Improvement of Adaptive EDF

Kiyofumi Tanaka
School of Information Science
Japan Advanced Institute of Science and Technology (JAIST)
Email: kiyofumi@jaist.ac.jp

***Abstract*—Rate monotonic (RM) and earliest deadline first (EDF) are representative scheduling algorithms for real-time periodic tasks. RM has a merit that tasks with high-priority (short-period) have small jitters and short response times. However, it is impossible that processor utilization reaches 100% while maintaining schedulability or that tasks are given importance independent of their periods. On the other hand, EDF can utilize processors by 100% while it cannot give tasks fixed priorities or importance. Therefore, it is difficult to keep jitters or response times of particular tasks short. The author proposed an EDF-based method, Adaptive EDF, of shortening response times of a task that is important for the system by introducing predictive execution times (PET) and dividing execution of the task into two sub instances based on PET [1]. This paper shows improved techniques which can further reduce response times by extending the adaptive EDF.**

I. Introduction

Rate monotonic (RM) and earliest deadline first (EDF) are representative scheduling algorithms for real-time periodic tasks [2]. RM is one of fixed-priority algorithms and executes tasks with shorter periods preferentially. Although RM has a merit that tasks with high-priority (short-period) have small jitters and short response times, processor utilization cannot reaches 100% while maintaining schedulability. EDF is one of dynamic-priority algorithms and executes tasks with earlier absolute deadlines preferentially. Under EDF, processor utilization of 100% can be achieved with schedulability, while it is impossible to give tasks fixed priorities, which means that it is difficult to keep jitters or response times of particular tasks short. In the near future, systems consisting of different types or criticality levels of tasks would be common [3], [4]. Therefore, this study explores algorithms based on EDF and aims to reduce response times of a particular periodic task that is important for the system, while not affecting schedulability.

In the past research, the author proposed a method, "Adaptive EDF", of shortening response times of a particular periodic task [1]. This method introduced predictive execution times (PET) in addition to Worst-Case Execution Times (WCET), divides execution of a particular task (most important for the system) into two sub instances, and gives the first instance an earlier deadline, which has a possibility of shortening the response time when the actual execution time is shorter than the WCET. This paper makes two improvements, Residual Bandwidth Appropriation and Incremental Deadline Update, in the adaptive EDF to further reduce response times.

II. The Adaptive Earliest Deadline First Algorithm

This section summarizes the adaptive EDF proposed in [1].

The paper was presented at APRES 2014.

A. Worst-Case vs. Actual Execution Times

In real-time scheduling and its schedulability analysis, task execution is supposed to spend its WCET. However, in most cases, actual execution time (AET) is shorter than WCET. Since exact WCET is difficult to obtain [5], [6] and therefore pessimistically estimated, the gap between AET and WCET tends to be large. By exploiting this gap, the adaptive EDF reduces response times. That is, it assumes PET instead of WCET and sets (tentative) short deadline based on the PET.

In [1], the following formula was used as predictive execution times.

$$\begin{aligned} C_{i_k}^{PET} &= \alpha \times C_{i_{k-1}}^{PET} + (1-\alpha) \times C_{i_{k-1}}^{AET} \\ C_{i_0}^{PET} &= C_i^{WCET} \end{aligned} \quad (1)$$

Here, $C_{i_k}^{PET}$ means PET for the kth ($k = 0, 1, \ldots$) instance of a periodic task, τ_i. $C_{i_{k-1}}^{AET}$ is the execution time actually spent for the previous instance of τ_i. The initial value of PET, $C_{i_0}^{PET}$, is equal to WCET of the task (C_i^{WCET}). This formula calculates as the predictive execution time an weighted average of the previous PET and the previous AET with the weighting coefficient α ($0 \leq \alpha \leq 1$).

B. Definition of the Adaptive EDF

In RM, it is possible to keep response times of important tasks short by giving short periods to them. However, it is impossible to set up the importance of tasks independent of the periods. On the other hand, EDF uses dynamic priorities and therefore cannot reflect static importance in the schedules. The adaptive EDF introduces fixed importance of tasks independent of the tasks' periods.

The adaptive EDF targets a set of periodic tasks. Each task has a relative deadline that is equal to the period. An execution of an important periodic task is divided into two sub instances. That is, if τ_i is regarded as the most important task, the kth execution instance, J_{i_k}, of τ_i is divided into two parts, $J_{i_k}^{PET}$ and $J_{i_k}^{REST}$. $J_{i_k}^{PET}$ corresponds to the execution from the beginning of J_{i_k} to the predicted finishing time. $J_{i_k}^{REST}$ corresponds to the execution after the predicted finishing time. Let the WCET of τ_i be C_i^{WCET}, the PET of J_{i_k} be $C_{i_k}^{PET}$, and the execution time of $J_{i_k}^{REST}$ be $C_{i_k}^{REST}$. Depending on the actual execution time, $C_{i_k}^{REST}$ can be between zero at a minimum and $C_i^{WCET} - C_{i_k}^{PET}$ at a maximum. In the following, the worst-case scenario, $C_{i_k}^{REST} = C_i^{WCET} - C_{i_k}^{PET}$, is assumed.

$J_{i_k}^{PET}$ and $J_{i_k}^{REST}$ are given the absolute deadlines, $d_{i_k}^{PET}$ and $d_{i_k}^{REST}$, respectively, that are calculated by the following formulas. (In the formulas, T_i is a period of τ_i and U_i is

the processor utilization by τ_i which is calculated by $U_i = C_i^{WCET}/T_i$.)

$$d_{i_k}^{PET} = k \times T_i + \frac{C_{i_k}^{PET}}{U_i} \quad (2)$$

$$d_{i_k}^{REST} = d_{i_k}^{PET} + \frac{C_{i_k}^{REST}}{U_i} = (k+1) \times T_i = d_{i_k}^{WCET} \quad (3)$$

After the deadlines are given, the two sub instances are scheduled based on EDF. $J_{i_k}^{PET}$ is executed before $J_{i_k}^{REST}$, since $d_{i_k}^{PET}$ is earlier than $d_{i_k}^{REST}$. If the execution of $J_{i_k}^{PET}$ finishes at or before the PET, $J_{i_k}^{REST}$ does not exist. ($C_{i_k}^{REST}$ is zero and therefore $J_{i_k}^{REST}$ is not executed.) In such a case, the response time can be expected to be shorter since the task might be scheduled earlier due to the earlier deadline ($d_{i_k}^{PET} \leq d_{i_k}^{WCET}$).

An example of the division of a task into two sub instances is shown in Fig. 1. The horizontal axis is time in ticks. A task, τ_i, has $T_i = 8$ and $C_i^{WCET} = 2$. The utilization by τ_i is 2/8 = 0.25. The execution of an instance of τ_i, J_{i_5}, is shown in the figure. $C_{i_5}^{PET}$ is assumed to be 1. Therefore, $d_{i_5}^{PET}$ becomes $40+1/0.25 = 44$. $J_{i_5}^{PET}$ starts at tick 41. When the execution spends $C_{i_5}^{PET}$, it is suspended if the task execution has not finished yet. Then, the remaining execution restarts at tick 46 in this example.

C. Schedulability

Schedulability of the adaptive EDF can be discussed as follows. By letting U_i be the processor utilization (based on the WCET) by the most important periodic task τ_i and dividing an instance of τ_i into two (aperiodic) sub instances, the adaptive EDF becomes the same as Total Bandwidth Server (TBS) [7] with the server bandwidth of U_i. (TBS is a scheduling algorithm for mixed task sets of periodic and aperiodic tasks. In TBS, each aperiodic instance is given deadline according to its WCET and the server bandwidth. After the deadline assignment, the whole task set is scheduled by EDF.)

The two sub instances made of the periodic instance are executed as aperiodic requests. The deadline assignment for the two sub instances leads to being the same as that in TBS. That is, when execution times of the two sub instances are $C_{i_k}^{PET}$ and $C_{i_k}^{REST}$, the deadlines given by TBS are equal to the formula (2) and (3). Therefore, the schedulability test for TBS in the literature [7] can be directly applied to the adaptive EDF, that is, if and only if the total processor utilization is equal to or smaller than 1 ($U \leq 1$), the whole task set is schedulable by EDF.

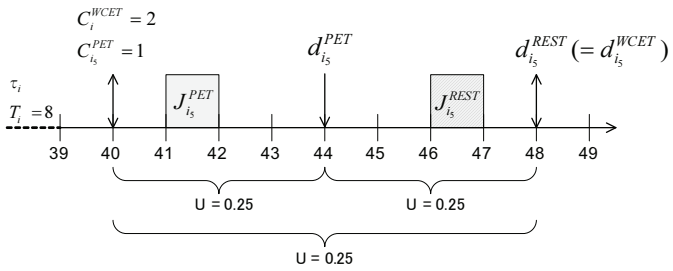

Fig. 1. Two sub instances from a task.

III. Two Improvements

To improve the adaptive EDF, two techniques, residual bandwidth appropriation (RBA) and incremental deadline update (IDU), are proposed in this section.

A. Residual Bandwidth Appropriation (RBA)

The basic idea of the adaptive EDF is that an important task is executed by TBS. Here, when a deadline of a sub instance is calculated by using the formula (2), the utilization of the task, U_i, is used as the bandwidth given to the sub instance. However, considering the characteristic of TBS, schedulability is guaranteed with $U_p + U_s \leq 1$. (U_p is utilization by all periodic tasks and U_s is the server bandwidth.) Therefore, $1 - (U_p - U_i)$ can be used as the corresponding bandwidth instead of U_i. This means that, when U_p is smaller than 1, the residual bandwidth, $1 - U_p$, can be used for the execution of the target important task.

Consequently, the formula (2) is changed into the following formula (4). If $U_p \leq 1$, $1 - (U_p - U_i) \geq U_i$. Therefore, the calculated deadline would be earlier and the execution of the target task can be expected to be advanced.

$$d_{i_k}^{PET} = k \times T_i + \frac{C_{i_k}^{PET}}{1 - (U_p - U_i)} \quad (4)$$

This technique is residual bandwidth appropriation (RBA). Fig. 2 shows an example of the RBA. In the figure, there are two periodic tasks, τ_1 and τ_2. τ_1 has $T_1 = 4$ and $C_1^{WCET} = C_1^{AET} = 2$. τ_2 has $T_2 = 6$ and $C_2^{WCET} = C_2^{AET} = 1$. In this example, τ_2 is supposed to be an important task. In the original EDF, the first instance of τ_2 has the response time of three. On the other hand, in the RBA, the bandwidth that can be used by τ_2 becomes $1 - 2/4 = 0.5$, and by the formula (4), earlier deadlines depicted as the dashed arrows in the figure are given. Consequently, the same instance has the response time of one.

B. Incremental Deadline Update (IDU)

In the adaptive EDF, when PET is overestimated, the deadline would not be short enough, and the response time would not be short. In contrast, when PET is underestimated, the remaining execution part after the predicted execution time is scheduled according to the deadline equal to the period timing, which would not lead to short response time. The

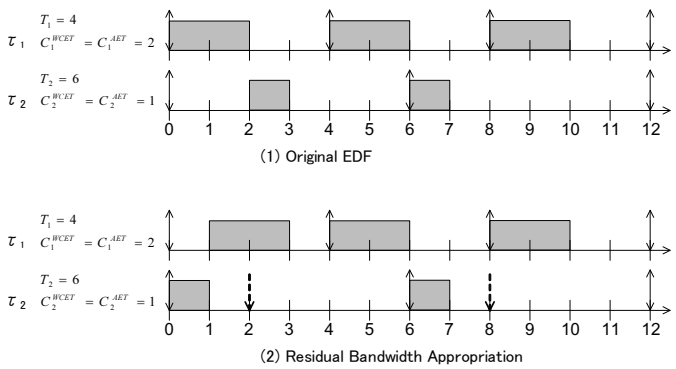

Fig. 2. An example of residual bandwidth appropriation.

accuracy of PET determines the effectiveness in reducing the response times.

The problems above can be solved as follows. The minimum value, one tick, is used as the first PET when a periodic instance is invoked. When the PET has expired, the (original) adaptive EDF resets the deadline to the period timing. On the other hand, in the technique, incremental deadline update (IDU), the deadline is updated one by one. This means, at a time, the remaining execution time is predicted as one tick. Repeatedly, whenever the PET (one tick) has expired, the same update to the deadline is done.

This idea is formalized as follows. An instance, J_{i_k}, is divided into one or more sub instances, $J^1_{i_k}, J^2_{i_k}, J^3_{i_k}, ...$ The first sub instance, $J^1_{i_k}$, corresponds to the execution from the beginning of J_{i_k} to one tick later. $J^j_{i_k}$ corresponds to the execution from $j-1$ ticks to j ticks. If J_{i_k} finishes in j ticks, $J^{j+1}_{i_k}$ and the later sub instances do not exist. In the kth period of τ_i, $J^j_{i_k}$ is given the following absolute deadline.

$$d^1_{i_k} = k \times T_i + \frac{1}{U_s} \quad (5)$$

$$d^j_{i_k} = d^{j-1}_{i_k} + \frac{1}{U_s} \quad (j > 1) \quad (6)$$

Fig. 3 shows schedules by (1) original EDF, (2) adaptive EDF with PET overestimated, (3) adaptive EDF with PET underestimated, and (4) IDU. τ_3 is an important (target) task and has $C_3^{WCET} = 4$ and $C_3^{AET} = 2$. In the original EDF, the response time is six. In the adaptive EDF with overestimated PET(=3), the tentative deadline is set to tick 9 and the response time becomes six. In the adaptive EDF with underestimated PET(=1), the tentative deadline is first set to tick 3, then reset to tick 12, and the response time is finally six. In the IDU, the first deadline is set to tick 3 by using the first PET of one. Then the second deadline becomes tick 6 by using the second PET of one for the remaining execution. Consequently, the response time is four, which is shorter than all the other results.

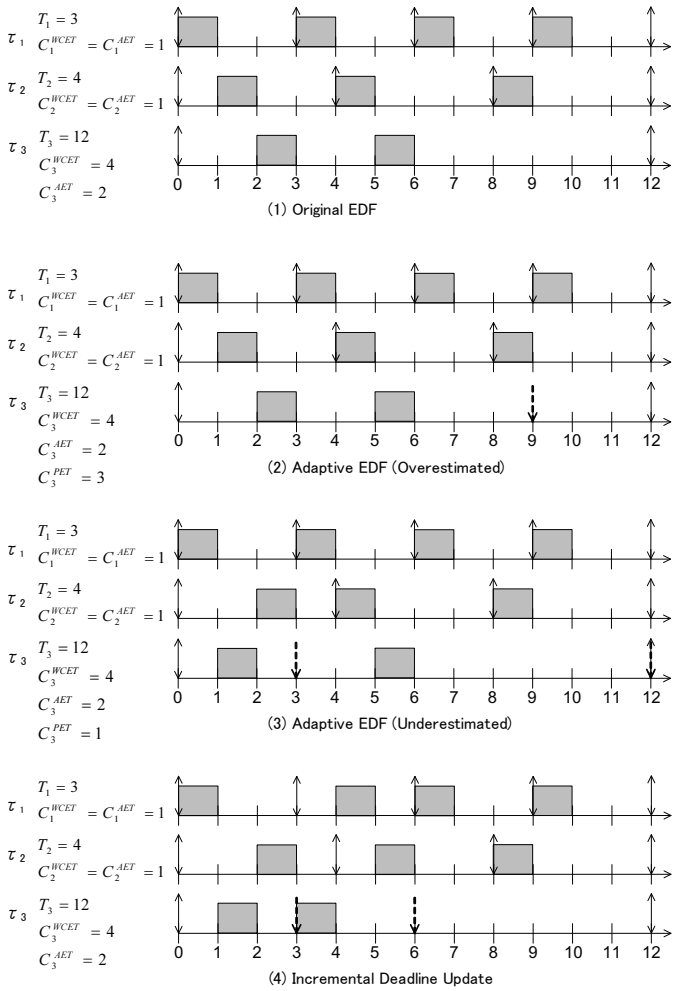

Fig. 3. Schedule examples.

IV. Evaluation

In this section, effects of the two improvement techniques are shown by comparing them with the original EDF, RM, deadline monotonic (DM)[8], and the original adaptive EDF in the simulations.

DM is a method that weakens the constraint in RM, "deadline is equal to its period." By exploiting this quality, similar to the RBA, the extra bandwidth can be given to an important task and relative deadline shorter than its period can be assigned. That is, the deadline of an important task is given as follows.

$$d_{i_k} = k \times T_i + \frac{C_i^{WCET}}{U_{ub} - (U_p - U_i)} \quad (7)$$

U_{ub} is the upper bound for RM and set to 0.9 in the simulation. (Although this is optimistic, no deadline misses occurred in the simulations.) Other tasks are supposed to have deadline equal to their period.

In the simulations, task sets consist of periodic tasks with the total CPU utilization (U_p) from 70% to 100% at intervals of 5%. U_p is based on WCETs of all tasks. For each periodic task, its period is decided by uniform distribution between 1 and 100 ticks and its WCET is decided by uniform distribution between 1/10 and 1/3 of the period. Actual execution times (AETs) of instances of each task is decided by uniform distribution between 1/3 and 1/1 of the WCET. (Instances from the same task have different execution times.) The observation period is 100,000 ticks.

For each U_p, twenty periodic task sets were simulated and the average values of response times are shown. For the original adaptive EDF, the weighting coefficient for PET calculation (α in the section II-A) is 0.5. In the simulations for all task sets, no deadline misses occurred even in RM and DM with $U_p = 0.9$, since actual processor utilization based on AETs was lower than U_p that was based on WCETs.

A. Results

Fig. 4 shows average response times of an important task when a task with the shortest period is regarded as the important task. "AEDF" is the original adaptive EDF. "AEDF+R" is the adaptive EDF with RBA. "AEDF+I" is the adaptive EDF with IDU. "AEDF+RI" is the adaptive EDF with both RBA and IDU. The horizontal axis indicates U_p and the vertical axis indicates average response times of the target task

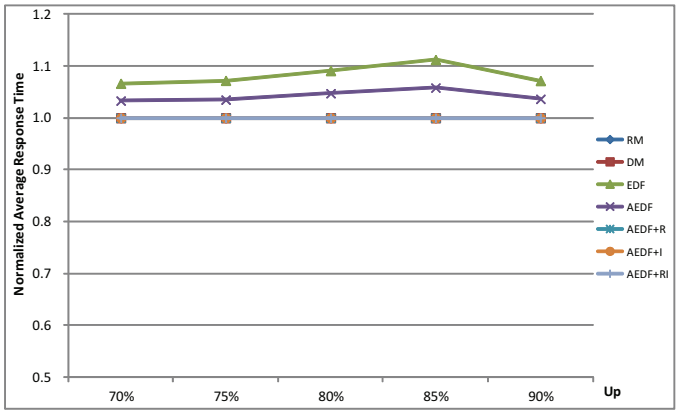

Fig. 4. Results – Target is the task with shortest period.

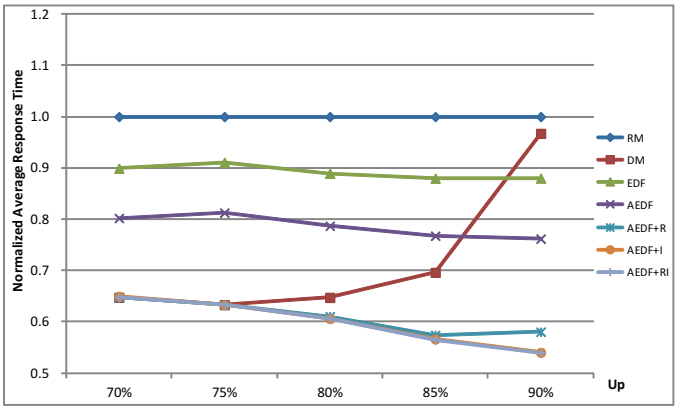

Fig. 6. Results – Target is the task with longest period.

normalized to the results of RM. In the figure, RM and DM exhibited shortest response times. This is natural since they always give the highest priority to the important task (with the shortest period). EDF exhibited longer response times since it does not consider the importance of tasks. Compared to EDF, AEDF alleviates the performance degradation. The results of AEDF+R and AEDF+I are almost the same as RM, which means both techniques worked well. (The response times were longer than RM by, at a maximum, 0.0056% in AEDF+R and 0.0061% in AEDF+I.) Finally, AEDF+RI had the same results as RM. (RM, DM, AEDF+R, AEDF+I, and AEDF+RI overlap with each other in the figure.)

Fig. 5 shows the results where a task with a medium-length period is regarded as the important one. RM and EDF were almost the same. DM outperformed RM when U_p is low, but it was almost the same as RM when U_p is 90%. This is because the residual bandwidth exploited from U_{ub} reaches zero when U_p is 90%. Although AEDF improved the response times, the amount of improvement was not high. AEDF+R and AEDF+I improved much more than AEDF. The combination of the two techniques exhibited the best results, improvement by about 19.1% against AEDF when U_p is 90%.

Fig. 6 is when a task with the longest period is the important task. Except that EDF outperformed RM, a trend similar to Fig.5 appeared, although the improvement rates against RM are much higher. Compared to AEDF, AEDF+RI improved by about 29.2% when U_p is 90%. From the results, it is shown that the proposed two improvement techniques can shorten response times of the important task especially when a task with a long period is important.

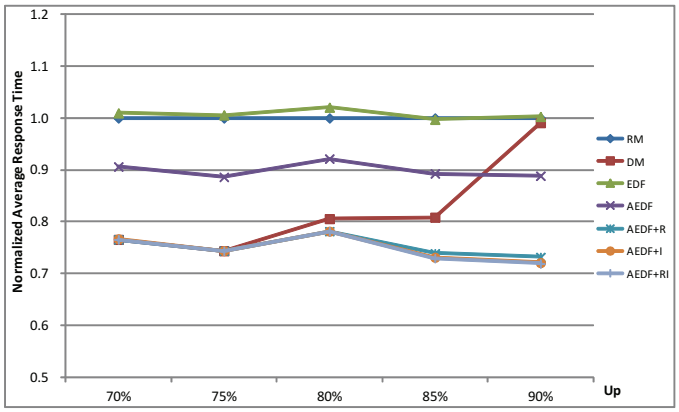

Fig. 5. Results – Target is the task with medium period.

V. Conclusion

In this paper, two improvement techniques, residual bandwidth appropriation (RBA) and incremental deadline update (IDU), are proposed. RBA achieves adaptation to extra processor utilization and IDU achieves adaptation to varying execution times of the same task. These are applied to the adaptive EDF that shortens response times of a task with importance independent of its period. In the simulation-based evaluation, with the combination of the two techniques, the average response times of an important task were reduced by 29.2% at a maximum.

The evaluation in this paper used task sets that were generated using probability distribution. To reflect actual situations where different instances of the same task spend different execution times, evaluation with actual program codes is desired. In the future, evaluation with actual codes including scheduling overheads by operating systems should be performed.

References

[1] K. Tanaka, "Adaptive EDF: Using Predictive Execution Time," 5th Workshop on Adaptive and Reconfigurable Embedded Systems (APRES), April 2013.

[2] C. L. Liu and J. W. Layland, "Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment," Journal of the Association for Computing Machinery, Vol. 20, No. 1, pp. 46–61, January 1973.

[3] D. de Niz, K. Lakshmanan, and R.Rajkumar, "On the Scheduling of Mixed-Criticality Real-Time Task Sets," Proc. of IEEE Real-Time Systems Symposium, pp. 291–300, December, 2009.

[4] S. Baruah, H. Li, and L.Stougie, "Towards the Design of Certifiable Mixed-Criticality Systems," Proc. of IEEE Real-Time and Embedded Technology and Application Symposium, pp. 13–22, Apr, 2010.

[5] T. Lundqvist and P. Stenström, "Timing Anomalies in Dynamically Scheduled Microprocessors," Proc. of IEEE Real-Time Systems Symposium, pp. 12–21, December 1999.

[6] R. Wilhelm, et al., "The Worst-Case Execution Time Problem – Overview of Methods and Survey of Tools," ACM Trans. on Embedded Computing Systems, Vol. 7, No. 3, Article No. 36, April 2008.

[7] M. Spuri and G. C. Buttazzo, "Efficient Aperiodic Service under Earliest Deadline First Scheduling," Proc. of IEEE Real-Time Systems Symposium, pp. 2–11, December 1994.

[8] J. Leung and J. Whitehead, "On the Complexity of Fixed-Priority Scheduling of Periodic Real-Time Tasks," Performance Evaluation, Vol. 2, No. 4, pp.237–250, December 1982.