Proceedings of the American Control Conference
Anchorage, AK May 8-10, 2002

Feedback Control for Real-time Scheduling[1]

Deepak R. Sahoo, S. Swaminathan, R. Al-Omari,
Murti V. Salapaka, G. Manimaran, Arun K. Somani[2].
Department of Electrical and Computer Engineering,
Iowa State University, Ames, Iowa, 50011

Abstract

Most of real-time scheduling algorithms are open-loop algorithms as the scheduling decisions are based on the worst-case estimates of task parameters. They do not continuously observe the performance of the system and do not dynamically adjust the system parameters to improve performance. In many cases, it is preferable to base scheduling decisions on *average-case* workload parameters and be ready to deal with bounded transient overloads dynamically. In recent years, the "closed-loop" scheduling has gained importance due to its applicability to many real-world problems wherein the feedback information can be exploited efficiently to adjust task and/or scheduler parameters, thereby improving the system's performance.

In this paper, we discuss an open-loop dynamic scheduling algorithm that employs a notion of task overlap in the scheduler in order to provide some flexibility in task execution time. Then we present a novel closed-loop approach for dynamically estimating the execution time of tasks based on both deadline miss ratio and task rejection ratio in the system. This approach is highly preferable for firm/soft real-time systems since it provides a firm performance guarantee in terms of deadline misses while achieving a high guarantee ratio. We design proportional-integral controller and H-infinity controller for closed loop scheduling. We evaluate the performance of the open-loop and the closed-loop approaches using simulation studies. We show that the closed-loop dynamic scheduling offers a better performance over the open-loop scheduling under all practical conditions.

1 Introduction

Modeling a computing system as a dynamic system or as a controller is an approach that has proved to be fruitful in many cases. In this paper we will develop a similar notion for dynamic task schedulers for a real-time system. In real-time systems, the system's performance depends not only on the logical correctness of the result, but also on the time at which the results are produced [1]. Based on the time criticality, these systems are classified into : hard, firm or soft real-time systems, in the decreasing order of damages caused due to missing a deadline. So, in hard real-time systems, missing a deadline can lead to catastrophic conditions whereas in firm/soft real-time tasks, missing a deadline will not have such serious consequences. The problem of scheduling real-time tasks in multiprocessor systems is to determine when and on which processor a given task is executed [1]. Though significant results have been achieved in real-time scheduling with algorithms such as RMS, EDF and Spring Scheduling, these algorithms perform well when the workload of the system can be accurately modeled, and does not perform well for unpredictable workloads. Further, all the scheduling algorithms discussed, work based on the worst case execution time (WCET). Practically, identifying the correct estimate for WCET is not easy and many analysis tools come up with over-estimation strategies. Thus, scheduling algorithms based on WCET leads to under-utilized system. In many cases, it is preferable to base scheduling decisions on average execution time and to be ready to deal with bounded transient overloads dynamically. This approach is especially preferable in firm/soft real-time systems since it provides a firm performance guarantee in terms of deadline misses while achieving high utilization and throughput (guarantee ratio) at the same time. In an unpredictable environment, it is impossible for a system to achieve 100% utilization and a 0% miss ratio all the time and a tradeoff between miss ratio and utilization is unavoidable. The approach proposed in [6], uses the closed-loop scheduling based on average case estimation, where a low (but possibly non-zero) miss ratio is maintained with high utilization. When high misses happen due to underestimation of the execution time, the scheduler corrects the system's state back to the satisfactory state, i.e., a state with low miss ratio and high utilization and throughput.

The key issue addressed in this paper is the relaxation of the requirement on a known *worst-case* workload parameters. Specifically, in this paper, we use an open-loop dynamic (where task's characteristics are not known apriori) planning based scheduling algorithm (e.g. Spring Scheduling) that employs a notion of task overlap in the scheduler in order to provide some flexibility in task execution times. This algorithm, dynamically guarantees incoming firm tasks via on-line admission control and planning based on their *AvCET*. We use feedback control theory to design a closed-loop scheduling algorithm derived from the open-loop algorithm that estimates the execution time of a task. The loop is closed by feeding back both deadline miss ratio and task rejection ratio. The result is that this scheduling paradigm has low task miss ratio and high guarantee ratio thereby improving the productivity of

[1]Supported in part by NSF grant ECS-9733802 and CCR-0098354
[2]Email: {deepak, swamis, romari, murti, gmani, arun}@iastate.edu

the firm/soft real-time systems.

The rest of the paper is organized as follows: In Section 2, we define the system model, which includes the task model and scheduler model, and the performance metrics of the system. In Section 3, we present the related work and also discuss the system model and its limitations on performance. In Section 4, we discuss the design of the PI controller and a H_∞ controller as a part of the closed loop scheduling. We also study the response of the PI controller and H_∞ controller and verify the models. In Section 5 we study the performance of the closed loop algorithms and compare it with the open loop scheduler. Then we conclude in section 6.

2 System Model

2.1 Task Model

Tasks are assumed to be aperiodic, i.e., task characteristics are not known a priori. Every task T_i has the following attributes: arrival time (a_i), ready time (r_i), worst-case execution time ($WCET_i$), best-case execution time ($BCET_i$), and firm deadline (d_i). Further, tasks are assumed to be non-preemptable, i.e., when a task starts execution on a processor, it finishes to its completion and tasks may have resource and/or precedence constraint.

2.2 Scheduler Model

In our dynamic multiprocessor scheduling, all the tasks arrive at a central processor called the scheduler, from where they are dispatched to other processors in the system for execution. These processors are identical and are connected through a shared medium. The communication between the scheduler and the processors is through the dispatch queues. Each processor has its own dispatch queue and the scheduler will be running in parallel with the processors, scheduling the newly arriving tasks and periodically updating the dispatch queues. The scheduler considered in this paper is a dynamic scheduler, which does an admission check, wherein the tasks are rejected if they are found non-schedulable and if the task is accepted by the admission controller, then the scheduler constructs the appropriate schedule and arranges the tasks in their appropriate dispatch queues.

2.3 Performance Metrics

Task hit ratio (HR): This is the ratio of the number of admitted tasks that meet their deadlines to the total number of tasks admitted in the system. Though the schedulability check of tasks are performed while admitting them, tasks can still miss their deadline when their actual execution time (AET) is greater than their estimated execution time (EET) or due to the occurrence of unanticipated faults in the system. Task miss ratio (MR) is the ratio of the number of admitted tasks that missed their deadlines to the total number of tasks admitted in the system which is equal to one minus the hit ratio.

Task guarantee ratio (GR): This is the ratio of the number of tasks admitted into the system to the total number of tasks that arrived in the system. The rejection of tasks happens at the scheduler and depends on the schedulability check algorithm, estimated execution time, and the time at which the schedulability check is performed for a task. Task rejection ratio (RR) is the ratio of the number of tasks rejected to the total number of tasks arrived in the system which is equal to one minus the guarantee ratio.

Task effective ratio (ER): This is the ratio of the number of tasks that meet their deadlines to the total number of tasks arrived in the system which is equal to the product of the task hit ratio (HR) and the task guarantee ratio (GR).

3 Feedback Controlled Scheduling

Estimation of task execution time is very important in dynamic scheduling of firm real time tasks. An underestimation of execution times may jeopardize the stability of the system and an overestimation will cause performance degradation by wasting system resources [9]. The actual execution time of a task varies between its $BCET$ and $WCET$ due to non deterministic behavior of several low-level processor mechanisms (e.g. caching, prefetching, and DMA data transfer), and also due to the fact that the actual execution time for these tasks are function of the system state, and the amount, nature, and the value of input data [7]. In this section, we present a novel approach in which the actual execution time of tasks is dynamically estimated based on the current deadline miss ratio (MR) and task rejection ratio (RR) in the system. This output feedback scheme improved the scheduling performance of the algorithm by accepting significantly more number of tasks and meeting more deadlines. Thereby it can improve the productivity of the firm/soft real-time systems.

3.1 Related Work

The idea of feedback has been used informally in scheduling algorithms for applications where the dynamics of the computation workload cannot be characterized accurately for a long time, such as in [10, 11]. Feedback scheduling in real-time systems has been a relatively new area and very few works have been done in this area. In [12] and [13] the authors propose to integrate the design of the system controller with the scheduling of real-time control systems and present offline open loop algorithms. A notable work in real-time feedback scheduling is [4, 5] where the authors propose the use of a PID controller as an on-line scheduler under the notion of Feedback Control-EDF (FC-EDF). In [9] the authors proposed an approach in which task periods can be dynamically adjusted based on the current load. Load is estimated by monitoring the actual computation time of each task. When the estimated load is found to be greater than a certain threshold (e.g. 1 under EDF), the elastic theory is used to enlarge the task periods to find a feasible configuration. In both works [5, 9] the authors assume that each task has multiple versions that differ in their execution time, the higher the execution time the better the quality (imprecise computation). Instead, our work assumes only one version for each task which is more general and realistic. Moreover, in [5] the feedback mechanism is used to reject tasks in order to keep the total number of missed deadlines below a desired value. Instead, in our work the feedback mechanism is used to obtain a trade off between the guarantee ratio and the hit ratio

in order to improve the effective ratio. Also, in this work the admission controller uses the $WCET$ of task versions to perform the schedulability test. Instead, in our work we use an estimated execution time for each task to perform the schedulability test. The estimated execution time of the tasks are adjusted using the feedback mechanism. In [9] the control mechanism has not been analyzed to prove its correctness and stability. Instead, in our work we analyze the control mechanism and we present the block diagram for the system and the analytical equations that connect the miss ratio and the rejection ratio with the estimated execution time of tasks.

3.2 System and Model Identification

Before applying feedback control techniques, we present the measured variables, the control output and the open-loop characteristics of the system.

3.2.1 Measured Variables: The choice of the measured variables depends on the system goal. The performance of firm real-time systems usually depends on: (1) how many tasks it admits (rejects); (2) how many tasks among the accepted task make (miss) their deadlines. Therefore the deadline miss ratio (MR) and the tasks rejection ratio (RR) are natural choices of the measured variables.

The instantaneous miss ratio (MR_{kT}) and instantaneous rejection ratio (RR_{kT}) are calculated every sample period (T) for the time interval $[(k-1)T, kT]$. In contrast, the average rejection ratio (RR) is defined as the time average of the instantaneous rejection ratio (RR_{kT}) for the entire run-time.

3.2.2 Control Output: The control output must be able to affect the value of the measured variable. In the non-preemptive multiprocessor firm real-time system, it is a widely known fact that the deadline miss ratio and the tasks rejection ratio highly depend on the estimated execution time of tasks. Thus the estimated execution time (EET) of tasks is an appropriate choice for the control output. The estimated execution time (EET_i) of task (T_i) is calculated using the following equation:

$$EET_i = AvCET_i + etf_{kT}\,[AvCET_i - BCET_i], \quad (1)$$

where etf_{kT} is the estimation factor at time instant kT which can have a value in the interval $[-1, 1]$.

In the case of open-loop scheduling algorithms etf is fixed which is equal to 0 for the algorithms that use the $AvCET$ of tasks as an estimation for their actual execution time, and it is equal to 1 for the algorithms that use the $WCET$ of tasks as an estimation for their actual execution time. In the case of closed-loop scheduling algorithms the estimated factor (etf) is determined using the controller.

3.2.3 Open-Loop Characteristics: After identifying the measured variables and control output, we analyze the open-loop characteristics of the system.

From simulation the open-loop characteristic of the real-time task scheduler system (figure 1) was obtained. We observe that the relation of the estimation factor (etf) with task rejection ratio (RR) and deadline miss ratio (MR) is not linear for any task load (L). Also the nonlinear relation varies with the system task load. It can be noticed that MR starts from a maximum value when $etf = -1$ and reduces quadratically as etf increases. For all task loads that are plotted MR reaches zero when ($etf > 0.75$). RR starts from approximately zero when $etf = -1$ and increases quadratically to a maximum value as etf increases to 1.

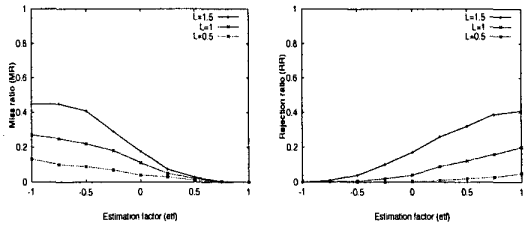


Figure 1: MR (left) and RR (right) vs. etf

The open loop scheduler is designed for a firm real-time system and works according to the algorithm given in [6]. The open loop algorithm schedules the incoming tasks by taking into account their $AvCET$. The closed loop algorithm schedules the incoming tasks by estimating their EET. The goal of our closed loop scheduler design is to trade off between MR and RR in order to achieve a high value of hit ratio and guarantee ratio. In that case, the MR and RR values will be close to each other and the system will operate in the linear region of the curve near $etf = 0$. Then the corresponding system model in that region is given by,

$$\begin{aligned} MR(z) &= -(mgf(L))etf(z) + mdf(L), \\ RR(z) &= (rgf(L))etf(z) + rdf(L), \end{aligned} \quad (2)$$

where mgf and rgf are miss ratio gain factor and rejection ratio gain factor respectively. mdf and rdf are miss ratio disturbance factor and rejection ratio disturbance factor respectively. The values for the mgf, rgf, mdf and rdf varies with, the scheduling algorithm that is used, the system load, and the system parameter (e.g. number of processors). The values of these factors can be found by studying the simulation model.

This linear model approximation may not be accurate since scheduling system typically contains non-linear factors which are not presented in the current simulator. These figures have been generated using the simulation model. For each point in Figures 1, the system was simulated for 10,000 tasks. This number of tasks have been chosen to have a 98% confidence interval within ± 0.0017, and ± 0.0022 around each value of MR, and RR, respectively.

3.2.4 Limitation on Performance: From equation 2 it can be derived that

$$\frac{MR(z)}{mgf(L)} + \frac{RR(z)}{rgf(L)} = \frac{mdf(L)}{mgf(L)} + \frac{rdf(L)}{rgf(L)}. \quad (3)$$

Since $MR(z)$ and $RR(z)$ must satisfy equation (3) for a given task load, we can not simultaneously make them arbitrarily small. This is a limitation on performance exerted by the system for any task-load. However we can trade off between the miss ratio and rejection ratio by applying feedback control or by designing a closed loop dynamic scheduler and obtain an optimal performance of the scheduler in terms of effective ratio. Note that since the parameters mgf, mdf, rgf and rdf are functions of load L, miss ratio and rejection ratio after trade off remains high or low according to the task-load. From equation (3), it is clear that a low rejection ratio corresponds to a high miss ratio and vice versa. In other words a high guarantee ratio corresponds to a low hit ratio and vice versa. For a stable real-time system both the guarantee ratio and hit ratio should be high or effective ratio should be high. Therefore for closed loop scheduler design we feedback both miss ratio and rejection ration to obtain an high effective ratio for optimal performance.

4 Developing Closed-Loop Scheduling Algorithms with Controllers

In this section, we will incorporate control theory into the open loop scheduling algorithm to develop the dynamic closed loop scheduling algorithms. We will design PI and H_∞ controllers to develop the closed loop model.

The closed loop architecture is given in Figure 2. The new scheduler is composed of a controller, an execution time estimator and an overlap scheduler. In this approach the task rejection ratio (RR_{kT}) and deadline miss ratio (MR_{kT}) are periodically fed back to the controller after each sample time T. The controller computes the estimation factor etf and calls the execution time estimator to change estimated execution time of tasks. The overlap scheduler schedules the arrived tasks according to their estimated execution time.

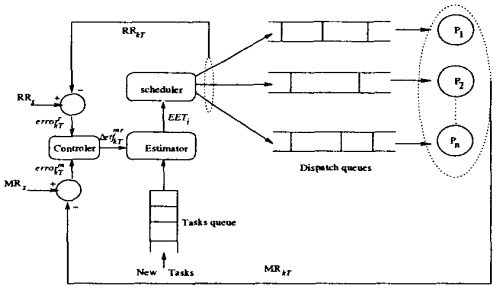


Figure 2: Architecture of Closed Loop Algorithms

4.1 PI-Controller

Our primary goal behind using a feedback controller is to obtain a trade off between hit ratio and guarantee ratio, by controlling/estimating the execution time of tasks, in order to improve the effective ratio of the system. The closed loop system with PI-Controller is given in figure 3.

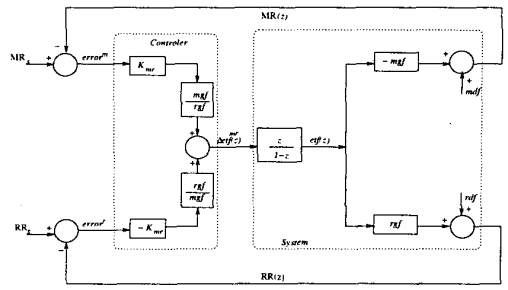


Figure 3: Block diagram of the Closed Loop System with the PI-Controller

The suggested controller in Z-domain is given by,

$$etf(z) = \frac{z}{1-z}\Delta etf(z), \text{ where} \quad (4)$$
$$\Delta etf(z) = f_{m/r}\, K_{mr}\, err^m(z) - f_{r/m}\, K_{mr}\, err^r(z)$$

and $f_{r/m} = \frac{rgf}{mgf}$, $f_{m/r} = \frac{mgf}{rgf}$ are normalization factors for task rejection ratio and deadline miss ratio respectively. The idea behind using these normalization factors is that the sensitivity of miss ratio (MR_{kT}) to some variation in the estimated factor (Δetf_{kT}) is different from the sensitivity of rejection ratio (RR_{kT}) to the same variation.

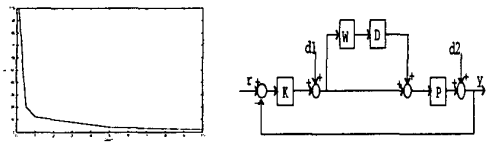
Note that in s-domain the PI controller is given by using bilinear-transforms as, $etf(s) = \frac{-\frac{1}{F}-\frac{1}{2}s}{s}\Delta etf(s)$, where F is the sampling frequency of the scheduler.

The control rule is obtained from equation (4) by using inverse Z-transform and is given by,

$$\Delta etf_{kT}^{mr} = f_{m/r}\, K_{mr}\, err_{kT}^m - f_{r/m}\, K_{mr}\, err_{kT}^r$$
$$etf_{kT} = etf_{(k-1)T} - \Delta etf_{kT},$$

The PI controller is an ad hoc design and it needs tuning of controller gain, K^{mr} and sampling period, T for optimal performance. Note that the bandwidth of the system is decided by sampling period, T. If T is low, the bandwidth will be high, the transient response will be fast i.e. the settling time of the system will be low. But in our system it is not possible to obtain arbitrarily high bandwidth as the system will become unstable when T is smaller than a certain value $T_0 = \max(\theta, C)$, where C is the mean computation time of the system and θ is the mean arrival time of tasks. That is because, in the case of feeding back the rejection ratio, if the sample period is less than the mean arrival time of tasks, there is a chance that the number of tasks arrived in the sample interval $[kT, (k+1)T]$ is zero or a very small number. This results in feeding back an inaccurate estimation for the rejection ratio. In the case of feeding back the miss ratio, if the sample period is less than the mean computation time of the system , there is a chance that the number of tasks finished in the sample interval $[kT, (k+1)T]$ is zero or a very small number. This results in feeding back an inaccurate estimation for the miss ratio. Apart from the sample

period T, the controller gain (K^{mr}) also needs to be tuned in order to obtain better performance. Figure 4 shows that a low value of controller gain can make the system extremely sluggish. After tuning for small settling time and peak overshoot, we chose $K^{mr} = 0.5$ for our simulations.



(a) (b)

Figure 4: (a) Settling time Vs Controller Gain (b) Closed Loop Block Diagram with the H_∞ Controller where $G = [-mgf,\ 0;\ 0,\ rgf]$, $d_2 = [mdf\ rdf]'$, $y = [MR\ RR]'$, $r = [MR_s\ RR_s]'$ and $u = etf$.

4.2 H_∞ Controller

A model for open-loop real-time scheduler behaviour is obtained through extensive simulation so that closed loop scheduling algorithms can be designed in order to improve performance of the scheduler. The PI-controller is the first step in designing such a closed loop algorithm. Note that effective ratio is taken as the performance metric which is related to hit ratio and guarantee ratio of the incoming tasks. From the model of the system it can be derived that maximum effective ratio is obtained when $\frac{HR}{mgf} = \frac{GR}{rgf}$. The PI-controller designed is not able to satisfy this condition. Therefore it is necessary to design advance controllers using modern control theory to achieve desired optimal performance.

The open loop characteristics and the system model is given by figure 1 and equation 2 respectively. Under varying load condition the system model can be treated under H_∞ controller design framework. Change in task-load corresponds to perturbations in the system parameters mgf, rgf, mdf and rdf. Therefore a generalized parametric uncertainty model will best describe the given system as shown figure 4(b). For simplicity we considered a linearized model around an operating point with uncertainty (See equation 5) to simulate the varying load condition. Here the underlying assumption is that in steady state under constant load condition the estimation factor, etf remains constant. The H_∞ controller was designed using the function 'hdfsyn' in matlab.

$$\begin{aligned} MR(z) &= -(mgf_0)etf(z) + mdf_0 + \Delta mdf, \\ RR(z) &= (rgf_0)etf(z) + rdf_0 + +\Delta rdf. \end{aligned} \quad (5)$$

5 Model Verification and Performance Evaluation

In this section, we have verified the closed-loop scheduling models by comparing the results from the matlab simulink models and the results from the simulator for the real-time scheduler. The simulator software simulates the open loop and closed loop algorithms (PI and H_∞) for the real-time schedulers for a multiprocessor firm real-time system . The tasks for the simulator are generated as per the rules given below:

The best-case execution time($BCET_i$) of a task T_i are chosen uniformly between 10 secs and 20 secs. The worst-case execution time ($WCET_i$) of a task T_i is chosen to be 4 times its $BCET_i$. The average-case execution time ($AvCET_i$) of a task T_i is the average of its $BCET_i$ and $WCET_i$. The actual execution time is computed as an uniform random variable in the interval [BCET, WCET]. The firm deadline of a task T_i is uniformly chosen between $2+WCET_i$ and $4*WCET_i$. The arrival time of tasks follow exponential distribution with mean θ. The task load L is defined as the expected number of task arrivals per mean service time and is approximately equal to C/θ, where C is the mean computation time of the system.

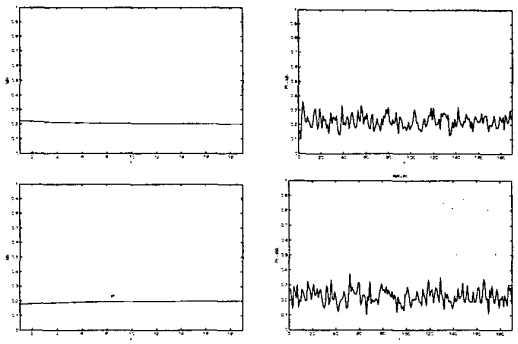


Figure 5: Plots from Matlab (left) and Simulator (right) of MR_{kT} (top) and RR_{kT} (bottom) for the PI-Controller based closed loop algorithm.

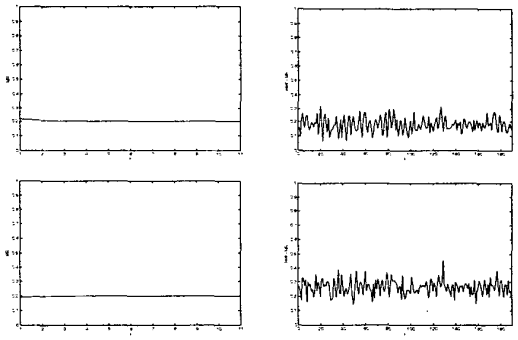


Figure 6: Plots from Matlab (left) and Simulator (right) of MR_{kT} (top) and RR_{kT} (bottom) for the H_∞-Controller based closed loop algorithm.

Figure 5 and 6 show the time response of the closed loop schedulers implemented in the matlab simulink and the real-time simulator. They verify that closed loop scheduler algorithm in the simulator behaves consistently as predicted by the closed loop model in simulink. The instantaneous values of steady state miss ratio (MR_{kT}) and rejection ratio (RR_{kT}) are used as the performance metrics to verify accuracy of the modeling.

5.1 Effect of Task Load on HR, GR and ER

Experiments were carried out to study the performance of the closed-loop scheduling algorithms as compared to the open-loop scheduling algorithm under varying load condition. The performance metrics are chosen to

be hit ratio, guarantee ratio and effective ratio of the system. The results obtained from simulation are plotted in figure 7. Each point in the plots is the average of 20 runs. In each run the system was simulated with 10,000 tasks. This number of runs has been chosen to have a 98% confidence interval within ± 0.013 and ± 0.015 around each value of MR and RR respectively.

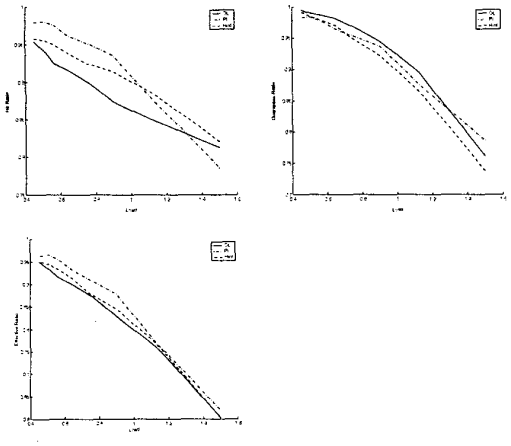


Figure 7: Hit Ratio (top left), Guarantee Ratio (top right) and Effective Ratio (bottom) of the open loop and the two closed loop algorithms.

As shown in the figure 7, all the three algorithms perform equally well when the system load is less than 80%, with respect to GR, whereas the open loop scheduler's HR decreases with increase in task load L. This implies during overload cases the open loop schedulers give a poor MR, as it is insensitive to the change in workload conditions. Since the closed loop schedulers observe the change in system dynamics by constant feedback of the MR and RR, they give a better performance during overload scenarios. In the context of real-time systems, hit ratio is important and completing all the accepted tasks successfully is highly desirable, which is done efficiently by the closed loop schedulers. Among the closed loop schedulers, the PI controller gives a better HR and GR than the H_∞ controller for all task loads. The PI-controller is an ad hoc design and it can adapt to the system dynamics that changes with the task load. Whereas the H_∞-controller designed, which is a model based design, performs well near the nominal task load $L = 1$ only. Figure 7 shows the effective ratio under different load conditions for all the three algorithms. It can be seen that the closed loop schedulers perform better than the open loop schedulers for all load scenarios. This proves the effectiveness of the closed scheduling algorithms developed in this paper.

6 Conclusion

In this paper, we have viewed the problem of scheduling firm real-time tasks in multiprocessor systems as an input/output system. We have used feedback control theory to design two closed-loop scheduling algorithms, which work based on estimated execution time, estimated by feeding back both the deadline miss ratio and task rejection ratio. We have shown that the proposed closed-loop dynamic scheduling algorithms perform better than the open-loop scheduling algorithm for all the stated performance metrics. Further, we would also like to point out that the open loop scheduling algorithms will perform well only when the tasks execute with average case execution time and will perform poorly when the task's execution times are more or less than its average execution time. However, the closed loop schedulers will react to the change in the task workload and will estimate the task load characteristics resulting in the correct estimation of the task's execution time, resulting in a better system performance under dynamic workload settings. Further, we have shown through our experiments that the closed loop algorithms perform better than the open loop algorithm for various task loads. Thus, we feel that closed loop scheduling algorithms are a proper solution to real-time scheduling, leading to improved performance, especially when the workload cannot be characterized easily.

References

[1] K. Ramamritham and J.A. Stankovic, "Scheduling algorithms and operating system support for real-time systems", *Proc. IEEE,* vol. 82, no. 1, pp. 55-67, Jan. 1994.

[2] Liu, C. L. and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard real-time environment," *Journal of the ACM,* 20:1, pp. 40-61, 1973.

[3] J.A. Stankovic and K. Ramamritham, "The Spring kernel: A new paradigm for real-time operating systems," *ACM SIGOPS, Operating Systems Review,* vol. 23, no. 2, pp. 77-83, Jan. 1995.

[4] J. Stankovic, C. Lu, S. Son, and G. Tao, "The case for feedback control real-time scheduling," *EuroMicro Conference on Real-Time Systems,* pp. 11-20, June 1999.

[5] C. Lu, J. A.Stankovic, G. Tao, and S. H. Son, "Design and evaluation of feedback control EDF scheduling algorithm", *Real-Time Systems Symp. (RTSS),* pp. 56-67 Dec. 1999.

[6] Ra'ed Mohammad S. Al-Omari, "Controlling schedulability-realiability trade-offs in real-time systems", Ph.D thesis, Iowa State University, 2001.

[7] K.-E. Aºrzén, B. Bernhardsson, J. Eker, A. Cervin, K. Nilsson, P. Persson, and L. Sha, "Integrated control and scheduling", *Internal report TFRT-7582,* Department of Automatic Control, Lund University, Aug. 1999.

[8] G. Manimaran and C. Siva Ram Murthy, "An efficient dynamic scheduling algorithm for multiprocessor real-time systems," *IEEE Transactions on Parallel and Distributed Systems,* vol. 9, no. 3, pp. 312-319, Mar. 1998.

[9] Giorgio Buttazzo and Luca Abeni, "Adaptive rate control through elastic scheduling," *Proceedings of the 39th IEEE Conference on Decision and Control,* Sydney, Australia, Dec. 2000.

[10] P. R. Blevins and C. V. Ramamoorthy, "Aspects of a dynamically adaptive operating systems," *IEEE Transactions on Computers,* 25(7), pp. 713-725, July 1976.

[11] David C. Steere, et. al., "A Feedback-driven proportion allocator for real-rate scheduling," *Operating Systems Design and Implementation (OSDI),* Feb. 1999.

[12] D. Seto, et. al., "On task schedulability in real-time control systems," *IEEE Real-Time Systems Symposium,* Dec. 1996.

[13] M. Ryu and S. Hong, "Toward automatic synthesis of schedulable real-time controllers," *Integrated Computer-Aided Engineering,* 5(3) pp. 261-277, 1998.

[14] K. G. Shin and C. L. Meissner, "Adaptation and graceful degradation of control system performance by task reallocation and period adjustment," *EuroMicro Conference on Real-Time Systems,* June 1999.