Tech Report: Best-Harmonically-Fit Best-Harmonically-Fit Periodic Task Assignment Algorithm on Multiple Periodic Resources

Chunhui Guo, *Student Member, IEEE*, Xiayu Hua, *Student Member, IEEE*, Hao Wu, *Student Member, IEEE* and Shangping Ren, *Senior Member,IEEE*

I. Introduction

As computer hardware technology advances, both the number and the computational capabilities of processors used in real-time systems are increasing. To take advantages of the increased physical resource's capacity, multiple groups of real-time applications are deployed on the same physical platform. However, since each group of real-time applications may have different time granularity [1], when multiple groups of applications share the same processors, traditional real-time system models and theoretic results may not be sufficient or even applicable to guarantee that each group of real-time applications will not have time interference among each other and that they will satisfy their timing requirements.

In order to study the issues of scheduling multiple groups of real-time applications on the same physical resources, the concept of virtual real-time resources is proposed [1]. Virtual real-time resources are an abstraction of physical resources where the physical resources are shared by real-time application groups [1]. With the concept of virtual real-time resources, each group of real-time applications has its own isolated and independent virtual resource, hence avoiding interference among different groups of real-time applications. However, from an application perspective, virtual real-time resources are not continuously available to the application. Instead, virtual resources are periodic, i.e., they periodically provide certain amount of processing capability to the applications [1], [2], [3], [4], [5], [6], [7].

The study of periodic resources can be traced back to 1999 when the concept of periodic resource was first formally defined [4]. It has recently drawn more attention in the community [1], [7], [8], [9], [10], [11], [12], [13], [14]. However, until now, most of the studies about periodic resources focus on task set schedulability analysis on a *single* periodic resource. There has not been much work, if any, in the literature dealing with the task assignment problem on multiple periodic resources. In this paper, we study the task assignment problem in the context of assigning multiple periodic tasks to multiple periodic resources. The main goal is to decide on a task assignment strategy so that the resource capacity provided by every periodic resource is maximally utilized.

The authors are with Computer Science department, Illinois Institute of Technology, 10 W 31st street, Room 013C, Chicago, IL 60616, USA
Emails: {cguo13, xhua, hwu28}@hawk.iit.edu, ren@iit.edu

To achieve the goal, we first study under what relationship between the task period and resource period can the resource capacity be fully utilized. Intuitively, the more harmonically related the tasks and the periodic resources are, the better resource utilization rate we can achieve. We formally prove that, in fact, if a harmonic task set is also harmonic with the resource, the resource capacity can be fully (100%) utilized under the RM (rate-monotonic) scheduling algorithm. Second, based on the harmonicity property, we present the Best-Harmonically-Fit (BHF) task assignment algorithm to assign a periodic task set to multiple periodic resources. We then empirically evaluate the BHF algorithm's performance by comparing it with commonly used multiprocessor task assignment approaches in the literature, i.e., the Best-Fit, the First-Fit, and the Worst-Fit approaches [15], [16], [17]. We also evaluate the BHF algorithm by comparing it with the optimal task assignment found through exhaustive search for a small-sized task set and resource set.

The rest of the paper is organized as follow: Section II discusses related work. Section III defines system models, presents preliminary results, and formulates the problem we are to address. Section IV presents the harmonic utilization bound for a single periodic resource and task set harmonic transformation with respect to a periodic resource. In Section V, we introduce the Best-Harmonically-Fit task assignment algorithm. Section VI discusses the experimental results. We conclude the work in Section VII.

II. Related Work

Since the rate-monotonic (RM) and the earliest deadline first (EDF) scheduling algorithms were first analyzed by Liu and Layland in 1973 [18], the real-time scheduling problem has been studied extensively. To this day, the rate-monotonic scheduling algorithm is still considered the most significant fixed-priority scheduling algorithm for scheduling periodic tasks on a single dedicated resource. The RM utilization bound for a preemptive system is $N(2^{1/N}-1)$ with its limit of 69.3% on a single dedicated resource, where N is the number of tasks [18]. Mossé *et al.* [19], [20] proposed the R-Bound based on the Liu and Layland bound. The R-Bound takes the difference of task periods into consideration and tightens the utilization bound to $(N-1)(r^{1/(N-1)}-1)+2/r-1$ with limit $\ln r+2/r-1$, where $r=T_{\max}/T_{\min}$ and $1\leq r<2$. Another improvement on the RM bound was made by Han *et al.* [21].

They proved that the utilization bound can reach 100% with the RM scheduling algorithm if the task set is harmonic.

Another major research area in the real-time community is scheduling tasks on multiprocessors. The goal of multiprocessor scheduling is to schedule as many tasks as possible (from the total task utilization perspective) on a given number of processors while still guaranteeing task set deadline satisfication [22], [17], [23]. However, as stated by Liu and Layland in [18], scheduling periodic tasks on multiprocessors is much harder than on a single processor. The utilization bound for a multiprocessor system is much lower than for a single processor. Many algorithms have been proposed to improve the utilization bound for multiprocessors. However, it has been proven that the optimal utilization bound for a multiprocessor scheduling algorithm is only $(M+1)/2$ [22], where M is the number of processors. Recently, researchers have improved the utilization bound for multiprocessors under certain conditions. Wang *et al.* proved that the utilization bound for multiprocessors can reach Liu and Layland bound for a single processor if tasks can be split [17]. Fan and Quan proposed a harmonic-aware scheduling approach to improve schedulability on multiprocessors [23].

However, the literature mentioned above is based on the assumption that resources are dedicated resources, i.e., they are constantly available to application tasks. When multiple groups of real-time applications share the same physical resources, the assumption that resources are constantly available to the application becomes invalid. Hence, the concept of virtual real-time resources is proposed to handle such scenario. A virtual resource is often represented as $\gamma = (\Pi, \Theta)$, where Π is the virtual resource period and Θ is the processing time available to applications [1], [2], [3], [4], [7].

Shirero *et al.* [4] first defined periodic resources and proposed a real-time round robin scheduling algorithm in 1999. They also introduced the concept of *resource regularity*. Based on resource regularity, they proposed schedulability bounds for periodic tasks on a single periodic resource. Mok *et al.* [7], [24] extended Shirero's work and proposed a more comprehensive schedulability analysis for periodic resources under both EDF and RM scheduling algorithms. However, both Shirero's and Mok's periodic resource models have a constraint that either the resource available pattern within each period or the resource regularity is known and does not change at run-time. Shin *et al.* later removed the constraint on the periodic resource model and provided the schedulability analysis under a relaxed model where resource pattern can be arbitrary and can change at run-time. They give schedulability bounds under the relaxed model for both EDF and RM [8], [9], [10], [11].

Hua *et al.* [14] recently studied how multiple periodic resources may be integrated into an equivalent single periodic resource so that existing real-time scheduling theorems on a single periodic resource can be applied. They further extended schedulability tests of periodic tasks on a single periodic resource from the continuous time domain given in [11] to a discrete time domain so that the schedulability tests given in [11] can be applied in practice.

However, as of today, much of the work in the literature has been on schedulability analysis of a task set on a single periodic resource. To our best knowledge, there is no prior work studying the task assignment problem on multiple periodic resources. In this paper, we are to address the problem: for a given set of periodic tasks and a given set of periodic resources, if the task set is not schedulable on any single periodic resource in the resource pool, how to assign tasks to the available periodic resources so that the utilization rate of used periodic resources is maximized. Though the problem we are to address is similar to the work of assigning a periodic task set to processors so as to minimize the number of processors used [25], it is in a different context where resources are not dedicated processors, but rather they are of periodic nature.

III. Problem Formulation

A. Models and Definitions

Task Model

The task model considered in this paper is similar to the one defined by Liu and Layland [18]. A task set $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_N\}$ has N independent periodic tasks that are all released at time 0. Each task $\tau_i \in \Gamma$ is a 2-tuple (T_i, C_i), where T_i is the *inter-arrival time* between any two consecutive jobs of τ_i (also called *period*), and C_i is the *worst-case execution time*. We further assume that a task period is an integer, i.e., $T_i \in \mathbb{N}^+$. The *utilization* of each task τ_i is defined as $U_{\tau_i} = C_i/T_i$, and the *utilization* of the task set Γ is denoted as U_Γ, where

$$U_\Gamma = \sum_{\tau_i \in \Gamma} U_{\tau_i}. \tag{1}$$

We use $U_{\max}$ and $T_{\min}$ to denote the maximum task utilization and the minimum task period of the task set Γ, respectively, i.e.,

$$U_{\max} = \max\{U_{\tau_i} | \forall \tau_i \in \Gamma\}$$
$$T_{\min} = \min\{T_i | \forall \tau_i \in \Gamma\}.$$

Resource Model

For a resource model, we adopt the one defined by Shin [8]. A resource set $\mathcal{R} = \{\gamma_1, \gamma_2, \ldots, \gamma_M\}$ consists of M periodic resources that are all available at time 0. Each resource $\gamma_j \in \mathcal{R}$ is characterized by a 2-tuple (Π_j, Θ_j), where Π_j is the resource *period*, and Θ_j is the *allocation time*. We further assume that both a resource period and a resource allocation time are integers, i.e., $\Pi_j, \Theta_j \in \mathbb{N}^+$, and satisfy $0 < \Theta_j \leq \Pi_j$. The *capacity* of each resource γ_j is defined as $\mathcal{C}_{\gamma_j} = \Theta_j/\Pi_j$, and the *capacity* of a resource set $\mathcal{R}$ is denoted as $\mathcal{C}_\mathcal{R}$, where

$$\mathcal{C}_\mathcal{R} = \sum_{\gamma_j \in \mathcal{R}} \mathcal{C}_{\gamma_j}. \tag{2}$$

We use $\tau \mapsto \gamma$ to denote that task τ is assigned to periodic resource γ, and say that γ is the *host resource* for task τ. For a resource γ and a task set Γ, we use $\Gamma_\gamma \subseteq \Gamma$ to denote the task subset containing all tasks assigned to resource γ, i.e.,

$$\Gamma_\gamma = \{\tau_i | \tau_i \in \Gamma \wedge \tau_i \mapsto \gamma\}. \tag{3}$$

Similarly, the number of tasks in Γ_γ is denoted as N_γ, and the *utilization* of Γ_γ is denoted as

$$U_{\Gamma_\gamma} = \sum_{\tau_i \in \Gamma_\gamma} U_{\tau_i}. \tag{4}$$

A periodic resource γ is called an *unused resource* if there are no tasks assigned to it; otherwise it is called a *used resource*. For a resource set $\mathcal{R}$, we use $\mathcal{R}_{\texttt{used}} \subseteq \mathcal{R}$ to denote the resource subset containing all used resources; $M_{\texttt{used}}$ is the size of $\mathcal{R}_{\texttt{used}}$.

Definition 1. [Resource Utilization Rate ($\texttt{UR}_\gamma$)] *The utilization rate of a periodic resource γ is defined as the capacity percentage used by tasks assigned to γ, i.e.,*

$$\texttt{UR}_\gamma = \begin{cases} \frac{U_{\Gamma_\gamma}}{\mathcal{C}_\gamma} = \frac{\sum_{\tau_i \in \Gamma_\gamma} U_{\tau_i}}{\Theta/\Pi} & \texttt{if} \quad \Gamma_\gamma \neq \emptyset \\ 0 & \texttt{if} \quad \Gamma_\gamma = \emptyset \end{cases} \tag{5}$$

□

Definition 2. [Resource Set Utilization Rate ($\texttt{UR}_\mathcal{R}$)] *The utilization rate of a resource set $\mathcal{R}$ is defined as the total capacity percentage used by tasks assigned to $\mathcal{R}$, i.e.,*

$$\texttt{UR}_\mathcal{R} = \frac{\sum_{\gamma_j \in \mathcal{R}_{\texttt{used}}} U_{\Gamma_{\gamma_j}}}{\sum_{\gamma_j \in \mathcal{R}_{\texttt{used}}} \mathcal{C}_{\gamma_j}}. \tag{6}$$

□

B. Preliminary Results

For self-containment, we introduce a few terms and schedulability analysis results from [11], and give our corollaries derived from the theorems in [11].

Theorem 1. *[11] Given a task set Γ and a single periodic resource $\gamma = (\Pi, \Theta)$, if $\forall i, 1 \leq i \leq N, T_i \geq 2\Pi - \Theta$, the task utilization bound under RM scheduling is*

$$\texttt{UB}_\gamma = \mathcal{C}_\gamma \cdot N_\gamma \left[\left(\frac{2k + 2(1 - \mathcal{C}_\gamma)}{k + 2(1 - \mathcal{C}_\gamma)} \right)^{1/N_\gamma} - 1 \right] \tag{7}$$

where $k = \max\{k \in \mathbb{N}^0 | (k+1)\Pi - \Theta < T_{min}\}$. □

From Theorem 1, we can derive the following corollary.

Corollary 1. *Given a task $\tau = (T, C)$ and a single periodic resource $\gamma = (\Pi, \Theta)$ with condition $T \geq 2\Pi - \Theta$, the task τ is guaranteed to be schedulable on resource γ with the RM scheduling policy if*

$$\mathcal{C}_\gamma \cdot \frac{k}{k + 2(1 - \mathcal{C}_\gamma)} \geq U_\tau \tag{8}$$

where $k = \max\{k \in \mathbb{N}^0 | (k+1)\Pi - \Theta < T\}$. □

Proof. Task τ is guaranteed to be schedulable on resource γ under RM scheduling if

$$\texttt{UB}_\gamma \geq U_\tau$$

Since there is only one task to be scheduled on resource γ, the utilization bound of γ can be obtained by substituting N_γ in formula (7) with 1. Hence, we have

$$\texttt{UB}_\gamma = \mathcal{C}_\gamma \cdot \frac{k}{k + 2(1 - \mathcal{C}_\gamma)}$$

where $k = \max\{k \in \mathbb{N}^0 | (k+1)\Pi - \Theta < T\}$.

■

Definition 3. [Abstraction Overhead (O)][11] *For a single periodic resource γ and a task set Γ, the abstraction overhead is defined as*

$$O = \frac{\mathcal{C}_\gamma - U_\Gamma}{U_\Gamma} \tag{9}$$

Theorem 2. *[11] Given a single periodic resource γ and a task set Γ which is schedulable on γ under RM, the lower abstraction overhead bound is 0.443 when $k \to +\infty$, i.e.,*

$$\texttt{OB} = \frac{\mathcal{C}_\gamma - U_\Gamma}{U_\Gamma} \geq 44.3\% \tag{10}$$

where $k = \max\{k \in \mathbb{N}^0 | (k+1)\Pi - \Theta < T_{min}\}$.

□

From Theorem 2, we can derive the following corollary:

Corollary 2. *Given a single periodic resource γ and a task set Γ which is schedulable on γ with RM, the resource utilization rate up bound of γ is 0.693 when $k \to +\infty$, i.e.,*

$$\texttt{UR}_\gamma \leq 69.3\% \tag{11}$$

where $k = \max\{k \in \mathbb{N}^0 | (k+1)\Pi - \Theta < T_{min}\}$. □

Proof. According to Theorem 2, we have

$$\mathcal{C}_\gamma \geq 1.443 U_\Gamma \tag{12}$$

Since Γ is schedulable on γ with RM, by Definition 1, we have

$$\texttt{UR}_\gamma = \frac{U_{\Gamma_\gamma}}{\mathcal{C}_\gamma} = \frac{U_\Gamma}{\mathcal{C}_\gamma} \leq 69.3\% \tag{13}$$

■

To achieve the above utilization rate of 69.3%, a precondition is that $k \to +\infty$, which implies that the period of the resource is infinitely small. When the resource period becomes infinitely small, the periodic resource approaches the continuous resource. In this case, the utilization bound for a single periodic resource is actually the Liu and Layland RM utilization bound for continuous resources [18]. However, since a resource with an infinitely small period is difficult to implement in reality, Shin's periodic resource utilization bound is hardly achievable in practice.

C. Problem Formulation

The problem to be addressed in this paper is that under the condition that a given periodic task set is scheduable on a given set of periodic resources, how to assign the task set to the periodic resource set so that the used resource set utilization rate is maximized and all tasks are schedulable on their assigned resources. It is formally defined as follows:

Given a task set $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_N\}$ where $\tau_i = (T_i, C_i)$, and a resource set $\mathcal{R} = \{\gamma_1, \gamma_2, \ldots, \gamma_M\}$ where $\gamma_j = (\Pi_j, \Theta_j)$, assign Γ to $\mathcal{R}$ such that:

$$\begin{aligned}
&\textbf{Object:} && \max \mathtt{UR}_{\mathcal{R}} \\
&\textbf{Subject to} && \\
&\textit{Constraint 1:} && \forall j\ 1 \le j \le M,\ \mathcal{C}_{\gamma_j} \cdot \tfrac{k}{k+2(1-\mathcal{C}_{\gamma_j})} \ge U_{\max} \\
&\textit{Constraint 2:} && M \ge N \\
&\textit{Constraint 3:} && \forall j\ 1 \le j \le M,\ 2\Pi_j - \Theta_j \le T_{\min}
\end{aligned}$$

where $k = \max\{k \in \mathbb{N}^0 | (k+1)\Pi_j - \Theta_j < T_{min}\}$.

Constraint 1 guarantees that each resource is large enough to schedule any single task in the given task set (Corollary 1). Constraint 2 together with Constraint 1 ensure that the task set is schedulable on the resource set without task splitting. Constraint 3 is the pre-condition of Constraint 1 (needed by Theorem 1).

We take two steps to address the problem. First, we analyze the periodic resource harmonic utilization bound under the RM scheduling policy, and present task set harmonic transformation with respect to a periodic resource (Section IV). Second, we present the Best-Harmonically-Fit (BHF) algorithm (Section V) which assigns tasks to resources with the goal of maximizing the resource set utilization rate while guaranteeing task set schedulability.

IV. Harmonic Property

A. Utilization Bound for Harmonically Related Task Set and Periodic Resource

If a harmonic task set is also harmonic with a given periodic resource, then the schedulable utilization bound of the task set can be as high as the resource capacity. To prove this property, we first give a lemma to show that if one task is harmonic with a given periodic resource, then the task can fully utilize the resource. We then prove a theorem stating that if a harmonic task set and a periodic resource are harmonic, the resource can be fully utilized by the task set under RM scheduling.

Lemma 1. *Given a task $\tau = (T, C)$ and a periodic resource $\gamma = (\Pi, \Theta)$, if the task and the resource are harmonic, i.e., $T = K \cdot \Pi$ $(K \in \mathbb{N}^+)$, then task τ is schedulable on γ if and only if $\frac{C}{T} \le \frac{\Theta}{\Pi}$.* □

Proof. Since $T = K \cdot \Pi$ $(K \in \mathbb{N}^+)$ and γ and τ all start at time 0, each task period T contains K complete resource periods. In other words, in each period, task τ obtains $K \cdot \Theta$ allocation time from resource γ. Hence, to guarantee that in each period of τ there is at least C allocation time, if and only if the following condition holds: $K \cdot \Theta \ge C$, which means $\frac{C}{T} \le \frac{K \cdot \Theta}{T}$, i.e. $\frac{C}{T} \le \frac{\Theta}{\Pi}$. ■

Lemma 2. *Given a task set Γ with two harmonic tasks $\tau_1 = (T, C_1)$ and $\tau_2 = (K \cdot T, C_2)$, and a periodic resource $\gamma = (\Pi, \Theta)$ which is also harmonic with the task set Γ. Let task $\tau = (T, C_1 + \frac{C_2}{K})$, if task τ is schedulable on resource γ, then the task set Γ is also schedulable on γ with RM scheduling.* □

Proof. Since τ is harmonic with γ and is schedulable, according to Lemma 1, we have $\frac{C_1 + C_2/K}{T} \le \frac{\Theta}{\Pi}$.

In task set Γ, based on RM, τ_1 has the highest priority. In addition, since τ_1 and τ have the same period, both release at time 0, and $C_1 < C_1 + \frac{C_2}{K}$, if τ is schedulable, it guarantees that τ_1 is schedulable.

Since τ_2 also releases at time 0, within each period of τ_2, there are K instances of τ_1. The total resource demand is $C_2 + KC_1$. Furthermore, within each period of τ_2, there are also K instances of τ, with total execution time of $K \times (C_1 + \frac{C_2}{K}) = KC_1 + C_2$. As τ is schedulable on resource γ, τ_2 is also schedulable on resource γ. ■

Based on Lemma 1 and Lemma 2, we have the following theorem.

Theorem 3. *Given a harmonic task set $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_N\}$ with period $T_i = T_1 \cdot p^{i-1} (p \in \mathbb{N}^+)$ and a periodic resource $\gamma = (\Pi, \Theta)$ which is harmonic with the task set Γ, i.e., $T_1 = K \cdot \Pi (K \in \mathbb{N}^+)$, the harmonic utilization bound under RM scheduling is*

$$\mathtt{HUB}_\gamma = \frac{\Theta}{\Pi} \tag{14}$$

□

Proof. To prove Theorem 3 is equivalent to proving that the task set Γ is schedulable on the resource γ under RM scheduling if

$$\sum_{i=1}^{N} \frac{C_i}{T_i} \le \frac{\Theta}{\Pi} \tag{15}$$

holds.

We use induction on the number of tasks (n) in the task set to prove the theorem.

- Base case $n = 1$, based on Lemma 1, the theorem holds.
- Assume when $n = N$, if $\sum_{i=1}^{n} \frac{C_i}{T_i} \le \frac{\Theta}{\Pi}$, the task set is schedulable on γ with RM scheduling.
- We prove that if $n = N + 1$ and $\sum_{i=1}^{N+1} \frac{C_i}{T_i} \le \frac{\Theta}{\Pi}$, then the task set is schedulable on the resource by RM.
  Without loss of generality, we assume the task set is $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_{N-1}, \tau_N, \tau_{N+1}\}$ and $T_i \le T_{i+1}, \forall i \in \{1, 2, ..., N\}$. Since Γ is harmonic, $\frac{T_{N+1}}{T_N} = p$ where $p \in \mathbb{N}^+$, we replace τ_N and τ_{N+1} by a single task $\tau = (T_N, C_N + \frac{C_{N+1}}{p})$ and denote the new task set as $\Gamma^* = \{\tau_1, \tau_2, ..., \tau_{N-1}, \tau\}$. The utilization of U_{Γ^*} is calculated as:

$$U_{\Gamma^*} = \sum_{i=1}^{N-1} \frac{C_i}{T_i} + \frac{C_N + \frac{C_{N+1}}{p}}{T_N} = \sum_{i=1}^{N+1} \frac{C_i}{T_i} = U_\Gamma \le \frac{\Theta}{\Pi}$$

  Hence, for Γ^*, its task number $n = N$ and its total utilization $U_{\Gamma^*} \le \frac{\Theta}{\Pi}$. Then, according to the induction assumption, Γ^* is schedulable, which indicates τ along with the task set $\{\tau_1, ..., \tau_{N-1}\}$ is schedulable by RM. According to Lemma 2, τ_N and τ_{N+1} are also schedulable if τ is schedulable. Hence, the task set Γ is schedulable by RM.

■

B. Task Set Harmonic Transformation with Respect to Periodic Resource

As discussed in the previous sub-section, the harmonic relation among a set of tasks and a resource brings the advantage that the resource capacity can be fully utilized by the task set. However, in the real world this criteria, i.e., tasks and resource are pairwise harmonic, is difficult to achieve. In order to have the advantage brought by a harmonic relation, we need to transform an arbitrary task set into a task set that meets the criteria.

Han *et al.* [21] developed the DCT (Distance-Constrained Tasks) algorithm that can transform an arbitrary task set into a harmonic task set. In particular, given a task set $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_N\}$, if we use task τ_1's period T_1 as the base for transformation, the harmonic periods of the other tasks transformed by the DCT Algorithm are

$$T_i' = \begin{cases} T_1 \cdot \lfloor T_i/T_1 \rfloor & \text{if } T_i \geq T_1 \\ \frac{T_1}{\lceil T_1/T_i \rceil} & \text{if } T_i < T_1 \end{cases} \tag{16}$$

where $2 \leq i \leq N$.

Han *et al.* [21] also proved that such transformation does not change the schedulability of the given task set as shown in the following theorem.

Theorem 4. *[21] Given a task set Γ, if there exists another task set Γ' such that $T_i' \leq T_i$ and $C_i' = C_i$, for $1 \leq i \leq N$, and Γ' is schedulable by RM, then Γ is also schedulable by RM.* □

For the problem we are to address (defined in Section III-C), the smallest task period in the task set and the resource must satisfy *Constraint 3*, i.e., $2\Pi - \Theta \leq T_{\min}$, which implies that $\Pi < T_{\min}$ holds. Based on (16) and *Constraint 3*, we give the definition of a task's harmonic transformation with respect to a resource.

Definition 4. [Task Harmonic Transformation (τ')] *Given a task $\tau = (T, C)$ and a periodic resource $\gamma = (\Pi, \Theta)$, let $\tau' = (T', C)$ where the period is*

$$T' = \Pi \cdot \lfloor T/\Pi \rfloor \tag{17}$$

Then τ' is called task τ's harmonic transformation with respect to resource γ. □

According to Theorem 3, tasks and the resource must be pairwise harmonic. When assigning more than one task to the same resource, the harmonic transformation must be not only harmonic with respect to the resource period, but also harmonic with respect to the periods of the tasks that are already assigned to the resource. We use a recursive definition to define a task set's harmonic transformation with respect to a given resource.

Definition 5. [Task Set Harmonic Transformation (Γ')] *Given a task set $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_N\}$ and a periodic resource $\gamma = (\Pi, \Theta)$,*

$$\Gamma' = \{\tau_i' = (T_i', C_i) | \forall i \ 1 \leq i \leq N\}$$

is task set Γ's harmonic transformation with respect to resource γ, where

$$T_1' = \Pi \cdot \lfloor T_1/\Pi \rfloor \tag{18}$$

$$T_i' = \begin{cases} T_{i-1}' \cdot \lfloor T_i/T_{i-1}' \rfloor & \text{if } T_i \geq T_{i-1}' \\ \frac{T_{i-1}'}{\lceil T_{i-1}'/T_i \rceil} & \text{if } T_i < T_{i-1}' \end{cases}, \quad 2 \leq i \leq N \tag{19}$$

□

In Definition 5, (18) guarantees the first task is harmonic with the periodic resource, and (19) guarantees that tasks, except the first one, are all harmonic with prior tasks assigned to the resource. We use an example to illustrate the task set harmonic transformation.

Example 1. *Given a task set $\Gamma = \{\tau_1 = (13, 2), \tau_2 = (25, 4), \tau_3 = (20, 3)\}$, and a periodic resource $\gamma = (6, 4)$, we are to compute Γ's harmonic transformation with respect to γ according to Definition 5.*

The period of τ_1's harmonic transformation τ_1' is calculated by (18), so $\tau_1' = (12, 2)$. Then use (19) to calculate T_2' ($= 12 \cdot \lfloor 25/12 \rfloor$ = 24) and T_3' ($= \frac{24}{\lceil 24/20 \rceil} = 12$), and obtain $\tau_2' = (24, 4)$ and $\tau_3' = (12, 3)$.

The harmonic transformation of Γ with respect to γ is $\Gamma' = \{(12, 2), (24, 4), (12, 3)\}$. After the transformation, tasks in Γ' and the resource γ are pairwise harmonic. □

Theorem 5. *Given a periodic resource $\gamma = (\Pi, \Theta)$, a task set Γ, and its harmonic transformation Γ' with respect to γ, the task set Γ is schedulable on resource γ under RM scheduling if $U_{\Gamma'} \leq \Theta/\Pi$.* □

Proof. Definition 5 indicates $T_i' \leq T_i$ and $C_i' = C_i$, for $1 \leq i \leq N$. Based on Definition 5 and Theorem 4, under RM scheduling, if Γ' is schedulable, then Γ is also schedulable.

According to Theorem 3, $U_{\Gamma'} \leq \Theta/\Pi$ indicates that Γ' is schedulable on γ under RM scheduling. Hence, the task set Γ is schedulable on resource γ by RM. ■

Definition 6. [Task and Resource Harmonicity ($H(\tau, \gamma)$)] *Given a task $\tau = (T, C)$ and a periodic resource $\gamma = (\Pi, \Theta)$, assume the task's harmonic transformation with respect to the resource is $\tau' = (T', C)$. The harmonicity between task τ and resource γ is defined as*

$$H(\tau, \gamma) = \frac{T'}{T}. \tag{20}$$

□

Definition 7. [Task Set and Resource Set Harmonicity ($H(\Gamma, \mathcal{R})$)] *Given a task set $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_N\}$ and a periodic resource set $\mathcal{R} = \{\gamma_1, \gamma_2, \ldots, \gamma_M\}$, the harmonicity between task set Γ and resource set $\mathcal{R}$ is defined as the average value of each task's harmonicity with every resource, i.e.,*

$$H(\Gamma, \mathcal{R}) = \frac{\sum_{\tau_i \in \Gamma, \gamma_j \in \mathcal{R}} H(\tau_i, \gamma_j)}{N \cdot M}. \tag{21}$$

□

Lemma 3. *The harmonicity $H(\tau,\gamma)$ of a task $\tau=(T,C)$ and a periodic resource $\gamma=(\Pi,\Theta)$ is in the range $(0.5,1]$, i.e., $0.5<H(\tau,\gamma)\leq 1$.* □

Proof. By the harmonic transformation definition and harmonicity definition, we have $T'\leq T$, hence $H(\tau,\gamma)\leq 1$.

We prove $H(\tau,\gamma)>0.5$ by contradiction. Since the period of a task is a positive number, the harmonicity must be larger than 0. Assume to the contrary, we have $0<H(\tau,\gamma)\leq 0.5$. According to the harmonic transformation definition, assume $T'=K\cdot\Pi$, where $K\in\mathbb{N}^+$. By the harmonicity definition, we have the following inequality

$$H(\tau,\gamma)=\frac{T'}{T}=\frac{K\cdot\Pi}{T}\leq 0.5 \tag{22}$$

hence

$$T\geq 2K\cdot\Pi \tag{23}$$

By Definition 4 and (23), the period of τ' should be at least $2K\cdot\Pi$, which contradicts the assumption that $T'=K\cdot\Pi$. Hence, the assumption $0<H(\tau,\gamma)\leq 0.5$ does not hold, i.e., $H(\tau,\gamma)>0.5$.

Therefore, we prove that $0.5<H(\tau,\gamma)\leq 1$. ■

For a task τ, the utilization of its harmonic transformation is

$$U_{\tau'}=C/T'=U_\tau/H(\tau,\gamma) \tag{24}$$

and the utilization increment is

$$\Delta U_\tau=U_{\tau'}-U_\tau=(\frac{1}{H(\tau,\gamma)}-1)\cdot U_\tau \tag{25}$$

which defines the utilization difference between a task and its harmonic transformation.

Lemma 4. *Given a task τ and a periodic resource γ, the utilization increment caused by task harmonic transformation is less than 100%, i.e.,*

$$0\leq\frac{\Delta U_\tau}{U_\tau}<1. \tag{26}$$

□

Proof. It can be directly derived from Lemma 3 and formula (25). ■

It is not difficult to see that there is a trade-off: when a task set is transformed to a harmonic task set with respect to the given periodic resource, the transformed task set can utilize the resource to its capacity with guaranteed schedulibility under RM. But on the other hand, the transformation itself increases the task set's utilization. Therefore, to reduce the cost of utilizing the harmonicity property, we need to select tasks and periodic resources that are best harmonically fit.

V. Best-Harmonically-Fit (BHF) Task Assignment Algorithm

As discussed in Section IV, the *harmonicity* measures how harmonically related a task (set) is to a resource (set). The higher the harmonicity, the smaller the task utilization increment ΔU_τ caused by harmonic transformation. In this section, we present the Best-Harmonically-Fit (BHF) task assignment algorithm that utilizes the harmonicity characteristics between tasks and periodic resources to maximize the periodic resource utilization rate.

When assigning tasks to a resource, we have to also ensure that all tasks assigned to the resource are schedulable, i.e., the total task utilization must not exceed the utilization bound. As we have discussed in the previous section, Shin's utilization bound applies to a general task set, and is therefore relatively low. The harmonic utilization bound only applies to a harmonic task set that is also harmonic to the resources. Only when the harmonicity condition holds, can the task set fully, i.e., 100%, utilize the resource's capacity. For an arbitrary task set, in order to take advantage of the higher harmonic utilization bound, the task set has to be transformed to a harmonic task set. However, such transformation may result in an increased task utilization. Hence, to guarantee schedulability and also maximize the resource utilization rate, when deciding if a task set's utilization exceeds the resource's capacity, both Shin's utilization bound (7) and the harmonic utilization bound (14) are checked. The task set is schedulable on the resource if either bound is satisfied. Theorem 6 gives the combined schedulability condition.

Theorem 6. *Given a periodic resource γ, and a set of tasks Γ_γ assigned to the resource γ. For a new task τ, let τ' and Γ'_γ be the harmonic transformations of task τ and task set Γ_γ with respect to γ, respectively, the task τ is schedulable on γ if the following schedulability condition $SC(\tau,\gamma)$ is satisfied:*

$$SC(\tau,\gamma):U_{\Gamma'_\gamma}+U_{\tau'}\leq \texttt{HUB}_\gamma\vee U_{\Gamma_\gamma}+U_\tau\leq \texttt{UB}_\gamma \tag{27}$$

□

Proof. The conclusion can be directly derived from Theorem 1 and Theorem 3. ■

We use an example to illustrate that both bounds are needed in deciding task schedulability on a given resource.

Example 2. *Given a periodic resource $\gamma=(7,5)$, assume there is no other task assigned to the resource. We need to decide if a task τ can be assigned to the resource.*

***Case** 1: $\tau=(12,5.2)$*

The task's utilization $U_\tau=5.2/12$ is smaller than Shin's utilization bound $\texttt{UB}_\gamma=5/9$. Hence, it is schedulable on the resource. However, task τ's harmonic transformation with respect to γ is $\tau'=(7,5.2)$, with utilization $U_{\tau'}=5.2/7$ which is larger than the resource's harmonic utilization bound, i.e., resource capacity $\texttt{HUB}_\gamma=5/7$. In other words, task τ's harmonic transformation is not schedulable on the given resource. In this case, we need to use Shin's utilization bound (Theorem 1) to decide schedulability.

***Case 2:** $\tau=(15,9)$*

The task's utilization $U_\tau=9/15$ is larger than Shin's utilization bound $\texttt{UB}_\gamma=5/9$. Hence, it is not schedulable on the resource according to Shin's bound. However, task τ's harmonic transformation with respect to γ is $\tau'=(14,9)$, with utilization $U_{\tau'}=9/14$ which is smaller than the resource's harmonic utilization bound $\texttt{HUB}_\gamma=5/7$. In other words, task τ's harmonic transformation is schedulable on the resource.

Hence, τ is also schedulable on γ. In this case, we need to use the harmonic utilization bound (Theorem 3) to decide schedulability.

***Case** 3: $\tau = (15, 5.2)$*

The task's utilization $U_\tau = 5.2/15$ is smaller than Shin's utilization bound $\texttt{UB}_\gamma = 5/9$. Hence, it is schedulable on the resource. Furthermore, task τ's harmonic transformation with respect to γ is $\tau' = (14, 5.2)$, with utilization $U_{\tau'} = 5.2/14$ which is also smaller than the resource's harmonic utilization bound $\texttt{HUB}_\gamma = 5/7$. In other words, task τ's harmonic transformation is also schedulable on the resource. In this case, we can use either Shin's utilization bound (Theorem 1) or the harmonic utilization bound (Theorem 3) to check schedulability. □

When a heuristic approach is used to assign tasks to periodic resources, the order in which tasks are assigned may impact the resource utilization rate. We use an example to illustrate this.

Example 3. *Given a task set $\Gamma = \{\tau_1 = (13, 3), \tau_2 = (23, 8), \tau_3 = (27, 6), \tau_4 = (17, 0.5)\}$ and a resource set $\mathcal{R} = \{\gamma_1 = (6, 3), \gamma_2 = (5, 2), \gamma_3 = (7, 3.5)\}$, we assign the task set Γ to the resource set $\mathcal{R}$.*

If we first assign task τ_2 to resource γ_1, then the optimal way to assign the rest of the tasks is to assign τ_3 and τ_4 to γ_2, and τ_1 to γ_3. The resource utilization rate of such assignment is 59.3%. However, if we first assign task τ_1 to resource γ_1, the optimal way to assign the rest of the tasks is to assign τ_3 to γ_1, and τ_2 and τ_4 to γ_3. Such assignment only uses two resources and increases the resource utilization rate to 83%. □

In this example, it is not difficult to see that τ_1 and γ_1 have the highest harmonicity among all task and resource pairs. The example also indicates that assigning the most harmonically related task and resource pair first leads to a higher resource utilization rate. Based on the observations and the discussion in Section IV that assigning tasks to their most harmonically related resources can improve the resource utilization rate, we present the Best-Harmonically-Fit task set assignment algorithm. We first introduce the concept of Best-Harmonically-Fit-Task (BHFT) and Best-Harmonically-Fit-Pair (BHFP). Then, we show how to find the BHFT and the BHFP, respectively.

Definition 8. *Given a task set $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_N\}$ and a periodic resource γ, the Best-Harmonically-Fit-Task $\texttt{BHFT}(\Gamma, \gamma)$ with respect to resource γ is τ_i, i.e., $\texttt{BHFT}(\Gamma, \gamma) = \tau_i$, if task τ_i satisfies the following condition:*

$$\begin{aligned}\tau_i \in \Gamma \wedge SC(\tau_i, \gamma) \\ \wedge \forall j\ 1 \le j \ne i \le N\ H(\tau_i, \gamma) \ge H(\tau_j, \gamma) \\ \wedge H(\tau_i, \gamma) = H(\tau_j, \gamma) \rightarrow U_{\tau_i} \ge U_{\tau_j}\end{aligned} \tag{28}$$

□

In other words, the $\texttt{BHFT}(\Gamma, \gamma)$ is the task in the task set Γ that satisfies: (1) it is schedulable on the resource, i.e., $SC(\tau_i, \gamma) = \texttt{true}$; (2) it has the highest harmonicity with resource γ, i.e., $\forall j\ 1 \le j \ne i \le N\ H(\tau_i, \gamma) \ge H(\tau_j, \gamma)$; (3) if more than one task in the task set has the same harmonicity with the given resource, it has the highest utilization, i.e., $\forall j\ 1 \le j \ne i \le N\ H(\tau_i, \gamma) = H(\tau_j, \gamma) \rightarrow U_{\tau_i} \ge U_{\tau_j}$.

We extend the best harmonically fit task with respect to a given resource to a given resource set and have the best harmonically fit task and resource pair.

Definition 9. *Given a task set $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_N\}$ and a resource set $\mathcal{R} = \{\gamma_1, \gamma_2, \ldots, \gamma_M\}$, the Best-Harmonically-Fit-Pair $\texttt{BHFP}(\Gamma, \mathcal{R})$ is (τ_i, γ_j), i.e., $\texttt{BHFP}(\Gamma, \mathcal{R}) = (\tau_i, \gamma_j)$, if the pair (τ_i, γ_j) satisfies the following condition:*

$$\begin{aligned}\tau_i \in \Gamma, \gamma_j \in \mathcal{R} \\ \wedge SC(\tau_i, \gamma_j) \\ \wedge \forall m \forall k\ 1 \le m \ne i \le N 1 \le k \ne j \le M \\ H(\tau_i, \gamma_j) \ge H(\tau_m, \gamma_k) \\ \wedge H(\tau_i, \gamma_j) = H(\tau_m, \gamma_k) \rightarrow U_{\tau_i} \ge U_{\tau_m}\end{aligned} \tag{29}$$

□

Algorithm 1 and Algorithm 2 give the pseudo code that find $\texttt{BHFT}(\Gamma, \gamma)$ and $\texttt{BHFP}(\Gamma, \mathcal{R})$, respectively. In Algorithm 1, the `for` loop (Line 4 to Line 10) compares all tasks in the task set against the given resource γ and finds the maximum $H(\tau_i, \gamma)$ and U_{τ_i}. The complexity of the algorithm is $O(N)$.

Algorithm 2 calls Algorithm 1 to find the BHFT for each resource in the resource set, and selects the pair of task and resource that has the maximum harmonicity and task utilization. The complexity of Algorithm 2 is $O(NM)$.

Algorithm 1 SEARCH-BHFT(Γ, γ)

Input: A task set $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_N\}$ and a periodic resource γ.
Output: The BHFT $\tau_{\texttt{BHF}}$.

```
1: τ_BHF ← NULL
2: H_max ← 0
3: U_max ← 0
4: for i ← 1 to N do
5:    if SC(τ_i, γ) ∧ (H(τ_i, γ) > H_max ∨ (H(τ_i, γ) = H_max ∧
      U_τi > U_max)) then
6:       τ_BHF ← τ_i
7:       H_max ← H(τ_i, γ)
8:       U_max ← U_τi
9:    end if
10: end for
11: return τ_BHF
```

Once the Best-Harmonically-Fit-Task and Best-Harmonically-Fit-Pair are found, we are ready to introduce the Best-Harmonically-Fit (BHF) task assignment algorithm shown in Algorithm 3. In Algorithm 3, the `for` loop (Line 1 to Line 4) calculates the initial harmonicity value of each pair of tasks and resources. If there are tasks in the task set Γ, find the Best-Harmonically-Fit-Pair (Line 7). Once we have the best harmonically fit pair (τ, γ), assign task τ to resource γ (Line 9), remove the task from the given task set (Line 10), and update the harmonicity value between each unassigned task and resource γ (Line 11 to Line 13). Once resource γ is used, assign remaining tasks that are best harmonically fit to

Algorithm 2 SEARCH-BHFP($\Gamma, \mathcal{R}$)

Input: A task set $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_N\}$ and a periodic resource set $\mathcal{R} = \{\gamma_1, \gamma_2, \ldots, \gamma_M\}$.
Output: The BHFP $(\tau_{\texttt{BHF}}, \gamma_{\texttt{BHF}})$.

1: $\tau_{\texttt{BHF}} \leftarrow$ NULL
2: $\gamma_{\texttt{BHF}} \leftarrow$ NULL
3: $H_{\max} \leftarrow 0$
4: $U_{\max} \leftarrow 0$
5: **for** $i = 1$ to M **do**
6: $\quad \tau_{tmp} \leftarrow$ SEARCH-BHFT(Γ, γ_i)
7: $\quad$ **if** $H(\tau_{tmp}, \gamma_i) > H_{\max} \vee (H(\tau_{tmp}, \gamma_i) = H_{\max} \wedge U_{\tau_{tmp}} > U_{\max})$ **then**
8: $\quad\quad \tau_{\texttt{BHF}} \leftarrow \tau_{tmp}$
9: $\quad\quad \gamma_{\texttt{BHF}} \leftarrow \gamma_i$
10: $\quad\quad H_{\max} \leftarrow H(\tau_{tmp}, \gamma_i)$
11: $\quad\quad U_{\max} \leftarrow U_{\tau_{tmp}}$
12: $\quad$ **end if**
13: **end for**
14: **return** $(\tau_{\texttt{BHF}}, \gamma_{\texttt{BHF}})$

resource γ until γ is full and then remove resource γ from the resource set $\mathcal{R}$ (Line 14 to Line 23). The complexity of Algorithm 3 is $O(NM)$.

Algorithm 3 BHF($\Gamma, \mathcal{R}$)

Input: A task set $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_N\}$ and a resource set $\mathcal{R} = \{\gamma_1, \gamma_2, \ldots, \gamma_M\}$.

1: **for** $\tau_i \in \Gamma \wedge \gamma_j \in \mathcal{R}$ **do**
2: $\quad T'_i \leftarrow \Pi_j \cdot \lfloor T_i / \Pi_j \rfloor$
3: $\quad H(\tau_i, \gamma_j) \leftarrow T'_i / T_i$
4: **end for**
5: $\tau \leftarrow$ NULL, $\gamma \leftarrow$ NULL
6: **while** $\Gamma \neq \emptyset$ **do**
7: $\quad (\tau, \gamma) \leftarrow$SEARCH-BHFP($\Gamma, \mathcal{R}$)
8: $\quad \Gamma'_\gamma \leftarrow \emptyset$
9: $\quad \tau \mapsto \gamma$
10: $\quad$ Remove τ from Γ
11: $\quad \tau' \leftarrow (T \cdot H(\tau, \gamma), C)$
12: $\quad \Gamma'_\gamma \leftarrow \Gamma'_\gamma \cup \tau'$
13: $\quad$ UPDATE-H($\Gamma, \gamma, \Gamma'_\gamma$)
14: $\quad \tau \leftarrow$SEARCH-BHFT(Γ, γ)
15: $\quad$ **while** $\tau! =$ NULL **do**
16: $\quad\quad \tau \mapsto \gamma$
17: $\quad\quad$ Remove τ from Γ
18: $\quad\quad \tau' \leftarrow (T \cdot H(\tau, \gamma), C)$
19: $\quad\quad \Gamma'_\gamma \leftarrow \Gamma'_\gamma \cup \tau'$
20: $\quad\quad$ UPDATE-H($\Gamma, \gamma, \Gamma'_\gamma$)
21: $\quad\quad \tau \leftarrow$SEARCH-BHFT($\Gamma, \gamma$)
22: $\quad$ **end while**
23: $\quad$ Remove γ from $\mathcal{R}$
24: **end while**

It is worth pointing out that in Algorithm 3, Line 13 and Line 20 update the harmonicity value between each unassigned task and the resource. According to Definition 5 and Definition 6, if a task is harmonically related to a resource, the task is also harmonically related to all the tasks assigned to the resource. Hence, we need to update tasks' harmonicity after each task assignment. The harmonicity update algorithm is shown in Algorithm 4.

In Algorithm 4, the pseudo code from Line 3 to Line 7 uses formula (19) to find the task's harmonic transformation that is not only harmonic with each task assigned to γ, but also harmonic with the resource. The complexity of the algorithm is $O(N)$.

Algorithm 4 UPDATE-H($\Gamma, \gamma, \Gamma'_\gamma$)

Input: A task set $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_N\}$, a periodic resource $\gamma = (\Pi, \Theta)$ and the harmonic transformation Γ'_γ of all tasks assigned to the resource with respect to γ.

1: **for** $i = 1$ to N **do**
2: $\quad \tau'_k \leftarrow$ the last task in Γ'_γ
3: $\quad$ **if** $T_i \geq T'_k$ **then**
4: $\quad\quad T'_i = T'_k \cdot \lfloor T_i / T'_k \rfloor$
5: $\quad$ **else**
6: $\quad\quad T'_i = \frac{T'_k}{\lceil T'_k / T_i \rceil}$
7: $\quad$ **end if**
8: $\quad H(\tau_i, \gamma) \leftarrow T'_i / T_i$
9: **end for**

We use an example to illustrate the BHF procedure.

Example 4. *Consider the resource set and task set given in Example 3. We are to use the proposed BHF algorithm to assign the task set to the resource set.*

We first calculate the harmonicity for each task and resource pair. The results are shown in Table I(a).

The Best-Harmonically-Fit-Pair in Table I(a) is (τ_1, γ_1). *Hence,* τ_1 *is assigned to* γ_1. *After assigning* τ_1, *the harmonicity between each unassigned task and* γ_1 *is updated and shown in Table I(b). From Table I(a) and Table I(b), it can be seen that the harmonicity of* τ_2 *decreases from* $18/23$ *to* $12/23$.

Then, we find the Best-Harmonically-Fit-Task for resource γ_1, *which is* τ_3. *Task* τ_3 *is assigned to resource* γ_1.

Once τ_3 *is assigned to* γ_1, γ_1 *does not have enough capacity to host* τ_2 *or* τ_4. *Hence, we find the next Best-Harmonically-Fit-Pair which is* (τ_2, γ_3), *and assign* τ_2 *to* γ_3. *The harmonicity of the only unassigned task* τ_4 *with respect to* γ_3 *is updated to* $7/17$ *after assigning* τ_2.

Finally, we assign τ_4 *to resource* γ_3. *The harmonicity between each task and each resource is shown in Table I(c).*

The task assignment resulting from our algorithm for this example matches the optimal assignment given in Example 3, with a resource utilization rate of 83%. □

The BHF algorithm is a heuristic and uses a sufficient utilization bound to check schedulability. The scenario where a task set violates the bound but is still schedulable exists and the necessary utilization bound for RM scheduling on periodic resources is yet to be found.

Next section, we experimentally evaluate how well it performs when it is compared with other heuristic approaches and the optimal solution.

TABLE I
HARMONICITY FOR Γ AND $\mathcal{R}$

(a) Initial State (Before Assignment)

$H(\tau_i, \gamma_j)$	γ_1	γ_2	γ_3
τ_1	12/13	10/13	7/13
τ_2	18/23	20/23	21/23
τ_3	24/27	25/27	21/27
τ_4	12/17	15/17	14/17

(b) After Assigning τ_1 to γ_1

$H(\tau_i, \gamma_j)$	γ_1	γ_2	γ_3
τ_1	12/13	10/13	7/13
τ_2	**12/23**	20/23	21/23
τ_3	24/27	25/27	21/27
τ_4	12/17	15/17	14/17

(c) Final State (Assignment Done)

$H(\tau_i, \gamma_j)$	γ_1	γ_2	γ_3
τ_1	12/13	10/13	7/13
τ_2	**12/23**	20/23	21/23
τ_3	24/27	25/27	21/27
τ_4	12/17	15/17	**7/17**

VI. PERFORMANCE EVALUATION

In this section, we empirically evaluate the performance of the proposed Best-Harmonically-Fit algorithm. We compare the BHF task assignment algorithm with three commonly used multiprocessor task assignment algorithms, namely Best-Fit (BF), First-Fit (FF), and Worst-Fit (WF) algorithms [15], [16], [17].

When BF, FF, and WF are used, Shin's utilization bound [11], i.e., formula (7), is used to decide if a task set assigned to a resource is schedulable.

- Best-Fit (BF) [15]: Assign task $\tau \in \Gamma$ to the periodic resource $\gamma \in \mathcal{R}$ so that the remaining capacity percentage is minimal after the assignment, i.e.,

$$U_{\Gamma_\gamma} + U_\tau \leq \mathtt{UB}_\gamma \quad \wedge$$
$$\frac{\mathtt{UB}_\gamma - U_{\Gamma_\gamma} - U_\tau}{U_\gamma} = \min\{\frac{\mathtt{UB}_{\gamma_j} - U_{\Gamma_{\gamma_j}} - U_\tau}{U_{\gamma_j}} | \forall \gamma_j \in \mathcal{R}\}$$

- First-Fit (FF) [16]: Assign task $\tau \in \Gamma$ to the first periodic resource $\gamma \in \mathcal{R}$ that satisfies τ's schedulability condition, i.e.,

$$U_{\Gamma_\gamma} + U_\tau \leq \mathtt{UB}_\gamma$$

- Worst-Fit (WF) [17]: Assign task $\tau \in \Gamma$ to the periodic resource $\gamma \in \mathcal{R}$ so that the remaining capacity percentage is maximal after the assignment, i.e.,

$$U_{\Gamma_\gamma} + U_\tau \leq \mathtt{UB}_\gamma \quad \wedge$$
$$\frac{\mathtt{UB}_\gamma - U_{\Gamma_\gamma} - U_\tau}{U_\gamma} = \max\{\frac{\mathtt{UB}_{\gamma_j} - U_{\Gamma_{\gamma_j}} - U_\tau}{U_{\gamma_j}} | \forall \gamma_j \in \mathcal{R}\}$$

The performance of a task assignment algorithm is evaluated in two aspects, i.e., (1) resource utilization rate $\mathtt{UR}_\mathcal{R}$, and (2) total number of periodic resources used $M_{\mathtt{used}}$. The higher the $\mathtt{UR}_\mathcal{R}$ and the smaller the $M_{\mathtt{used}}$, the better the performance of the algorithm.

In the following experiments, the task sets and the resource sets are generated using the UUniFast algorithm [26] which gives an unbiased distribution of utilization values.

A. Harmonicity Impact

This set of experiments is to evaluate harmonicity impact on the performance of task assignment algorithms.

Experiment Settings

- Task number: 10
- Task period range: $[11639628, 200,000,000]$
- Task utilization range: $[0.1, 1.0]$
- Resource number: 10
- Resource period: 11639628
- Resource capacity range: $[0.468, 1.0]$
- Resource set capacity: 6.5
- Harmonicity: varies in the range of $[0.55, 1.0]$ with step 0.05

The reason we choose such large task periods and resource period is to guarantee that for each harmonicity value in the set $\{0.55, 0.6, 0.65, \ldots, 1.00\}$, we are able to generate a resource set and a task set that satisfy the harmonicity requirement and *Constraint 1 to Constraint 3* given in Section III-C.

Based on the task set and resource set harmonicity definition, and the integer requirement of both task periods and resource periods, the resource period has to be dividable by every harmonicity value H in the set $\{0.55, 0.6, 0.65, \ldots, 1.00\}$. The smallest value that is divisible by $H \in \{0.55, 0.6, 0.65, \ldots, 1.00\}$ is 11639628. Hence, the resource period is set to be 11639628.

According to Constraint 3 in Section III-C, a task's period must be no less than a resource's period. To guarantee that a task period has enough available values to choose from with a given harmonicity, we set the task period range to be $[11639628, 200000000]$.

It is worth pointing out that the harmonicity value for a given task set and a given resource set is the average harmonicity among all tasks and all resources in the given sets. Hence, when resource periods are set to be constant and we guarantee that the harmonicity between a task and any resource in the sets is the same, then the harmonicity between the task set and the resource set is the same as harmonicity between the task set and a resource in the sets.

Testing Procedure

The following steps are used to generate valid task sets and resource sets:

- The UUniFast algorithm [26] is used to generate resource sets which contain 10 resources with total capacity $U_\mathcal{R}$ equal to 6.5 and each resource's capacity U_{γ_j} within $[0.468, 1.0]$.
- Resource period Π_j is set as 11639628, and resource allocation time Θ_j is set as $\Pi_j \cdot U_{\gamma_j}$.
- The UUniFast algorithm [26] is used to generate task sets with 10 tasks that satisfy the following two conditions: (1) each task's utilization U_{τ_i} is within $[0.1, 1.0]$, and (2) the task set satisfies *Constraint 1, Constraint 2,* and *Constraint 3* given in Section III-C.
- We fix the utilization of each task and adjust task period T_i to be within $[11639628, 200000000]$ such that the harmonicity between the resource set and the task set is one of the values in the harmonicity variable set $\{0.55, 0.6, 0.65, \ldots, 1.00\}$.

- Task execution time C_i is set to $T_i \cdot U_{\tau_i}$.

We use the above generating procedure to randomly generate 200 resource sets and 10 task sets for each resource set. For each test case, we apply the BF, the FF, the WF, and the BHF algorithms to assign the generated tasks to resources. The average value of $200 * 10$ repeats is used to represent the performance of each algorithm.

Fig. 1(a) shows the resource utilization rate under different harmonicities. From Fig. 1(a), we have the following observations:

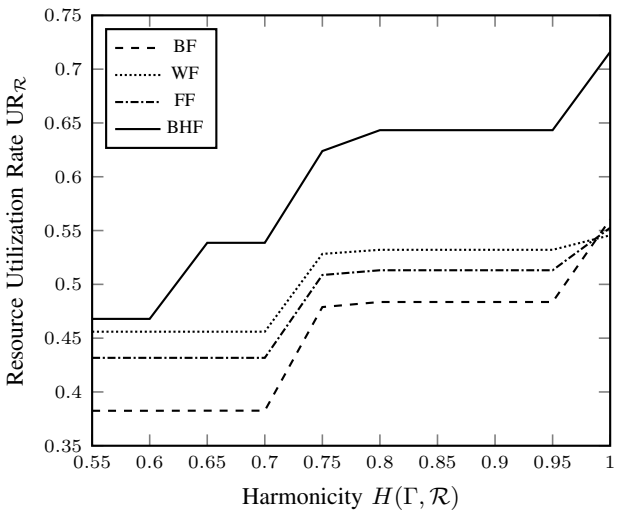
1) For all four task assignment algorithms, the resource utilization rate increases when harmonicity increases;
2) The BHF algorithm is more sensitive to harmonicity change than the other three algorithms;
3) The BHF algorithm results in a much higher resource utilization rate, as much as 40.77% more than the other three algorithms.

Fig. 1(b) depicts the number of periodic resources used by different algorithms. Similar observations can be made:

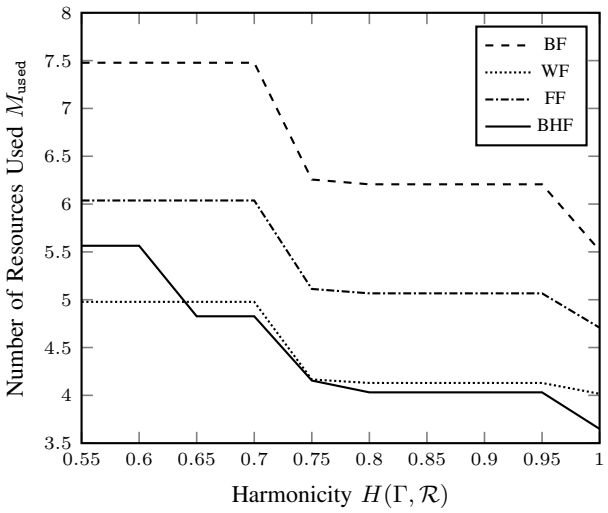
1) For all four task assignment algorithms, the number of periodic resources used decreases when harmonicity increases;
2) The BHF algorithm is more sensitive to harmonicity change than the other three algorithms;
3) The BHF algorithm needs less number of resources, as much as 54.92% less than the other three algorithms.

In addition, from Fig.1, we can also see that the BHF algorithm is more sensitive to harmonicity than the other three algorithms. The behavior is consistent with the design of the BHF algorithm, i.e., BHF is based on harmonicity. The sensitivity of BHF is demonstrated with the following two aspects:

1) discrepancy of the resource utilization rate between BHF and the other three algorithms increases with an increase in harmonicity,
2) BHF has a larger increase of the resource utilization rate than the other three algorithms.



(a) Utilization Rate under Different Harmonicity



(b) Number of Resources Used under Different Harmonicity

Fig. 1. Harmonicity Impact

B. Task Set Utilization Impact

The second set of experiments evaluates the performance of the proposed BHF task assignment algorithm under different task set utilizations. The experiment parameters are given below.

Experiment Settings

- Task number: 10
- Task period range: $[20, 50]$
- Task utilization range: $[0.1, 1.0]$
- Task set utilization: random
- Resource number: 10
- Resource period range: $[1, 20]$
- Resource capacity range: $[0.325, 1.0]$
- Resource set capacity: 6.5

Testing Procedure

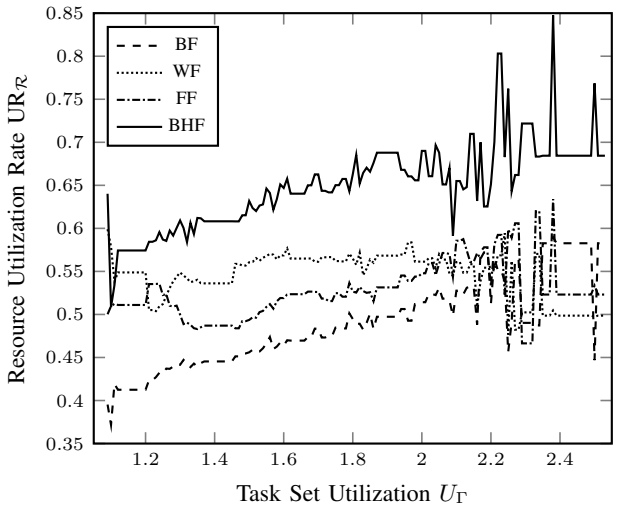
The following steps are taken to generate valid task sets and resource sets:

- The UUniFast algorithm [26] is used to generate a resource set with 10 individual periodic resources with total capacity $U_{\mathcal{R}}$ equal to 6.5 and each resource's capacity U_{γ_j} within $[0.325, 1.0]$.
- Resource period Π_j is randomly selected from $[1, 20]$, and resource allocation time Θ_j is set to $\Pi_j \cdot U_{\gamma_j}$.
- The UUniFast algorithm [26] is used to generate a task set with 10 tasks that satisfy the following two conditions: (1) each task's utilization U_{τ_i} is within $[0.1, 1.0]$, and (2) the task set satisfies the three constraints in Section III-C.
- Task period T_i is randomly selected from $[20, 50]$, and task execution time C_i is set to $T_i \cdot U_{\tau_i}$.

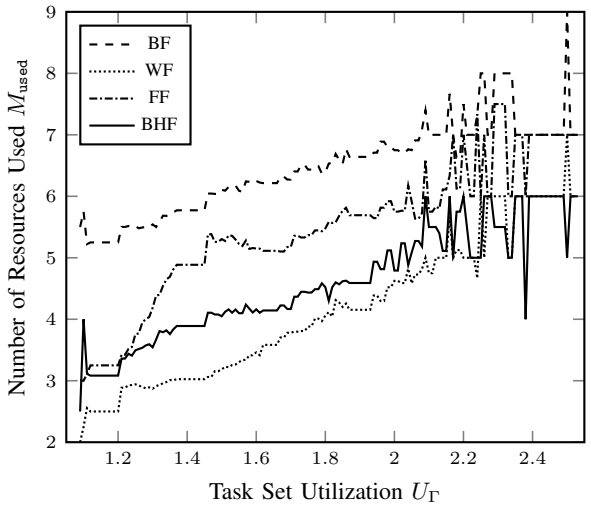
We use the above generating procedure to randomly generate 200 resource sets and 100 task sets for each resource set. For each test case, we apply the four different task assignment algorithms. We run the $200 * 100$ test cases, and combine the results based on task set utilizations rounded to the nearest hundredth. The average value is used to represent the performance of each algorithm.

Fig. 2(a) depicts the average resource utilization rate under different task set utilizations. As shown in Fig. 2(a), in general, when the task set utilization increases, the resource utilization rate also increases. Among the four task assignment algorithms, the BHF algorithm has the highest resource utilization rate. The resource utilization rate resulting from the BHF algorithm is always above 50.86%. When task set utilization is above 2, variations increase, and the average resource utilization rate for BHF is 64.95%, while BF is 48.99%, FF is 52.73%, and WF is 54.49%.

Fig. 2(b) depicts the number of periodic resources used under different task set utilizations. From Fig. 2(b), we observe that BF uses the most number of resources, which is consistent with the observation that BF has the lowest resource utilization rate. The WF, on the other hand, uses less number of resources than the proposed BHF approach, but also has a lower utilization rate, which seems to be counter intuitive.



(a) Utilization Rate



(b) Number of Resources Used

Fig. 2. Task Set Utilization Impact ($\frac{\Theta}{\Pi} \in [0.325, 1.0]$)

Further study reveals that since the WF algorithm always selects the largest resource to assign tasks to, when resource capacity is relatively small, the remaining portion of the resource resulting from the WF algorithm would have a higher possibility than other approaches to host other tasks. Therefore, the WF algorithm results in less number of resources used. To verify this reasoning, we repeat the above experiments but with a higher capacity resource set: each resource capacity is in the range of $[0.8, 1.0]$ and we do not fix the total resource set capacity. The results are shown in Fig. 3.

As shown in Fig. 3, BHF has better performance (higher resource utilization rate and smaller number of resources used) than the other three algorithms. When task set utilization is between 2.5 and 5.0, the advantages of the BHF algorithm over the other three approaches are significant. In particular, the BHF algorithm uses 15.49% less number of resources than the other three approaches on average. Another observation from Fig. 3 is that the performance difference among BF, FF, and WF algorithms is small when individual resource capacity is large. The experiment also confirms our observation that when individual resource capacity is small, the WF algorithm may have some advantage with respect to the number of resources needed.

C. BHF and Optimal Solution Comparison

The third set of experiments compares the performance of the proposed BHF task assignment algorithm with the optimal solution obtained by brute-force search. The experiment parameters are given below.

Experiment Settings

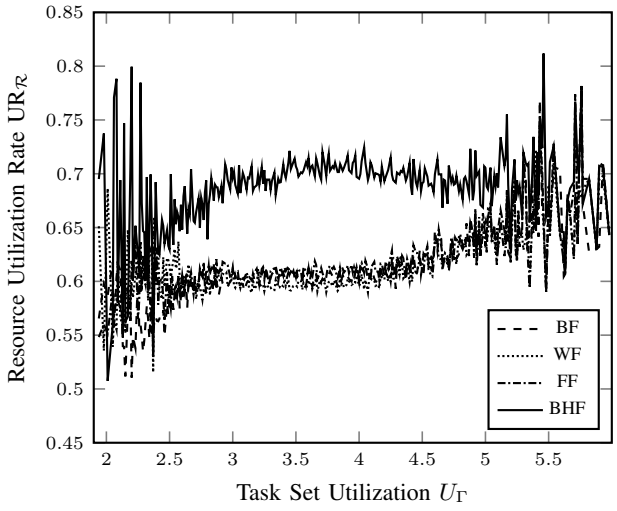
- Task number: 3
- Task period range: $[20, 50]$
- Task utilization range: $[0.1, 1.0]$
- Task set utilization: random
- Resource number: 3
- Resource period range: $[1, 20]$
- Resource capacity range: $[0.325, 1.0]$
- Resource set capacity: 1.95

Testing Procedure

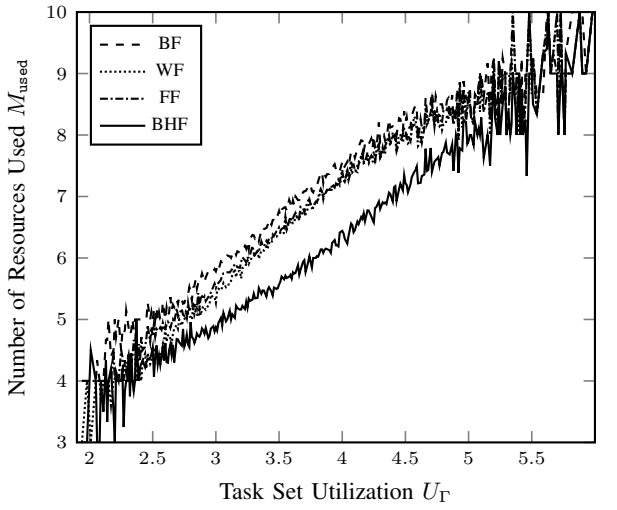
In this experiment, the task set and resource set generating procedure is the same as in Section VI-B except that the parameter ranges are different. We run the $20,000$ test cases, and combine the results based on task set utilizations rounded to the nearest hundredth. The average value is used to represent the performance of each algorithm.

For each test case, we apply the four different task assignment algorithms, and consider all possible task assignments. We choose the one with the largest resource utilization rate as the optimal solution. It is worth noting that when searching for the optimal solution, we use both Shin's utilization bound (Theorem 1) and the harmonic utilization bound (Theorem 3) to check schedulability. We consider tasks to be schedulable if either bound is satisfied.

Fig. 4(a) depicts the average resource utilization rate. Compared with the optimal solution, on average, the BHF, BF, FF, and WF result in a 13.02%, 18.94%, 26.82%, and 21.81% lower resource utilization rate, respectively.

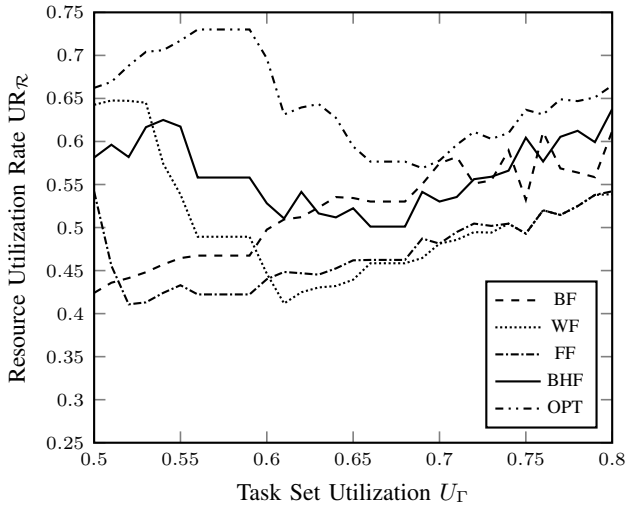


(a) Utilization Rate

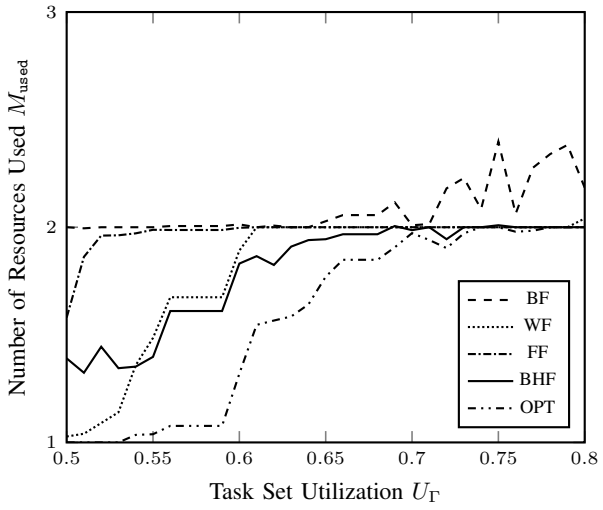


(b) Number of Resources Used

Fig. 3. Task Set Utilization Impact ($\frac{\Theta}{\Pi} \in [0.8, 1.0]$)



(a) Utilization Rate



(b) Number of Resources Used

Fig. 4. BHF and Optimal Solution Comparison

Fig. 4(b) depicts the number of periodic resources used. Compared with the optimal solution, on average, the BHF, BF, FF, and WF use 18.47%, 40.95%, 34.47%, and 17.16% more number of resources, respectively.

In addition, from an average performance aspect, the WF uses less number of resources than the proposed BHF approach, but also has a lower utilization rate. This scenario is studied in the second experiment of Section VI-B (Fig. 3).

VII. Conclusion

Periodic resource models and their scheduling problems have drawn more attention in real-time community in recent years. However, to our best knowledge, there has not been much work, if any, in the literature dealing with the task assignment problem on multiple periodic resources. In this paper, we study the task assignment problem in the context of assigning multiple periodic tasks to multiple periodic resources. Specifically, we first study the harmonic properties between periodic tasks and periodic resources. We prove that if a harmonic task set is also harmonic with the resource, the task set can 100% utilize the resource's capacity under the RM scheduling algorithm. Then we propose a heuristic Best-Harmonically-Fit (BHF) task assignment algorithm to maximize the resource utilization rate based on the harmonic properties between periodic tasks and periodic resources. We compare the performance of Best-Harmonically-Fit with Best-Fit (BF), First-Fit (FF), Worst-Fit (WF) task assignment algorithms, and the optimal task assignment (found through exhaustive search for a small-sized task set and resource set). The experimental results show that, on average, the BHF results in a 32.58%, 23.17%, and 19.2% higher resource utilization rate than the Best-Fit (BF), the First-Fit (FF), and the Worst-Fit (WF) task assignment algorithms, respectively, and a 13.02% lower resource utilization rate than the optimal solution.

Acknowledgement

The research is supported in part by NSF under grant number CAREER 0746643 and CNS 1018731.

REFERENCES

[1] Aloysius K Mok, Xiang Feng, and Deji Chen. Resource partition for real-time systems. In *Real-Time Technology and Applications Symposium, 2001. Proceedings. Seventh IEEE*, pages 75–84. IEEE, 2001.

[2] Yu Li, Albert MK Cheng, and Aloysius K Mok. Regularity-based partitioning of uniform resources in real-time systems. In *Embedded and Real-Time Computing Systems and Applications (RTCSA), 2012 IEEE 18th International Conference on*, pages 368–377. IEEE, 2012.

[3] Jaewoo Lee, Sisu Xi, Sanjian Chen, Linh TX Phan, Chris Gill, Insup Lee, Chenyang Lu, and Oleg Sokolsky. Realizing compositional scheduling through virtualization. In *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2012 IEEE 18th*, pages 13–22. IEEE, 2012.

[4] S Shirero, Matsumoto Takashi, and Hiraki Kei. On the schedulability conditions on partial time slots. In *Real-Time Computing Systems and Applications, 1999. RTCSA'99. Sixth International Conference on*, pages 166–173. IEEE, 1999.

[5] Xiayu Hua, Hao Wu, Zheng Li, and Shangping Ren. Enhancing throughput of the hadoop distributed file system for interaction-intensive tasks. *Journal of Parallel and Distributed Computing*, 74(8):2770–2779, 2014.

[6] Xiayu Hua, Hao Wu, and Shangping Ren. Enhancing throughput of hadoop distributed file system for interaction-intensive tasks. In *Proceedings of the 2014 22nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, pages 508–511. IEEE Computer Society, 2014.

[7] Aloysius K Mok and Xiang Alex. Towards compositionality in real-time resource partitioning based on regularity bounds. In *Real-Time Systems Symposium, 2001.(RTSS 2001). Proceedings. 22nd IEEE*, pages 129–138. IEEE, 2001.

[8] Insik Shin and Insup Lee. Periodic resource model for compositional real-time guarantees. In *Real-Time Systems Symposium, 2003. RTSS 2003. 24th IEEE*, pages 2–13, Dec 2003.

[9] Arvind Easwaran, Insik Shin, Oleg Sokolsky, and Insup Lee. Incremental schedulability analysis of hierarchical real-time components. In *Proceedings of the 6th ACM &Amp; IEEE International Conference on Embedded Software*, EMSOFT '06, pages 272–281, New York, NY, USA, 2006. ACM.

[10] A. Easwaran, Insup Lee, Insik Shin, and O. Sokolsky. Compositional schedulability analysis of hierarchical real-time systems. In *Object and Component-Oriented Real-Time Distributed Computing, 2007. ISORC '07. 10th IEEE International Symposium on*, pages 274–281, May 2007.

[11] Insik Shin and Insup Lee. Compositional real-time scheduling framework with periodic model. *ACM Transactions on Embedded Computing Systems (TECS)*, 7(3):30:1–30:39, May 2008.

[12] Nathan Fisher and Farhana Dewan. Approximate bandwidth allocation for compositional real-time systems. In *Real-Time Systems, 2009. ECRTS'09. 21st Euromicro Conference on*, pages 87–96. IEEE, 2009.

[13] Farhana Dewan and Nathan Fisher. Approximate bandwidth allocation for fixed-priority-scheduled periodic resources. In *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2010 16th IEEE*, pages 247–256. IEEE, 2010.

[14] Xiayu Hua, Zheng Li, Hao Wu, and Shangping Ren. Scheduling periodic tasks on multiple periodic resources. In *International Conference on Advanced Communications and Computation, 2014. INFOCOMP 2014. 4th IARIA*. IARIA, 2014.

[15] Yingfeng Oh and Sang H. Son. Tight performance bounds of heuristics for a real-time scheduling problem. Technical report, Charlottesville, VA, USA, 1993.

[16] Sudarshan K. Dhall and C. L. Liu. On a real-time scheduling problem. *Operations Research*, 26(1):127–140, 1978.

[17] Nan Guan, M. Stigge, Wang Yi, and Ge Yu. Fixed-priority multiprocessor scheduling with liu and layland's utilization bound. In *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2010 16th IEEE*, pages 165–174, April 2010.

[18] C. L. Liu and James W. Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *J. ACM*, 20(1):46–61, jan 1973.

[19] A. Burchard, J. Liebeherr, Yingfeng Oh, and S.H. Son. New strategies for assigning real-time tasks to multiprocessor systems. *Computers, IEEE Transactions on*, 44(12):1429–1442, Dec 1995.

[20] Sylvain Lauzac, Rami Melhem, and Daniel Mossé. An efficient rms admission control and its application to multiprocessor scheduling. In *Parallel Processing Symposium(IPPS/SPDP), 1998*, pages 511–518. IEEE, 1998.

[21] Ching-Chih Han and Hung ying Tyan. A better polynomial-time schedulability test for real-time fixed-priority scheduling algorithms. In *Real-Time Systems Symposium, 1997. Proceedings., The 18th IEEE*, pages 36–45, Dec 1997.

[22] Björn Andersson, Sanjoy Baruah, and Jan Jonsson. Static-priority scheduling on multiprocessors. In *Real-Time Systems Symposium, 2001.(RTSS 2001). Proceedings. 22nd IEEE*, pages 193–202. IEEE, 2001.

[23] Ming Fan and Gang Quan. Harmonic-aware multi-core scheduling for fixed-priority real-time systems. *Parallel and Distributed Systems, IEEE Transactions on*, PP(99):1–11, 2013.

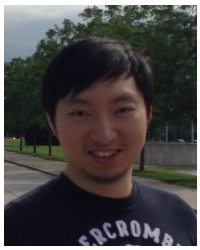
[24] Xiang Feng and Aloysius K Mok. A model of hierarchical real-time virtual resources. In *Real-Time Systems Symposium, 2002. RTSS 2002. 23rd IEEE*, pages 26–35. IEEE, 2002.

[25] Jan Korst, Emile Aarts, JanKarel Lenstra, and Jaap Wessels. Periodic multiprocessor scheduling. In Emile H.L. Aarts, Jan van Leeuwen, and Martin Rem, editors, *PARLE '91 Parallel Architectures and Languages Europe*, volume 505 of *Lecture Notes in Computer Science*, pages 166–178. Springer Berlin Heidelberg, 1991.

[26] Enrico Bini and GiorgioC. Buttazzo. Measuring the performance of schedulability tests. *Real-Time Systems*, 30(1-2):129–154, 2005.

Chunhui Guo is now a Ph.D candidate in the Computer Science Department at Illinois Institute of Technology. He received both B.E. and M.E. degrees in Electrical Engineering from Shandong University, China, in 2010 and 2013, respectively. His current research interests mainly focus on reliable real-time systems, real-time scheduling, distributed systems, and Cyber-Physical System.

Xiayu Hua is now a Ph.D candidate in the Computer Science Department at Illinois Institute of Technology. His research interests are in distributed file system, virtualization technology, real-time scheduling and cloud computing. He earned his B.S. degree from the Northwestern Polytechnic University, China, in 2008 and his M.S. degree from the East China Normal University, China, in 2012.



Hao Wu is now a Ph.D candidate in the Computer Science Department at Illinois Institute of Technology. He received B.E. in Information Security from Sichuan University, Chengdu, China, 2007. He received M.S. in Computer Science from University of Bridgeport, Bridgeport, CT, 2009. His current research interests mainly focus on cloud computing, real-time distributed open systems, Cyber-Physical System, parallel and distributed systems, and real-time applications.

Dr. Shangping Ren is an associate professor in the Computer Science Department at the Illinois Institute of Technology. She earned her Ph.D from UIUC in 1997. Before she joined IIT in 2003, she worked in software and telecommunication companies as a software engineer and then lead software engineer. Her current research interests include coordination models for real-time distributed open systems, real-time, fault-tolerant and adaptive systems, Cyber-Physical System, parallel and distributed systems, cloud computing, and application-aware many-core virtualization for embedded and real-time applications.