International Journal of Engineering Innovation & Research
Volume 2, Issue 6, ISSN: 2277 – 5668

Real-Time Scheduling Algorithm Based on RMZL and Schedulability

Ken Yanai, MyungryunYoo, 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 RM.RMZLPD 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 $n(2^{1/n}\text{-}1)$, where, n is the total number of tasks. Unfortunately, this optimality of RM breaks down onmultiprocessor 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 miss.

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 τ of n periodic tasks to be scheduled on $m$symmetric processors using a global algorithm.

Each taskτ_k= $(C_k,T_k) \in \tau$ is characterizedby a $(C_k,\ T_k)$ tuple, where C_k is its worst-case computation time and T_k is its period. The utilization ofτ_k is defined by $U_k = C_k/T_k$. The system utilization is defined by$U = \sum_{\tau k \in \tau}(U_k)/m$. A task$\tau_k$has the sequence of jobs J_{jk}, where each job ischaracterized by an arrival time r_{jk} and a finish time f_{jk}. Moreover, each job has an absolute deadline $d_{jk} = r_{jk} + T_k$. The laxityof a job at time $t$is defined as $L_{jk} = d_{jk} - t - C_{jk}(t)$, where, $C_{jk}(t)$ is a remaining execution time of job J_{jk} at time t. Fig.1showsthe laxityof J_{jk} at time t.

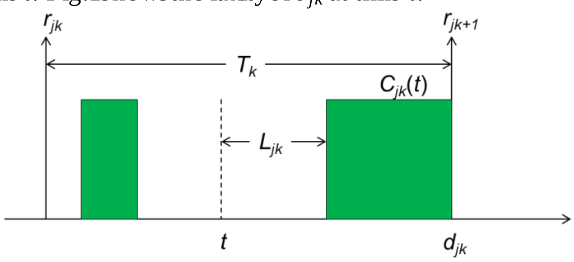

Fig.1. The laxityof a job

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

International Journal of Engineering Innovation & Research
Volume 2, Issue 6, ISSN: 2277 – 5668

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.2showsthe scheduling exampleof global RM, when three periodic tasks, τ_1 =τ_2 =τ_3= (2, 3) are submitted on two processors. As shown in the Fig.2, τ_3 misses a deadline at time 3.

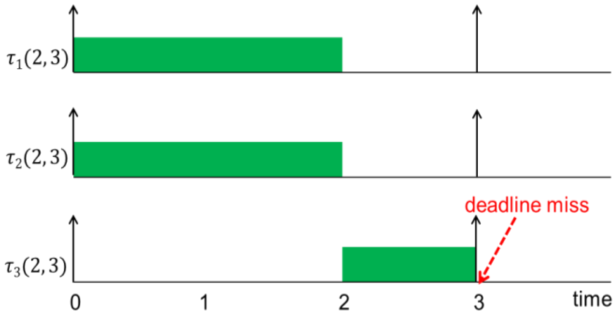

Fig.2. Example of RM schedule

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

RMZL [4] is based on global RM. Under RMZL, jobs are scheduled according to thefixed priority of their associated task, until a situation isreached where the remaining execution time of a job is equalto the time to its deadline. Such a job has zero laxity and willmiss its deadline unless it executes continually untilcompletion.

RMZL gives the highestpriority to such zero-laxity jobs. The schedules produced by RMZL and global RM scheduling are identical until the latter fails to execute a taskwith zero laxity. Such a task will subsequently miss its deadline. Hence RMZL dominates global RM scheduling, inthe sense that all priority ordered tasksets that areschedulable according to global RM scheduling are alsoschedulable according to RMZL. Fig.3showsthe 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 τ_3 is promoted to the top at time 1 due to zero laxity.

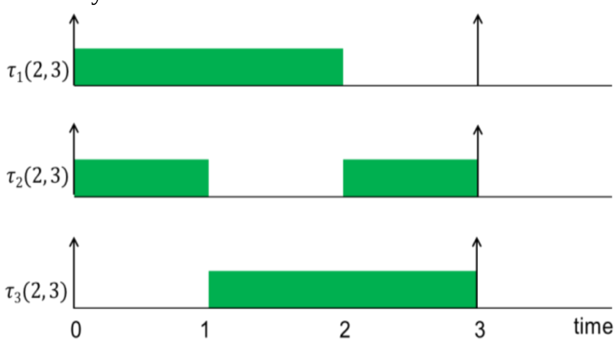

Fig.3. Example I of RMZL schedule

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.5show the scheduling exampleof RMZL and LP-RMZL, when four periodic tasks, τ_1 =τ_2 = (1, 2),τ_3= (1, 4) and τ_4 = (6, 8) are submitted on two processors. As shown in the Fig.4, τ_4 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 τ_4 is not preempted at time 2, 4, and 6.

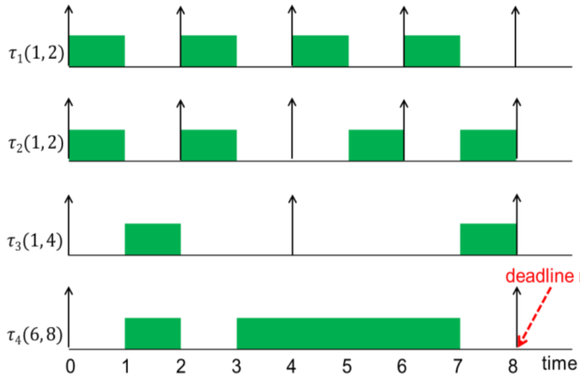

Fig.4. Example II of RMZL schedule

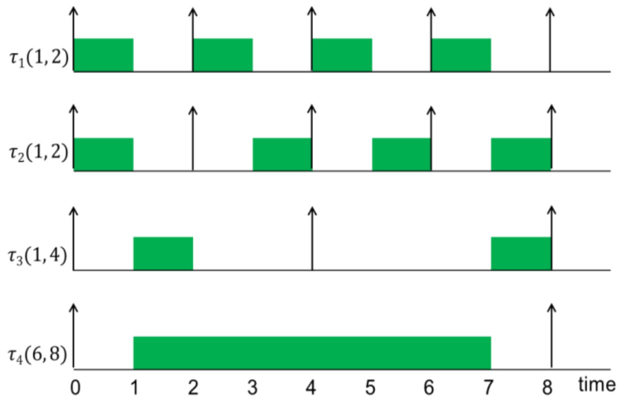

Fig.5. Example I of LP-RMZL schedule

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 isreached where the remaining pseudo execution time of a job is equalto the time to its pseudo deadline. Such a job has pseudo zero laxity and willmiss its pseudo deadline unless it executes continually until its pseudo deadline.

RMZLPD gives the semi highestpriority 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.7showsthe scheduling exampleof LP-RMZL and RMZLPD, when five periodic tasks, τ_1 = τ_2 = τ_3= (1, 4) and τ_4 = τ_5 = (6, 12) are submitted on two processors. As shown in the Fig.6, τ_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 τ_5 has the semi highest priority at time 5 due to pseudo zero laxity.

International Journal of Engineering Innovation & Research
Volume 2, Issue 6, ISSN: 2277 – 5668

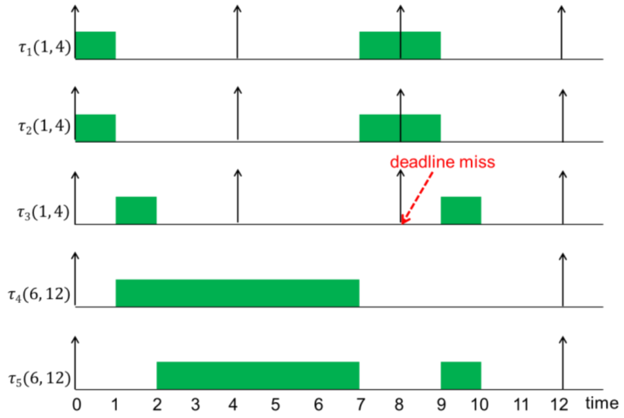

Fig.6. Example II of LP-RMZL schedule

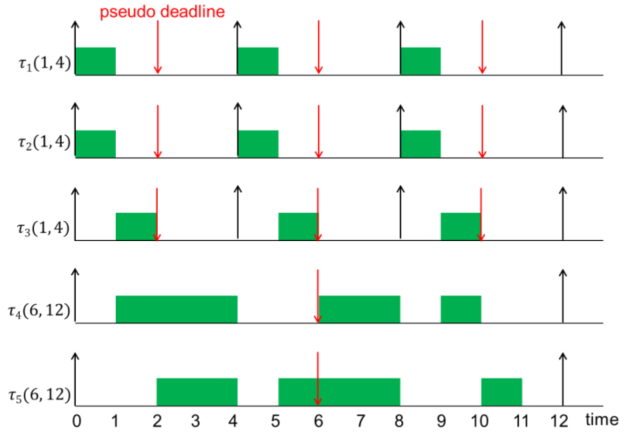

Fig.7. Example of RMZLPD schedule

V. SCHEDULABILITY

In this section, we derive sufficient schedulability test for RMZLPD with Response Time Analysis (RTA) [7].

A. Interference and Workload

To analyze the schedulability, we define two parameters: the *interference* and the *workload*.

Interference: The interference $I_k(a, b)$ on τ_k over an interval $[a, b]$ is the cumulative length of all intervals inwhich τ_k is backlogged but cannot be scheduled on any processordue to the contemporary execution of m higher prioritytasks.We also define the interference $I_k^i(a, b)$ of a task τ_i on a task τ_k over an interval $[a, b]$ as the cumulative length ofall intervals in which τ_k is backlogged but cannot be scheduledon any processor, while τ_i is executing.

Workload: The workload $W_k(a, b)$ of a task τ_k in an interval $[a, b)$ represents the amount of computation that the taskrequires in $[a, b)$ on a given situation.

As for interference, the following lemma 1 is showed.

Lemma 1: For any global scheduling algorithm, the following equation (1) is established [8]:

$$I_k(a,b) \ge x \Leftrightarrow \sum_{i \ne k} \min(I_k^i(a,b), x) \ge mx \tag{1}$$

The flow of an algorithm analysis is shown below.

Let J_k^* be the job of τ_k with maximum interference. The upper bound of I_k, I_k^{ub}, can be calculated over an interval $[r_k^*, r_k^* + R_k^{ub})$ from arrival time to response time of J_k^*. Also, the upper bound of response time of τ_k, R_k^{ub}, is derived from I_k^{ub} and the execution time of τ_k. And then, we can calculate the lower bound of laxity of τ_k, L_k^{lb} from the following equation (2).

$$L_k^{lb} = T_k - R_k^{ub} \tag{2}$$

The scedulability of an algorithm is analyzed using L_k^{lb}.

In here, we have lemma 2 as for I_k^{ub}.

Lemma 2: $I_k^i(a, b)$ is always smaller than $W_i(a, b)$.

Therefore, the upper bound of the interference of τ_k, I_k^{ub} can be calculated by calculating the upper bound of the workload of τ_i, W_i^{ub}.

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, τ_k is interfered with not only higher static priority tasks but also lower static priority tasks.If τ_k is higher static priority than τ_i, the workload of τ_i over an interval $[r_k^*, r_k^* + R_k^{ub})$ is represented in Fig.8.

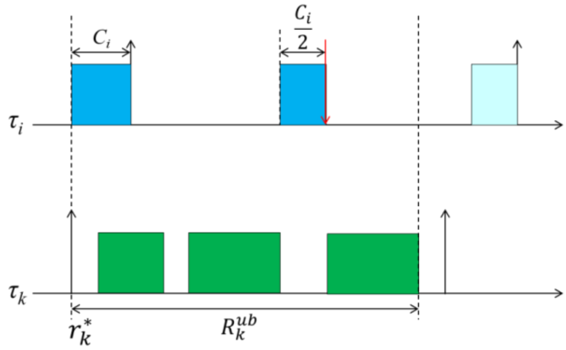

Fig.8. The workload on RMZLPD ($i>k$)

Thus, the upper bound of the workload, $W_i^{ub}(r_k^*, r_k^* + R_k^{ub}) = W_i^{ub}(R_k^{ub})$ is calculated as following equation (3).

$$W_i^{ub}(R_k^{ub}) = C_i + \min\left(\frac{C_i}{2}, \max\left(0, R_k^{ub} - \frac{T_i}{2} - \frac{C_i}{2}\right)\right) \quad if \quad (i > k) \tag{3}$$

If τ_k is lower static priority than τ_i, the workload of τ_i over an interval $[r_k^*, r_k^* + R_k^{ub})$ is represented in Fig.9.

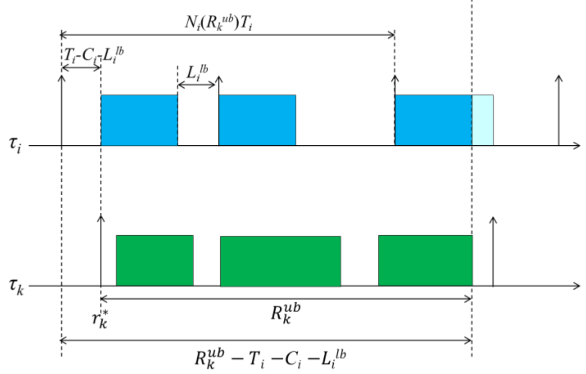

Fig.9. The workload on RMZLPD ($i<k$)

Thus, the upper bound of the workload, $W_i^{ub}(r_k^*, r_k^* + R_k^{ub}) = W_i^{ub}(R_k^{ub})$ is calculated as following equation (4).

$$W_i^{ub}(R_k^{ub}) = n_i(R_k^{ub})C_i + \min\left(C_i, R_k^{ub} + T_i - C_i - L_i^{lb} - n_i(R_k^{ub})T_i\right) \quad if \quad (i < k) \tag{4}$$

International Journal of Engineering Innovation & Research
Volume 2, Issue 6, ISSN: 2277 – 5668

where $n_i(R_k^{ub})$ is the maximum number of jobs of task τ_i that contribute all of their execution time in the interval and it is calculated as following equation (5).

$$n_i(R_k^{ub}) = \left\lfloor \frac{R_k^{ub} + T_i - C_i - L_i^{lb}}{T_i} \right\rfloor \tag{5}$$

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

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

$$R_k^{ub} \leftarrow C_k + \left\lfloor \frac{1}{m} \sum_{i \neq k} \hat{I}_k^i (R_k^{ub}) \right\rfloor \tag{6}$$

Where, $\hat{I}_k^i(R_k^{ub})$ is calculated as following equation (7).

$$\hat{I}_k^i(R_k^{ub}) = \min(W_i^{ub}(R_k^{ub}), R_k^{ub} - C_k + 1) \tag{7}$$

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 τ_k, and it holds strictly (<) for at least one of them:

$$L_k^{lb} \leq 0 \tag{8}$$

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

$$Schedulability = \frac{schedulable\ \ task\ \ set}{submitted\ \ all\ task\ set} \tag{9}$$

The proposed RMZLPD has two priority promotion chances. Therefore, RMZLPD is more interfered with static lower priority tasks than RMZL and LP-RMZL.

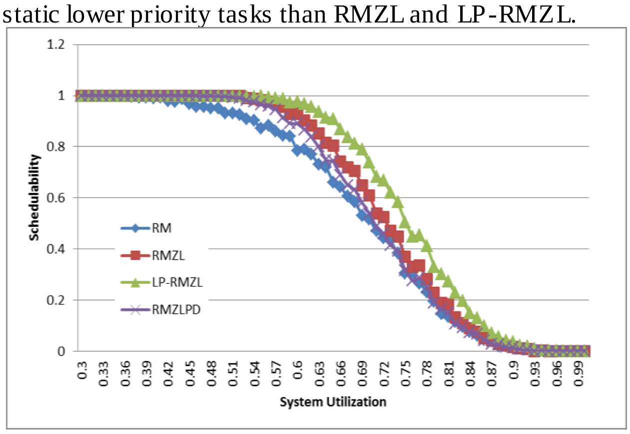

Fig.10. Schedulability (M=4)

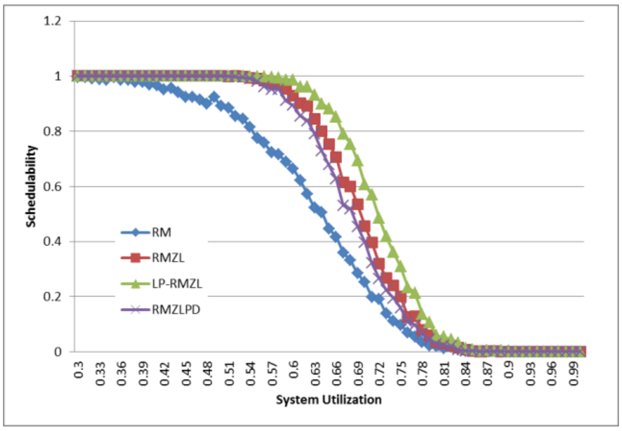

Fig.11. Schedulability (M=8)

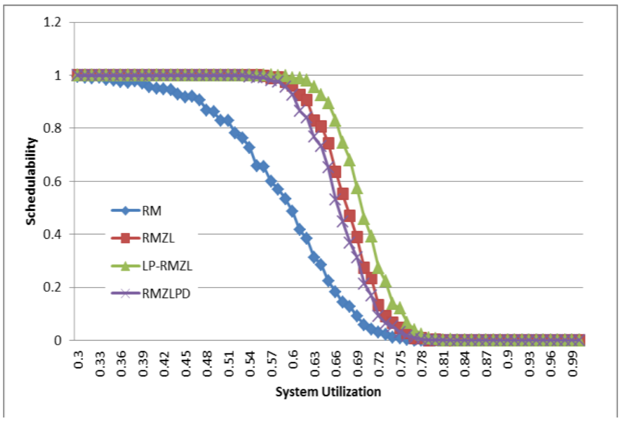

Fig.12. Schedulability (M=16)

B. Success Ratio

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

$$Success\ Ratio = \frac{successful\ \ task\ \ set}{submitted\ \ all\ task\ set} \tag{10}$$

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

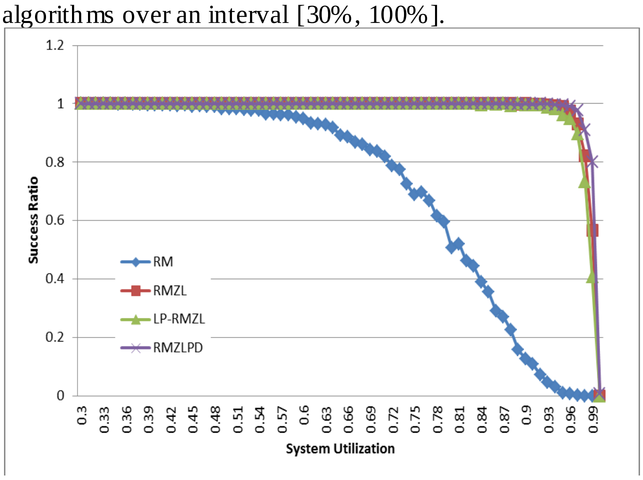

Fig.13. Success Ratio (M=4)

International Journal of Engineering Innovation & Research
Volume 2, Issue 6, ISSN: 2277 – 5668

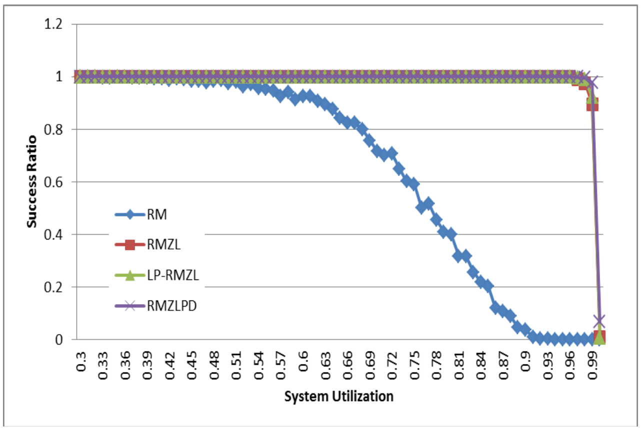

Fig.14. Success Ratio (M=8)

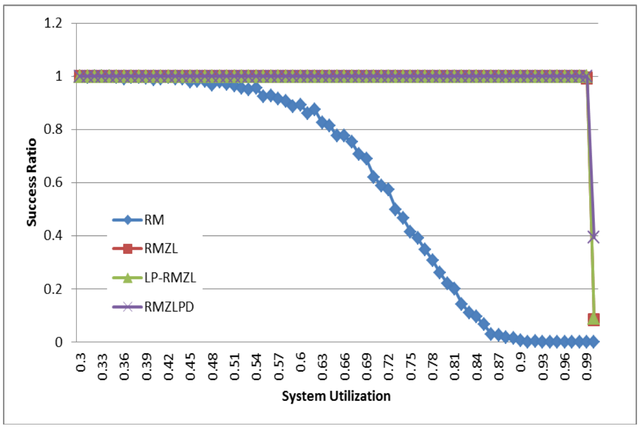

Fig.15. Success Ratio (M=16)

C. Preemption

Fig.16, 17, 18show the result of simulation about preemptions, when 1,000task set (system utilization 0.3 ~ 1.0) are submitted on4 processors, 8 processors and 16 processors. The number of preemptions is calculated as following equation (11).

$$Number\ of\ Preemption\ =\ \frac{toal\ number\ \ of\ preemptions}{submitted\ \ all\ task\ \ set} \tag{11}$$

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.

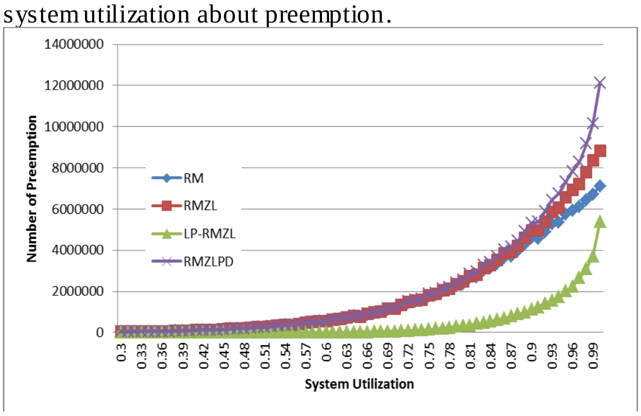

Fig.16. Preemption (M=4)

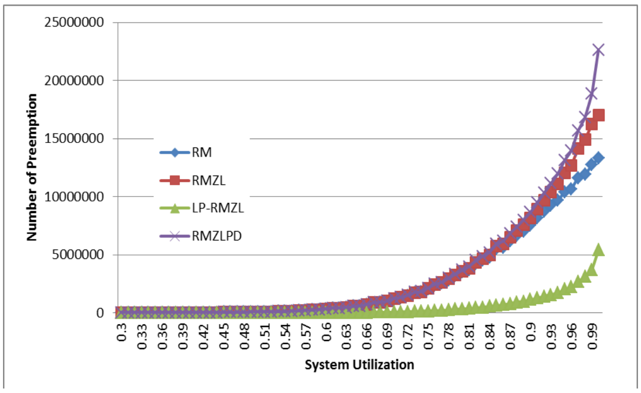

Fig.17. Preemption (M=8)

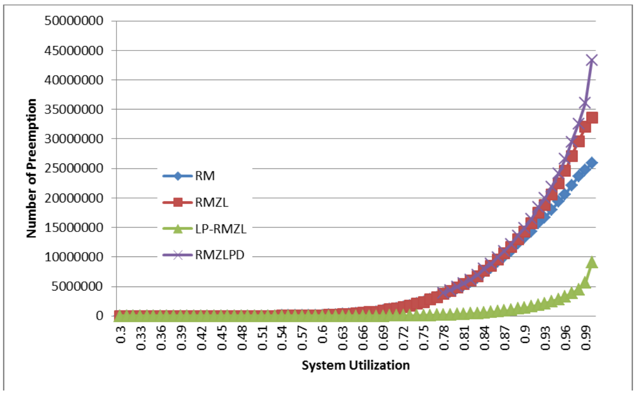

Fig.18. Preemption (M=16)

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 Number24500046.

References

[1] Devi, U.C. and J. H. Anderson, “Tardiness Bounds under Global EDF Scheduling on aMultiprocessor,”*Proceedings of the 26th IEEE International Real-Time Systems Symposium*, 2005, pp. 330-341.

[2] Guan, N. and W. Yi, “Fixed-Priority Multiprocessor Scheduling: Critical Instant, Response Time and Utilization Bound,”*IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum*, 2012, pp. 2470 – 2473.

[3] Dhall, S. K. and C. L. Liu, “On a real-time scheduling problem,”*Operations Research*, Vol. 26, No. 1, 1978, pp. 127-140.

[4] Takeda, A., S. Kato, and N. Yamasaki, “Real-time scheduling based on rate monotonic for multiprocessors,”*IPSJ CPSY*, Vol. 2, No. 1,2009,pp. 64–74.
[5] Davis, R. I. and A. Burns, “Fpzl schedulability analysis,”*17th IEEE Real Time and Embedded Technology and Applications Symposium*, 2011, pp. 245-256.
[6] Nishigaki, K., M. Yoo, and T. Yokoyama, “A Proposal of Real-Time Scheduling Algorithm based on RM and Schedulability Analysis,”*IEICED*, Vol. J95, No. 6, 2012, pp. 1347-1355.
[7] Bertogna, M. and M. Cirinei,“Response-time analysis for globally scheduled symmetric multiprocessorplatforms,”*28th IEEE International Real-Time Systems Symposium*, 2007, pp. 149-160.
[8] Bertogna, M., M. Cirinei and G. Lipari, “Improved schedulability analysis of edf on multiprocessor platforms,”*Proc. 17th Euromicro Conference on Real Time Systems*, 2005, pp. 209-218.

AUTHOR’S PROFILE

Ken Yanai

received B. E. form 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 PohangUniversity of science & Technology, Korea in 1996and Ph.D. degree from Graduate School ofInformation, Production & Systems, WasedaUniversity, Japan in 2006. She joined Tokyo CityUniversity as Associate Professor in 2006. Her research field is Real-TimeSystem, 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 fromTohoku University in 1981, 1983, and 2002respectively. He joined Hitachi, Ltd. in 1983. HejoinedMusashi Institute of Technology in 2004. He isnow a professor of Tokyo City University. Hisresearch interest includes embedded systems, distributed systems and software engineering. He is amember of IEEE, ACM, IPSJ.
Email: yokoyama@cs.tcu.ac.jp