the transaction; if the event that releases the $p^{th}$ job of $\Gamma_i$ occurs at $t_0$, then the $p^{th}$ instances of its tasks are released at or after $t_0$. We call $r_i$ the release time of the first job of transaction $\Gamma_i$ and $r_j = r_i + O_j^i$ the release time of the first instance of $\tau_{ij}^i$.

Tasks in a transaction are related by precedence dependency. A precedence dependency between two tasks, denoted $\tau_{ip} \prec \tau_{ij}$, is a constraint that means that $\tau_{ip}$ (predecessor) must finish before $\tau_{ij}$ (successor) can be released.

A periodic task’s shared resource critical section is a tuple $(\tau, R, S, B)$ where $\tau$ is the task using the resource, $R$ the resource, $S$ the resource allocation time and $B$ the resource blocking time.

Each periodic task is defined by some parameters [17]. $C_j^i$ is the WCET, $O_j^i$ is the offset, $d_j^i$ is the deadline (and value $O_j^i + d_j^i$ is called the global deadline), $J_j^i$ is the jitter, $B_j^i$ is the worst case blocking time [14], $\text{prio}(\tau_j^i)$ is the priority, and $\text{proc}(\tau_j^i)$ is the processor of $\tau_j^i$.

**B. DGMF To Transaction**

To perform scheduling analysis of DGMF tasks, we transform them to transactions in several steps. The full transformation algorithm, with proofs, can be found in [6].

**Step 1** of the transformation is an extension of [12]. Each DGMF task $G_i$ is transformed into transaction $\Gamma_i$ with period $T_i \gets \sum_{j=1}^{N_i} P_j^i$ and release time $r_i$ equal to the release time of $G_i$. Frames are transformed into periodic tasks. Frame parameters $E_j^i, D_j^i, P_j^i$ are expressed with periodic task parameters $C_j^i \leftarrow E_j^i, O_j^i \leftarrow \sum_{h=1}^{j-1} P_h^i, d_j^i \leftarrow D_j^i, \text{prio}(\tau_j^i) \leftarrow \text{prio}(F_j^i)$.
In Step 2, critical sections in the DGMF model are expressed in the resulting transaction set from Step 1.

In Step 3, precedence dependencies in the DGMF model, are modeled according to the transaction model [11]. First, precedence dependencies between frames of different DGMF tasks are expressed in the resulting transaction set. A precedence dependency is also added between two tasks representing consecutive frames of a same DGMF task (i.e. \( \tau_{ij} \prec \tau_{ij+1} \), \( j < N_t \)). We then enforce successors to be released after predecessors: when \( \tau_{ij} \prec \tau_{ij} \) and \( \tau_{pq} + C_{pq} > \tau_{ij} \) then offset and deadline are updated as: \( O_{ij} \leftarrow O_{ij} + \tau_{ij} + C_{ij} - \tau_{ij} \) and \( d_{ij} \leftarrow d_{ij} - r_{ij} + C_{ij} - \tau_{ij} \). Afterwards we ensure that precedence dependent tasks are delayed from the same event, with respect to the definition of transactions. Tasks in different transactions, are assigned into a same transaction \( \Gamma_m \) if:

\[
\exists \tau_{ij}^q, \tau_{ij}^p \mid (\Gamma_i \neq \Gamma_p) \land \left( \tau_{ij}^q \prec \tau_{ij}^p \lor \tau_{ij}^q \prec \tau_{ij}^p \right)
\]

Tasks \( \tau_{ij}^q \) in \( \Gamma_m \) were originally in \( \Gamma_o \). The release time of the first job of \( \Gamma_m \) is then \( r_m \leftarrow \min(r_m, r_o + O_{ij}) \). The offset of each task \( \tau_{ij}^q \) is then \( O_{ij} \leftarrow r_o + O_{ij} - r_m \). Finally for a specific task \( \tau_{ij}^q \), the transformation reduces predecessors \( \tau_{ij}^q \) that have a global deadline (i.e. \( O_{ij}^q + d_{ij}^q \)) smaller than the offset \( O_{ij}^q \) of \( \tau_{ij}^q \). At least one predecessor of \( \tau_{ij}^q \) is kept.

C. Assessing Schedulability of Resulting Transactions

To enforce schedulability of the resulting transactions, we use a schedulability test [5] based on [13]. The schedulability test is applicable to tree-shaped transactions with non-immediate tasks. A non-immediate task is one that is not necessarily immediately released by its predecessor. We obtain tree-shaped transactions with non-immediate tasks from the DGMF to transaction transformation, according to the following theorem proved in [6]:

\textbf{Theorem 1:} A DGMF task set with the Unique Predecessor property (Property 1) is transformed into a transaction set without tasks that have more than one predecessor.

V. EXPERIMENTS

The DGMF to transaction transformation is implemented in Cheddar [15], a real-time scheduling analysis tool. Experiments are conducted to assess the transformation’s correctness, time performance, and the applicability of the approach on a real TDMA SRP from Thales.

A. Transformation Correctness

A total of 25600 DGMF task sets were randomly generated to evaluate the transformation’s correctness. Each of them was transformed to a transaction set. Both DGMF and transaction sets were simulated and we observed that each DGMF set’s schedule was strictly the same as the resulting transaction set’s schedule. Along with the proofs in [6], this simulation further enforces the transformation’s correctness.

B. Transformation Time Performance

We now verify the transformation’s complexity which depends on two parameters: \( n_F \) the number of frames, and \( n_D \) the number of task dependencies (both precedence and shared resource). The complexity of the transformation is \( O(n_D^2 + n_F) \). We check this complexity by varying \( n_D \) (resp. \( n_F \)) and setting \( n_F \) (resp. \( n_D \)) to a constant value.

Fig. 2 shows the transformation duration by the number of precedence dependencies. The number of frames is set to 1000, the number of DGMF tasks to 100, and the number of shared resource dependencies to 0. The minimum number of precedence dependencies starts at 900, due to precedence dependencies in the transaction model that represent consecutive frames of a DGMF task. From Fig. 2 we see that the transformation duration is polynomial when the number of precedence dependencies vary. This result is consistent with the complexity \( O(n_D^2) \), when \( n_D \) is the varying parameter.

Fig. 3 shows the transformation duration by the number of frames. The number of precedence dependencies is set to 0 and the other parameters remains the same. Fig. 3 shows that the duration is polynomial when the number of frames varies. One can think that this result is inconsistent with the transformation’s complexity \( O(n_F) \). In practice, the implementation in Cheddar introduces a loop to verify that a task is not already present in the system’s task set. Thus the time complexity of the implementation is \( O(n_F^2) \).

Overall we see that a system with no dependency, 1000 frames, and 100 DGMF tasks, takes less than 120ms to be
In our system there are three tasks: $G_1$, $G_2$, and $G_3$. For $G_2$, GMF WCRT analysis gives a lower WCRT than DGMF analysis but this value is underestimated. Indeed GMF WCRT analysis considers that $F^2_2$ is only interfered by $F^3_1$ and $F^2_1$, without considering the fact that $F^2_1$ is released after $F^1_1$. In conclusion DGMF analysis determines a schedulable system, and precedence dependencies must be considered by the analysis, in order to not underestimate WCRTs.

VI. CONCLUSION

In this article we expected to improve scheduling analysis of systems with TDMA communications and task dependencies, using TDMA SRPs as an illustration. The DGMF task model was proposed to model such a system, and we proposed to transform DGMF tasks to transactions, for which an applicable scheduling analysis technique was chosen. Experiments on a real TDMA SRP from Thales showed that DGMF improves schedulability compared to both the GMF and periodic models.

In the future, we will integrate the proposed analysis technique in a SRP development process at Thales.

REFERENCES