On Task Schedulability in Real-Time Control Systems *

Danbing Seto[1], John P. Lehoczky[2], Lui Sha[1], and Kang G. Shin[3]

[1] Software Engineering Institute
[2] Dept. of Statistics
Carnegie Mellon University
Pittsburgh, PA 15213

[3] Real-Time Computing Laboratory
Dept. of Electrical Eng. and Computer Sci.
The University of Michigan
Ann Arbor, MI 48109

Abstract

Most real-time computer-controlled systems are built in two separate steps, each in isolation: controller design and its digital implementation. Computational tasks that realize the control algorithms are usually scheduled by treating their execution times and periods as unchangeable parameters. Task scheduling therefore depends only on the limited computing resources available. On the other hand, controller design is primarily based on the continuous-time dynamics of the physical system being controlled. The set of tasks resulting from this controller design may not be schedulable with the limited computing resources available. Even if the given set of tasks is schedulable, the overall control performance may not be optimal in the sense that they do not make a full use of the computing resource. In this paper, we propose an integrated approach to controller design and task scheduling. Specifically, task frequencies (or periods) are allowed to vary within a certain range as long as such a change doesn't affect critical control functions such as maintenance of system stability. We present an algorithm that optimizes task frequencies and then schedules the resulting tasks with the limited computing resources available. The proposed approach is also applicable to failure recovery and reconfiguration in real-time control systems.

1 Introduction

The control of physical systems has changed from analog to digital technology, and computer control has been applied to perform more complex, higher-level functions. In applications ranging from flight control to micro-surgery, real-time control plays a crucial role in the coordination of the dynamics of these systems. Although the application domain of digital real-time control has been enlarged significantly, there still remain many issues in control implementation before its full potential can be realized. For example, the design of controllers and the scheduling of control tasks are usually considered separately, and this can result in suboptimal system designs. In this paper we investigate the interaction between control task performance and task scheduling.

Task scheduling is a fundamental issue in real-time control algorithm implementation. A seminal contribution was made by Liu and Layland [2] who developed optimal static and dynamic priority scheduling algorithms for hard real-time task sets. They showed for such task sets that dynamic priority scheduling algorithms can achieve 100% schedulable utilization, while the optimal static priority algorithm, the rate monotonic algorithm, has a least upper bound of 69% on its schedulable utilization. Nevertheless, over the last two decades, significant progress has been made on generalizing these algorithms and making them suitable for real applications. Still, nearly all of these developments have assumed that the task set characteristics (e.g., computation times, periods and deadlines) are fixed and known.

There have been several papers which relate task scheduling and system performance. For example, Gerber, Hong and Saksena [1] addressed the issue of task design in relation to system performance; however, their focus was on distributed systems and they did not use the performance index approach we present. Task scheduling and system performance have also been addressed by Locke [3] and other authors using best-effort scheduling. This approach is especially designed to handle transient overloads, and its premise is that a

*This research was supported in part by the Office of Naval Research under contract N00014-92-J-1524, by the Software Engineering Institute of Carnegie Mellon University, by the NSSN program, by the ISC program, and by the JSF program.

task will obtain a value which depends on the time at which it is completed. Again, this work did not focus on any particular application area such as control algorithm scheduling nor were performance indices introduced.

The control law is usually derived based on physical system properties under the assumption that it will always be implementable on a digital controller computer. A digital control algorithm could be designed to optimize some system performance index (PI). With a digital implementation, a controller can be designed by either *direct digital design* which first discretizes the associated continuous-time system dynamics and then design a control algorithm for the discretized system, or *continuous-time design and then digitization* in which the control algorithm is designed based on the continuous-time system dynamics and the resulting control law is digitized for implementation on a computer. However, neither of these design strategies takes into account any limitations on the available computing resources.

A better approach to real-time control system design would be to optimize the global system performance considering *both* control performance and computing resources. Such an integrated approach requires knowledge of the relationship between control system performance and sampling frequencies (task periods). In this paper, we assume that (i) the control algorithm is developed with the continuous-time design and then digitization and (ii) the resulting algorithm is "optimal" (in the sense of a certain given objective function). To implement such a control strategy, we would like the sampling frequency to be as high as possible in order to make a better match between the (optimal) continuous-time control and its digital implementation. Note, however, that the limitations on computing resources shared among multiple tasks imposes an upper bound on the sampling frequency for each periodic task. These upper bounds, one for each periodic task, must be considered in the integrated design approach, while they need not in standard control designs. On the other hand, to correctly capture and control the system dynamics, the sampling frequencies are normally chosen to be 5–10 times the corresponding system's characteristic frequencies. This requirement gives a lower bound on the sampling frequency from the control system point of view.

By allowing the sampling frequencies to vary within the ranges defined by the lower and upper bounds mentioned above, we can actually enhance periodic task schedulability. That is, one can change the periods of some of tasks in the given set (within these ranges) so that all of the periodic tasks in the set may become schedulable, if the originally-given task periods make the tasks unschedulable. We will adjust the task frequencies to optimize the overall system control performance, subject to two constraints: (1) the lower bounds on task frequencies and (2) the underlying scheduling algorithm and the limitations on available computing resources. This approach can also be used to re-allocate tasks when some of the processors fail (but the details of this extension are not within the scope of this paper; we will report on them in a future paper).

The paper is organized as follows. In Section 2, we briefly review some of the basic concepts in control theory, especially the optimization of control system performance and digital control implementation. We describe in detail the rationale for combining control design and its digital implementation (i.e., task scheduling) in the design of real-time control systems. The main results of this paper are presented in Section 3, where we derive an algorithm for choosing the optimal task frequencies such that all the tasks are schedulable, and the system performance using the digital control implementation is optimized for the limited computing resources. The conclusions are drawn in Section 4.

2. Control Design and Its Digital Implementation

Here we will briefly review some of the relevant concepts in control theory. In particular, we will first present an example to demonstrate the relationship between the control system performance index and the control task frequency when the control input is to be produced by a digital computer. Next we discuss the issues of optimizing the system performance and digital control implementation in general.

To illustrate the effect of sampling frequency on system performance, we have chosen an actual real-time control application, a bubble control system. Such a system is a simplified model designed to study diving control in submarines. For a discussion of real-time control systems in real submarines, the reader may refer to [5]. The bubble control system considered here consists of a tank filled with water and a diver, an inverted cup partially filled with air and immersed in the water. Depth control of the diver is achieved by adjusting the air volume inside the diver. A schematic diagram of the system is given in Fig. 1.

Let $x = (x_1, x_2, x_3) = (y, \dot{y}, h - h_e)$. The equations

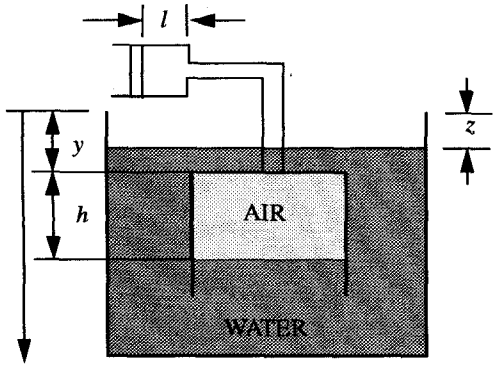

Figure 1. A schematic diagram of the bubble control system

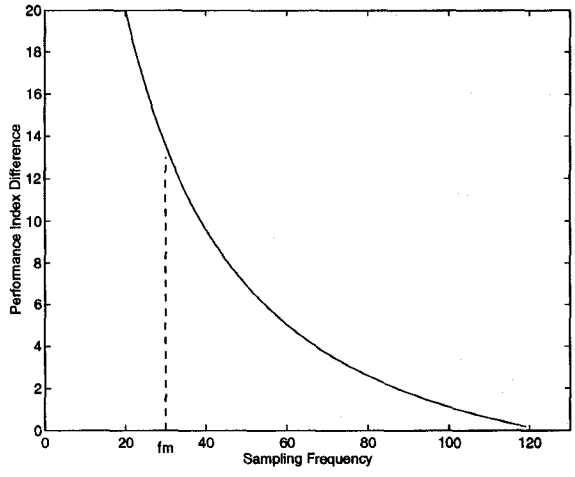

Figure 2. $\Delta J^*(f)$ for the bubble control system

of motion can be written as;

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -c_1|x_2|x_2 - c_2x_3 \\ \dot{x}_3 &= -a(x)x_2 - b(x)u \end{aligned} \tag{1}$$

where h_e is the air-volume height at equilibrium state, u is the control variable defined as the piston velocity $\dot{l}$, c_1 and c_2 are positive coefficients of the water resistance and buoyancy, respectively, and $a(x)$ and $b(x)$ are positive definite functions obtained from the law of conservation of mass. Suppose the control objective is to drive the diver to move at a given speed v_d. Then, a tracking problem can be formulated, for example to have the diver follow the reference trajectory

$$y_r = y(0) + v_d t,$$

and the optimization problem can be stated as:

$$J^* = \min_u J(u) = \int_0^{t_f} [ru^2 + (y - y_r)^T Q(y - y_r)]dt$$

subject to: Eq. (1)
$H_0 \le y \le H - H_c, \quad 0 \le h \le H_c$
$|u| \le u_{\max}$

where H_0, H_c and H are the water level, the height of the diver, and the height of the tank, respectively, and $u_{\max}$ is the maximum piston velocity. The functional $J(u)$ is called system performance index. Its physical meaning can be interpreted as a measure of the total cost of control and tracking error generated in the time period $[0, t_f]$ by the control u. The optimal control problem is to find a control u achieving the minimum cost and tracking error. The optimal control function u is denoted by u^*, and its performance index is denoted by J^*. When any control u is implemented digitally, the system performance will depend on the sampling frequency, that is J is a function of the sampling frequency f, $J_D(u, f)$. Fig. 2 shows the simulation result of $\Delta J^*(f) = J_D^*(f) - J^*$ with f_m the lower bound of frequency.

Remark 2.1 Fig. 2 demonstrates that there is a wide range over which the sampling frequency can vary. Specifically, any frequency above the lower bound f_m will keep the system running satisfactorily. Furthermore, the performance index is a convex function, and it is this convexity property that allows us to schedule the tasks by choosing proper frequencies for them. This will be elaborated on in the next section.

As shown in above example, the system performance is usually measured by a *Performance Index* (PI), and the control algorithm is often derived to optimize this index subject to the system dynamics and constraints. For example, the objective for a radar system is to track a target and the performance index would be some measure of the tracking error. For this example, one would like to design the control algorithm to minimize this performance index. For mechanical systems, on the other hand, the performance index might be some measure of the total work the system produces, and in this case we would want to maximize this performance index. Other examples may involve minimization of time (e.g., minimizing the system response time) or energy (e.g., minimizing the cost). The problem of optimizing the performance index can be stated

formally as follows.

$$(\max)\ \min_{u\in\Omega} J(u) = S(x(t_f), t_f) + \int_0^{t_f} L(x(t), u(t), t)dt \tag{2}$$

$$\text{subject to:} \quad \dot{x} = f(x(t), u(t), t) \tag{3}$$

$$c(x(t), u(t)) \leq 0 \tag{4}$$

where $J(u)$ is the system performance index, $[0, t_f]$ is the time interval of interest, $S(\cdot)$ and $L(\cdot)$ are the weighting (or cost) functions depending on system states, time and control inputs. Eq. (3) describes the dynamics of the underlying system with state $x(t) \in R^n$, control input $u(t) \in R^m$ for each $t > 0$, while Eq. (4) represents the constraints on the system trajectory and the control input with $c(\cdot) \in R^p$. The complete statement of the optimization problem can be summarized as: find the optimal control such that the performance index defined by Eq. (2) will be minimized (maximized) subject to the system dynamics and constraints given by Eq. (3) and Eq. (4).

The optimal control for the problem described above can be derived by direct digital design, or continuous-time design and then digitization. We adopt the latter design approach; similar results can be obtained for the direct digital design. Suppose the optimization problem in Eq. (2)–(4) can be solved with the optimal control $u^*(t)$ resulting in PI J^*. Then, the control implementation determines if we can obtain the performance for which the controller is designed. Discretizing the control input $u^*(t)$ in the time domain, we obtain the performance index as

$$J_D^*(f) = S(x^*(t_f), t_f) + \sum_{k=0}^{n-1} \int_{kT}^{(k+1)T} L(x^*(t), u^*(kT), t)dt \tag{5}$$

where $f = 1/T$ is the sampling frequency, $t_f = nT$, and $x^*(t)$ is determined by

$$\dot{x}^*(t) = f(x^*(t), u^*(kT), t), \quad kT \leq t \leq (k+1)T, \quad k = 0, ..., n-1$$

Eq. (5) shows that the performance index is a function of the sampling frequency and Fig. 3 illustrates possible performance indices. In this paper, we will consider only monotone, convex or concave functions as shown in Fig. 3. The physical meaning of these functions is clear: as the sampling frequency increases, the performance index with discrete-time optimal control (PIDTOC) will tend to converge to the performance index

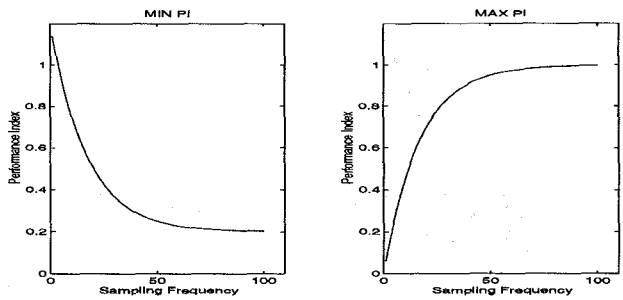

Figure 3. Control system performance indices versus sample frequency

with continuous-time optimal control (PICTOC). On the other hand, as the sampling frequency decreases, the difference between PIDTOC and PICTOC will increase, and eventually the system will become unstable. To prevent this from happening, a lower bound on the sampling frequency must be imposed. For convenience, we will consider the difference $\Delta J^*(f) = J_D^*(f) - J^*$. Clearly,

$$\lim_{f\to 0} |\Delta J^*| = \infty, \quad \text{and} \quad \lim_{f\to\infty} |\Delta J^*| = 0.$$

For a set of n control tasks with given $\Delta J_i^*(f_i)$, $i = 1, ..., n$, we will develop an algorithm to find the optimal choice of $f_1, ..., f_n$ such that $\sum_{i=1}^n \Delta J_i^*$ is minimized subject to the availability of the computing resource.

Remark 2.2 The algorithm developed in this paper is not restricted to the optimal control problem. Let J and $J_D(f)$ be the performance indices generated by continuous-time control which may not be optimal and its digital implementation at sampling frequency f, respectively. Then the algorithm can be applied provided $\Delta J(f) = J_D(f) - J$ is convex and monotonically decreasing. We shall therefore use $\Delta J(f)$ in the rest of the paper. Again, note that ΔJ is a functions of both the sampling frequency f and the control function u. Since our goal is to investigate the effect on sampling frequency of the performance index for any given control function u, we will omit u from the argument list of ΔJ.

Remark 2.3 The tasks to be scheduled may not all be control-related tasks. For example, some tasks might involve data processing and display. The periods for some of these tasks may not be changeable, while others may not have associated performance indices. When the system involves such tasks, we will schedule them as a part of the real-time task set using an appropriate real-time scheduling algorithm, but we will not be able to optimize their frequencies/periods.

In this paper, the algorithm used to choose the task frequencies is developed for a general class of control systems in which the functions $\Delta J(f)$ are monotonically decreasing and convex. Many control systems belong to this class; for example, the aircraft landing control application studied in [7] offers a second example. For this class of systems, we will approximate the function $\Delta J(f)$ to be an exponential decay function, i.e.,

$$\Delta J(f) = \alpha e^{-\beta f}$$

where α is the magnitude coefficient and β is the decay rate.

3. Task Scheduling

In this section, we address the issue of determining the task frequencies such that all the tasks are schedulable *and* the system performance indices are optimized. Specifically, for a given set of tasks, $\tau_1, ..., \tau_n$, with

$$\Delta J_i(f_i) = \alpha_i e^{-\beta_i f_i}, \quad i = 1, ..., n \tag{6}$$

we choose the frequencies f_i to minimize $\sum \Delta J_i$, to maintain system stability, and to satisfy the condition

$$\sum_{i=1}^{n} C_i f_i \leq A, \quad 0 < A \leq 1 \tag{7}$$

where C_i, $i = 1, ..., n$, are the task execution times. To maintain system stability, we need to choose the frequency f_i at least as large as the lower bound f_{mi} for each task.

To scheduling the tasks, we consider both dynamic and static priority assignment schemes, e.g., *Earliest Deadline First* (EDF) for dynamic assignment and *Rate-Monotonic Assignment* (RMA) for static assignment. We will choose a range of A such that EDF and RMA can be used for scheduling.

Remark 3.1 The tasks to be scheduled could be a mixture of two types: some may require the periods to be rigidly maintained whereas others do not. If this is the case, we only need to modify A in Eq. (7) by subtracting $\sum_{j \in J} C_j f_j$ from the schedulable utilization with J being the collection of indices of the tasks whose periods are fixed. As far as the determination of the frequencies is concerned, we only consider the tasks which have variable frequencies and performance indices characterized by Eq. (6).

The optimization problem we need to solve is a nonlinear programming problem [4]. More precisely, this problem is:

$$\min_{(f_1,..,f_n)} \Delta J = \sum_{i=1}^{n} w_i \Delta J_i = \sum_{i=1}^{n} w_i \alpha_i e^{-\beta_i f_i} \tag{8}$$

$$\text{subject to: } \sum_{i=1}^{n} C_i f_i \leq A, \quad 0 < A \leq 1 \qquad f_i \geq f_{mi}, \quad i = 1, ..., n \tag{9}$$

where $w_i, i = 1, ..., n$, are a set of weights.

Proposition 3.1 Given the objective function in Eq. (8) and constraints in Eq. (9), there exists a unique optimal solution given by

$$\begin{aligned} f_i &= f_{mi}, \quad i = 1, ..., p \\ f_j &= \frac{1}{\beta_i}(\ln \Gamma_j - Q), \quad j = p+1, ..., n, \end{aligned} \tag{10}$$

where f_ks are ordered as f_{mk} which are arranged as

$$\Gamma_1 e^{-\beta_1 f_{m1}} \leq \Gamma_2 e^{-\beta_2 f_{m2}} \leq \cdots \leq \Gamma_n e^{-\beta_n f_{mn}},$$

$p \in [1, .., n]$ is the smallest integer such that

$$\sum_{l=1}^{p} C_l f_{ml} + \sum_{l=p+1}^{n} \frac{C_l}{\beta_l} \left(\beta_p f_{mp} + \ln \frac{\Gamma_l}{\Gamma_p} \right) \geq A, \tag{11}$$

and

$$\Gamma_j = \frac{w_j \alpha_j \beta_j}{C_j},$$

$$Q = \frac{1}{\sum_{l=p+1}^{n} \frac{C_l}{\beta_l}} \left(\sum_{l=1}^{p} C_l f_{ml} + \sum_{l=p+1}^{n} \frac{C_l}{\beta_l} \ln \Gamma_l - A \right)$$

Proof: See the Appendix.

Remark 3.2 Proposition 3.1 is based on the fact that most control systems can have a flexible sampling frequency, provided this frequency is chosen above the lower bound. This feature was defined as the *control system deadline* and discussed in detail in [8] where the authors studied the effect of missing control updates in control systems. Proposition 3.1 provides a method for optimally determining the sampling frequency at each of the given levels of CPU utilization such that task schedulability is guaranteed. This is illustrated in the following examples.

Example 3.1 Consider an open-loop temperature control problem. Suppose there are five (5) room units whose temperatures need to be automatically controlled by one processor, and controlling the temperature in each unit is considered to be one task. The execution time C_i (ms) and given frequency f_i (Hz) of each task are listed in Table 1.

	C_i	f_i	Utilization (%)
unit 1	10	30	30
unit 2	15	20	30
unit 3	20	20	40
unit 4	25	10	25
unit 5	30	10	30

Table 1. Data for temperature control scheduling

According to the given data, the total utilization will be $1.55 > 1$, and therefore, the tasks are not all schedulable. However, by investigating the underlying physical systems, we may find that these tasks are schedulable with a set of "redesigned" frequencies. Suppose the temperature for each of the units is governed by the dynamic equation

$$\dot{\gamma}_i = -a_i\gamma_i + b_i u_i \tag{12}$$

where $\gamma_i(t)$ is the temperature difference between the i-th unit and the ambiance with $\gamma(0) = 0$; a_i and b_i are constants depending on the insulation of the unit; u_i is the rate of heat (cold air) supplied to the unit. Suppose we need to change the temperature in the i-th unit, and we require such a change to be done in not more than t_f time units and to consume a minimum amount of fuel. Let γ_{di} be the difference between the desired temperature and the ambient temperature. We also require that $|\gamma_i(t_f) - \gamma_{di}| \le \delta_i$. Then, the optimization problem can be formulated as

$$\min_{u_i} J_i = \frac{1}{2}p_i(\gamma_i(t_f) - \gamma_{di})^2 + \frac{1}{2}\int_o^{t_f} u_i^2(t)dt$$

where p_i is a weight coefficient. Then, the continuous-time optimal control and the final state can be determined as

$$u^*(t) = \frac{\gamma_{di}p_i a_i b_i e^{a_i t}}{a_i e^{a_i t_f} + p_i b_i^2 \sinh a_i t_f},$$

$$\gamma_i^*(t_f) = \frac{\gamma_{di}p_i b_i^2 \sinh a_i t_f}{a_i e^{a_i t_f} + p_i b_i^2 \sinh a_i t_f}$$

Digitizing $u^*(t)$ with sampling period T_i and choosing $p_i \gg 1$, we obtain the approximations:

$$\gamma_i^*(t_f) = \gamma_{di}, \quad \text{and} \quad J_{di} = \frac{1}{2}p_i\gamma_{di}^2\left(\frac{1-e^{-a_iT_i}}{1+e^{-a_iT_i}}\right)^2$$

To satisfy the condition $|\gamma_i(t_f) - \gamma_{di}| \le \delta_i$, we need

$$\gamma_{di}\left(\frac{1-e^{-a_iT_i}}{1+e^{-a_iT_i}}\right) \le \delta_i \quad \Rightarrow \quad T_i \le \frac{1}{a_i}\ln\frac{\gamma_{di}+\delta_i}{\gamma_{di}-\delta_i}$$

Furthermore, by approximating

$$1 + e^{-a_iT_i} \approx \frac{3}{2}, \quad 1 - e^{-a_iT_i} \approx e^{-\beta_i f_i}$$

with $\beta_i = 1/a_i$ and $f_i = 1/T_i$, we finally obtain

$$\Delta J_i = \frac{2}{3}e^{-\beta_i f_i} \quad \text{with} \quad f_{mi} = \frac{a_i}{\ln\frac{\gamma_{di}+\delta_i}{\gamma_{di}-\delta_i}}$$

For scheduling, we consider both EDF and RMA approaches. To use EDF, we set $A = 1$. For RMA, A must be chosen so that the tasks can all be scheduled by the chosen scheduling algorithm. Suppose the physical parameters are given in Table 2, where f_{mi} (Hz) is the lower bound on the sampling frequency of task i, and w_i is the weight assigned to task i.

	β_i	C_i	f_{mi}	w_i
unit 1	0.3	10	20	1
unit 2	0.4	15	12.5	2
unit 3	0.5	20	10	3
unit 4	0.6	25	6	4
unit 5	0.7	30	4	5

Table 2. Data for temperature control tasks

EDF Scheduling: We first check whether the task set is schedulable when all task frequencies are set to their minimum values. We find that

$$\sum_{i=1}^{5} C_i f_{mi} = 0.8575 < 1$$

which indicates that all five of the tasks are schedulable. Following the algorithm in Proposition 3.1, we determine the optimal frequency for each task as following. Let

$$F(p) = \sum_{i=1}^{p} C_i f_{mi} + \sum_{i=p+1}^{5} \frac{C_i}{\beta_i}\left(\beta_p f_{mp} + \ln\frac{\Gamma_i}{\Gamma_p}\right)$$

Then, a simple calculation shows

$$F(5) = 0.8875 < 1, \quad F(4) = 0.9 < 1, \quad F(3) = 1.04 > 1$$

Therefore, we will assign

$$f_i = f_{mi}, \quad \text{for } i = 1, 2, 3,$$
$$\text{i.e. } f_1 = 20 \text{ Hz}, \quad f_2 = 12.5 \text{ Hz}, \quad f_3 = 10 \text{ Hz},$$

and compute f_4 and f_5 as

$$f_4 = (\ln\Gamma_4 - Q)/\beta_4 = 7.97 \text{ Hz}$$
$$f_5 = (\ln\Gamma_5 - Q)/\beta_5 = 7.1 \text{ Hz}.$$

This choice of frequencies yields a total utilization 99.97%, and the final set of tasks is schedulable.

RMA Scheduling: Let t_c (ms) be the completion time of the first invocation of a periodic task and D (ms) be

its deadline. For utilization $A = 1$, $A = 0.86$, and $A = 0.8575$, we compute the optimal frequencies for each unit by the algorithm described in Proposition 3.1 and list their deadlines in Table 3. By following Theorem 2 and the completion time test in [6], we obtain the completion time of each task for all three utilizations as shown in Table 3.

	$A=1$		$A=0.86$		$A=0.8575$	
	t_c	D	t_c	D	t_c	D
unit 1	10	50	10	50	10	50
unit 2	25	80	25	80	25	80
unit 3	45	100	45	100	45	100
unit 4	80	125	80	166	80	166
unit 5	155	141	235	245	235	250

Table 3. RMA scheduling table for temperature control tasks

The data in the above table shows that not all tasks can be scheduled by the RMA approach if $A = 1$, but all of them can be scheduled for $A = 0.86$ or $A = 0.8575$. When $A = 0.86$ $f_i = f_{mi}$, $i = 1, .., 4$, $f_5 = 4.0833$, and when $A = 0.8575$, $f_i = f_{mi}$, $i = 1, .., 5$.

By comparing two scheduling approaches presented above, we conclude that full utilization of computing resource can be achieved by using EDF and this yields a performance index $\Delta J = 0.00695$. RMA scheduling algorithm, on the other hand, does not guarantee full CPU utilization, and the performance index is $\Delta J = 0.2882$ at $A = 0.86$.

Example 3.2 Consider the bubble control system discussed in the last section. Suppose four such systems with different physical dimensions are installed on an underwater vehicle to control the depth and orientation of the vehicle, and they are controlled by one on-board processor. For each bubble control system bi, let C_i (ms) be the control task execution time in each sampling period, f_{mi} (Hz) be the lower bound on sampling frequency, and w_i be the weight assigned to system i. The following data are given for the control design and scheduling problem: $\Delta J_i = \alpha_i e^{-\beta_i f_i}$, $i = 1, ..., 4$, and

	α_i	β_i	C_i	f_{mi}	w_i	f_i
b1	1	0.5	10	15	5	-
b2	1	0.7	10	10	3	-
b3	1	0.3	10	18	2	-
b4	1	0.1	10	20	1	-
coordinator	-	-	5	-	-	10

Table 4. Data for bubble control scheduling

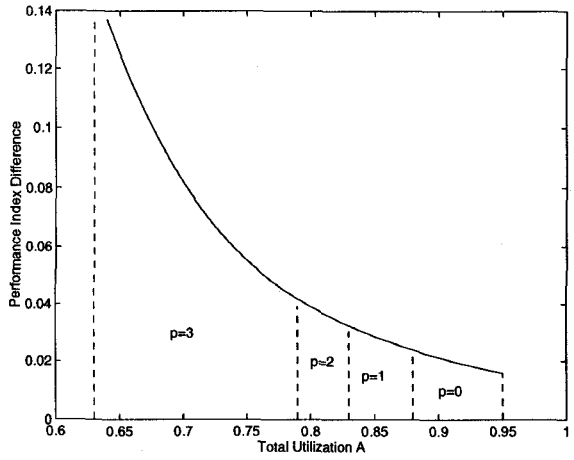

Figure 4. ΔJ versus total utilization A

where the frequencies, f_i (Hz), $i = 1, .., 4$, must be determined. The coordinator is the control unit which coordinates the bubble subsystems to perform a desired mission, for instance, changing depth, roll or pitch angles. The frequency for this particular task is fixed, and it is determined by the underlying coordination requests. For example, to keep the roll angle θ_r within a certain range, say $|\theta_r| < \theta_{rm}$, we may want to choose the task period to be $\theta_{rm}/\max(|\dot{\theta}_r|)$. A simple calculation shows that the total CPU utilization of the overall bubble system is 63% when the minimum task frequencies are assigned, and the total CPU utilization available for the bubble systems is 95%, excluding the utilization of the coordinator. These imply that for any $A \in [0.63, 0.95]$, all the tasks are schedulable with EDF, using the task frequencies determined by the algorithm described in Proposition 3.1. Fig. 4 shows the relation between $\Delta J = \sum_{i=1}^{4} w_i \alpha_i e^{-\beta_i f_i}$ and the value of A.

Again, let

$$F(p) = \sum_{i=1}^{p} C_i f_{mi} + \sum_{i=p+1}^{4} \frac{C_i}{\beta_i} \left(\beta_p f_{mp} + \ln \frac{\Gamma_i}{\Gamma_p} \right),$$

then we get

$$F(4) = 0.63, \quad F(3) = 0.7908,$$
$$F(2) = 0.8371, \quad F(1) = 0.8852$$

Hence we conclude (also as indicated in the figure): when $A \in [0.8852, 0.95]$, there is no frequency chosen to be the corresponding lower bound. When $A \in [0.8371, 0.8852)$, there is one frequency that needs to assume its minimum value, $f_1 = 15$ (Hz). When $A \in [0.7908, 0.8371)$, two frequencies are set to their

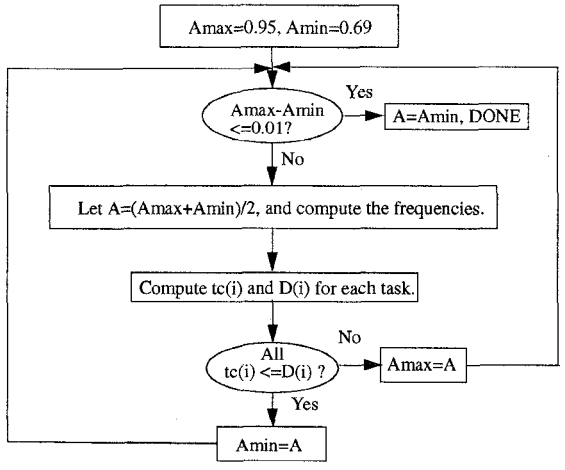

Figure 5. Flowchart for binary search for an optimal value of A

minimum value, $f_1 = 15$ (Hz), and $f_2 = 10$ (Hz). When A is reduced further to the range $(0.63, 0.7908)$, three frequencies must be set to their lower bounds, namely, $f_1 = 15$ (Hz), $f_2 = 10$ (Hz), and $f_3 = 18$ (Hz). Finally, when $A = 0.63$, all the frequencies must be assigned their minimum values.

We now consider scheduling the tasks by the RMA approach. While all the tasks can be scheduled by the EDF algorithm, for some value of $A \in [0.63, 0.95]$, this may not be true in case of RMA scheduling. For the given range of A, we would like to schedule all the tasks using RMA with the largest value of A, in which case we can get the best possible performance. To achieve this, we use a binary search to determine the value of A that will give a near-optimal solution. This search may not guarantee the convergence to the optimal A, because the frequencies vary for different A, and RMA schedules tasks based on the tasks' frequencies. The binary search is illustrated in Fig. 5, and the search results are listed in Table 5 with t_c (ms) the task completion time and D (ms) the task deadline. Apparently, the solution for RMA scheduling should be obtained by setting $A = 0.91$ and they are

$$f_1 = 15.3 \text{ Hz}, \quad f_2 = 10.68 \text{ Hz},$$
$$f_3 = 20.74 \text{ Hz}, \quad f_4 = 44.29 \text{ Hz}$$

Again, we found the performance indices $\Delta J = 0.0157$ at $A = 0.95$ by the EDF scheduling algorithm and $\Delta J = 0.02$ at $A = 0.91$ by the RMA scheduling algorithm.

	$A = 0.82$		$A = 0.89$		$A = 0.92$		$A = 0.91$	
	t_c	D	t_c	D	t_c	D	t_c	D
b4	10	26	10	23	10	22	10	23
b3	20	53	20	49	20	48	20	48
b1	40	67	40	66	40	65	40	65
b2	50	100	90	95	100	93	90	94

Table 5. RMA scheduling table for bubble control systems

Remark 3.3 Examples 3.1 and 3.2 demonstrate that for real-time control systems the choice of processors and scheduling algorithms can be important. It is clear that the faster the processing speed and the higher the schedulable utilization, the better will be the system performance that can be obtained. For a slow processor and low levels of schedulable utilization, great gains in the overall performance can occur from relatively small increases in computing speed or schedulable utilization. Consequently, the increases in schedulable utilization available from using EDF rather than RMA can significantly improve performance, as was the case in Example 3.1. However, a point of diminishing marginal returns will be reached. At this point, increases in processing speed and schedulable utilization begin to contribute relatively little to overall system performance. Consequently, EDF will offer only slightly better performance than RMA. This is illustrated in Example 3.2. These observations indicate that in designing control systems, it is very important to choose a processor speed for which the task set frequencies can be chosen above the point of decreasing marginal returns for all tasks. If this is done, then the choice of scheduling algorithms will be a relatively small (second-order) effect.

4. Conclusion

In this paper, we have considered the periodic task schedulability issue from a new viewpoint, the optimization of a real-time control system's performance. For a set of tasks from a class of control systems whose performance indices with digital control implementations are a monotonically decreasing and convex function of each task's frequency, an algorithm is proposed to determine the task frequencies such that all tasks are schedulable for both EDF and RMA.

The main contribution of this paper is two-fold. First, the proposed algorithm enhances the processor's task schedulability by allowing the task frequencies to change in the context of real-time control system performance. For a given set of tasks, they may not be schedulable with the given task periods, but they may

be schedulable if the periods are changeable and they are schedulable when their frequencies take values on their lower bounds. If this is the case, the algorithm guarantees that the tasks will be schedulable and the system performance with a digital control implementation will be optimized subject to the limitation on the computing resources. Second, even for a set of tasks which are originally schedulable, the proposed algorithm can still be used to improve the overall system performance. This feature distinguishes our algorithm from the others where system performance was not considered to be a factor in scheduling the tasks.

Future work along this line will include generalizing the algorithm developed here to address the problem of task re-allocation over a network of processors and optimizing the overall system performance.

References

[1] Gerber, R., Hong, S. and Saksena, M., "Guaranteeing end-to-end timing constraints by calibrating Intermediate Processes," *Proceedings of the IEEE Real-Time Systems Symposium*, December, 1994.

[2] Liu, C. L. Layland, J.W., "Scheduling algorithms for multiprogramming in a hard real-time environment," *Journal of Association for Computing Machinery*, Vol. 20, No.1, January 1973, 46-61.

[3] Locke, C. D., "Best-effort decision making for real-time scheduling." Ph.D. Dissertation, Computer Science Department, Carnegie Mellon University, 1986.

[4] Mangasarian, O. L., *Nonlinear Programming.* McGraw-Hill Book Company, 1969.

[5] Molini, J. J., Maimon, S. K. and Watson, P. H., "Real-Time System Scenarios," *Proceedings of the IEEE Real-Time Systems Symposium*, December, 1990.

[6] Sha, L., Rajkumar, R. and Sathaye, S. S., "Generalized rate-monotonic scheduling theory: A framework for developing real-time systems," *Proceedings of the IEEE*, Vol. 82, No. 1, January, 1994, 68-82.

[7] Shin, K. G., Krishna, C. M. and Lee, Y.-H., "A unified method for evaluating real-time computer controllers and its application," *IEEE Transactions on Automatic Control*, Vol. 30, No. 4, April, 1985, 357-366.

[8] Shin, K. G. and H. Kim, H., "Derivation and application of hard deadlines for real-time control systems," *IEEE Transactions on Systems, Manufacturing, and Cybernetics*, Vol. 22, No. 6, November/December, 1992, 1103-1413.

Appendix

Proof of Proposition 3.1: Introducing Lagrangian multipliers λ, λ_i, $i = 1, ..., n$, we write the Kuhn-Tucker condition as

$$\begin{cases} -\Gamma_i e^{-\beta_i f_i} + \lambda - \lambda_i / C_i = 0 \\ \sum_{i=1}^{n} C_i f_i - A \leq 0, \quad f_{mi} - f_i \leq 0 \\ \lambda \left(\sum_{i=1}^{n} C_i f_i - A \right), \quad \lambda_i (f_{mi} - f_i) = 0, \\ \lambda, \ \lambda_i \geq 0 \end{cases}$$

where $i = 1, ..., n$. Since $\lambda_i \geq 0$, $\Gamma_i > 0, \forall i$, we have

$$\lambda > 0, \quad \text{and} \quad \sum_{i=1}^{n} C_i f_i = A \tag{13}$$

Letting $f_i = f_{mi}$, $i = 1, ..., n$, then by schedulability assumption, we have

$$\sum_{i=1}^{n} C_i f_i < A$$

We now increase the frequencies starting with f_n. For each $f_i > f_{mi}$, we have $\lambda_i = 0$,

$$\lambda = \Gamma_i e^{-\beta_i f_i} \tag{14}$$

and λ decreases as f_i increases. Also, each f_i has to start increasing from f_{mi} when $\lambda < \Gamma_i e^{-\beta_i f_{mi}}$, otherwise equation

$$-\Gamma_i e^{-\beta_i f_i} + \lambda - \lambda_i / C_i = 0$$

would not hold for $\lambda_i \geq 0$. To determine which f_is are greater than f_{mi} and how much they can increase, we need to find the smallest integer p such that

$$\sum_{i=1}^{p-1} C_i f_{mi} + C_p f_{mp} + \sum_{i=p+1}^{n} \frac{C_i}{\beta_i} (\ln \Gamma_i - \ln \lambda) > A$$

with $\lambda = \Gamma_p e^{-\beta_p f_{mp}}$, which is equivalent to

$$\sum_{i=1}^{p} C_i f_{mi} + \sum_{i=p+1}^{n} \frac{C_i}{\beta_i} \left(\beta_p f_{mp} + \ln \frac{\Gamma_i}{\Gamma_p} \right) \geq A$$

as given in the proposition. After the integer p has been identified, we conclude that the frequencies $f_1, ..., f_p$ must be chosen to be $f_{m1}, ..., f_{mp}$ and the remaining frequencies are computed from Eqs. (13) and (14). Since the objective function in Eq. (8) is strictly convex, there is always one and only one minimum. Therefore, a unique set of optimal task frequencies is obtained.