Real-Time Scheduling Algorithm Based on RMZL and Schedulability

Ken Yanai, Myungryun Yoo, Takanori Yokoyama

Abstract – Recently, multiprocessor platform is generally used in embedded real time systems. The scheduling of recurrent real-time tasks on multiprocessors has been one of the primary subjects in the real time systems, ever since Dhall and Liu demonstrated the RM (Rate Monotonic) scheduling is no longer optimal on multiprocessors. Several algorithms based on RM are proposed. However the optimal scheduling algorithm on multiprocessors is not established yet. In this study, we propose a scheduling algorithm on multiprocessors, called RMZL with Pseudo Deadline (RMZLPD) based on Rate Monotonic until Zero Laxity(RMZL) applied zero-laxity rule to RMLRMZLPD can realize high parallelism. Through simulation, RMZLPD has shown the high schedule success ratio. The schedulability of proposed algorithm also is shown by response time analysis.

Keywords – Multiprocessor, Rate Monotonic, Real Time System, Schedulability, Scheduling.

I. INTRODUCTION

Real time systems are characterized by computational activities with timing constraints. Timing constraints in real time applications are predominantly soft in that deadlines may be missed as long as the long run fraction of the processing time allocated to each task in the application is in accordance with its utilization. A system design that can guarantee that deadline misses, if any, are bounded by constant amounts is sufficient to provide guarantees on long term processor shares. Hence, scheduling methods that ensure bounded deadline misses and that can be applied when other methods cannot are of considerable value and interest [1].

Multiprocessor scheduling are usually categorized into two paradigms: global scheduling, in which each task can execute on any available processor at runtime, and partitioned scheduling in which each tasks is assigned to a processor beforehand, and at runtime each task can only execute on this particular processor. Partitioned scheduling enjoys relatively easier design and analysis. On the other hand, global scheduling on average utilizes computing resource better, and is more robust in the presence of timing errors [2]. Global scheduling algorithms are based on widely optimal uniprocessor scheduling algorithms like RM (Rate Monotonic) and EDF (Earliest Deadline First) by Dhall [3]. In this paper, we focus in global scheduling algorithm based on RM.

It is widely known that a set of independent periodic tasks, in which the deadline of each task is equal to its period, is always successfully scheduled by RM on single processor if the total utilization of the tasks does not exceed \( \tau = \frac{1}{n} \cdot (1 + \frac{1}{2^n}) \), where, \( n \) is the total number of tasks. Unfortunately, this optimality of RM breaks down on multiprocessor systems. Takeda et al. proposed RMZL (Rate Monotonic until zero laxity) based on RM [4][5] and Nishigaki et al. proposed LP-RMZL (Limited Preemptive-RMZL) based RMZL [6]. These algorithms show higher schedule success ratio than that of RM. However, there are still reducible deadline misses.

In this paper, we propose a new scheduling algorithm, called RMZLPD (RMZL with Pseudo Deadline). The proposed algorithm dominates global RM scheduling, RMZL and LP-RMZL. Also, we analyze schedulability of RMZLPD using Response Time Analysis (RTA).

The rest of the paper is organized as follows: In Section 2, we explain system model. In Section 3, global RM, RMZL and LP-RMZL are explained more detail. Section 4 introduces proposed RMZLPD. In section 5, a schedulability of RMZLPD is analyzed. Then, the experimental results are illustrated and analyzed in Section 6. Finally, Section 7 provides discussion and suggestions for further work on this problem.

II. SYSTEM MODEL

The notation is described in this Section. We consider a set \( J \) of \( n \) periodic tasks to be scheduled on \( m \) symmetric processors using a global algorithm.

Each task \( t = (C_k, T_k) \in J \) is characterized by a \( (C_k, T_k) \) tuple, where \( C_k \) is its worst-case computation time and \( T_k \) is its period. The utilization of \( t \) is defined by \( U_k = C_k/T_k \). The system utilization is defined by \( U = \sum_k U_k/j \). A task \( t \) has the sequence of jobs \( J_k \), where each job is characterized by an arrival time \( r_k \) and a finish time \( f_k \). Moreover, each job has an absolute deadline \( d_k = r_k + T_k \). The laxity of a job at time \( r \) is defined as \( \ell_k = d_k - r - C_k(t) \), where, \( C_k(t) \) is a remaining execution time of job \( J_k \) at time \( t \).

![Fig.1. The laxity of a job](image)

III. RELATED WORKS

In this paper, the proposed scheduling algorithm is based on global RM. This section describes global RM and several scheduling algorithms based on global RM. A. Global RM (Rate Monotonic)

Global RM is preemptive fixed priority scheduling algorithm. Tasks with higher request rates will have higher
priorities in global RM. But, owing to low schedulability, it is known that global RM which is applied for multiprocessor platforms is not optimal scheduling algorithm [3]. Fig.2 showsthe scheduling exampleof global RM, when three periodic tasks, \( r_1 = r_2 = r_3 = (2, 3) \) are submitted on two processors. As shown in the Fig.2, \( r_3 \) misses a deadline at time 3.

![Fig.2. Example of RM schedule](image)

**B. RMZL (Rate Monotonic until zero laxity)**

RMZL [4] is based on global RM. Under RMZL, jobs are scheduled according to the fixed priority of their associated task, until a situation is reached where the remaining execution time of a job is equal to the time to its deadline. Such a job has zero laxity and will miss its deadline unless it executes continually until its completion.

RMZL gives the highest priority to such zero-laxity jobs. The schedules produced by RMZL and global RM scheduling are identical until the latter fails to execute a task with zero laxity. Such a task will subsequently miss its deadline. Hence RMZL dominates global RM scheduling, in the sense that all priority ordered tasksets that are schedulable according to global RM scheduling are also schedulable according to RMZL. Fig.3 showsthe scheduling exampleof RMZL with same task set as the example of global RM in Fig.2. As shown in the Fig.3, all the three tasks are successfully scheduled by RMZL, since the priority of \( r_3 \) is promoted to the top at time 1 due to zero laxity.

![Fig.3. Example I of RMZL schedule](image)

**C. LP-RMZL (Limited Preemptive-RMZL)**

LP-RMZL [6] is based on RMZL. Under LP-RMZL, running jobs are not preempted by higher priority tasks except for zero-laxity tasks. Compared to RMZL, LP-RMZL reduced preemption and improved success ratio and schedulability. Fig.4 and Fig.5 show the scheduling exampleof RMZL and LP-RMZL, when four periodic tasks, \( r_1 = r_2 = r_3 = (1, 4) \) and \( r_4 = r_5 = (6, 8) \) are submitted on two processors. As shown in the Fig.4, \( r_2 \) misses a deadline at time 8 in RMZL. On the other hand, all the four tasks are successfully scheduled by LP-RMZL in Fig.5, since the \( r_2 \) is not preempted at time 2, 4, and 6.

![Fig.4. Example II of RMZL schedule](image)

![Fig.5. Example I of LP-RMZL schedule](image)

**IV. RMZLPD (RMZL with Pseudo Deadline)**

We propose RMZLPD added pseudo deadline to RMZL. Under RMZL, jobs are scheduled according to RMZL, until a situation is reached where the remaining pseudo execution time of a job is equal to the time to its pseudo deadline. Such a job has pseudo zero laxity and will miss its pseudo deadline unless it executes continually until its pseudo deadline.

RMZLPD gives the semi highest priority to such pseudo zero-laxity jobs until its pseudo deadline. Pseudo Deadline is set on the half deadline. Pseudo execution time also is set on the half execution time. Fig.6 and Fig.7 showsthe scheduling exampleof LP-RMZL and RMZLPD, when five periodic tasks, \( r_1 = r_2 = r_3 = (1, 4) \) and \( r_4 = r_5 = (6, 12) \) are submitted on two processors. As shown in the Fig.6, \( r_3 \) misses a deadline at time 3 in LP-RMZL. On the other hand, all the five tasks are successfully scheduled by RMZLPD in Fig.7, since \( r_3 \) has the semi highest priority at time 5 due to pseudo zero laxity.
The flow of an algorithm analysis is shown below.

As for interference, the following lemma 1 is showed.

Therefore, the upper bound of the interference of $\tau_k$, $I_{ub}^k$ can be calculated by calculating the upper bound of the workload of $\tau_k$, $W_{lb}^k$.  

B. Schedulability of RMZLPD

Under RMZLPD scheduling, if the laxity or the pseudo laxity of a job reaches zero then it is given the highest or the semi highest priority. Therefore, $\tau_k$ is interfered with not only higher static priority tasks but also lower static priority tasks. If $\tau_k$ is higher static priority than $\tau_i$, the workload of $\tau_i$ over an interval $[r^*_k, r^*_k + R_k^{ub}]$ is represented in Fig.8.

Thus, the upper bound of the workload, $W_{lb}^k (r^*_k, r^*_k + R_k^{ub}) = W_{lb}^k (R_k^{ub})$ is calculated as following equation (3).

Therefore, the upper bound of the interference of $\tau_k$, $I_{ub}^k$ is calculated as following equation (4).

The schedulability of an algorithm is analyzed using $L_m^b$.

In here, we have lemma 2 as for $I_{ub}^k$.

**Lemma 2:** $I^k_{ub} (a, b)$ is always smaller than $W(a, b)$.
where \( n_i(R_i^{ub}) \) is the maximum number of jobs of task \( \tau_i \) that contribute all of their execution time in the interval and it is calculated as following equation (5).

\[
n_i(R_i^{ub}) = \left\lceil \frac{R_i^{ub} + T_i - C_i - R_i^{lb}}{r_i} \right\rceil
\]

From the above equations, the following two theorems are obtained.

**Theorem 1 (RTA for RMZLPD):** An upper bound on the response time of a task \( \tau_k \) in a multiprocessor system scheduled with RMZLPD can be derived by the fixed point iteration on the value \( R_k^{ub} \) of the following equation (6), starting with \( R_k^{ub} = C_k \):

\[
R_k^{ub} \leftarrow C_k + \frac{1}{m} \sum_{i=1}^{m} I'_k(R_i^{ub})
\]

where, \( I'_k(R_i^{ub}) \) is calculated as following equation (7).

\[
I'_k(R_i^{ub}) = \min W_i^{ad}(R_i^{ub}, R_i^{ub} - C_k + 1)
\]

**Theorem 2 (Schedulability for RMZLPD):** A periodic task system \( \tau = \{\tau_1, \ldots, \tau_n\} \) is schedulable by RMZLPD on \( m \) symmetrical processors unless the following inequality (8) holds for least \( m + 1 \) different tasks \( \tau_k \), and it holds strictly (<) for at least one of them:

\[
I_k^{ad} \leq 0
\]

**VI. EXPERIMENTAL EVALUATION**

In order to see how well the RMZLPD algorithm and the above schedulability test perform, a series of experiments were conducted with the same simulation environment [4]. We compared proposed RMZLPD with RM by Dhall and Liu [3], RMZL by Takeda et al. [4] and LP-RMZL by Nishigaki et al. [6]. Numerical tests are performed with a randomly generated task set.

**A. Schedulability**

Fig. 10, 11, 12 show the result of simulation about schedulability when 1,000 task set (system utilization 0.3 – 1.0) are submitted on 4 processors, 8 processors and 16 processors. The schedulability is calculated as following equation (9).

\[
Schedulability = \frac{\text{scheduled task set}}{\text{submitted all task set}}
\]

It indicates that RMZLPD is superior to the other algorithms over an interval [30%, 100%].

**B. Success Ratio**

Fig. 13, 14, 15 show the result of simulation about success ratio, when 1,000 task set (system utilization 0.3 – 1.0) are submitted on 4 processors, 8 processors and 16 processors. The success ratio is calculated as following equation (10).

\[
Success Ratio = \frac{\text{successful task set}}{\text{submitted all task set}}
\]
C. Preemption

Fig. 16, 17, 18 show the result of simulation about preemptions, when 1,000-task set (system utilization 0.3 ~ 1.0) are submitted on 4 processors, 8 processors and 16 processors. The number of preemptions is calculated as following equation (11).

\[
\text{Number of Preemption} = \frac{\text{total number of preemptions}}{\text{submitted all task set}}
\]  

(11)

In Fig. 16, 17, 18, the number of preemption of proposed RMZLPD is a little bit more than those of the other three algorithms on high system utilization. However, RMZLPD is equally evaluated as the other three algorithms in low system utilization about preemption.

VII. CONCLUSIONS

A new tasks scheduling algorithm on real time multiprocessor system is proposed in this paper. The schedulability of the proposed algorithm, RMZLPD, is analyzed using RTA. RMZLPD has high schedule success ratio and can realize high parallelism. From the numerical results, the results of the proposed RMZLPD are better than that of other algorithms. However, there are a gap between the schedulability and the success ratio.

This determines the next step of our study. We plan to analysis the schedulability with another method other than RTA.

ACKNOWLEDGMENT

This work is supported in part by JSPS KAKENHI Grant Number 24500046.

REFERENCES


AUTHOR’S PROFILE

Ken Yanai
received B. E. from Tokyo City University, Japan in 2011. He is a student in master course, Tokyo City University. His research field is Real-Time System, Scheduling, Multimedia System, etc.
Email: ar138432@gmail.com

Myungryun Yoo
received B.E from Andong National University, Korea in 1994, M.S from Pohang University of Science & Technology, Korea in 1996 and Ph.D. degree from Graduate School of Information, Production & Systems, Waseda University, Japan in 2006. She joined Tokyo City University as Associate Professor in 2006. Her research field is Real-Time System, Scheduling, Multimedia System, etc.
Email: yoo@cs.tcu.ac.jp

Takanori Yokoyama
received B.E. degree, M.E. degree, and Dr. degree in information science from Tohoku University in 1981, 1983, and 2002 respectively. He joined Hitachi, Ltd. in 1983. He joined Musashi Institute of Technology in 2004. He is now a professor of Tokyo City University. His research interest includes embedded systems, distributed systems and software engineering. He is a member of IEEE, ACM, IPSJ.
Email: yokoyama@cs.tcu.ac.jp

Copyright © 2013 IJEIR, All right reserved