Comments on “Gang EDF Schedulability Analysis”

Pascal Richard*  Joël Goossens†  Shinpei Kato ‡

Abstract
This short report raises a correctness issue in the schedulability test presented in [6]: “Gang EDF Scheduling of Parallel Task Systems”, 30th IEEE Real-Time Systems Symposium, 2009, pp. 459-468.

1 Introduction
We raise a correctness issue in the schedulability test in the paper: “Gang EDF Scheduling of Parallel Task Systems”, Shinpei Kato and Yutaka Ishikawa, presented at RTSS’09. This paper presents a Gang scheduling algorithm (Gang EDF) and its schedulability test, named [KAT] hereafter.

[KAT] considers $n$ preemptive sporadic gang tasks (also known as rigid parallel tasks [5]), to be executed upon $m$ identical processors. Each task $\tau_i = (v_i, C_i, D_i, T_i)$, $1 \leq i \leq n$, is characterized by the number $v_i$ of used processors, a worst-case execution time $C_i$ when executed in parallel on $v_i$ processors, a minimum inter-arrival time $T_i$ and a constrained relative deadline $D_i$ (i.e., $D_i \leq T_i$). The utilization of $\tau_i$ is $U_i = C_i / T_i$. Each task generates an infinite sequence of jobs. The execution of a job of $\tau_i$ is represented as a $C_i\times v_i$ rectangle in time $\times$ processor space. Every job must be completed by its deadline.

Gang EDF applies the well-known EDF policy to the gang scheduling problem: jobs with earlier deadlines are assigned higher priorities. Since tasks use several processors, the earliest deadline rule is extended in Gang EDF to take into account the spacial limitation on the number of available processors by using a first fit based strategy. The reader can refer to [6] for a complete description of the scheduling algorithm. Next, we only focus on the schedulability test for analyzing gang tasks scheduled by Gang EDF.

---

*LIAS-University of Poitiers, pascal.richard@univ-poitiers.fr
†Université Libre de Bruxelles, Belgium, joel.goossens@ulb.ac.be
‡Nagoya University, Japan, shinpei@is.nagoya-u.ac.jp
2 [KAT] Schedulability Test Principles

Basically, the test follows the principles of [BAR] test defined in [1] for tasks using at most one processor. It is based on a necessary schedulability condition for a task to miss a deadline. Then, the contrapositive condition yields a sufficient schedulability condition for the considered scheduling algorithm.

[KAT] test considers any legal sequence of job requests on which a deadline is missed by Gang EDF. Assume that $\tau_k$ is generating the problem job at time $t_a$ that must be completed by its deadline at time $t_d = t_a + D_k$. Let $t_0$ be the latest time instant before or at $t_a$ at which at least $v_k$ processors are idled and $\Delta_k = t_d - t_0$. A necessary condition for the problem job to miss its deadline is: higher priority tasks are blocking $\tau_k$ for strictly more that $D_k - C_k$ in the interval $[t_a, t_d)$. Since $\tau_k$ requires $v_k$ processors, it is blocked while $m - v_k + 1$ processors are busy. This minimum interference necessary for the deadline miss is defined by the interference rectangle whose width $w_k$ and height $h_k$ are respectively given by: $w_k = \Delta_k - C_k$ and $h_k = m - v_k + 1$.

Let $I(\tau_i, \Delta_k)$ be the worst-case interference against the problem job over $[t_0, t_d)$, meaning that it blocks the problem job over $[t_a, t_d)$ and is executed over $[t_0, t_a)$. In [6], it is claimed that: If the problem job misses its deadline, then the total amount of work that interferes over $[t_0, t_d)$ necessarily exceeds the interference rectangle:

$$\sum_{\tau_i \in \tau} I(\tau_i, \Delta_k) > w_k \times h_k \quad (1)$$

It is important to understand exactly what Equation (1) means: if a task misses a deadline, then the condition defined in (1) is satisfied. Thus, it represents necessary conditions for task $\tau_k$ to miss a deadline $A_k = \Delta_k - D_k$ units after an instant at which at least $v_k$ processors are idled. It is also important to notice that the necessary condition only exploits the area of the interference rectangle (i.e., $w_k \times h_k$) for defining a bound on the processor demand for task $\tau_k$. Finally, it is also worth noticing that this necessary condition is not formally proved in [6].

As in [BAR], the contrapositive of the previous necessary condition allows to define the following sufficient schedulability condition in [KAT]: if the contrapositive of Equation (1) is satisfied, then deadlines are met.

The interference must take into account carry-in jobs in the interference rectangle who arrive before $t_0$ and have not completed execution by $t_0$. The [KAT] test distinguishes the interference coming from tasks without or with a carry-in job for bounding the overall interference $I(\tau_i, \Delta_k)$, respectively denoted $I_1(\tau_i, \Delta_k)$ and $I_{\text{carry-in}}$. The schedulability test [KAT] checks every task using the contrapositive of the necessary condition (1) that yields a sufficient schedulability condition in Theorem 1.

**Theorem 1** [6] It is guaranteed that a task system $\tau$ is successfully scheduled by Gang EDF upon $m$ processors, if the following condition is satisfied for all tasks
\( \tau_k \in \tau \) and all \( \Delta_k \geq D_k \):

\[
\sum_{\tau_i \in \tau} I_1(\tau_i, \Delta_k) + I_{\text{carry-in}} \leq w_k \times h_k \tag{2}
\]

Several bounds have been proposed in [6] to evaluate the accumulated interference \( \sum_{\tau_i \in \tau} I_1(\tau_i, \Delta_k) \). We limit ourselves to use the first proposed bound:

\[
I_1(\tau_i) = \min(hbf(\tau_i, \Delta_k), w_k) \times \min(v_i, h_k) \quad \text{if } i \neq k
\]

\[
I_1(\tau_i) = \min(hbf(\tau_i, \Delta_k) - C_k, A_k) \times \min(v_i, h_k) \quad \text{if } i = k
\]

where \( A_k = \Delta_k - D_k \) defines the maximum contribution of \( \tau_k \) in the feasibility interval \([l_0, t_d]\) and \( hbf(\tau_i, L) = \max(0, \left[ \frac{L - D_i}{2} \right] + 1) \times C_i \) is the horizontal-demand bound function.

Similarly, the contribution of task \( \tau_i \) with a carry-in job to the interference rectangle can be bounded by [6]:

\[
I_2(\tau_i) = \min(hbf'(\tau_i, \Delta_k), w_k) \times \min(v_i, h_k) \quad \text{if } i \neq k
\]

\[
I_2(\tau_i) = \min(hbf'(\tau_i, \Delta_k) - C_k, A_k) \times \min(v_i, h_k) \quad \text{if } i = k
\]

where \( hbf'(\tau_i, L) = \left( \frac{\Delta_i}{2} \right) \times C_i + \min(C_i, L \mod T_i) \). \( I_{\text{diff}} = I_2(\tau_i) - I_1(\tau_i) \) is the contribution of \( \tau_i \) by its carry-in job to the interference rectangle (defined in [6]). \( I_{\text{diff}} \) is used to compute the total amount of work contributed by the carry-in parts, that is at most \( I_{\text{carry-in}} \) in Equation (2). We do not detail it here since it will not be used in the remainder.

Notice that \( h_k \) is fixed while \( w_k \) is not (since \( \Delta_k \) is not determined). Thus, the previous condition must be checked for all values of \( \Delta_k \). As a consequence, in order to use the test (Theorem 1), \( \Delta_k \) must be bounded to define a finite time interval to test possible values for \( \Delta_k \) (Theorem 2).

**Theorem 2** [6] If Condition (2) is to be violated for any \( \Delta_k \), then it is violated for some \( \Delta_k \geq D_k \) satisfying Condition (2), where \( C_{\text{carry-in}} \) denotes \( \sum_{\tau_i \in \tau_{\text{carry-in}}} C_i \).

\[
\Delta_k \leq \frac{h_k C_k - \sum_{\tau_i \in \tau}(D_i - T_i) U_i \times \min(v_i, h_k) + C_{\text{carry-in}}}{h_k - \sum_{\tau_i \in \tau} U_i \times \min(v_i, h_k)} \tag{3}
\]

where \( \tau_{\text{carry-in}} \) is the set of tasks with a carry-in job.

### 3 Correctness Issues

In this section, we show through a counterexample that problems arise when the schedulability test presented in the previous section is applied to a counterexample task set. Then, we show that the necessary condition for a job to miss
a deadline is not valid (i.e., Equation (1)). This will allow us to conclude that Theorems 1 and 2 that both exploit the contrapositive of Equation (1) cannot define a valid sufficient schedulability test. We first present the counterexample task set.

Counter-example definition. Let us consider the feasible task set defined in Figure 1 and a platform with $m = 3$ processors. Both tasks require simultaneously two processors and have a deadline of 2 units of time. According to Gang EDF, tasks $\tau_1$ and $\tau_2$ have equal priority since (i) they have the same relative deadline and (ii) they both require two processors. With no loss of generality, we assume that Gang EDF tie breaker gives the higher priority to $\tau_1$. Therefore, task $\tau_2$ will miss its deadline. Clearly, this task set is infeasible upon a 3-processor platform.

3.1 Feasibility Interval

Applying the test on the counterexample. We analyze the task $\tau_2$. $\tau_2$ releases the problem job that misses its deadline at time 2 as depicted in Figure 1. The feasibility interval is delimited by: $t_0 = t_a = 0$ and $t_d = D_2 = 2$; $\Delta_2 = t_d - t_0 = 2$ and $A_2 = \Delta_2 - D_2 = 0$. The scenario $\Delta_2 = D_2$ is the first scheduling point considered in the feasibility interval when applying Theorem 1. The interference rectangle is: $w_2 = \Delta_2 - C_2 = 2 - 1 = 1$ and $h_2 = m - v_2 + 1 = 3 - 2 + 1 = 2$.

The first step in order to apply [KAT] test is to define the feasibility interval defined in Theorem 2. Next, we will only use the fact that $C_{\text{carry-in}} \geq 0$. In Theorem 2 the numerator has always a positive value since $D_i \leq T_i$, $1 \leq i \leq n$, and all used values are positive or zero. We will see that the denominator is not positive in the counterexample. As a consequence, the schedulability test has to be applied over a time interval which has surprisingly a negative length.

The task under analysis is $\tau_2$, thus we set $k = 2$ in the remainder. We need to bound $\Delta_2$ using Theorem 2. We prove hereafter that such a bound is

| Tasks | $v_i$ | $C_i$ | $D_i$ | $T_i$ |
|-------|-------|-------|-------|-------|
| $\tau_1$ | 2     | 2     | 2     | 2     |
| $\tau_2$ | 2     | 1     | 2     | 2     |

Figure 1: Counter-Example: infeasible task set for 3-processor platforms.

3See Section 3 in [6] for a complete presentation of Gang EDF.

4The inexistence of carry-in jobs for the considered task set will be proved in the remainder of this report.
negative for the counterexample. Consider the denominator of Inequality (3):
\[ h_k - \sum_{\tau_i \in \tau} U_i \times \min(v_i, h_k). \]
As shown previously, we have \( h_2 = 2 \); this implies that \( \min(v_i, h_2) = 2 \), \( 1 \leq i \leq n \) and thus \( h_2 - \sum_{\tau_i \in \tau} U_i \times 2 = 2 - (\frac{2}{2} + \frac{1}{2}) \times 2 = -1 \). As a consequence the denominator is negative. Thus, the upper bound computed by Theorem 2 of the time interval while checking the schedulability of a task has a negative length.

Since, \( \Delta_k < 0 \) implies that there is no \( \Delta_k \) such that \( \Delta_k \geq D_k \), then the sufficient schedulability test defined by Theorem 1 cannot be applied. According to our interpretation, such a situation cannot be interpreted as Theorem 1 is respected by default but raises instead a correctness issue.

**Insight.** In [6], the last mathematical derivations performed to prove Theorem 2 is incorrect. Precisely, if \( (h_k - \sum_{\tau_i \in \tau} U_i \times \min(v_i, h_k)) \geq 0 \) then:

\[
\Delta_k \leq \frac{h_k C_k - \sum_{\tau_i \in \tau} \{(D_i - T_i) \times \min(v_i, h_k)\} + C_{\text{carry-in}}}{h_k - \sum_{\tau_i \in \tau} U_i \times \min(v_i, h_k)}
\]  
(4)

- Otherwise:

\[
\Delta_k > \frac{h_k C_k - \sum_{\tau_i \in \tau} \{(D_i - T_i) \times \min(v_i, h_k)\} + C_{\text{carry-in}}}{h_k - \sum_{\tau_i \in \tau} U_i \times \min(v_i, h_k)}
\]  
(5)

Unfortunately most of the time we are in the second case and [KAT] cannot be applied. To clearly illustrate that purpose, Figure 2 depicts the number of task sets for which [KAT] cannot be used.

### 3.2 Necessary schedulability condition is incorrect

**Applying the test on the counterexample.** In order to show that the necessary condition defined in Equation (1) for a task to miss its deadline is incorrect, we consider once again the task \( \tau_2 \) in the counterexample. First, we compute the interference bounds for the tasks without carry-in jobs:

\[
I_1(\tau_1, \Delta_2) = \min \left( \left( \left\lfloor \frac{\Delta_2 - D_1}{T_1} \right\rfloor + 1 \right) \times C_1, w_2 \right) \times \min(v_1, h_2)
\]

\[
= \min \left( \left( \left\lfloor \frac{2 - 2}{2} \right\rfloor + 1 \right) \times 2, 1 \right) \times \min(2, 2) = 1 \times 2 = 2
\]

\[
I_1(\tau_2, \Delta_2) = \min \left( \left( \left\lfloor \frac{\Delta_2 - D_2}{T_2} \right\rfloor + 1 \right) \times C_2 - C_2, A_2 \right) \times \min(v_2, h_2)
\]

\[
= \min \left( \left( \left\lfloor \frac{2 - 2}{2} \right\rfloor + 1 \right) \times 1 - 1, 0 \right) \times \min(2, 2) = 0 \times 2 = 0
\]

---

\(^5\) Synthetic task sets with 10 tasks and 6 processors have been generated using the Stafford's algorithm.
Since both tasks have identical periods and deadlines, there is no carry-in job (i.e., jobs of $\tau_1$ and $\tau_2$ are released and have deadlines within the interval $[t_0, t_d]$). To prove this, let us show that $I_{diff}(\tau_i) = 0, 1 \leq i \leq 2$. Let us compute $I_2(\tau_i)$:

$$I_2(\tau_1, \Delta_2) = \min \left( \frac{\Delta_2}{T_1} \times C_1 + \min(C_1, \Delta_2 \mod T_1), w_2 \right) \times \min(v_1, h_2)$$

$$= \min \left( \frac{2}{2} \times 2 + \min(2, 2 \mod 2), 1 \right) \times \min(2, 2) = 1 \times 2 = 2$$

$$I_2(\tau_2, \Delta_2) = \min \left( \frac{\Delta_2}{T_2} \times C_2 + \min(C_1, \Delta_2 \mod T_1) - C_2, A_2 \right) \times \min(v_2, h_2)$$

$$= \min \left( \frac{2}{2} \times 1 + \min(1, 2 \mod 2) - 1, 0 \right) \times \min(2, 2) = 0 \times 2 = 0$$

Hence there is no carry-in job since $I_{diff}(\tau_i) = I_2(\tau_i) - I_1(\tau_i) = 0, 1 \leq i \leq 2$.

Since there is no carry-in job (i.e., $I_{carry-in} = 0$), the total cumulative interference is $\sum_{\tau_i \in \tau} I_1(\tau_i) = 2$ and thus consequently the necessary condition defined in Equation (1) is not satisfied since:

$$\sum_{\tau_i \in \tau} I_1(\tau_i) = 2 = w_2 \times h_2$$

To summarize, there is a deadline miss but the necessary condition defined in Equation (1) is not satisfied. As a consequence, we conclude that Equation

Figure 2: Task sets for which [KAT] cannot be applied
[1] is not a valid necessary condition for a task to miss its deadline. Hence, Theorems [1] and [2] are not valid since they both use the necessary schedulability condition defined in Equation [1] as an initial claim.

Insight. This problem is in fact inherited from the [BAR] test. In a footnote in [3, 4], a technical problem has been exhibited and a simple solution is proposed to fix it. The detection of the error in [BAR] ([1]) was concurrent to the publication of [KAT].

The bound of the interference is not correct but can be easily fixed by:

- adding an $\epsilon$ in the interference bounds in $I_1$ and $I_2$ (resp. Equations (3) and (5) in [1]),
- or changing the inequality (8) in [1] into a strict inequality.

Hence, using a strict inequality in Equation (2) (in Theorem [1]) will fix the problem that has been exhibited the previous counter-example while using [KAT].

4 Discussion

This short note raises a major correctness issue in the schedulability test presented in [6]. The first problem comes from the feasibility interval that may have negative length, even for infeasible task sets. Through numerical experiments, we shown that [KAT] cannot be applied. The second problem is inherited from the [BAR] that uses incorrect interference bounds. This latter problem can be corrected in [KAT] following the same principles used for correcting the similar problem in [BAR].

It is worth noticing that the detected problem in the necessary condition does not exist if the interference rectangle reaches its maximum height (i.e it is equal to $m$). This case only arises if every gang tasks uses exactly one processor at a time. In this precise case, [KAT] schedulability test is valid and equivalent to [BAR] [1] as explained in [6].

References

[1] BARUAH, S. Techniques for multiprocessor global schedulability analysis. In 28th IEEE International Real-Time Systems Symposium, 2007 (December 2007), pp. 119–128.

[2] BARUAH, S. K., BERTOGNA, M., AND BUTTAZZO, G. C. Multiprocessor Scheduling for Real-Time Systems. Embedded Systems. Springer, 2015.

\(^6\)Notice that in [2] the erroneous version of [BAR] is presented.
[3] Bertogna, M. Evaluation of existing schedulability tests for global EDF. In *ICPP Workshops* (2009), IEEE Computer Society, pp. 11–18.

[4] Bertogna, M., and Baruah, S. K. Tests for global EDF schedulability analysis. *Journal of Systems Architecture - Embedded Systems Design* 57, 5 (2011), 487–497.

[5] Dutot, P.-F., Mounié, G., and Trystram, D. Scheduling parallel tasks: Approximation algorithms. *Handbook of scheduling: Algorithms, models, and performance analysis* (2004), 26.1–26.24.

[6] Kato, S., and Ishikawa, Y. Gang EDF scheduling of parallel task systems. In *30th IEEE Real-Time Systems Symposium, 2009* (December 2009), pp. 459–468.