An efficient mixed scheduling algorithm for the hybrid task set on heterogeneous multiprocessor

Hui Wang
School of Information Science
and Engineering, Hunan University
Lushan South Rood, Changsha
410082, P.R. China
Email: wanghui2007321@163.com

Cheng Xu
School of Information Science
and Engineering, Hunan University
Lushan South Rood, Changsha
410082, P.R. China
Email: cheng_xu@yeah.net

Lining Zeng
School of Information Science
and Engineering, Hunan University
Lushan South Rood, Changsha
410082, P.R. China
Email: will120120@126.com

***Abstract*—In this paper we study the schedulability conditions of multiple types of real-time tasks (periodic real-time tasks, sporadic real-time tasks and aperiodic soft real-time tasks) in a heterogeneous multiprocessing environment. With the practical application of complicated environment, we propose a mixed scheduling algorithm in this paper, which itegrates UEDF algorithm with Task-Centric with Slack Defragmentation algorithm in the heterogeneous multiprocessor. Due to its characteristics that making the most of the processor which is already in use at first in the process of scheduling in order to reserve more free resources for future tasks, this algorithm can effectively improve the success rate of the entire hybrid task set scheduling. The results in simulation experiments show that the performance of the proposed algorithm scheduling on both the monotonic performance ordering and the non-monotonic performance ordering resource have obvious advantages over that of the current other combination algorithms.**

***Index Terms*—hybrid task set, sporadic real-time task, heterogeneous multiprocessor, UEDF algorithm**

I. Introduction

With the rapid development and popularization of computer technology, real-time system has been an indispensable part of human production and life. Real-time system means the correctness of computation highly depends on not only accurate of logic result but also the generation time of the result. As the advanced development of real-time system, it is increasingly common that multiple types of real-time tasks coexist in the system so that the complexity of real-time system is consistently increased. Therefore the scheduling of the hybrid real-time task set has become a hot research problem, it represents the future direction of real-time systems. Scheduling algorithm is the critical issue of real-time task scheduling, and there are strong theoretical and practical values in the aspect of studying hybrid real-time task scheduling algorithm.

Our work is to study a hybrid task set which consists of the periodic hard real-time tasks, sporadic hard real-time tasks and aperiodic soft real-time tasks in heterogeneous multi-processor environment. The scheduling problem of hybrid task sets has been a hot and difficult research, therefore proposing a new hybrid task scheduling algorithm makes the success rate of hybrid task set scheduling increase in this paper.

II. Related Work

The scheduling problem of sporadic task has become a hot and difficult issue since it was a widespread phenomenon in the real-time system. S.L.Vieira focused on the scheduling of sporadic real-time tasks in a single processor environment by simplifying the sporadic tasks as a special class of periodic real-time tasks to schedule using classical algorithm named EDF in[4]; However, due to the pessimistic view of the problem ,this method resulted in a waste of resource reserve in the real situation. Kevin Jeffay considered not only the presence of periodic real-time task on single-processor environment, but also sporadic real-time task coexisting in it. Then they schedule a set of periodic real-time tasks or sporadic real-time tasks in the conditions that idle time could not been inserted into the system and without preemption, and also proposed a necessary and sufficient condition that a set of periodic or sporadic real-time tasks released at any time is schedulable[5].

Manuel Coutinho considered the hybrid task set consisting of a set of periodic real-time tasks and a set of sporadic real-time tasks coexisted in a system, but they only regarded the sporadic real-time tasks as periodic real-time tasks which reach according to the maximum frequency in [6]. Jiafu wan studied a scheduling problem about a hybrid task set which consists of the time-based triggered periodic real-time tasks and the event-based triggered aperiodic real-time tasks in a preemptive priority system in[7], the innovation of the hybrid scheduling task set is divided into two-layer scheduling scheme, the uniform of the second layer uses EDF algorithm to schedule, this will help reducing the complexity of the hybrid task scheduling, and fully takes the real-time and the characteristic of different tasks into account. It considers the real-time performance of the tasks and the characteristics of the distributed operating platform in [8].

Geoffrey Nelissen presented a single-processor classical EDF algorithm to be promoted another form U-EDF in a multi-processor environment in [1], which is different from most forms of promotion on a multi-processor G-EDF which allows early to start the arrived task execution, while U-EDF algorithm tries to execute tasks on fewer processors when it ensures the task set to meet their deadlines. Hsiang-

Kuo Tang has studied hard periodic and soft aperiodic real-time task on heterogeneous processors, and has proposed the good scheduling algorithm, such as HEDF or Fastest First (FF) algorithm for periodic real-time task , Resource-Centric (RC), Task-Centric (TC) and Task-Centric with Slack Defragmentation (TCSD) algorithm for aperiodic soft real-time task in [3].

III. Problem description

A. Task Model

We consider the task set which contains periodic hard real-time task set τp_i which is composed of n periodic hard real-time tasks, sporadic real-time task set τs_i which is composed of t sporadic real-time tasks and aperiodic soft real-time task set τa_i which is composed of q aperiodic tasks ,which is denoted by $TS = \{\tau p, \tau s, \tau a\}$.

Periodic hard real-time task $\tau p = \{\tau p_1, \tau p_2, ..., \tau p_n\}$

Any task τp_i of a periodic task set is denoted by the arrival time ap_i, the deadline Dp_i, the period Tp_i and the computation time Cp_{ij} (define by computing time of task τp_i on the resource R_j).

Sporadic hard real-time task $\tau s = \{\tau s_1, \tau s_2, ..., \tau s_t\}$

Any task τs_i of sporadic real-time task set, the arrival time as_i , the deadline Ds_i, the minimum arrival time Ts_i and the computation time Cs_{ij}(define by computing time of task τs_i on a resource R_j) .

Aperiodic soft real-time task $\tau a = \{\tau a_1, \tau a_2, ..., \tau a_q\}$

Any task τa_i of aperiodic soft real-time task set, the arrival time aa_i, deadline Da_i, the minimum arrival interval time Ta_i and the computation time Ca_{ij} (defined by computing time of task τa_i on a resource R_j).

B. System Model

1) monotonic performance ordering resources: We summarize the potential heterogeneity into two different types: monotonic performance ordering and non-monotonic performance ordering. Considering a system with two processors A and B, a monotonic performance ordering processor resources mean that all tasks on processor A are better than them on processor B. There is a case that more one task execute faster on processor A, but another task executing on the processor B is better than that on processor A, which does not exist.

2) non-monotonic performance ordering resources: There is a non-monotonic performance ordering system which is not defined the same order to keep all tasks meet the requirement as (1). Consider a system with two types of resources: P is an instruction-based processor and F is the FPGA logic. Some tasks may obtain more acceleration on resources F than these on the resource P, while the execution of other tasks which have complex control flow or data-dependent behavior is relatively slow on resources F. This system is non-monotonic performance ordering.

The below experiment is conducted on the monotonic performance ordering resources and non-monotonic performance ordering resources to schedule the hybrid task set, and it verifies the performance of the proposed mixed scheduling algorithm.

IV. Algorithm

A. the scheduling algorithm of periodic real-time takes

Centralized scheduling algorithms: U-EDF

U-EDF algorithm is another generation form of a classical algorithm EDF on multiprocessors. Periodic tasks in accordance with the deadline arrive in ascending order into a global queue, the global task scheduler checks to see whether the front of the queue is assigned at first to execute the task on this resource which is already in use without missing deadlines. It is different from main features of the H-EDF algorithm which allows scheduling the arrived tasks as soon as possible to begin the task. But on the premise that all tasks are required to meet the deadline, using as fewer processors as possible to schedule the execution in order to generate enough idle processor time to execute the task which will arrive in future.

B. Sporadic or aperiodic task scheduler

Task-Centric with Slack Defragmentation (TCSD)[3]

The purpose of scheduling periodic tasks assigned to each task execution on a resource is to determine its latest start time. Because it exists that the slack time between the arrival time of many periodic tasks and their latest starting time. TCSD algorithm uses this fact that moving the periodic task instances earlier than its latest starting time to execute, and doing this can create a more idle time window after periodic task is completed in order to schedule aperiodic task. When aperiodic task arrives, TCSD algorithm first checks whether there is a large enough idle time interval to execute this task; If not, check each periodic task which is surrounded by two idle time window and decide to move periodic task to execute earlier, and the next window will be extended to accommodate aperiodic tasks. If the extended idle time window can not still meet the requirement of the aperiodic tasks, it will be rejected.

Theorem 1 The periodic task set $\tau_p = \{Tp_1, Tp_2, ..., Tp_n\}$ and the sporadic task set $\tau_s = \{Ts_1, Ts_2, ..., Ts_q\}$ are scheduled on processor resource set $R = \{R_1, R_2, ..., R_m\}$. Task TP_i executed on the processor resource R_j can be denoted by (Cp_{ij}, Tp_i), similarly using (Cs_{ij}, Ts_i) represents any sporadic real-time task. If they are schedulable then

(1)

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\frac{Cp_{ij}}{Tp_i}Loc_{ij} + \sum_{i=1}^{n}\sum_{j=1}^{m}\frac{Cs_{ij}}{Ts_i}Loc_{ij} < m,$$

assuming any task only executes on a processor, therefore

$$\sum_{j=1}^{m} Loc_{ij} = 1$$

means only the value of a variable among Loc_{ij} that is 1, and any processor R_k meets the condition:

$$\sum_{i=1}^{n}\frac{Cp_{ik}}{Tp_i}Loc_{ik} + \sum_{i=1}^{q}\frac{Cs_{ik}}{Ts_i}Loc_{ik} < 1$$

whereLoc_{ij}indicates whether any task τ_i executes on the processor R_j.
(2) Periodic real-time task sort by their periods in non-decreasing order,$\forall i, 1 < i \leq n$; Periodic real-time task sort by their periods in non-decreasing order, all periodic real-time tasks which execute on any processor R_k need to meet the condition:

$$\forall L, Tp_1 < L < Tp_i; L \geq Cp_{ik} + \sum_{j=1}^{i-1} \lfloor \frac{L-1}{Tp_j} \rfloor Cp_{jk}$$

Condition 1 can be considered as a necessary condition for a hybrid tasks set that can be scheduled to ensure that the processor is not overloaded. The cumulative processor utilization of all tasks in the hybrid task set can not exceed the sum of the processing power of multiple processors, and the sum of the utilization of the tasks on each processor must be strictly less than 1, otherwise the task set must not be scheduled to execute.

Condition 2 means the load of the processor within a certain time interval L, which needs to satisfy the conditions. In order that a tasks set can be scheduled, the demand within the time interval L must always be less than the length of the interval L.

Theorem 2 let τs is a set of sporadic real-time task $\{(Cs_1, Ts_1), (Cs_2, Ts_2), ...(Cs_n, Ts_n)\}$, they are sorted by the period of the task in non-decreasing order. If a task set τ_s which meets condition 1 and condition 2 from theorem 1, then EDF algorithm can schedule any sporadic real-time task set generated from the task set τ_s.

Theorem 3 a task set $\Gamma^S = \{\tau p_i(ap_i^0, Cp_i, Dp_i, Tp_i), i = 1..n\}$ is schedulable, Only when any pair of task (τ_i, τ_j) which derives from the task set meets the condition:

$$Cp_i \leq (ap_j^0 - ap_i^0) mod\ g_{ij} - Cp_j (1)$$

g_{ij} is the Greatest Common Divisor of two period tasks $\tau p_i, \tau p_j . ap_i^0$ and ap_j^0 denote task arrival time.

When a periodic task has been scheduled to execute in order to study the schedulability of sporadic task, we must decide a critical moment which is the worst-case response time of a sporadic task [2].

The following lemma is given the calculation requirement of time t for sporadic tasks that release at the time $S \in \Psi$.

Theorem 4 Consider that a strict periodic task set Γ^S and a sporadic task set $hp^{NS}(i)$ have been scheduled to execute on each processor. Let τs_i represents any sporadic task whose release time is at $S \in \Psi$, the sum of execution time at time t is given by the following formula:

$$W_i(t) = Cs_i + \sum_{\tau_i \in hp^{NS}(i)} \lceil \frac{t}{Ts_j} \rceil Cs_j + \sum_{\tau_j \in \Gamma^S} \lceil \frac{t - ap_j}{Tp_j} \rceil Cp_j$$

According to the release time S, ap_j is equivalent to the start time ap_j^k.

$$ap_j = ap_j^0 + \lceil \frac{S - ap_j^0}{Tp_j} \rceil Tp_j - S$$

Proof

Consider that a sporadic task τs_i is the first release of task at time S $\in \Psi$, the total of execution time at time t is the accumulative of the execution time about the following case:
1. The execution time of a sporadic task τs_i at start time S: Cs_i
2. All strict periodic task (all task have higher priority than the task τs_i) : $\sum_{\tau p_i \in \Gamma^S} \lceil \frac{t - ap_j}{Tp_j} \rceil$
3. All the sporadic tasks which have higher priority than the sporadic task τs_i :$\sum_{\tau s_i \in hp^{NS}(i)} \lceil \frac{t}{Ts_j} \rceil Cs_j$

The following lemma gives a necessary and sufficient schedulability condition of a set of sporadic tasks [2].

Lemma 3 Consider that a strict periodic task set Γ^S have been scheduled, a sporadic task set Γ^{NS} is schedulable, if and only if $\forall \tau s_i \in \Gamma^{NS} : R_i \leq D_i$, where R_i is the worst response time of a task , which is the solution of $R_i = W(R_i)$ computed by the iteration.

Proof

The proof is identical to the one given in [10] which states that the worst Case Response Time of any task should be less than or equal to its deadline.

An example of GEDF or UEDF algorithm

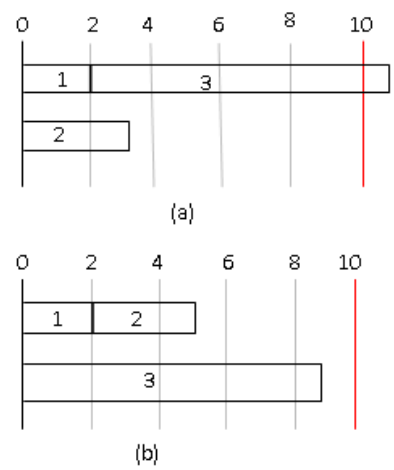

Fig. 1. Use GEDF algorithm and UEDF algorithm to schedule three different tasks

Assuming that there are three tasks which reach at time 0, namely the task τ_1, τ_2, τ_3. Their execution time are two time units, three time units, nine time units respectively, their deadline at the time instant 10. In (a) part of figure 1 it uses G-EDF algorithm to schedule two of these three tasks on the processors in order to let each task begin to execute as soon as possible, it will schedule at first task τ_1 and τ_2 on the processor 1 and processor 2 for beginning immediately to perform respectively. Then the task τ_3 only waits for the first two tasks to complete before we can use a processor to execute the task τ_3, we can put the task on any one of two processors to execute as soon as possible, but task τ_3 misses its deadline.

We use the U-EDF algorithm to schedule these three tasks

on two processor resources for executing in part (b) of Figure 1. The characteristic of U-EDF algorithm differs from one of G-EDF algorithm that each task executes as soon as possible, while it ensures that all tasks is on the premise that they meet their deadline, we use minimal processor resources to execute the current tasks. It will help to ensure that the task which arrives afterwards can have enough spare resources to execute successfully.We use UEDF algorithm to schedule the tree task on the two processor successfully.

From the above example, it is obvious that U-EDF algorithm has some advantage over the widely used G-EDF algorithm, it reserves more free processor resource for tasks arriving in future in order to facilitate scheduling them to meet the deadline.

V. Experiment Simulation

The paper considers two kind of different periodic tasks (synchronous and asynchronous periodic task) and three kind of different aperiodic tasks (task deadline is less than the task period (constrained sporadic task), the task deadline is equal to the task period (fixed sporadic task), the task deadline is greater than the task periodic (any sporadic tasks)). In this paper, the main study is a hybrid tasks set consisting of synchronized periodic real-time tasks, constrained sporadic real-time tasks and aperiodic soft real-time tasks, compared the performance by using the four kinds of combination-type scheduling algorithms: H-EDF/RC, FF/RC, H-EDF/TC, FF/TC algorithm in [3] with that by using this paper proposing combination-type scheduling algorithms: U-EDF/RC, U-EDF/TC, U-EDF/TCSD algorithm. We also need to consider the characteristic of processor resources, resources are mainly divided into two categories such as monotonic performance ordering and non-monotonic performance ordering. It analyzes each processor utilization by using different scheduling algorithms to schedule the same task set.

A. Task Set Generation

We generate randomly task sets. They are scheduled for execution on a heterogeneous multiprocessor platform which provides enough computing capacity to execute hard deadline periodic tasks, sporadic hard real-time tasks and aperiodic soft real-time tasks in the applications. Depending on the utilization of periodic tasks, we will have 50 different periodic task sets. Similar to the former, we also have 10 different sporadic real-time task sets and aperiodic soft real-time task sets. It can form 500 different combinations from periodic real-time task sets and sporadic real-time task sets, soft real-time aperiodic task sets.

Below generated periodic real-time tasks, Sporadic real-time and aperiodic soft real-time tasks in accordance with the above requirements, we assume that scheduling them on three heterogeneous processors in this experiment. In the case of monotonic performance ordering, these three resources are respectively slow processor resources, general speed processor resources, fast processor resources. In the case of non-monotonic performance ordering, it is three different speed processing resources for each task, not as monotonic as a unified performance ordering corresponding relationship.

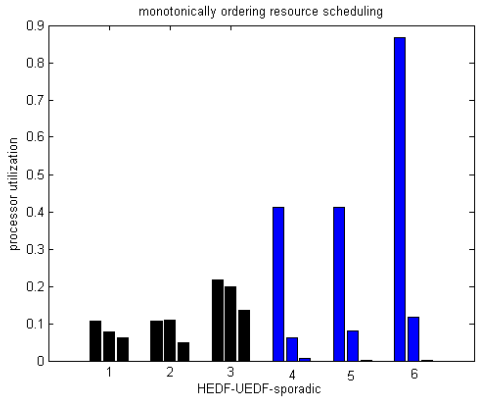

Fig. 2. The total utilization of any one of processors on the monotonic performance ordering processors(sporadic task)

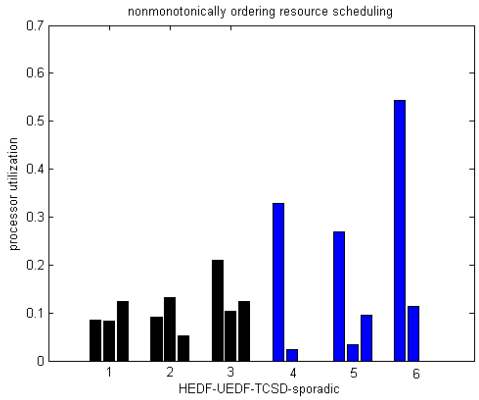

Fig. 3. The total utilization of any one of processors on the non-monotonic performance ordering processors(sporadic task)

Experimental results in the previous three groups in Figure 2 show 3 heterogeneous processor utilization distribution in the case of different total utilization of the hybrid task set in the monotonic performance ordering processor resources by using HEDF algorithms to schedule periodic real-time tasks and using TCSD algorithms to schedule sporadic real-time tasks and aperiodic soft real-time tasks. The latter three groups of results in the experiment shows 3 heterogeneous processor utilization distribution in the same situation with the previous three groups ones by using UEDF algorithm to schedule periodic real-time tasks and using RC or TCSD algorithm to schedule sporadic real-time tasks and aperiodic soft real-time tasks. Figure 3 is similar to Figure 2, which shows that the hybrid tasks set is scheduled on a non-monotonic performance ordering processor to execute and display the utilization of each processor. Combining Figure 2 with Figure 3, we find that HEDF algorithm allows each processor to be early used, which results in a premature start of each processor; the UEDF algorithm tries to consider the task assignment on the

processor which has been used to perform, let processors as idle as possible in order to set aside enough free resources for executing tasks which arrive afterwards.

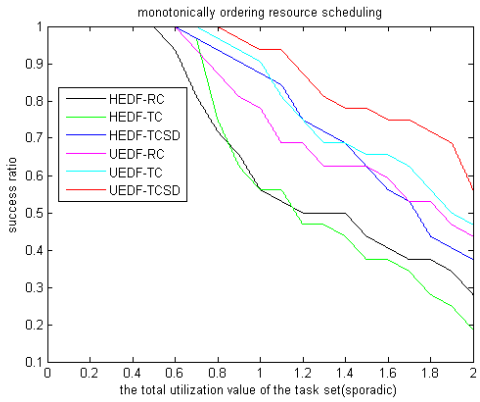

Fig. 4. The success ratio of the task set scheduling on the monotonic performance ordering processors(sporadic task)

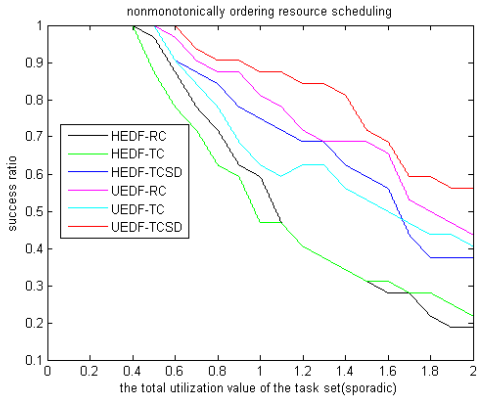

Fig. 5. The success ratio of the task set scheduling on the non-monotonic performance ordering processors(sporadic task)

Considering that under different the total utilization of task sets, we use several different mixed scheduling algorithms to schedule the hybrid task set on monotonic performance ordering resources. We will find that it has a higher success ratio than other combination algorithms that combining UEDF algorithm with TCSD algorithm schedule the hybrid real-time task set consist of periodic real-time task, sporadic real-time tasks and aperiodic soft real-time task in Figure 4.The experiments in Figure 5 are that schedule a hybrid task set on non-monotonic performance ordering resource, its result is similar to the result in Figure 4. It has higher success rate than other combination scheduling algorithms that a combination of UEDF algorithm and TCSD algorithm schedule the hybrid task set whether on a monotonic performance ordering processors or on non-monotonic performance ordering processors in this paper.

VI. Conclusion

The paper considers that the periodic real-time task, sporadic real-time tasks and aperiodic soft real-time tasks scheduling are composed of the hybrid task scheduling in heterogeneous processor multiprocessor environment. When the sporadic task arrives, we need to validate the schedulability of the entire task set, and then take advantage of the better aperiodic soft real-time task scheduling algorithms to steal free time to schedule the sporadic tasks, which effectively guarantees the deadline of periodic real-time tasks and sporadic real-time tasks. Characteristics of the new scheduling algorithm UEDF is that use minimal processor to execute the entire task set on the premise of guaranteeing the deadline, and each tries to assigned it on the processor which is already used to execute tasks. It is conducive to be ready to set aside the idle processor resources for scheduling the task which arrives in future successfully. It verifies both in theory and in the experiment very well. This paper presents that a new mixed scheduling algorithm-combined UEDF algorithm with TCSD algorithm schedules the hybrid task sets which generate in accordance with the appropriate rules, and we find that the proposed mixed scheduling algorithm significantly have a higher success rate than the combination of HEDF algorithms and TCSD algorithms. While the scheduling problem about the hybrid task set is improved in heterogeneous multiprocessor environment, but more complex task sets need to be considered for further research.

References

[1] Geoffery Nelissen, Vandy Berten, Vincent Nelis, Joel Goossens, Dragomir Milojevic."U-EDF: An Unfair but Optimal Multiprocessor Scheduling Algorithm for Sporadic Tasks",24th Euromicro Conference on Real-Time System,2012

[2] Mohamed MAROUF, Laurent GEORGE, Y ves SOREL. "Schedulability analysis for a combination of non-preemptive strict periodic tasks and preemptive sporadic tasks", 18th IEEE International Conference on Emerging Technologies & Factory Automation, 2012.

[3] Hsiang-Kuo Tang, Parmesh Ramanathan, Katherine Compton."Combining Hard Periodic and Soft Aperiodic Real-Time Task Scheduling on Heterogeneous Compute Resources", 2011 International Conference on Parallel Processing, 2011.

[4] S.L.Vieira, M.F. Magalhaes. "On-line Sporadic Task Scheduling in Hard Real-Time System*", IEEE, 1994.

[5] Kevin Jeffay*, Donald F.Stanat, Charles U.Martel**. On Non-Preemptive Scheduling of Periodic and Sporadic Tasks, IEEE, 1991.

[6] Manuel Coutinho, Jose Rufino, Carlos Almeida. "Response Time Analysis of Asynchronous Periodic and Sporadic Tasks scheduled by a Fixed-Priority Preemptive Algorithm", Euromicro Conference on Real-Time Systems,2008.

[7] Jiafu wan el at. "A Two-level Hierarchical Scheduling Scheme for Hybrid Tasks in Priority-Based Preemptive Systems", Networking, Sensing and Control Conference , pp.32-36,2008

[8] Gautam H.Thaker, Patrick J. Lardieri, Dr. Donald K. Krecker, and Michael Price. "Empirical Quantification of Pessimism in State-of-the-Art Scheduling Theory Techniques for Perioidic and Sporadic DRE Tasks", Proceedings of the 10th IEEE Real-Time and Embedded Technology and Application Symposium, 2004.

[9] Joel Goossens, Christophe Macq. "Limitation of the Hyper-Period in Real-Time Periodic Task Set Generation", Proc. In RTS, 2001,pp. 133-148.

[10] M.Joseph and P. K. Pandya. "Finding response times in a real-time system", Comput.J., 29(5):390-395, 1986.