Feedback-Controlled Server for Scheduling Aperiodic Tasks

Shinpei Kato and Nobuyuki Yamasaki

***Abstract*—This paper proposes a scheduling scheme using feedback control to reduce the response time of aperiodic tasks with soft real-time constraints. We design an algorithm based on the proposed scheduling scheme and Total Bandwidth Server (TBS) that is a conventional server technique for scheduling aperiodic tasks. We then describe the feedback controller of the algorithm and give the control parameter tuning methods. The simulation study demonstrates that the algorithm can reduce the mean response time up to 26% compared to TBS in exchange for slight deadline misses.**

***Keywords*—Real-Time Systems, Aperiodic Task Scheduling, Feedback-Control Scheduling, Total Bandwidth Server**

I. Introduction

Recent real-time systems are required to deal with not only periodic procedures but also aperiodic requests with real-time constrants. In such systems, periodic tasks need to be completed before their deadline while aperiodic tasks need to be completed as soon as possible. Several work have been done so as to meet those requirements. We focus on dynamic priority systems in this paper, since a scheduling algorithm such as Earliest Deadline First (EDF) [6] can achieve the schedulable system utilization of 100%.

Lehoczky *et al.* proposed server techniques, Deferrable Server (DS) and Priority Exchange (PE) [5], which is executed as a periodic task to service aperiodic request. Sprunt *et al.* investigated other server techniques based on the above servers, Extended Priority Exchange (EPE) [11] and Sporadic Server (SS) [12]. Slack Stealer [4], [10], proposed by Lehoczky and Ramos-Thuel, is an optimal technique which improves response time substantially over the servers described above. Whereas Slack Stealer is optimal and its performance is excellent, it is widely known that it needs a huge memory space to store the slack available at each task arrival. Dynamic Slack Stealer [2] or Dual Priority Scheduling [3] overcomes this problem and is still optimal or nearly optimal, however, the runtime overhead is quite expensive.

All the techniques described above are basically used with Rate Monotonic (RM) [6] scheduling algorithm. The lowest bound of the CPU utilization in RM is only about 69% whereas dynamic priority scheduling such as Earliest Deadline First (EDF) [6] can achieve 100%. Spuri *et al.* proposed a number of aperiodic service techniques under dynamic priority scheduling [13], [14]. They first proposed Dynamic Sporadic Server (DSS) and Dynamic Priority Exchange (DPE) that are the dynamic versions of Deferrable Server (DS) [5] and Priority Exchange (PE) [5] for the fixed priority systems. They also proposed a simple and efficient algorithm, Total Bandwidth Server (TBS), which is more suitable for deadline-based scheduling. Moreover, they invented an optimal algorithm, Earliest Deadline as Late as possible (EDL) server, and a suboptimal algorithm, Improved Priority Exchange (IPE). EDL is an dynamic extension of Slack Stealer [4] for fixed priority systems. IPE is derived from DPE and is modified to make efficient use of idle times in EDL.

The performance of reasonable algorithms such as DSS, DPE and TBS drops dramatically in high-utilized systems. Meanwhile the sophisticated algorithms such as EDL and IPE perform very well even in high-utilized systems, however those algorithms are too complicated to implement into real environments and not practical in the computational overhead point of view. The trade-off between the performance and practicality stands because the above algorithms are designed for hard real-time systems. In fact, most of current embedded systems are constructed as soft real-time systems. We consider that the trade-off can be mitigated if some periodic jobs are allowed to miss their deadline.

We explore aperiodic task scheduling for soft real-time systems in this paper. We apply feedback control to reduce the response time of aperiodic tasks with soft real-time constraints. The main idea of our algorithm is to execute the aperiodic tasks for a certain interval immediately when they arrive at the system in order to reduce the response time. Although some periodic jobs would miss their deadline as a result, we can bound its frequency by feedback control. In other words, the response time can be reduced in exchange for slight deadline misses. This paper considers only dynamic-priority systems.

The rest of this paper is organized as follows. The next section explains the system model assumed in this paper. Section 3 describes our algorithm FC-TBS. Section 4 evaluates the performance of FC-TBS compared to the existing algorithms. In section 5, we conclude our work and give an insight of our future work.

II. System Model

We assume the following system model and notations in this paper. There are n periodic tasks $(\tau_1, \tau_2, ..., \tau_n)$ with soft real-time constraint in the system. Each task τ_i is denoted as a topple of (T_i, C_i). T_i is a period and C_i is a worst-case execution time. The deadline of each task is equal to the beginning of its next period. All the periodic tasks can be preempted at any time. The kth job of τ_i is denoted by $\tau_{i,k}$. The release time and the deadline of $\tau_{i,k}$ is described by $r_{i,k}$

Shinpei Kato and Nobuyuki Yamasaki are with the School of Science for Open Environmental Systems, Keio University, Yokohama, Japan, email: {shinpei, yamasaki}@ny.ics.keio.ac.jp

and $d_{i,k} = r_{i,k} + T_i$ respectively. Also the start time and the finish time of $\tau_{i,k}$ is described by $s_{i,k}$ and $f_{i,k}$ respectively. The total utilization of all the periodic tasks, i.e. the periodic load, is denoted by $U_p = \sum_{i=1}^{n} \frac{C_i}{T_i}$.

Each aperiodic job α_k is denoted as a topple (r_k, E_k). r_k is an arrival time and E_k is a worst-case execution time. The finish time of α_k is described by f_k. The response time of α_k is described by $R_k = f_k - r_k \geq E_k$. For all α_k, $r_k < r_{k+1}$ is satisfied. The aperiodic load is denoted by $U_a = \frac{\lambda}{\mu}$ where μ is the average service rate and λ is the average arrival rate.

The periodic task set is scheduled on a single processor by EDF. In order to avoid the domino effect, the execution of a periodic task which missed the deadline is terminated immediately and a new instance is released on the spot.

III. FC-TBS

In this section, we propose the *Feedback Controlled Total Bandwidth Server* (FC-TBS) algorithm which integrates feedback control with the conventional TBS algorithm. We first discuss the approach then design the algorithm. We also describe the feedback control mechanism of FC-TBS in the viewpoint of the control theory.

A. Approach

As we already explained, TBS is known to be a reasonable algorithm in the viewpoints of implementation complexity, memory requirement, runtime overhead and so on. The basic idea of TBS is to assign the possible earliest deadline to each aperiodic task. Once the deadline is assigned, the aperiodic task is scheduled as a periodic task by EDF. The deadline of the kth aperiodic task, d_k, is obtained by Equation (1) where U_s is the server bandwidth configured by a system designer.

$$d_k = \max(r_k, d_{k-1}) + \frac{E_k}{U_s} \tag{1}$$

Let us assume that there are two periodic tasks, $\tau_1 = (6, 3)$ and $\tau_2 = (8, 2)$. When three aperiodic tasks, $\alpha_1 = (3, 1)$, $\alpha_2 = (9, 2)$ and $\alpha_3 = (14, 1)$, arrive at the system, EDF with TBS schedules the tasks as shown in Figure 1.

Note that, in Figure 1, α_2 and α_3 can be executed immediately when they arrive without causing any periodic tasks to miss their deadline. However the EDF scheduler does not execute the aperiodic tasks immediately even if it can, since it schedules all the tasks based on the assigned deadline. The improved version of TBS [1] which shortens the assigned deadline using recursive calculation can overcome this circumstance. However the recursive calculation incurs a significant runtime overhead.

In this paper, we propose a new scheduling scheme that is simple and able to improve the response time for TBS. Our approach provides the special time interval called *Immediate Execution Amount* (IEA) in which the aperiodic tasks can be executed immediately when they arrive. It is obvious that this approach can reduce the response time of the aperiodic tasks. However it is also obvious that some periodic tasks would miss the deadline, since this approach jeopardizes the schedule of EDF. Although a few jobs are allowd to miss their deadline

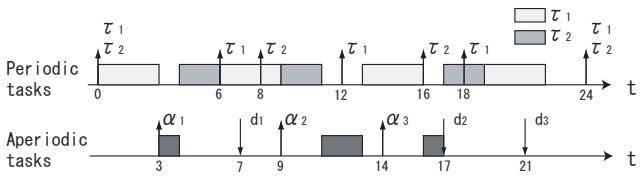

Fig. 1. Schedule example of TBS

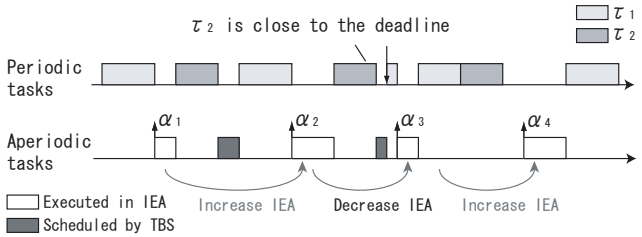

Fig. 2. Schedule example of FC-TBS

in soft real-time systems, the quality of the system degrades as they miss their deadline. Therefore we need to adjust the IEA so as not to increase the deadline misses.

We make use of the feedback control theory to realize our approach. In feedback control, we need to define several variables such as *controlled variable*, *set point* and *manipulated variable*. The controlled variable is the output of the system and is monitored at every sampling period. Then it is controlled so as to be equal to the set point. The difference between the controlled variable and the set point is defined as *error*. A control function adjusts the manipulated variable based on the error so as to have the appropriate controlled variable. We discuss how to apply feedback control to our algorithm in the next section.

B. Algorithm Design

First of all, we need to choose the controlled variable and its set point. Since our goal is to improve the response time with few deadline misses, the controlled variable must be related to the deadline misses. Stankovic *et al.* introduced feedback control to real-time scheduling and they chose the deadline miss ratio as a controlled variable [16], [7], [15]. Al-Omari *et al.* also chose the deadline miss ratio for their feedback control scheduling [9]. However the deadline miss ratio can be measured only after some jobs miss their deadline. As a result, a control function works tardily and the quality of the system may degrade. In this paper, we choose the maximum lateness of the periodic jobs as a controlled variable. Then we choose the IEA as a manipulated variable. The lateness of periodic job $\tau_{i,k}$ is calculated by $f_{i,k} - d_{i,k}$. Let P be the interval of the sampling period, the maximum lateness in the jth sampling period, $L(j)$, is described by Equation (2).

$$L(j) = \max\{L_{i,k} \mid s_{i,k} \geq (j-1)P, f_{i,k} < jP\} \tag{2}$$

Let L_s be the set point of the maximum lateness. In the case of $L(j) > L_s$, since the periodic tasks are about to miss their deadline, FC-TBS decreases the IEA to avoid missing the deadlines. Meanwhile, in the case of $L(j) < L_s$, there is

an enough slack for the periodic tasks, so FC-TBS increases the IEA to reduce the response time. Setting a negative value to L_s, we can control the deadline misses before they happen. Consequently it can provide a higher quality of soft real-time systems.

FC-TBS schedules the tasks as shown in Figure 2. It executes the aperiodic job immediately when it arrives. After the IEA is consumed, the aperiodic job is scheduled based on EDF and TBS. When the maximum lateness is larger than the set point, it means that the job may miss the deadline as the second job of τ_2 in Figure 2, so the controller decreases the value of the IEA. Meanwhile when the maximum lateness is smaller than the set point, the controller increases the value of the IEA.

C. PID Controller

The maximum lateness can take both a positive and a negative. Hence a little overshoot of the controlled variable is acceptable. Therefore we make use of PID (Proportional-Integral-Derivative) control that can bring the controlled variable close to the set point quickly and smoothly. Now we explain how to obtain the value of the IEA in the jth sampling period that is denoted by $A(j)$. Let $E(j)$ be the error in the jth sampling period. $E(j)$ is obtained as follows.

$$E(j) = L_s - L(j) \tag{3}$$

Then the magnitude of the IEA change for the next sampling period is denoted by $\Delta A(j)$ and is calculated by Equation (4) where K_p, K_i and K_d are the parameters for the feedback control gain that is described in the next section in detail. Also I is the time period in which the integral calculation is valid and D is the time period in which the derivative calculation is valid.

$$\Delta A(j) = K_p E(j) + K_i \sum_{k=j-I}^{j} E(k) + K_d \frac{E(j) - E(j-D)}{D} \tag{4}$$

The value of the IEA in the $j+1$th sampling period is obtained by Equation (5).

$$A(j+1) = A(j) + \Delta A(j) \tag{5}$$

D. Parameter Tuning

In feedback control, it is very important to set the proper parameters for the control gain. In the case of FC-TBS, we need to obtain the appropriate parameters in Equation (4). If the parameters are not set properly, the controlled variable does not converge and the system cannot be stable. In the control engineering point of view, there are two methods to determine the feedback control parameters.

1) Modeling and Analysis
2) Tuning

Most of the previous work on feedback control scheduling adopt the modeling and analysis method to determine the parameters. This method requires to define transfer function $P(j)$ that satisfies $L(j) = P(j)\Delta I(j)$. However, in FC-TBS, we need to conduct several experiments to obtain $P(j)$ since it highly depends on the runtime EDF schedule. If we need several experiments, it is easier to take the tuning method to determine the parameters. There are various ways in the tuning method. For example, we have a way that conducts thousands of experiments and find the best parameters. However this way costs a lot of time. In order to reduce the time cost, there are two popular ways to determine the parameters, *step response method* and *ultimate sensitivity method*, proposed by Ziegler and Nichols [18]. Those methods require only a few simple experiments. Based on the foundation above, we take the tuning method to determine the parameters in this paper.

TABLE I
ULTIMATE SENSITIVITY METHOD

K_p	K_i	K_d
$0.6K_u\left(1-\frac{1}{T_u}\right)$	$1.2\frac{K_u}{T_u}$	$\frac{3K_u}{40T_u}$

Although the step response method and the ultimate sensitivity method were originally designed for continuous-time systems, Takahashi *et al.* extended those methods to discrete-time systems [17]. Since the step response method requires to approximate the target system to a transfer function composed of the delay component and the dead time component. Hence we make use of the ultimate sensitivity method for simplicity. The ultimate sensitivity method empirically applies P control to the target system and increases the P gain by degrees until the control reaches the stability limit. Once P gain K_u and vibration period T_u at the stability limit are obtained, we can compute the PID control parameters from Table III-D according to Takahashi *et al.*.

IV. PERFORMANCE EVALUATION

This section evaluates the advancement of FC-TBS by simulation. First of all, we compare FC-TBS with three algorithms, Background Server, TBS, EDL and M/M/1 model process in the mean response time point of view. The M/M/1 model process is that the aperiodic tasks are executed immediately when they arrive and never preempted until it is completed. Secondly we measure the deadline miss ratio of FC-TBS. Finally we show how the feedback control function of FC-TBS works.

A. Simulation Model

The periodic task set is generated as follows. A uniform distribution is used to determine the utilizations of the periodic tasks in the range of $[1\%, 10\%]$. The period is chosen from $\{100, 200, 300, ..., 800\}$. Once the period is determined, the execution time can calculated with the utilization and the period. The simulation runs for the hyperperiod of the periodic tasks. If the range of the periods is not restricted as $\{100 \sim 800\}$, the simulation time becomes a huge amount because the hyperperiod is the least common multiple of all the task periods then we cannot even simulate EDL algorithm. The periodic load, i.e. the total utilization of the periodic tasks, is set 60% in this simulation.

TABLE II
THE PID CONTROL PARAMETERS

	K_p	K_i	K_d
FC-TBS-L(-5)	0.0270	0.0180	0.00112
FC-TBS-L(-10)	0.0315	0.0210	0.00131

As for the aperiodic workload, we refer to the M/M/1 queuing model to determine the arrival times and the execution times of the aperiodic tasks. Namely, Poisson distribution is used to calculate the arrival times and the exponential distribution is used to calculate the execution times. We prepare two average service rates, $\mu = 0.2$ and $\mu = 0.1$. Then, for each μ, the simulation runs from $U_P + U_A = 80\%$ to $U_P + U_A = 98\%$ by changing the average arrival rate.

We do not design a sophisticated deadline miss handler which deals with the periodic tasks which missed the deadline. The tasks missing the deadline are automatically terminated for simplicity in this paper.

B. Parameter Setup

In order to observe the impact of the set point, we prepare two set points, $L_s = -5$ and $L_s = -10$. We denote FC-TBS(-5) and FC-TBS(-10) respectively. As for the PID control parameters, we make use of the ultimate sensitivity method. The parameters are determined as follows. We set $K_p = 0$ at first. Then K_p is gradually increased until the control reaches the stability limit. As a result, we found $\{K_u, T_u\} = \{0.06, 4\}$ for FC-TBS(-5) and $\{K_u, T_u\} = \{0.07, 4\}$ for FC-TBS(-10). Substituting those K_u and T_u to the formulas in Table III-D, we have the PID control parameters as shown in Table IV-B. The feedback sampling period is 800 since the maximum task period is 800. The the integral interval and the derivative interval in Equation (4) are as $I = 10$ and $D = 1$ respectively.

C. Mean Response Time

Figure 3 shows the mean response time (MRT) for each algorithm when the periodic load is 60% and the average service rate is 0.1, that is, the average execution time of the aperiodic tasks is 10. We can observe that the proposed algorithm, FC-TBS, outperformed the conventional algorithms, Background and TBS. Another algorithm, EDL, had better performance than FC-TBS, however its overhead is much higher than FC-TBS as we discuss later. Since the performance for each algorithm was almost same until the system utilization of 90%, FC-TBS is more effective than Background and TBS in a higher system utilization. FC-TBS(-5) and FC-TBS(-10) reduced the mean response time up to about 23% and 20% respectively compared to TBS.

Figure 4 shows the mean response time for each algorithm when the average service rate is 0.2, that is, the average execution time of the aperiodic tasks is 5. At the system utilization of 98%, FC-TBS(-5) and FC-TBS(-10) reduced the mean response time about 26% and 22% compared to TBS. The M/M/1 model that executes the aperiodic tasks immediately when they arrive regardless of the periodic tasks highly outperformed the other algorithms, however it cannot be used in real-time systems since it causes many deadline misses as we show in the next section.

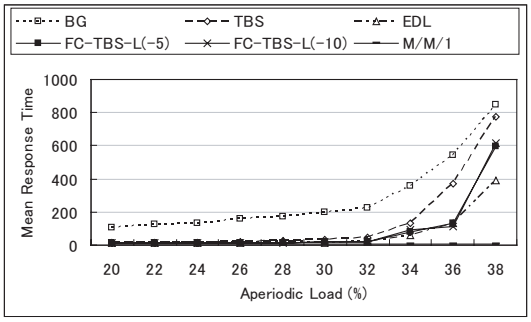

Fig. 3. MRT with $U_p = 60\%, \mu = 0.1$

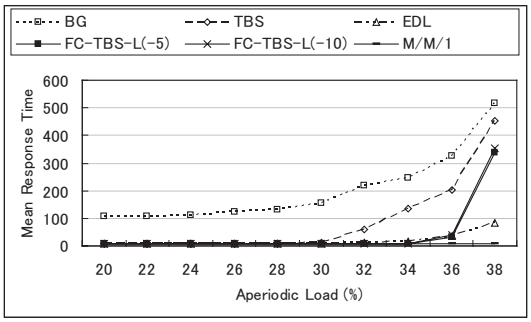

Fig. 4. MRT with $U_p = 60\%, \mu = 0.2$

From those simulations, we can observe the impact of executing the aperiodic tasks immediately when they arrive to reduce the response time. Comparing FC-TBS(-5) and FC-TBS(-10), we can observe that FC-TBS can reduce the mean response time by decreasing the set point of the maximum lateness.

D. Deadline Miss Ratio

This section evaluates the deadline miss ratio for FC-TBS(-5), FC-TBS(-10) and the M/M/1 model. Since the other algorithms are designed to meet all the deadlines, we did not include those algorithms to simulation.

Figure 5 and Figure 6 show that the deadline miss ratio of FC-TBS(-10) was the lowest in the three algorithms and it was at most only 0.5%. FC-TBS(-5) also kept the deadline miss ratio to be quite low and it was at most 1.2%. The M/M/1 model, on the other hand, caused the deadline miss ratio of more than 8%.

From those simulations, we can observe the impact of using feedback control to reduce the number of deadline misses as well as reducing the mean response time as shown in the previous section. Lu *et al.* mentioned that the deadline miss ratio of 2% is acceptable in real systems [8]. Hence FC-TBS is practical for real systems since its deadline miss ratio was less than 2%.

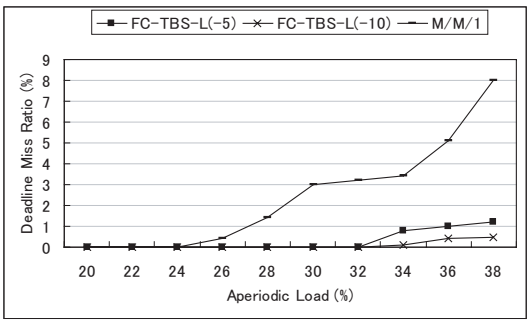

Fig. 5. DMR with $U_p = 60\%, \mu = 0.1$

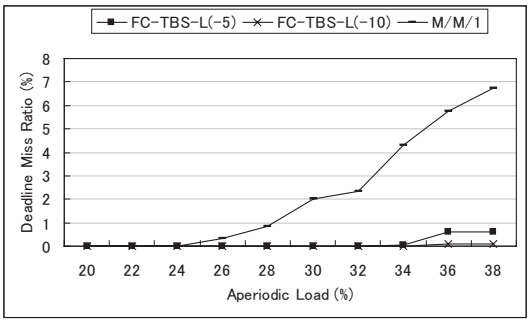

Fig. 6. DMR with $U_p = 60\%, \mu = 0.2$

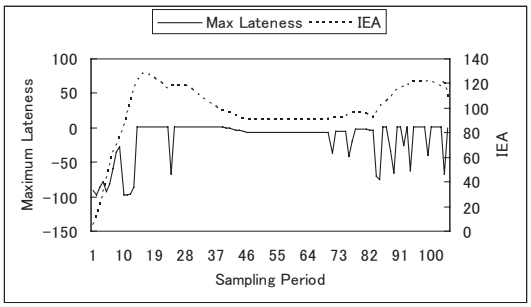

Fig. 7. FC-TBS(-5) with $U_p = 60\%, \mu = 0.1$

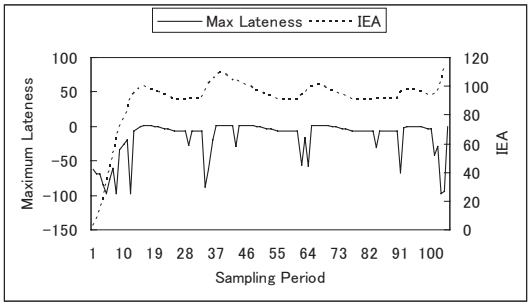

Fig. 8. FC-TBS(-5) with $U_p = 60\%, \mu = 0.2$

E. Feedback Control

The previous two sections showed the mean response time and the deadline miss ratio for each algorithm. This section shows how FC-TBS adjusted the IEA and controlled the maximum lateness at runtime. It was measured when the system utilization was 98%.

Figure 7 and Figure 8 show the feedback control of FC-TBS(-5). Since FC-TBS makes use of PID control, it quickly brought the maximum lateness to the set point (-5) around the 10th sampling period. After the maximum lateness reached the set point, it was controlled to keep stable. When the maximum lateness dropped out the set point, the PID controller brought it back to the set point quickly and smoothly by adjusting the IEA. Figure 9 and Figure 10 show the feedback control of FC-TBS(-10). FC-TBS(-10) also kept the maximum lateness around the set point (-10).

From those results, we can observe that FC-TBS properly controls the maximum lateness. Even if the maximum lateness dropped out the set point, it can bring it back to the stable condition quickly and smoothly.

F. Overhead

We finally evaluate the average overhead for TBS, EDL and FC-TBS in Table IV-F. The overhead here means the total calculation time except for the execution time of the tasks. The overhead time for each algorithm is normalized by that for TBS. The result shows that the overhead of FC-TBS is almost same as that of TBS. It means that the overhead of using feedback control can be ignored in real systems. Meanwhile EDL takes overhead about thirteen times as much as TBS and FC-TBS. Although the performance of EDL in the response time point of view was superior to the other algorithms, it is difficult to use EDL in real systems due to a significant overhead.

TABLE III
THE AVERAGE OVERHEAD

TBS	EDL	FC-TBS
1.00	13.89	1.01

V. CONCLUSION AND FUTURE WORK

This paper described a new scheme for scheduling aperiodic tasks in soft real-time systems. We designed the FC-TBS algorithm that integrates feedback control with the conventional TBS algorithm. We also explained how to determine the feedback control parameters.

The simulation study demonstrated that FC-TBS can reduce the mean response time up to $22\% \sim 26\%$ compared to TBS in exchange for the deadline miss ratio less than $0.5\% \sim 1.2\%$. We also showed how FC-TBS adjusts the manipulated variable and controls the controlled variable in feedback control.

We consider the following future work. Although this paper did not take the deadline miss handler into account, we need to design the handler for real systems. Otherwise the proposed algorithm is not available for sophisticated multimedia applications such as MPEG, though it can deal with simple multimedia application such as JPEG2000. Also we only discussed the step response method and the ultimate sensitivity method to determine the PID control parameters. However there are more sophisticated methods such as the

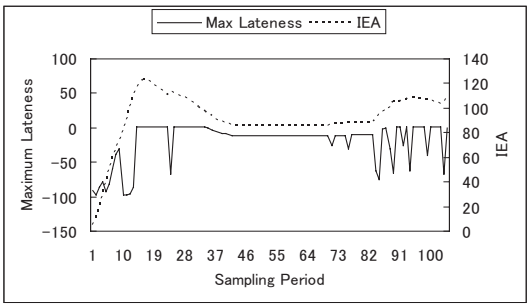

Fig. 9. FC-TBS(-10) with $U_p = 60\%, \mu = 0.1$

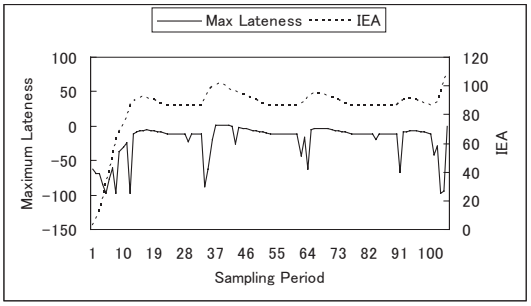

Fig. 10. FC-TBS(-10) with $U_p = 60\%, \mu = 0.2$

auto tuning method and the self tuning method. Hence we will try various parameter tuning methods to improve the quality of the algorithm.

ACKNOWLEDGMENT

This work is supported by the fund of Research Fellowships of the Japan Society for the Promotion of Science for Young Scientists. This work is also supported in part by the fund of Core Research for Evolutional Science and Technology, Japan Science and Technology Agency.

REFERENCES

[1] G.C. Buttazzo and F. Sensini. Deadline Assignment Methods for Soft Aperiodic Scheduling in Hard Real-Time Systems. *IEEE Trans. on Computers*, 48:1035–1052, 1999.

[2] R.I. Davis, K.W. Tindell, and A. Burns. Scheduling Slack Time in Fixed Priority Pre-emptive Systems. In *Proceedings of IEEE Real-Time Systems Symposium*, pages 222–231, 1993.

[3] R.I. Davis and A. Wellings. Dual Priority Scheduling. In *Proceedings of IEEE Real-Time Systems Symposium*, pages 100–109, 1995.

[4] J.P. Lehoczky and S. Ramos-Thuel. An Optimal Algorithm for Scheduling Soft-Aperiodic Tasks in Fixed-Priority Preemptive Systems. In *Proceedings of IEEE Real-Time Systems Symposium*, pages 110–123, 1992.

[5] J.P. Lehoczky, L. Sha, and J.K. Strosnider. Enhanced aperiodic responsiveness in hard real-time environments. In *Proceedings of IEEE Real-Time Systems Symposium*, pages 261–270, 1987.

[6] C.L. Liu and J.W. Layland. Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment. *Journal of ACM*, 20:46–61, 1973.

[7] C. Lu, J.A. Stankovic, G. Tao, and S.H. Son. Design and Evaluation of a Feedback Control EDF Scheduling Algorithm. In *Proceedings of Real-Time Systems Symposium*, pages 56–67, 1999.

[8] C. Lu, J.A. Stankovic, G. Tao, and S.H. Son. Feedback Control Real-Time Scheduling: Framework, Modeling, and Algorithms. *Journal of Real-Time Systems, Special Issue on Control-theoretical Approaches to Real-Time Computing*, 23:85–126, 2002.

[9] R.A. Omari, G. Manimaran, M.V. Salapaka, and A.K. Somani. Novel Algorithms for Open-loop and Closed-loop Scheduling of Real-time Tasks in Multiprocessor Systems Based on Execution Time Estimation. In *Proceedings of IEEE International Parallel and Distributed Proceesing Symposium*, pages 7–14, 2003.

[10] S. Ramos-Thuel and J.P. Lehoczky. On-line Scheduling of Hard Deadline Aperiodic Tasks in Fixed-Priority Systems. In *Proceedings of IEEE Real-Time Systems Symposium*, pages 160–171, 1993.

[11] B. Sprunt, J. Lehoczky, and L. Sha. Exploiting Unused Periodic Time for Aperiodic Service using the Extended Priority Exchange Algorithm. In *Proceedings of IEEE Real-Time Systems Symposium*, pages 251–258, 1988.

[12] B. Sprunt, L. Sha, and J. Lehoczky. Aperiodic Task Scheduling for Hard Real-Time Systems. *Journal of Real-Time Systems*, 1:27–60, 1989.

[13] M. Spuri and G. Buttazo. Efficient Aperiodic Service under Earliest Deadline Scheduling. In *Proceedings of IEEE Real-Time Systems Symposium*, pages 2–11, 1994.

[14] M. Spuri and G.C. Buttazo. Scheduling Aperiodic Tasks in Dynamic Priority Systems. *Journal of Real-Time Systems*, 10:179–210, 1996.

[15] J.A. Stankovic, T. He, T. Abdelzaher, M. Marley, G. Tao, S.H. Son, and C. Lu. Feedback Control Scheduling in Distributed Real-Time Systems. In *Proceedings of Real-Time Systems Symposium*, pages 59–70, 2001.

[16] J.A. Stankovic, C. Lu, S.H. Son, and G. Tao. The Case for Feedback Control Real-Time Scheduling. In *Proceedings of Euromicro Conference on Real-Time Systems*, pages 11–20, 1999.

[17] Y. Takahashi and C.S. Chan. Parametereinstellung bei linearen DDC-Algorithmen. *Regelungstechnik u. Prozess-Datenverarbeitung*, 19:237–284, 1971.

[18] J.G. Ziegler and N. B. Nichols. Optimum Setting for Automatic Controllers. *Transation of ASME*, 64:759–768, 1942.