IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 28, 2017, accepted October 25, 2017, date of publication November 1, 2017, date of current version November 28, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2768559

A Heuristic Task Periods Selection Algorithm for Real-Time Control Systems on a Multi-Core Processor

HONGYA FU, JIANKANG LIU, ZHENYU HAN, AND ZHONGXI SHAO
School of Mechatronics Engineering, Harbin Institute of Technology, Harbin 150001, China

Corresponding author: Zhongxi Shao (lordopt@163.com)

This work was supported in part by the National Science and Technology Major Project of China under Grant 2013ZX04013-011-09 and in part by the China Post-Doctoral Science Foundation under Grant LBH-Z13209.

ABSTRACT The performance optimization problem is investigated for discrete-time control systems on a multi-core platform. An integrated approach which considers both control performance and real-time scheduling aspects is applied to allocate optimal periods to controller tasks. A real-time control system is modeled as a set of directed acyclic graphs with weighted edges in this paper. The system allows producer/consumer relationship between tasks, and the data dependence relationships between tasks are uncoupled by attaching harmonic constraints to task periods. The period assignment problem is formulated as an optimization problem, which minimizes the system performance loss index under multi-core schedulability constraints. A heuristic search algorithm is proposed to solve this optimization problem and select periods for real-time tasks scheduled by rate-monotonic scheduling algorithm. Experimental results demonstrate that the proposed heuristic algorithm is capable of finding a high quality local optimal solution with fast computing speed. The proposed method is applicable to online failure recovery and reconfiguration in real-time control systems.

INDEX TERMS Control systems, multi-core processor, performance optimization, period selection, real-time control system.

I. INTRODUCTION

Modern discrete-time control systems, such as avionics system, automotive electronics and robot manipulator, are usually characterized by high complexity, high-integrity and high-order nonlinearity. The control algorithm is normally implemented as a set of periodic tasks with real-time constraints. The design and optimization of such systems should consider both control aspects and real-time execution platform aspects at the same time. Recently, a lot of work has been focused on complex nonlinear systems with highly uncertainties and dynamic disturbances. Learning system (such as neural network, fuzzy logic, heuristic, etc.) based approaches are proved to be particularly useful in designing such systems (see [1]–[6] and the references therein). Their common basic conception is to ensure the system output is converged to given trajectories by adopting the approximation ability of neural networks and other adaptive techniques. For discrete-time systems in practical engineering, considering execution platform aspects, the output tracking performance and system stability are often degraded because of the existence of time delays. Therefore, a lot of work has been focused on control design of time-delay systems, and there have been many interesting results reported in [7]–[10]. For example, in [7], to handle nonlinear dynamical systems with time delays, an adaptive neural controller is systemically designed which guarantees the semi-globally uniform boundedness of all closed-loop signals. Apart from the work focused on the control design of time-delay systems, the optimization of the control system has also received a great deal of attention [11], [12]. As discussed in [13], performance of a discrete-time control system depends heavily on its sampling frequency. With more frequent sensing and actuation, more accurate control results can be obtained. Since a shorter task execution period implies shorter input-output delay, which means more recent sensing data have been used to produce the actuation value, it can provide higher control performance.

On the other hand, although the system performance can be enhanced by assigning the control tasks with shorter periods, we have to notice the fact that the set of tasks resulting from this controller 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 full use of computing resources. Hence the control performance and real-time schedulability have a trade-off relation in terms of practical implementation [14]. As a consequence, the design of the whole system should consider both the control aspects and the real-time scheduling aspects at the same time. Such an integrated approach is referred to as control/scheduling co-design, and a lot of work has been focused on it [14]–[32]. A detailed discussion on related works is presented in the next section. It is observed that most of the prior works have been focused on uniprocessor platform with independent task model. The period selection problem for data dependent tasks on a multi-core processor has gained little attention.

Since multi-core architecture has improved processor computing power and parallel processing capacity, multiple real-time applications can be integrated into a single multi-core processor platform, which conserves hardware resources and improves the overall system performance. With the increasing of integration and complexity, the design and optimization of multi-core processor based control system has become a more challenging work.

Motivated by the aforementioned observations, we study the control system performance optimization problem on a multi-core platform from a perspective of task period assignment in this research. Compared with the existing works on period selection problem, the main features of this paper lie in the facts that: 1) the control system considered allows data dependence relationships between tasks and runs on a multi-core platform and 2) compared with the existing period assignment methods, the proposed heuristic is with lower computational complexity and suitable to be implemented for on-line use.

The rest of this paper is structured as follows. Section 2 introduces some background knowledge and related works. Section 3 describes the system model and assumptions used throughout this paper. A formal description of our problem is presented in Section 4. Section 5 presents the proposed heuristic algorithm for the period selection problem. The experimental results are presented in Section 6. Finally, Section 7 concludes this paper.

II. BACKGROUND AND RELATED WORKS

A. SCHEDULABILITY CONSTRAINTS

To guarantee the timing requirements in real-time systems, multiple periodic tasks are usually scheduled with a real-time scheduling policy, such as rate monotonic scheduling (RM) or earliest deadline first scheduling (EDF) [33]. Each scheduling policy has its corresponding schedulability analysis method.

Historically, there have been two distinct approaches for schedulability test: tests based on the notion of processor utilization and tests based on response times [34]. Utilization is proven to be a sufficient measure of the schedulability of real-time systems [33]. For multi-core processor systems, there are several utilization bounds proposed under RM scheduling policy:

- (Andersson *et al.* [35]) A task set can be successfully scheduled on a m-processor with rate monotonic scheduling if the utilization of every individual task do not exceed $m/(3m-2)$, and the total utilization is at most $m^2/(3m-1)$.
- (Baruah and Goossens [36]) Any periodic task system in which each taskąŕs utilization is no more than $1/3$, and the total utilizations of all the tasks is no more than $m/3$, is successfully scheduled by Algorithm RM upon m-processors.
- (Baker [37]) On a m-core processor, when deadline equals period and priorities are rate monotonic, any set of tasks with maximum individual task utilization u_{max} and minimum individual task utilization u_{min} is feasible if the total utilization does not exceed $m(1-u_{max})/2+u_{min}$.

B. PERFORMANCE INDEX

The performance of a real-time control system can be reflected by various evaluation indexes. Traditionally, transient response time and steady-state accuracy [38], or system error [23] are selected as the performance index to evaluate controllers. In some cases, other evaluation indexes such as energy consumption can also serve as the performance index.

The linear-quadratic-Gaussian (LQG) control problem is one of fundamental optimal control problems, and the performance of LQG controller tasks is formulated as a quadratic cost function Eq. (1) by Astrom and Wittenmark [39],

$$J = E\lim_{t_p\to\infty}\frac{1}{t_p}\int_0^{t_p}(x'Q_1x+u'Q_2u)dt \tag{1}$$

where x and u denote the state vector and the control vector, and t_p is the maximum time to be considered in the performance evaluation. Q_1,Q_2 are weighting matrices, and E denotes the expectation operator. J can be interpreted as the weighted sum of stationary variance of the plant state and the control signal. Higher value of J indicates worse control performance, so it is referred to as the performance loss index [19]. For discrete-time control, Eq.(1) can be written as a monotonic increasing function of the sampling period T:

$$J = J(T)$$

In some case, it has been proven that the cost-function Eq. (1) can be approximated by a quadratic function of the sampling period [40]

$$J \approx \alpha + \beta T + \gamma T^2 \tag{2}$$

or even a linear function of the sampling period [41]

$$J \approx \alpha + \beta T$$

C. RELATED WORKS

For real-time control systems, the co-design problem was initially formalized as an optimization problem by Seto *et al.* [14]. The system performance index is expressed as an exponential decay function, and period is selected as the optimization variable which is restricted within its feasible region to guarantee system schedulability. The proposed method is built on dynamic priority scheduling. Later, in [15], they presented another algorithm based on fixed priority scheduling methods to select optimal periods for the task set.

The optimization method proposed by Seto *et al.* [14] is further extended by many researchers. Palopoli *et al.* [16] developed optimal period assignment for a set of state feedback controllers using the stability radius as a performance criterion. Bini and Natale [18] provided a search-based algorithm that finds the task activation rates maximizing a performance function within the deadline constraints in systems scheduled by fixed priorities. Wu *et al.* [19] presented a methodology that selects task periods and deadlines under feasibility constraints. The interference generated by the concurrent execution of multiple tasks was considered when formalizing the optimization problem upon the convex approximation of EDF (Earliest Deadline First scheduling) deadline space.

The above methods of the control/scheduling co-design problem are time consuming and can be utilized off-line only. In order to improve the flexibility and robustness of the system, later work has moved the optimization algorithm from off-line to on-line use. Cervin *et al.* [21] proposed an on-line sampling period adjustment method based on estimates of current plant states and noise intensities. The optimization method is based on the expressions relating the expected cost over a finite horizon to the sampling period, the computational delay, and the amount of noise acting on the plant. Du *et al.* [22] proposed an analytic solution for schedulable situation of the optimization problem using the method of Lagrange multipliers. And an integrated method is proposed for the condition that the system is overloaded which has a deterministic time complexity and is suitable for on-line use. Recently, Cha *et al.* [23] proposed search-based heuristic algorithm which finds near-optimal feasible periods maximizing the overall control performance. Their algorithm has a linear complexity.

In the literature, period selection problem has been studied not only for control systems but also for general real-time systems. The most commonly used method to conduct period assignment is to assign harmonic periods to the tasks, because RM can guarantee 100% utilization if the periods are harmonic [24]. Also, harmonic period values can generate smaller hyperperiod which is calculated as the least common multiple (LCM) of the tasks' periods. The hyperperiod is the interval at which the entire schedule is repeated, the shorter the hyperperiod the shorter the scheduling table, and consequently the smaller the memory footprint. Nasri *et al.* [25], [26] presented a model to describe harmonic relations between ranges of period values, rather than between discrete numbers. Harmonic periods are assigned to the tasks by finding harmonic sub-intervals within given period ranges of the task set. Xu [27] proposed a method for adjusting the periods of periodic tasks to reduce the least common multiple (LCM) of the period lengths. Adjusted period lengths are closely harmonically related to each other, which makes it easier to generate a pre-run-time scheduling to schedule periodic tasks. Brocal *et al.* [28] and Ripoll and Ballester-Ripoll [29] both studied the method of minimizing hyperperiod when assigning periods to a set of periodic tasks.

The common assumption of the above researches about period assignment problem is that the tasks run independently on a uniprocessor platform. In practice, however, tasks in a real-time system have a variety of interactive relationships, such as producer/consumer relation [30]. Gerber *et al.* [31] considered a system with tasks connected by asynchronous channels, in which the endpoints are the system's external inputs and outputs. The system is rendered in an asynchronous task graph format with end-to-end timing constraints on its inputs and outputs. Periods are allocated to the tasks according to the end-to-end constraints.

Despite the above researches, the period assignment problem on a multi-core processor has gained little attention even though it is in a position to enhance the system computing power. Moreover, the data dependences between control tasks and other general real-time tasks were not considered in the period selection problem. In this paper, we address the period assignment problem on a multi-core processor. We consider a real-time control system consisted of a set of tasks connected by asynchronous channels.

III. SYSTEM MODEL AND ASSUMPTIONS

In this paper, a multi-core platform consists of m identical cores, $m \geq 2$, denoted as $\mathcal{P} = \{P_1, P_2, \ldots, P_m\}$. The system task set consists of N periodic tasks, denoted as $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_N\}$. Each task has three parameters (C_i, T_i, p_i). C_i is the worst case execution time of τ_i, T_i is the inter-arrival time (period) between any two consecutive jobs of τ_i, and p_i is the priority of τ_i. $u_i = C_i/T_i$ is the processor utilization of task τ_i, and $U = \sum_{i=1}^{n} u_i$ is the system total utilization. We assume that the deadline of a task is equal to the period.

To represent the data dependencies between tasks, Gerber *et al.* [31] denoted a real-time system as a directed acyclic graph $G(\Gamma, E)$, where Γ is the task set and E is a set of directed edges between tasks. An edge $\tau_i \rightarrow \tau_j$ denotes a producer/consumer relationship where τ_i produces data that τ_j consumes.

In this paper, the concept of directed acyclic graphs in real-time systems is further refined. In practice, it is common that the producer generates a plurality of data in one execution, and the consumer needs to execute multiple times to consume these data. For each producer/consumer pair we define a data transfer ratio which is denoted as r. A weighted edge $\tau_i \xrightarrow{r=x} \tau_j$ denotes that τ_i produces x data in one execution for τ_j to consume, where x is a positive, non-zero integer. In this

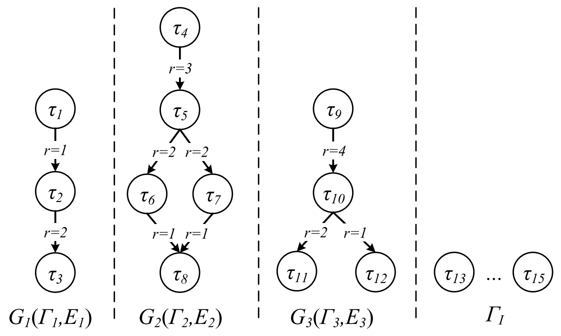


FIGURE 1. An example of real-time system.

work, we assume that the real-time data flow is paramount, which means that every data produced by the producer should be processed by the consumer in real-time. With this consideration, in order to ensure that the two tasks stay "in-phase", each producer/consumer pair $\tau_i \xrightarrow{r=x} \tau_j$ is constrained to have harmonic periods such that T_i is an integer multiple of T_j, and $T_i/T_j = x$.

A real-time system is represented by a set of irrelevant directed acyclic graphs $\{G_1, G_2, \ldots, G_N\}$ and a set of independent tasks Γ_I. Each task graph $G_i(\Gamma_i, E_i)$ is treated as a subsystem of the real-time control system, where Γ_i is a subset of the system task set, thus $\Gamma = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_n, \Gamma_I\}$. Figure 1 shows an example real-time system consists of three task graphs and a set of independent tasks.

In the remainder of this paper, the following additional notations are used.

Γ_{tail}	A set of tasks having no consumer tasks. They have no outgoing edges in the task graph.
Γ_{key}	A set of key tasks whose performances can represent a subsystem's performance profile. They are manually assigned with period constraints at the initial design stage. $\Gamma_K \subset \Gamma_{tail}$.

It is assumed that the tail tasks $\tau_i \in \Gamma_{tail}$ are some typical control tasks (i.e. closed loop control tasks), and the non-tail tasks are designed to produce necessary data for the control tasks. Usually, the period of a control task is constrained into an acceptable range during the design phase. However, if there is more than one tail task in a subsystem, it will cause conflict if all the tail tasks are manually assigned with period ranges due to the harmonic constraints. Note that in a subsystem task graph, once a tail task's period is determined, other tasks' periods can be calculated according to the harmonic relations between them. So only one tail task in a subsystem task graph is selected to be constrained with a period range, the other tasks' periods should be calculated according to the harmonic constraints. The selected task should have the most critical timing constraint in the subsystem, and the subsystem's performance can be represented by this task. To be specific, the periods T_i of each control task $\tau_i \in \Gamma_{key}$ are constrained into intervals (i.e. $\mathbf{T}_i = [T_i^{min}, T_i^{max}]$), where T_i^{min}, T_i^{max} are derived from the performance requirements.

IV. PROBLEM DESCRIPTION

This section gives a formal definition of the period selection problem. The following scenario is considered: On a multi-core processor platform, for the task set Γ of N real-time tasks where the periods T_i of key tasks $\tau_i \in \Gamma_{key}$ is given as a range $[T_i^{min}, T_i^{max}]$, find the optimal period for each task such that the performance loss index is minimized under schedulability constraints.

A. PERFORMANCE FUNCTION

The quadratic approximation cost function Eq.(2) is selected as the performance metric for each tail task $\tau_i \in \Gamma_{tail}$. In particular, we assume that $J_i(T_i)$ is a monotonous with respect to T_i when $T_i > 0$. The global system performance loss index is defined as a weighted sum of J_i as in the following:

$$J = \sum_{\tau_i \in \Gamma_{tail}} w_i J_i(T_i) \tag{3}$$

where w_i is a user-defined constant factor which denotes the importance of a task.

B. CONSTRAINTS

The period of a controller task is subject to constraints from both control theorem and real-time scheduling theory. The period range $[T_i^{min}, T_i^{max}]$ of a key task $\tau_i \in \Gamma_{key}$ is generated according to the required control performance, such as transient response and steady-state accuracy. For the entire task set Γ, periods are constraint by multi-core utilization bound to guarantee schedulability. With RM scheduling policy adopted, individual task utilization and system total utilization are limited within one of the schedulability bounds described above.

C. OPTIMIZATION PROBLEM

Summarizing the discussion above, the period selection problem in this work is formulated as a nonlinear constraint optimization problem Eq.(4).

$$min\ J = \sum_{\tau_i \in \Gamma_{tail}} w_i J_i(T_i)$$
$$s.t. \begin{cases} C_i < T_i & \\ T_i^{min} \le T_i \le T_i^{max} & (\tau_i \in \Gamma_{key}) \\ U \le U_{bound} & (\tau_i \in \Gamma) \end{cases} \tag{4}$$

V. THE OPTIMIZATION ALGORITHM

To find T_i with the minimum J, the simplest method is to explore the entire feasible domain for every task combination while checking the schedulability and calculating the performance function. However, this exhaustive search method is not applicable even with a small number of tasks due to its high complexity. Instead, we propose a heuristic search algorithm that finds an acceptable suboptimal solution with a very low computational complexity.

A. ANALYTIC SOLUTION AND NUMERICAL METHOD

As described above, the period selection problem is formalized as a nonlinear constrained multi-variable optimization problem. This kind of problem is soluble analytically using the method of Lagrangian multipliers by introducing relaxation factors to transform inequality constraints to equality constraints. However, when the number of tasks is large, it is very hard to solve the Lagrange equations. Therefore, numerical methods are introduced to solve this kind of problem.

There are also some search-based numerical methods for this kind of optimization problem, such as gradient descent method, simulated annealing, and genetic algorithm (GA) [32]. These techniques are available in some optimization modeling software such as MATLAB and LINGO. These numerical methods are very effective, but the problems are always solved offline. For example, LINGO has a set of powerful built-in solvers for linear, nonlinear optimization problems, but they cannot be integrated into a real-time control system.

In the case that a part of the system is reconstructed during runtime (e.g. the data transfer ratios are changed), the tasks' periods should be re-optimized in real-time. A fast and practicable heuristic search algorithm is proposed in the next subsection.

B. HEURISTIC ALGORITHM

In the optimization problem Eq.(4), the performance loss index J is expressed as a summation of all the control tasks' cost functions, while the cost function of a control task is modeled as a monotone increasing function of period. The optimization objective function is monotonous with respect to each task's period. As a result, the system performance loss index can be minimized by narrowing the periods of each task iteratively. A heuristic search algorithm is developed taking this observation into account.

In the first step of the heuristic algorithm, each task's period is initialized with the maximum value in the given range: $T_i^0 = T_i^{max}$, which means that the initial solution has the highest performance loss index. Then the heuristic algorithm iteratively decreases the periods of each task within the schedulability constraint.

The core of this heuristic algorithm is the rule of thumb for reducing the periods so as to minimize the system performance loss index. In the searching progress, a feasible solution is transferred to the next one following a direction which guarantees a decrease of the performance loss index. Notice that J has different partial derivatives for each T_i. Due to the property $\partial J/\partial T_i > 0$, a decrease of any task period will always achieve a lower performance loss index. With the decrease of separate tasks' period T_i, the value of J is reduced at different rates. For example, Figure 2 shows the cost functions of two tasks, where the current periods of task 1 and 2 are 4ms and 3ms. It's obvious that the cost function J has a larger partial derivative with respect to T_1. Compared to T_2, decreasing 1 unit time of T_1 could lead to a

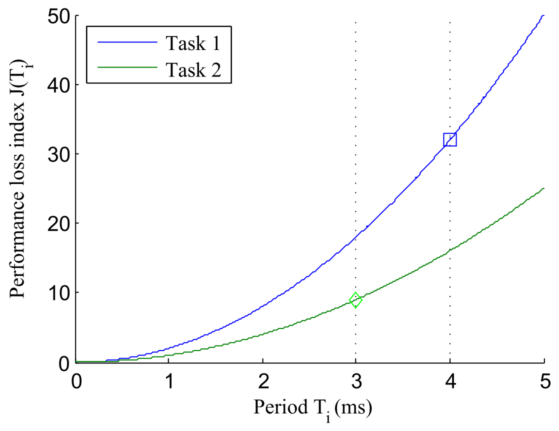


FIGURE 2. Example cost functions of two controller tasks.

greater degradation of J. So it is more effective in reducing the system cost J to decrease the period of a task which has larger partial derivative $\partial J/\partial T_i$ preferentially.

The cost function J is a monotone increasing with respect to every T_i. During the iteration, J is decreased along with the descending of T_i, and the system utilization is increased and converged on the schedulability bound. Baker's bound also changes with decreasing T_i because it is determined by u_{max} and u_{min}. u_{max} and u_{min} are increased as the number of iterations increases, and higher u_{max} can result in lower schedulability bound. Which means that if u_{max} is increased, the upper bound of the system total utilization is decreased, and there is less allowance for the task periods to be decreased. As a consequence, the searching progress probably converges to a solution which is far from being optimal. To combat this, u_{max} should not be increased during the iterations. On the other hand, higher u_{min} leads to higher schedulability bound, so tasks with lower utilization should decrease their periods more quickly.

Consequently, the rule of thumb in this heuristic search algorithm can be summarized as follows:

- Task with lager partial derivative $\partial J/\partial T_i$ is given preference to reduce its period T_i more quickly.
- Tasks with lower utilization are given preference to decrease their periods more quickly.
- The decrease of periods should avoid increasing of the maximum task utilization u_{max}.

The cutoff condition for iteration is that the utilization of the current solution is close enough to the schedulability bound. Decreasing amplitude of each task's period for the next iteration, to a large extent, is determined by the difference between current utilization and schedulability bound.

Let δ_U denote the difference between current utilization and bound.

$$\delta_U^n = U_{bound}^{n-1} - U^{n-1} \quad (n = 1, 2, \ldots) \tag{5}$$

where U^n is the system total utilization of n-th iteration, and $U_{bound}^n = m(1 - u_{max})/2 + u_{min}$ is Baker's schedulability bound.

In each iteration, the new period of key task τ_i is calculated using Eq.(6)

$$T_i^n = \frac{C_i}{u_i^n} \quad (i \in \{i|\tau_i \in \Gamma_{key}\}) \tag{6}$$

where u_i is increased iteratively using Eq.(7)

$$u_i^n = (1 + e\delta_U^n)u_i^{n-1} \tag{7}$$

where e is a proportional coefficient. Eq.(7) ensures that every task period is reduced and the total utilization is converged to the schedulability bound when $\delta_U \to 0$.

However, the solution is obtained by decreasing each task's period averagely and yet it is not optimized. In order to apply the rule of thumb described above, e is expanded as a function which is directly proportional to J'_{T_i} and inversely proportional to u_i, as shown in Eq.(8)

$$e_i^n = c_i \frac{J'_{T_i}}{max(J')} \frac{max(u^{n-1})}{u_i^{n-1}} \tag{8}$$

where c_i is a constant coefficient. $J'_{T_i} = \partial J/\partial T_i$ and $max(J') = max(J'_{T_1}, J'_{T_2}, \ldots J'_{T_N})$. u_i^n is the utilization of τ_i in n-th iteration. $max(u^n) = max(u_1^n, u_2^n, \ldots, u_N^n)$.

In order to prevent the schedulability bound from being decreased quickly, T_i of the task which has the maximum utilization is kept unchanged by setting e_i to 0. Let τ_j denote the task with max utilization, then $u_j = max(u_1, u_2, \ldots, u_N)$. If $\tau_j \in \Gamma_{key}$, let $e_j = 0$. However, if $\tau_j \notin \Gamma_{key}$, T_j is determined by the corresponding key task (τ_k) of the subsystem, to avoid u_j from being enlarged, T_k should remain unchanged in the current iteration (let $e_k = 0$). For example, in the sample system shown in Figure 1, τ_{11} is selected as the key task of subsystem $G_3(\Gamma_3, E_3)$. If τ_{10} has the maximum utilization in the current iteration, to prevent T_{10} from being decreased, e_{11} should be set to 0.

Algorithm 1 shows the pseudo-code of this heuristic iterative search method. The heuristic algorithm has a deterministic time complexity $O(Nn)$ with N tasks and the maximum iteration step is n.

Algorithm 1 Period Optimization Pseudo-Code

```
1: for all $i$ such that $\tau_i \in \Gamma_{key}$ do $T_i = T_i^{max}$;
2: end for
3: for all $i$ such that $\tau_i \notin \Gamma_{key}$ do
4:     Calculate $T_i$ according to harmonic constraints;
5: end for
6: Calculate $U$ and $U_{bound}$;
7: if $U > U_{bound}$ then
8:     Error: Task set is non-schedulable initially;
9: end if
10: while $U_{bound} - U > e$ do
11:     Calculate $\delta_U^n$ using Eq.(5);
12:     for $\tau_i$ in $\Gamma_{key}$ do
13:         Calculate $J'_{T_i}$;
14:         Calculate $e_i^n$ using Eq.(8);
15:         $(u_{max}, j) = max(u_1, u_2, \ldots, u_N)$;
16:         if $\tau_I \in \Gamma_{key}$ then
17:             $e_j = 0$;
18:         else
19:             $e_k = 0$;
20:         end if
21:         Calculate $T_i^{n+1}$ using Eq.(6) and Eq.(7);
22:         if $T_i < T_i^{min}$ then
23:             $T_i = T_i^{min}$;
24:         end if
25:     end for
26:     Calculate $T_i^{n+1}$ for $\tau_i \in \{\Gamma - \Gamma_{key}\}$;
27:     Calculate $U$ and $U_{bound}$;
28: end while
```

VI. EXPERIMENTAL RESULTS

In this section, the performance of the proposed algorithm is evaluated by comparison with some existing methods. The experimental tests are carried out on an Intel Celeron E3300 dual-core processor platform runs at 2.5GHz with 2GB RAM. The proposed heuristic algorithm is realized and tested on the MATLAB platform.

A. EXPERIMENT 1

Firstly, a uniprocessor case is tested under the same experiment settings as Du's [22]. In [22], the performance function of a controller task τ_i is modeled as Eq.(9), where w_i is a constant factor which denotes the importance of a task and $a_i + b_iT_i^2$ is the quadratic fitting cost-function. The system performance index is modeled as Eq.(10), the higher the better.

$$J_i(T_i) = w_i - (a_i + b_iT_i^2); \tag{9}$$

$$J = \sum_{i=1}^{n} J_i(T_i); \tag{10}$$

With RM scheduling policy adopted, the uniprocessor schedulability bound Eq.(11) [33] is applied as the optimization constraint.

$$\sum_{i=1}^{n} C_i/T_i \leq n(2^{1/n} - 1); \tag{11}$$

The coefficients and task parameters are shown in Table 1, as well as the periods selected by Du's method and our heuristic. Compared with Du's results, a different combination of task's periods is found by the proposed method. The performance index of the solution generated by our heuristic is 51.70, which is higher than Du's (51.39).

B. EXPERIMENT 2

In the second experiment, a dual-core processor based sample system which has the same architecture as Figure 1 is studied. The task parameters are listed in Table 2. Tasks $\Gamma_{key} = \{\tau_3, \tau_8, \tau_{11}, \tau_{13}, \tau_{14}, \tau_{15}\}$ are selected as the key tasks, and their periods are constrained into ranges. In this experiment, we implement a simplified cost function for each key task: $J_i = \alpha_i T_i^2$, and $w_i = 1$. Specifically, we assign α_i a value

TABLE 1. Parameters and periods of 8 tasks.

NO.	C_i (ms)	Coefficients			Periods (ms)	
		w_i	a_i	b_i	Du's	Heuristic
1	4	12.91	2.8	1677	46.47	43.92
2	4	11.18	2.8	1257.7	51.20	43.92
3	5	10.00	2.8	1006.2	59.42	54.90
4	6	9.13	2.8	838.5	65.75	65.88
5	7	8.45	2.8	7.8.7	70.97	76.86
6	7	7.91	2.8	628.9	75.87	77.00
7	8	7.45	2.8	559	80.52	88.82
8	8	7.07	2.8	503.1	84.82	91.18
Utilization					72.42%	72.41%
Performance Index J					51.39	51.70

TABLE 2. Example task parameters(μs).

τ_i	C_i	$[T_i^{min}, T_i^{max}]$
τ_3	16	[250,580]
τ_8	23	[200,496]
τ_{11}	16	[270,866]
τ_{13}	28	[220,925]
τ_{14}	19	[150,478]
τ_{15}	25	[200,1086]

	τ_1	τ_2	τ_4	τ_5	τ_6	τ_7	τ_9	τ_{10}	τ_{12}
C_i	31	37	48	56	20	38	47	23	22

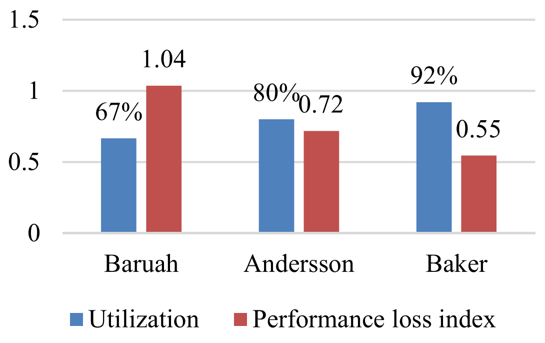


FIGURE 3. Comparison of three utilization bounds.

using the average value of the period range

$$\alpha_i = \frac{\overline{T}}{\overline{T_i}} \quad (\tau_i \in \Gamma_{key}) \tag{12}$$

where $\overline{T_i} = (T_i^{min} + T_i^{max})/2$, and $\overline{T} = mean(\overline{T_i})$.

1) COMPARISON OF DIFFERENT UTILIZATION BOUNDS

The optimization problem Eq.(4) is modeled and solved using LINGO with Baruah, Andersson and Baker's utilization bound adopted as constraints respectively. Three local optimal solutions are found by LINGO. Utilization and performance loss index of the found solutions is shown in Figure 3.

Final convergent utilization of Baker's bound is 92.08%, which is much higher than Baruah's (66.67%) and Andersson's (80%), and the corresponding solution has the minimum performance loss index (0.55). As a result, Baker's bound is chosen as the default schedulability bound in our heuristic algorithm.

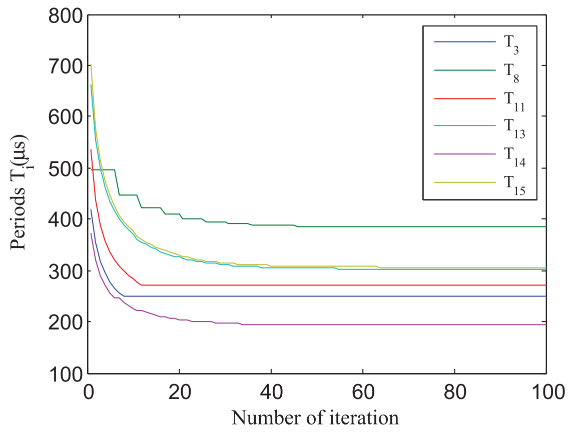


FIGURE 4. Period selection process.

2) CASE STUDY OF THE PROPOSED HEURISTIC ALGORITHM

In order to illustrate how the proposed heuristic algorithm works, the period assignment problem of the sample system is solved as a demonstration. Performance of the proposed heuristic algorithm is estimated in comparison with the local optimal solution found by LINGO. At the beginning of the searching process, all the task periods are initialized with the maximum value of the ranges. Then T_i are decreased using the heuristic algorithm for lower performance loss index and higher utilization. The period selection process of key tasks' periods with increasing number of iteration is shown in Figure 4. Figure 5 shows the selected periods using our heuristic and LINGO. Figure 6 and Figure 7 show the changing process of system utilization and performance loss index over the number of iteration.

As can be seen in Figure 4, with increasing number of iteration, the periods are decreased respectively with different rates. T_3 and T_{11} are decreased quickly and reach their minimum allowable value at 8th and 13th iteration. For task τ_8, there is a step-like descent on T_8, because τ_8 or its upstream tasks often have the maximum utilization in the task set, so its period is kept unchanged for several iterations.

In Figure 5, most tasks' period values derived from the two methods are close, however, some tasks' periods (e.g. T_9) differ considerably between the heuristic and LINGO. This is because the period difference between two methods is enlarged by the chained harmonic constraints. Take task τ_9 for example, T_9 is derived from the key task in the same sub-system (τ_{11}) according to the harmonic constraints between tasks. The harmonic constraint is transferred like: $T_9 \xrightarrow{4} T_{10} \xrightarrow{2} T_{11}$. Consequently $T_9 = 8T_{11}$. $T_{11} = 270$ in heuristic algorithm's solution, while the value is 296 in LINGO's solution. When it comes to task τ_9, this difference is enlarged 8 times.

In Figure 6 and 7, the initial system utilization is very low and the performance loss index J is high because all

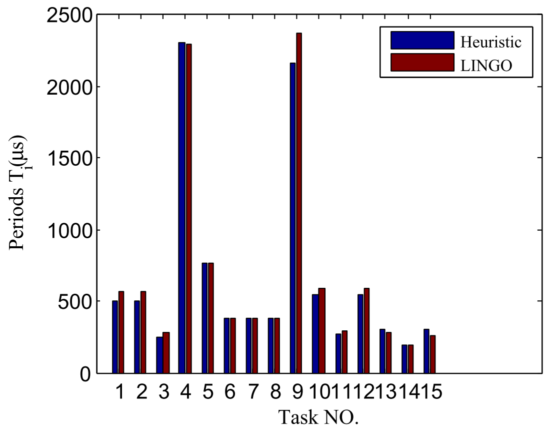


FIGURE 5. Period values of the final solution.

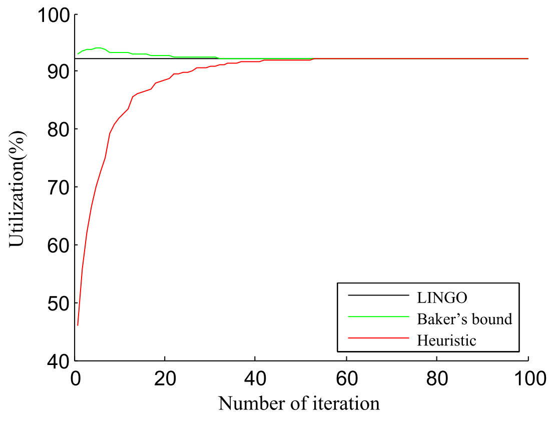


FIGURE 6. Changing process of system processor utilization.

the tasks are assigned with the maximum allowable period value initially. As the number of iteration increase, the system utilization keeps increasing and J keeps decreasing because the tasks' periods are decreased using the heuristic algorithm. Eventually, the system utilization is converged towards to the schedulability bound within 70 iterations. The final solution of the heuristic algorithm has a total utilization of 92.21% which is very close to LINGO's (92.08%). In the meanwhile, J is converged to 0.5385 which is even lower than LINGO's (0.5453).

C. EXPERIMENT 3

In this experiment, performance of the proposed heuristic period selection algorithm is compared with several existing methods:

- Traditional Gradient Descent method (GD).
- Function *fmincon* from MATLAB Optimization Toolbox with Active Set algorithm implemented.
- Optimization software tool LINGO.

LINGO is a professional optimization software tool, and it's capable of generating high quality local optimal solutions for the period selection problem. In this experiment, we take the local optimal solution produced by LINGO as the standard of comparison.

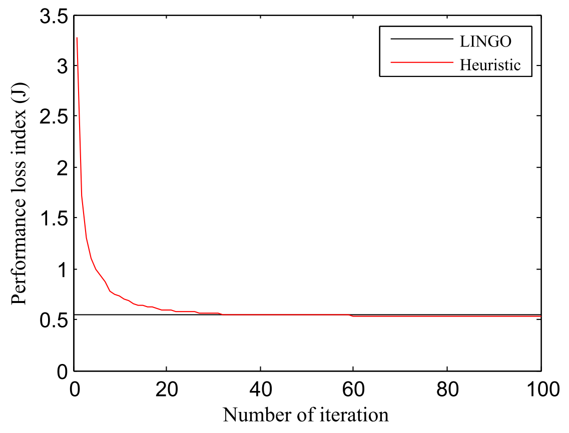


FIGURE 7. Performance loss index.

TABLE 3. Uniform distribution intervals (μs).

Parameter	Interval
C	[1,50]
T_i^{max}	[500,1000]
T_i^{min}	[50,200]
w_i	[0.2,4]

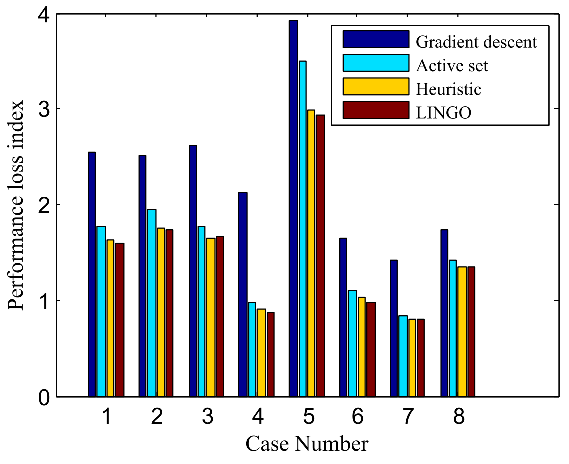


FIGURE 8. Performance loss index of different methods.

Multiple cases are tested in this experiment. In each case, periods of 10 tasks are selected using different methods. The task parameters are chosen randomly from continuous uniform distribution on the intervals shown in Table 3. Experimental results are reported in Figure 8 and Figure 9.

As shown in Figure 8, LINGO has generated the lowest performance loss index in most cases. Compared to Gradient Descent Method and MATLAB Active Set Algorithm, performance loss index produced by our heuristic is more close to LINGO's. Furthermore, LINGO and our heuristic have achieved higher system utilization than the other two methods, as shown in Figure 9.

To further evaluate the performance of our heuristic algorithm. More cases are tested. With LINGO's solution taken as

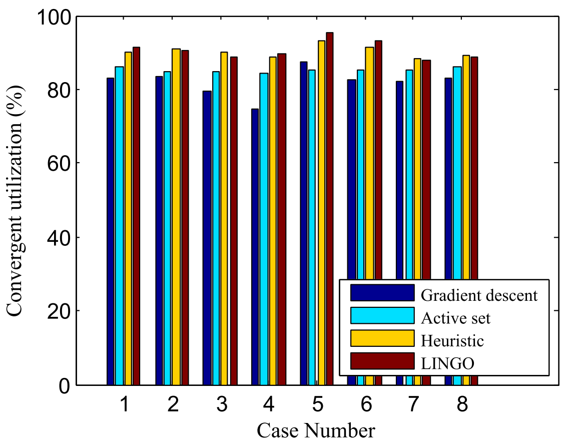


FIGURE 9. Utilization of different methods.

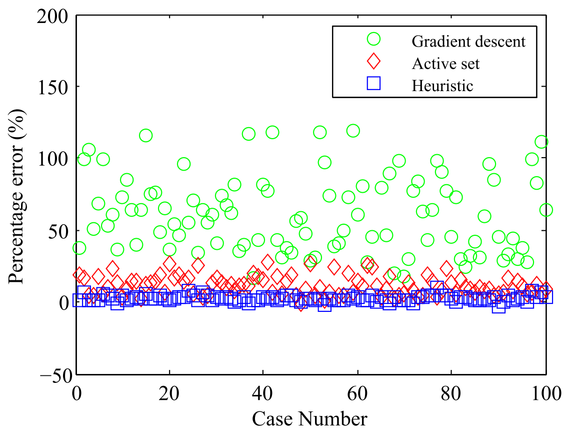


FIGURE 10. Percentage error of performance loss index.

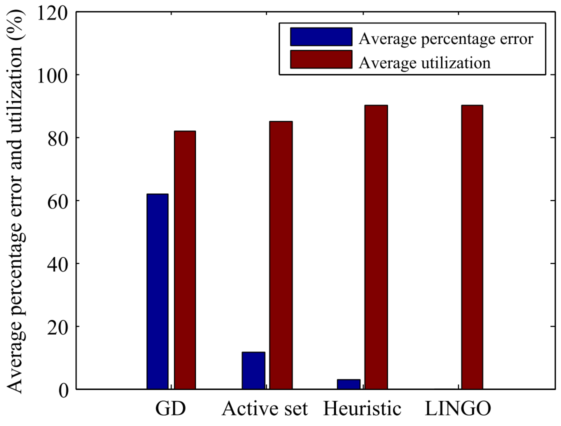


FIGURE 11. Average percentage error and utilization.

TABLE 4. Consumption time of different methods (s).

Number of tasks	GD	Active Set	LINGO	Heuristic
5	0.3269	1.0485	0.9305	0.0043
10	0.2334	0.9671	1.2815	0.0024
15	0.1862	1.1947	2.9312	0.0021
20	0.1757	1.4751	2.7072	0.0019
25	0.1536	1.7490	3.1890	0.0018
30	0.1445	2.0680	3.4959	0.0016
35	0.1085	2.7957	-[1]	0.0015
40	0.0922	2.7922	-	0.0013
45	0.0930	2.8500	-	0.0012
50	0.0650	3.2302	-	0.0011

[1] The model's dimensions exceed the capacity of free trial version of LINGO.

the standard of comparison, percentage error of performance loss index between the other three methods and LINGO is shown in Figure 10. Average percentage error and utilization of 100 cases are shown in Figure 11. The results show that the performance loss index generated by the heuristic is only 2.58% higher averagely than LINGO's while Active Set Algorithm is 11.5% higher and Gradient Descent Method is 61.8%. Average convergent utilization is 90%, which is very close to LINGO's (89.9%) and much higher than Active Set (84.8%) and Gradient Descent (81.6%).

The efficiency of this heuristic period selection algorithm is evaluated through measuring the consumption time to find a local optimal solution. Periods are selected for randomly generated task sets with different number of tasks using four methods. Consumption time of different methods is reported in Table 4. It is shown that our heuristic is much faster than other methods. The searching process is completed within less than 5ms, which is suitable for online use. There is a decrease in the consumption time of GD and our heuristic with the increasing number of tasks. Because when the task set becomes numerous, the initial system utilization will be greater. Consequently, it costs fewer iterations to increase system utilization towards the schedulability bound. Active Set Algorithm and LINGO become more time-consuming when the number of tasks is greater.

VII. CONCLUSION

Discrete-time control system performance optimization problem is studied from a perspective of task period allocation on a multi-core processor platform. The period selecting problem is generalized as an optimization problem which minimizes the system performance loss index under schedulability constraints. A heuristic search algorithm is proposed to solve this optimization problem. Task periods are selected using an empirical search method which decreases the task periods iteratively until a local optimal solution is found. Extensive simulations have been performed to compare the proposed methodology with other methods. The results demonstrate that the heuristic algorithm managed to keep the performance loss at an acceptable level while maintaining a very low time complexity.

In future work, the period assignment problem will be investigated with other forms of performance index and other scheduling policies (such as traditional fixed-priority scheduling and EDF scheduling) adopted.

REFERENCES

[1] X. Zhao, X. Wang, G. Zong, and X. Zheng, "Adaptive neural tracking control for switched high-order stochastic nonlinear systems," *IEEE Trans. Cybern.*, vol. 47, no. 10, pp. 3088–3099, Oct. 2017.

[2] X. Zhao, P. Shi, X. Zheng, and J. Zhang, "Intelligent tracking control for a class of uncertain high-order nonlinear systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 9, pp. 1976–1982, Sep. 2016.

[3] H. Wang, P. Shi, H. Li, and Q. Zhou, "Adaptive neural tracking control for a class of nonlinear systems with dynamic uncertainties," *IEEE Trans. Cybern.*, vol. 47, no. 10, pp. 3075–3087, Oct. 2017.

[4] H. Wang, X. Liu, and K. Liu, "Adaptive fuzzy tracking control for a class of pure-feedback stochastic nonlinear systems with non-lower triangular structure," *Fuzzy Sets Syst.*, vol. 302, pp. 101–120, Nov. 2016.

[5] H. Wang, X. Liu, K. Liu, B. Chen, and C. Lin, "Adaptive neural control for a general class of pure-feedback stochastic nonlinear systems," *Neurocomputing*, vol. 135, pp. 348–356, Jul. 2014.

[6] H. Wang, B. Chen, X. Liu, K. Liu, and C. Lin, "Robust adaptive fuzzy tracking control for pure-feedback stochastic nonlinear systems with input constraints," *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 2093–2104, Dec. 2013.

[7] H. Wang, W. Sun, and P. X. Liu, "Adaptive intelligent control of nonaffine nonlinear time-delay systems with dynamic uncertainties," *IEEE Trans. Syst. Man, Cybern., Syst.*, vol. 47, no. 7, pp. 1474–1485, Jul. 2017.

[8] S. K. Nguang, "Robust stabilization of a class of time-delay nonlinear systems," *IEEE Trans. Autom. Control*, vol. 45, no. 4, pp. 756–762, Apr. 2000.

[9] Z. Zhang, S. Xu, and B. Zhang, "Asymptotic tracking control of uncertain nonlinear systems with unknown actuator nonlinearity," *IEEE Trans. Autom. Control*, vol. 59, no. 5, pp. 1336–1341, May 2014.

[10] X.-J. Xie and L. Liu, "Further results on output feedback stabilization for stochastic high-order nonlinear systems with time-varying delay," *Automatica*, vol. 48, no. 10, pp. 2577–2586, Oct. 2012. [Online]. Available: http://dx.doi.org/10.1016/j.automatica.2012.06.061

[11] V. M. Becerra and P. D. Roberts, "Dynamic integrated system optimization and parameter estimation for discrete time optimal control of nonlinear systems," *Int. J. Control*, vol. 63, no. 2, pp. 257–281, 1996.

[12] T. Ikeda, M. Nagahara, and S. Ono, "Discrete-valued control of linear time-invariant systems by sum-of-absolute-values optimization," *IEEE Trans. Autom. Control*, vol. 62, no. 6, pp. 2750–2763, Jun. 2017.

[13] A. Cervin, D. Henriksson, B. Lincoln, J. Eker, and K.-E. Årzén, "How does control timing affect performance? Analysis and simulation of timing using Jitterbug and TrueTime," *IEEE Control Syst. Mag.*, vol. 23, no. 3, pp. 16–30, Jun. 2003.

[14] D. Seto, J. P. Lehoczky, L. Sha, and K. G. Shin, "On task schedulability in real-time control systems," in *Proc. 17th IEEE Real-Time Syst. Symp.*, Dec. 1996, pp. 13–21.

[15] D. Seto, J. P. Lehoczky, and L. Sha, "Task period selection and schedulability in real-time systems," in *Proc. 19th IEEE Real-Time Syst. Symp.*, Dec. 1998, pp. 188–198.

[16] L. Palopoli, C. Pinello, A. Bicchi, and A. Sangiovanni-Vincentelli, "Maximizing the stability radius of a set of systems under real-time scheduling constraints," *IEEE Trans. Autom. Control*, vol. 50, no. 11, pp. 1790–1795, Nov. 2005.

[17] X. Feng and Y.-X. Sun, "Control-scheduling codesign: A perspective on integrating control and computing," *Dyn. Continuous, Discrete Impulsive Syst. B*, vol. 13, no. S1, pp. 1352–1358, 2008.

[18] E. Bini and M. Di Natale, "Optimal task rate selection in fixed priority systems," in *Proc. 26th IEEE Int. Real-Time Syst. Symp.*, Dec. 2006, vol. 20. no. 5, pp. 403–413.

[19] Y. Wu, G. Buttazzo, E. Bini, and A. Cervin, "Parameter selection for real-time controllers in resource-constrained systems," *IEEE Trans. Ind. Informat.*, vol. 6, no. 4, pp. 610–620, Nov. 2010.

[20] A. Cervin and J. Eker, "Control-scheduling codesign of real-time systems: The control server approach," *J. Embedded Comput.*, vol. 1, no. 2, pp. 209–224, 2005.

[21] A. Cervin, M. Velasco, P. Martí, and A. Camacho, "Optimal online sampling period assignment: Theory and experiments," *IEEE Trans. Control Syst. Technol.*, vol. 19, no. 4, pp. 902–910, Jul. 2011.

[22] C. Du, L. Tan, and Y. Dong, "Period selection for integrated controller tasks in cyber-physical systems," *Chin. J. Aeronautics*, vol. 20, no. 3, pp. 56–62, 2015.

[23] H. J. Cha, W. H. Jeong, and J. C. Kim, "Control-scheduling codesign exploiting trade-off between task periods and deadlines," *Mobile Inf. Syst.*, vol. 2016, no. 2, 2016, Art. no. 3414816. [Online]. Available: http://dx.doi.org/10.1155/2016/3414816

[24] C.-C. Han and H.-Y. Tyan, "A better polynomial-time schedulability test for real-time fixed-priority scheduling algorithms," in *Proc. 18th IEEE Real-Time Syst. Symp.*, Dec. 1997, pp. 104–113.

[25] M. Nasri, G. Fohler, and M. Kargahi, "A framework to construct customized harmonic periods for real-time systems," in *Proc. 26th Euromicro Conf. Real-Time Syst. (ECRTS)*, 2014, pp. 211–220.

[26] M. Nasri and G. Fohler, "An efficient method for assigning harmonic periods to hard real-time tasks with period ranges," in *Proc. 27th Euromicro Conf. Real-Time Syst.*, 2015, pp. 149–159.

[27] J. Xu, "A method for adjusting the periods of periodic processes to reduce the least common multiple of the period lengths in real-time embedded systems," in *Proc. IEEE/ASME Int. Conf. Mechatronics Embedded Syst. Appl. (MESA)*, Jul. 2010, pp. 288–294.

[28] V. Brocal, P. Balbastre, R. Ballester, and I. Ripoll, "Task period selection to minimize hyperperiod," in *Proc. IEEE 16th Conf. Emerging Technol. Factory Autom.*, Sep. 2011, pp. 1–4.

[29] I. Ripoll and R. Ballester-Ripoll, "Period selection for minimal hyperperiod in periodic task systems," *IEEE Trans. Comput.*, vol. 62, no. 9, pp. 1813–1822, Sep. 2013.

[30] K. Jeffay, "The real-time producer/consumer paradigm: A paradigm for the construction of efficient, predictable real-time systems," *ACM/SIGAPP Symp Appl. Comput.*, 1993, pp. 796–804.

[31] R. Gerber, S. Hong, and M. Saksena, "Guaranteeing real-time requirements with resource-based calibration of periodic processes," *IEEE Trans. Softw. Eng.*, vol. 21, no. 7, pp. 579–592, Jul. 1995.

[32] M. Shin and M. Sunwoo, "Optimal period and priority assignment for a networked control system scheduled by a fixed priority scheduling system," *Int. J. Auto. Technol.*, vol. 8, no. 1, pp. 39–48, 2007.

[33] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," *J. ACM*, vol. 20, no. 1, pp. 46–61, 1973.

[34] C. J. Fidge, "Real-time schedulability tests for preemptive multitasking," *Real-Time Syst.*, vol. 14, no. 1, pp. 61–93, 1998.

[35] B. Andersson, S. Baruah, and J. Jonsson, "Static-priority scheduling on multiprocessors," in *Proc. IEEE Real-Time Syst. Symp.*, Dec. 2001, pp. 93–202.

[36] S. K. Baruah and J. Goossens, "Rate-monotonic scheduling on uniform multiprocessors," *IEEE Trans. Comput.*, vol. 52, no. 7, pp. 966–970, Jul. 2003.

[37] T. P. Baker, "An analysis of fixed-priority schedulability on a multiprocessor," *Real-Time Syst.*, vol. 32, nos. 1–2, pp. 49–71, 2006.

[38] G. Buttazzo, M. Velasco, and P. Mart, "Quality-of-control management in overloaded real-time systems," *IEEE Trans. Comput.*, vol. 56, no. 2, pp. 253–266, Feb. 2007.

[39] K. J. Astrom and B. Wittenmark, "Computer controlled systems: Theory and design," *IEE Rev.*, vol. 31, no. 31, pp. 237–248, 1997.

[40] S. M. Melzer and B. C. Kuo, "Sampling period sensitivity of the optimal sampled data linear regulator," *Automatica*, vol. 7, no. 3, pp. 367–370, 1971.

[41] B. Lennartson, "On the choice of controller and sampling period for linear stochastic control," *Automatica*, vol. 26, no. 3, pp. 573–578, 1990.

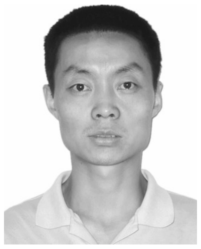
HONGYA FU was born in Harbin, China, in 1962. He received the B.S. and M.S. degrees in mechanical engineering from the Harbin Institute of Technology, Harbin, in 1987.

From 1992 to 2001, he was an Assistant Professor with the School of Mechanical Engineering, Harbin Institute of Technology, where he has been a Professor with the Mechanical Engineering Department, Harbin Institute of Technology, since 2002. He is the author of two books, more than 50 articles, and more than 20 inventions. His research interests include fiber winding molding technology, fiber automatic laying technology, and open CNC system.

JIANKANG LIU received the B.S. degree in mechanical engineering from Huaqiao University, Xiamen, China, in 2011, and the M.S. degree in mechanical engineering from the Harbin Institute of Technology, Harbin, China, in 2013, where he is currently pursuing the Ph.D. degree in mechanical engineering. His research interest includes real-time control system design and open CNC system.

ZHENYU HAN was born in Shandong, China, in 1978. He received the B.S., M.S., and Ph.D. degrees in mechanical engineering from the Harbin Institute of Technology, Harbin, in 2001 and 2005, respectively.

From 2008 to 2016, he was an Assistant Professor with the School of Mechanical Engineering, Harbin Institute of Technology, where he has been a Professor with the Mechanical Engineering Department, since 2017. He is the author of two books, more than 60 articles, and more than ten inventions. His research interests include fiber winding molding technology, fiber automatic laying technology, and compound material.



ZHONGXI SHAO was born in Heilongjiang, China, in 1978. He received the B.S. and M.S. degrees in mechanical engineering from Jiamusi University, Jiamusi, in 2006, and the Ph.D. degree in mechanical engineering from the Harbin Institute of Technology, Harbin, in 2010.

From 2010 to 2013, he was a Research Assistant with the Research Division of Numerical Control Technology, Harbin Institute of Technology, where he has been an Assistant Professor with the Mechanical Engineering Department, since 2014. His research interests include heavy duty machine tool design and industrial control system.