Execution Right Delegation Scheduling Algorithm Using Slack

SUZUKI Taka'HAL' (1310753)

School of Information Science,
Japan Advanced Institute of Science and Technology

08/06/2016

Keyword : Real-time scheduling, Rate monotonic, Deadline monotonic, Priority exchange

1 Introduction

In fixed-priority scheduling algorithms widely used in development of real-time systems, tasks with shorter periods have higher priorities. In contrast, ones with longer periods are likely to suffer from increased response times and jitters due to their lower priorities. This study aims at shortening response times of tasks which have relatively long periods but are important. We propose Execution Right Delegation (**ERD**) method, which introduces a high-priority server for particular tasks to be scheduled preferentially, and Slack Collection (**SC**) method, which provides particular tasks with slack time left by early completion of other tasks. These methods keep schedulability.

2 Related Work

Multiple papers have been published related to scheduling algorithms. Rate Monotonic (**RM**) [1] method is a fundamental algorithm of fixed-priority scheduling. In **RM**, tasks has period and execution time. The rule of it is a simple one that assigns priorities to tasks according to their period. In

this algorithm, tasks with shorter periods have higher priorities. Deadline of tasks is equal to their period.

Deadline Monotonic (**DM**) method [2] relaxes "period equals deadline" constraint of a static priority scheduling scheme. This algorithm provides the solution that assigns higher priority to tasks even if their period is longer.

In aperiodic task scheduling algorithms, Priority Execution (**PE**) method [3] is proposed. **PE** uses a high priority server to serve aperiodic requests. If aperiodic requests are pending and the server is runnable, then the requests are served using server's execution time; otherwise the priority of server and that of an active task are exchanged, so the active task executes at the priority level of the server. If aperiodic requests occur while an active task executes after priority exchange, the aperiodic requests are served by using server's capacity. The aim of this algorithm is to shorten the response time of aperiodic requests.

Dual Priority (**DM**) method [4] is another approach to shorten the response time of aperiodic requests. In this model, periodic tasks have two (different) priorities. In normal mode, periodic tasks are executed at lower priority. Aperiodic requests have middle level priority. If deadline of periodic task is near, then the higher priority becomes effective.

Estimation of feasibility is an important area of real-time scheduling algorithm. The feasibility analysis of RM was performed by Liu, et al. This study calculates processor utilization. This approach works in polynomial time but result is pessimistic.

Audsley et al., proposed response Time Analysis (RTA), which calculates the longest response time of a periodic task. This method gives the necessary and sufficient condition of feasibility in static priority scheduling.

3 Proposed Method

In this paper, we propose **ERD** method and **SC** method to shorten response time of specific task. First, we propose important theorems which show the feasibility.

If response time R_p of task τ_p is shorter than T_h, which is the period of higher priority task τ_h, then priority of τ_p is able to be higher than τ_h (Theorem 6).

The duration that any tasks' job was not executed is called idle-time. If there is an idle-time in period T_h of task τ_h, then a new task (whose execution time is idle-time and period is T_h) is able to be added to that task set (Lemma 10).

Using above-described theorem/lemma, two important techniques/strategies

are proposed. First, priority of a specific task is changeable, while keeping the schedulability of the whole task set. Second, execution of task is dividable. In **ERD** method, we introduce a virtual server to obtain these two techniques.

SC method is another approach to shorten response time and jitter. This method collects slack time, which is obtained from early completion of tasks, and gives it to a specific task.

4 Evaluation

We evaluate proposed methods by comparison with **RM** and **DM**. In the evaluation, random numbers from probability distribution are used to generate execution time and period of task sets.

From the result, response time of proposed methods is equal to/shorter than **DM** in a number of cases. For the task set whose maximum utilization of each task is higher, **ERD** and **SC** method win. By using divide-executable server, even if **DM** cannot set the deadline shorter, **ERD** and **SC** are able to decrease response time. On the other hand, some results show **DM** is effective than proposed method. This is because period of the virtual server and that of a specific task are different. In this case, there is the possibility of losing the server capacity.

Regarding the using of slack, there were no different between **SC**and **DM**. In our task set, tasks in **DM**are able to increase priority. As a result, if slack time is gaven, specific (important) task is runnable in many cases. On the other hand, the strategy to devide period and capacity of the virtual server helps the result in some cases. This is in anticipation of the extreme early termination. If the trend of slack generation is predictable/observable in real sysmtems, to shorten responce time is possible by adjust a virtual server.

5 Conclusion

In this study, we proposed **ERD** and **SC** methods for shortening response times of tasks which have relatively long periods but are important. And several important theorems which show the feasibility are proposed. From the results of evaluation with the random task sets, ability of proposed methods is equal to or greater than **DM** in a number of cases.

References

[1] C. L. Liu and J. W. Layland, Scheduling Algorithms for Multiprogramming in a Hard- Real-Time Environment *Journal of the ACM 20(1): 40-61.*, 1973

[2] N. C. Audsley, A. Burns, M. F. Richardson, and A. J. Wellings, Hard real-time scheduling: The deadline monotonic approach. *In IEEE Workshop on Real-Time Operating Systems*, 1992.

[3] J. P. Lehoczky, L. Sha, and J. K. Strosnider, Enhanced aperiodic responsiveness in hard real-time environments. *In Proceedings of the IEEE Real-Time Systems Symposium*, December 1987

[4] R. Davis, A. Wellings, Dual Priority Scheduling. *In Proceedings of the 16th IEEE RTSS*, 1995.