Accelerating exact schedulability analysis for fixed-priority pre-emptive scheduling

Yin Hang, Zhou Jiale
Dep. Intelligent embedded systems
Mälardalen University (MDH)
P.O. Box 883, SE-721 23 Västerås, Sweden
hyn08001@student.mdh.se, zje08001@student.mdh.se

Uğur Keskin, Reinder J. Bril
Dep. Mathematics and Computer Science
Technische Universiteit Eindhoven (TU/e)
Den Dolech 2, 5612 AZ Eindhoven, The Netherlands
U.Keskin@TUe.NL, R.J.Bril@TUe.NL

***Abstract*—The schedulability analysis for fixed-priority preemptive scheduling (FPPS) plays a significant role in the real-time systems domain. The so-called Hyperplanes Exact Test (HET) [1] is an example of an exact schedulability test for FPPS. In this paper, we aim at improving the efficiency of HET by combining it with initial values for exact response time analysis (RTA). We call the resulting improved test HETI and show by means of simulations that HETI is more efficient than HET.**

***Index Terms*—real-time, schedulability, workload, HET, HETI.**

I. Introduction

Fixed-priority preemptive scheduling (FPPS) [2], [3] is one of the main scheduling approaches in modern real-time computing systems. To determine whether or not all tasks of a task set are guaranteed to meet their deadlines, schedulability analysis is used. Despite the fact that exact schedulability tests for FPPS exist for over 25 years [4], there is still a steady flow of new research results reporting improvements in the efficiency of these tests. Today, the most common exact schedulability test is based on Response Time Analysis (RTA), and improvements of RTA mainly focus on new initial values for worst-case response time (WCRT) calculations of tasks [5]–[7], which can drastically accelerate these calculations.

Enrico Bini et al. [1] developed a so-called Hyperplanes Exact Test (HET), which is also an exact schedulability test. Unlike RTA, HET only provides boolean results, e.g. whether a task is schedulable or not, and does not determine WCRTs. Based on simulations, it is concluded in [1] that HET is typically (considerably) more efficient than RTA. Conversely, Davis et al. [7] infer from the data of execution time measurements that, in practice, RTA generally outperforms HET and by some significant margin in case of task sets with a broad spread of task periods.

In this paper, we aim at improving the efficiency of HET by combining it with initial values for RTA. We call the resulting improved test HETI, and compare the efficiency of HETI and HET by means of simulations.

The remainder of this paper is organized as follows. In Section 2, we introduce our scheduling model for FPPS and the notation used throughout the paper. We recapitulate HET and RTA in Section 3. Section 4 explains how HET can be improved by means of the initial values of RTA and presents the resulting HETI. In Section 5, HETI is compared with HET by means of simulations. In Section 6, we draw our conclusion.

II. Scheduling model and notation

In this section, we introduce our scheduling model for FPPS and the notation used throughout the paper. We consider a task set $\Gamma_n = \{\tau_1, ..., \tau_n\}$, where all tasks are (strictly) periodic and independent, have unique priorities, and do not suspend themselves. Tasks of Γ_n are indexed based on decreasing priority, i.e. τ_1 has highest and τ_n has lowest priority. A task τ_i consists of a sequence of instances (or jobs), where instance k is denoted as τ_{ik}. A task instance has its own release time r_{ik} and finalization time f_{ik}. Task τ_i of Γ_n is characterized by an initial activation time Φ_i (or phasing), a worst-case computation time C_i, a period T_i and a relative deadline D_i no greater than T_i. We assume arbitrary phasing for tasks. For a task τ_i, the initial value for calculating its worst-case response time R_i is denoted as ι_i. The utilization of τ_i is represented by $U_i^\tau = C_i/T_i$, and the utilization of the first i tasks is presented by $U_i = \sum_{j=1}^{i} U_j^\tau$. Theoretically speaking, it is safe to say that all the parameters are positive rational numbers, because the rational numbers provide a dense time domain. For the sake of simplicity, we use positive integers.

III. Recapitulation of HET and RTA

In this section, we briefly recapitulate HET and RTA. For HET, we first present the underlying notions of C-space and P set.

A. The P set in C-space

The essence of HET originates from the so-called C-space. For a given task set Γ_n with periods from T_1 to T_n, deadlines from D_1 to D_n and computation times from C_1 to C_n, the C-space $\mathbb{M}_n$ can be defined as:

$$\mathbb{M}_n(T_1, ..., T_n, D_1, ..., D_n) = \{(C_1, ..., C_n) \in R_+^n : \Gamma_n \text{ is schedulable by FP}\}. \quad (1)$$

Every coordinate of $\mathbb{M}_n$ is represented by the computation time C_i of a task τ_i, hence C-space. Based on C-space, the following theorem is presented in [1].

Theorem 1: The region of the schedulable task sets $\mathbb{M}_n$, as defined by (1), is given by

$$\mathbb{M}_n(T_1, ..., T_n, D_1, ..., D_n) = \{(C_1, ..., C_n) \in R_+^n : \\ \bigwedge_{i=1...n} \bigvee_{t \in P_{i-1}(D_i)} C_i + \sum_{j=1}^{i-1} \left\lceil \frac{t}{T_j} \right\rceil C_j \le t\}, \tag{2}$$

where $P_i(t)$ is defined by the following recurrent expression:

$$\begin{cases} P_0(t) = \{t\} \\ P_i(t) = P_{i-1}\left(\left\lfloor \frac{t}{T_i} \right\rfloor T_i\right) \bigcup P_{i-1}(t) \end{cases} \tag{3}$$

It is not hard to notice that the P set has a tree structure, where each node has two children nodes and these two branches could overlap. Apparently, each task is related to its own P set values. In this paper, the P set of τ_i will be expressed as P_i. P will be the set consisting of P set values of all the tasks (from high priority to low priority) in a given task set. The cardinality of (i.e. number of elements in the set) P_i and P is denoted by N_{P_i} and N_P, respectively. The theorem above together with the concept of P set represents the main thought of HET.

B. HET based on P set

HET is tightly related to P. However, before we discuss HET, it is necessary to know the concept of workload and ψ. A job τ_{ik} is said to be active at time t if $r_{ik} < t < f_{ik}$. The processor is *i-busy* at time t if there exists a job of a task τ_i in Γ_n active at t. The workload and ψ are defined as:

Definition 1: The worst-case workload $W_i(b)$ of the i highest priority tasks in $[0, b]$ is the total time the processor is i-busy in $[0, b]$. By extension, $W_0(b) = 0$ for all b.

Definition 2: Given the subset Γ_i of the i highest priority tasks, $\psi_i(b)$ is the last instant in $[0, b]$ in which the processor is not *i-busy*,that is:

$$\psi_i(b) = max\{t \in [0, b] \wedge t \notin Busy(\Gamma_i, b)\}.$$

Theorem 1 can be directly applied to a schedulability analysis, yet it is more efficient to convert it into the workload form, even though they are essentially equivalent. In the workload form, the schedulability condition in Theorem 1 can be expressed as

$$\forall i = 1...n \qquad C_i + W_{i-1}(D_i) \le D_i. \tag{4}$$

As a result, the P set calculation is transformed into workload calculation. [1] has deduced the final form of workload computation in a recurrent style:

$$W_i(b) = min\{b - f(T_i - C_i) + W_{i-1}(fT_i), cC_i + W_{i-1}(b)\} \tag{5}$$

where $f = \left\lfloor \frac{b}{T_i} \right\rfloor$ and $c = \left\lceil \frac{b}{T_i} \right\rceil$.

Besides, the relationship between workload and ψ is also deduced in [1]:

$$\begin{aligned} W_i(b) &= \min_{t \in [0,b]} \sum_{j=1}^{i} \left\lceil \frac{t}{T_j} \right\rceil C_j + (b - t) \\ &= \sum_{j=1}^{i} \left\lceil \frac{\psi_i(b)}{T_j} \right\rceil C_j + (b - \psi_i(b)). \end{aligned} \tag{6}$$

The schedulability test based on the recurrent form of (5) is called Hyperplanes Exact Test (HET). Even though (5) seems to be independent of P, yet P is implicitly generated during the workload calculation (please note that fT_i and b belong to P). The two branches of the workload are directly related with the two branches of P. Since P can recursively produce identical values from the two branches, the same is true of workload. This can often reduce complexity and speed up schedulability determination.

C. Initial values for RTA

The worst-case response time R_i of a task τ_i is determined by means of an iterative procedure starting with a lower bound as initial value, i.e.

$$\begin{cases} R_i^{(0)} = \iota_i \\ R_i^{(k)} = C_i + \sum_{j<i} \left\lceil \frac{R_i^{(k-1)}}{T_j} \right\rceil C_j, \end{cases} \tag{7}$$

where C_i is an appropriate value for ι_i. The procedure terminates when either $R_i^{(k)} = R_i^{(k-1)}$ or $R_i^{(k)} > D_i$. In the former case, $R_i^{(k)}$ will be the WCRT and in the latter case task τ_i is not schedulable. To reduce the number of required iterations, the most straightforward way is to start with a higher lower bound as initial value. An alternative initial value based on [5] for parameters from the positive integers is given by

$$\iota_i = \left\lceil \frac{C_i}{1 - U_{i-1}} \right\rceil. \tag{8}$$

When response times are determined in priority order and because $\iota_{i-1} + C_i$ is also an appropriate lower bound for R_i, it is also possible to use

$$\iota_i = \max\left\{\left\lceil \frac{C_i}{1 - U_{i-1}} \right\rceil, \iota_{i-1} + C_i\right\} \qquad (i > 1). \tag{9}$$

Because we aim at improving HET by using ι_i, values for ι_i that are based on R_{i-1} fall outside the scope of this paper.

In the following sections, ι_i will be derived using (9) and ι will be the set of ι_i for all tasks in Γ_n.

IV. P SET REDUCTION AND HETI

Let's go back to Theorem 1 and see how P works in the schedulability analysis. It is self-evident that the left part of (2) looks similar to the right part of (7). In fact, Theorem 1 can even be translated into the schedulability analysis based on both P and (7). For a task τ_i with corresponding P_i, what we need to test is to put all the elements in P_i into (7) as $R_i^{(k-1)}$. τ_i is schedulable if at least one element in P_i can yield a $R_i^{(k)}$ that is no larger than itself through (7).

The highly correlated nature between Theorem 1 and (7) enables ι to play a key role in pruning P, contributing faster schedulability determination. In WCRT calculation, ι_i draws a lower bound of R_i, and any $R_i^{(k-1)}$ smaller than ι_i will surely yield a larger $R_i^{(k)}$ through (7). Likewise, for τ_i, elements in P_i smaller than ι_i can be pruned because we are sure that they will not satisfy the condition in (2). Once P_i is pruned, it will have fewer elements. According to Theorem 1, fewer elements in P_i indicates that schedulability can be determined faster. Through the rest of this paper, we define P'_i as the pruned P_i and P' as the pruned P. Besides, $N_{P'_i}$ and $N_{P'}$ will be the cardinality of P_i and P' respectively.

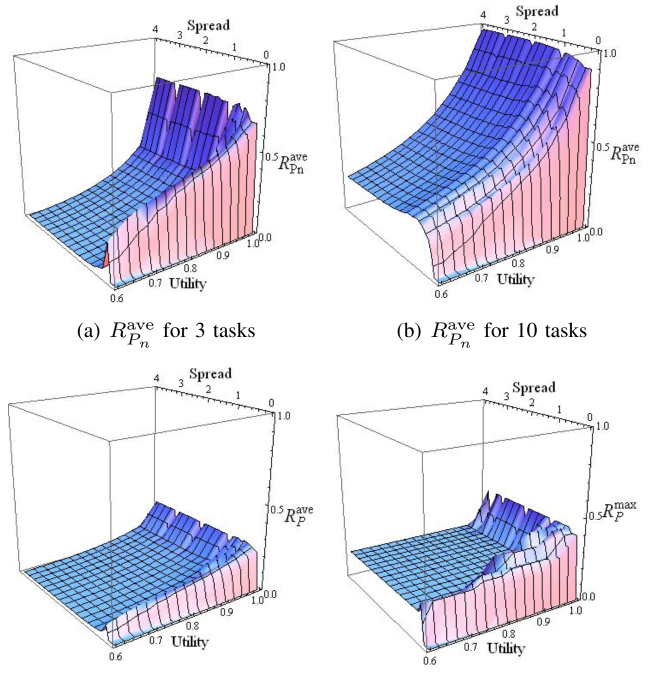
Now let's see how P set reduction can positively influence HET. On the one hand, we already know that both the two workload branches in (5) contain P set elements. On the other hand, (6) implies $\psi_i(b)$ in the interval $[0, b]$ contributes to the minimum $W_i(b)$, which corresponds to the minimal branch in (5) as the real workload. Furthermore, it has been proven in [1] that for interval $[0, b]$, $\psi_i(b) \in P_i(b)$. Consequently, P set reduction restricts the candidates of $\psi_i(b)$ further to $P'_i(b)$. In other words, $\psi_i(b)$ cannot be one of those P set elements whose values are smaller than ι_i. Thus we are sure that the workload branches in (5) that contain P set values smaller than ι_i will never produce a workload value smaller than the other workload branch and they should be pruned (Please note that only the smaller workload branch is preserved in (5)). Once those workload branches are pruned, schedulability will be determined even faster. We call this resulting improved test HETI.

V. Simulation

In this section, we will show how much P and P_i can be pruned by ι_i, and more importantly, how much HET can be improved by HETI. To this end, we consider the number of tasks, utility ($\sum_{i=1}^{n} \frac{C_i}{T_i}$) and spread ($lg(\frac{T_n}{T_1})$). As we shall explain in separate subsections, the performance is evaluated by P set reduction ratio (R_{P_n} and R_P) and workload reduction ratio (R_W). In both cases, tasks to be tested are generated according to the specified task number with uniform distribution of utility and spread [6]. To guarantee accuracy, we repeat our computation of R_{P_n}, R_P and R_W 10000 times for each combination of specified task number, utility and spread and then calculate their average, i.e. $R_{P_n}^{\text{ave}}$, R_P^{ave} and R_W^{ave}. This combination in our simulation is constrained to 3 and 10 tasks, with utility ranging from 60% to 100% and spread ranging from 0 to 4.

A. P set reduction ratio

We use P set reduction ratio, which consists of P_n reduction ratio (R_{P_n}) and P reduction ratio (R_P), to evaluate how much the P set can be pruned by the initial value. For a task τ_i, we define the P_i reduction ratio (R_{P_i}) as $(N_{P_i} - N_{P'_i})/N_{P_i}$. Higher R_{P_i} for τ_i means that more P_i values can be pruned by ι_i. Due to the fact that τ_n (the task with the lowest priority) is most likely to present high R_{P_i}, only R_{P_n} is tested.



(a) $R_{P_n}^{\text{ave}}$ for 3 tasks (b) $R_{P_n}^{\text{ave}}$ for 10 tasks

(c) R_P^{ave} for 3 tasks (d) $R_P^{\max}$ for 3 tasks

Fig. 1. R_{P_n} and R_P test

Furthermore, to evaluate the overall performance, we define the R_P of an entire task set as $(N_P - N_{P'})/N_P$. Apparently, R_P is influenced by the R_{P_i} of all tasks.

The simulation results of $R_{P_n}^{\text{ave}}$ for 3 tasks and 10 tasks are displayed in Figure 1(a) and 1(b), indicating that $R_{P_n}^{\text{ave}}$ increases for increasing task number and utility. For task sets with a spread lower than 1, the increase of spread will also lead to the increase of $R_{P_n}^{\text{ave}}$, which, however, will increase fairly slowly when the spread is higher than 1.

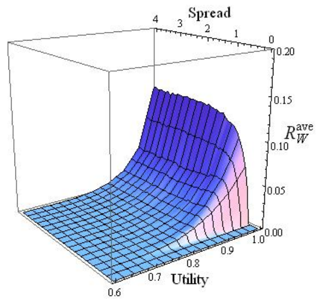
We also tested the R_P for all tasks in the same way. Figure 1(c) shows R_P^{ave} for 3 tasks, from which we can conclude that for a specified task set, compared with $R_{P_n}^{\text{ave}}$, R_P^{ave} is relatively lower. Moreover, it is also interesting to know the maximal case. Figure 1(d) shows the maximal R_P, i.e. $R_P^{\max}$, for 3 tasks, which does not change so radically with varying utilization.

B. Workload reduction ratio

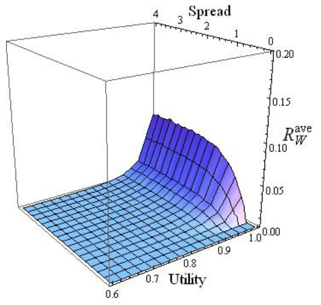
HETI has its advantage in fewer iterations by pruning workload branches containing a P_i value that is smaller than ι_i. Its simulation is conducted in the same way as the P set reduction ratio test. Let N_{HET} and N_{HETI} denote the number of steps to calculate $W_{i-1}(T_i)$ for τ_i for all tasks in the original HET and HETI, respectively. The workload reduction ratio R_W is now defined as

$$R_W = (N_{HET} - N_{HETI})/N_{HET}.$$

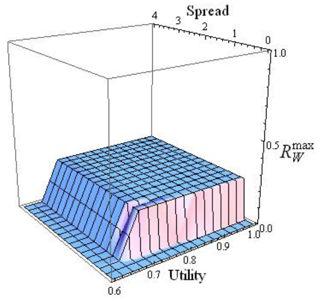
After defining R_W, we try to find its relationship with number of tasks, utility and spread. As is illustrated in Figure 2(a) and 2(b) for 3 and 10 tasks, the simulation result indicates that the



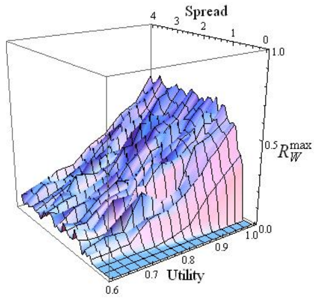
(a) R_W^{ave} for 3 tasks



(b) R_W^{ave} for 10 tasks



(c) $R_W^{\max}$ for 3 tasks



(d) $R_W^{\max}$ for 10 tasks

Fig. 2. R_W test

average R_W^{ave} increases for increasing utility, but decreases for an increasing number of tasks. Moreover, the improvements are not as remarkable as for R_P^{ave}. The reason is that the original HET keeps updating $\psi_i(b)$, which is non-decreasing, as the workload is calculated from the first task to the last task. For instance, if we have got $W_1(T_2)$ after testing τ_2 with the original HET, we will prune any workload branch with $W_i(b)$ where $b \leq T_2$. That is to say, $\psi_i(b)$ must be non-decreasing as its value is updated by HET of lower priority tasks. This accumulative effect of updated $\psi_i(b)$ has already been quite effective to prune redundant workload branches, in the same way as ι_i does. HETI based on ι_i is indeed better than the original HET, yet apparently any substantial enhancement will be rather challenging.

The maximal values for R_W for 3 and 10 tasks are illustrated in Figure 2(c) and 2(d). It is quite evident that the maximal value is much higher than the average value, almost 30% and 70% higher for 3 and 10 tasks respectively. Different from the average case, the maximum value $R_W^{\max}$ of R_W has an increasing trend as the number of tasks increases. By observing Figure 2, we can also conclude that R_W is fairly independent of the spread of a task set, while task utility plays a dominating role. The simulation result in maximal cases suggests that it is certainly worthwhile to implement HETI for schedulability analysis based on FPPS, even though the enhancement on average is not so conspicuous.

VI. Conclusion

We have proposed HETI as an improvement of HET by combining HET with initial values for exact response time analysis. Using the initial value to determine the WCRT of a task, we first studied the impact of pruning the P set for that task by removing all values from P smaller than the initial value. Based on our simulations, we conclude that the P set can be reduced by pruning, and that the reduction becomes more and more remarkable for increasing task set utilizations. Next, we studied pruning workload branches for values smaller than the corresponding initial value, being our proposed improvement of HET. Although on average HETI cannot speed up HET as much as the P set reduction, its advantage is not negligible in best cases, especially for a large number of tasks and high task set utilizations.

Because HETI does not significantly improve HET on average, we expect that the conclusions for HET drawn in [1], [7] will hardly change for HETI.

References

[1] E. Bini and G. C. Buttazzo, "Schedulability analysis of periodic fixed priority systems," *IEEE Transactions on Computers*, vol. 53, no. 11, pp. 1462–1473, November 2004.

[2] M. Klein, T. Ralya, B. Polak, R. Obenza, and M. G. Harbour, *A practitioner's Handbook for Real-Time Analysis-Guide to Rate Monotonic Analysis for Real-Time Systems*. Kluwer Academic Publishers, Boston, 1993.

[3] J. Liu, *Real-Time Systems*. Prentice Hall, 2000.

[4] P. Harter, "Response times in level-structured systems," Department of Computer Science, University of Colorado, USA, Tech. Rep. CU-CS-269-84, 1984, http://www.cs.colorado.edu/department/publications/reports/docs/CU-CS-269-84.pdf.

[5] M. Sjödin and H. Hansson, "Improved response-time analysis calculations," *Proc. 19th IEEE Real-Time Systems Symposium*, pp. 399–409, December 1998.

[6] R. J. Bril, W. F. Verhaegh, and E.-J. D. Pol, "Initial values for on-line response time calculations," *Proc. 15th Euromicro Conference on Real-Time Systems*, pp. 13–22, July 2003.

[7] R. Davis, A. Zabos, and A. Burns, "Efficient exact schedulability tests for fixed priority real-time systems," *IEEE Transactions on Computers*, vol. 57, no. 9, pp. 1261 –1276, September 2008.