Hindawi
Complexity
Volume 2021, Article ID 5580631, 16 pages
https://doi.org/10.1155/2021/5580631

Research Article

Analysis and Design of the Battery Initial Energy Level with Task Scheduling for Energy-Harvesting Embedded Systems

Xingyu Miao, Jiayuan Wei, and Yongqi Ge

School of Information Engineering, Ningxia University, Yinchuan 750021, China

Correspondence should be addressed to Yongqi Ge; geyongqi@nxu.edu.cn

Received 5 February 2021; Revised 18 March 2021; Accepted 30 March 2021; Published 16 April 2021

Academic Editor: Yongsheng Hao

When the energy-harvesting embedded system (EHES) is running, its available energy (harvesting energy and battery storage energy) seems to be sufficient overall. However, in the process of EHES task execution, an energy shortage may occur in the busy period such that system tasks cannot be scheduled. We call this issue the energy deception (ED) of the EHES. Aiming to address the ED issue, we design an appropriate initial energy level of the battery. In this paper, we propose three algorithms to judge the feasibility of the task set and calculate the appropriate initial energy level of the battery. The holistic energy evaluation (HEE) algorithm makes a preliminary judgment of the task set feasibility according to available energy and consumption energy. A worst-case response time-based initial energy level of the battery (WCRT-IELB) algorithm and an accurate cycle-initial energy level of the battery (AC-IELB) algorithm can calculate the proper initial battery capacity. We use the YARTISS tool to simulate the above three algorithms. We conducted 250 experiments on As Late As Possible (ALAP) and As Soon As Possible (ASAP) scheduling with the maximum battery capacities of 50, 100, 200, 300, and 400. The experimental results show that setting a reasonable initial energy level of the battery can effectively improve the feasibility of the task set. Among the 250 task sets, the HEE algorithm filtered 2.8% of them as infeasible task sets. When the battery capacity is set to 400, the WCRT-BIEL algorithm increases the success rates of the ALAP and ASAP by 17.2% and 26.8%, respectively. The AC-BIEL algorithm increases the success rates of the ALAP and ASAP by 18% and 26.8%, respectively.

1. Introduction

In recent years, embedded systems are fast becoming a key proportion of computer science and technology in different domains, such as driverless vehicles, medical implants, weather monitoring sensors, wearable devices, and so on [1–6]. Most embedded devices are battery-powered. The battery life determines the embedded system running time. However, the battery capacity is limited. Some embedded systems that are deployed in distant areas require long-term operation [7, 8]. In this case, these systems require periodic battery replacement to maintain running time. Battery replacement, however, is very difficult in general. Energy harvesting provides new insights into this issue. This technology harvests ambient energy and converts it into electrical energy for direct use in an embedded system or stores it in a storage module (i.e., battery) for future uses. The benefit of this approach is that it can increase system running time and eliminate the demand of battery replacement [9]. We refer to an embedded system that uses energy-harvesting technology as an energy-harvesting embedded system (EHES). The major problem in EHES is to ensure that the task calculation of the embedded system can obtain enough energy. The EHES generally consists of three parts, as shown in Figure 1, in which the energy source (such as sunlight, wind power, and vibration energy) is converted into electrical energy by the energy harvester and passed to the energy storage device (battery) for storage. Because of the unpredictability of the energy source (such as a period of overcast rain or the unavailability of solar power during the night), the task of harvesting energy is uncertain. Thus, the converted electrical energy cannot power the system stably.

In this work, we consider the issue of ED arising from scheduling algorithms in EHES. The main reason for ED is

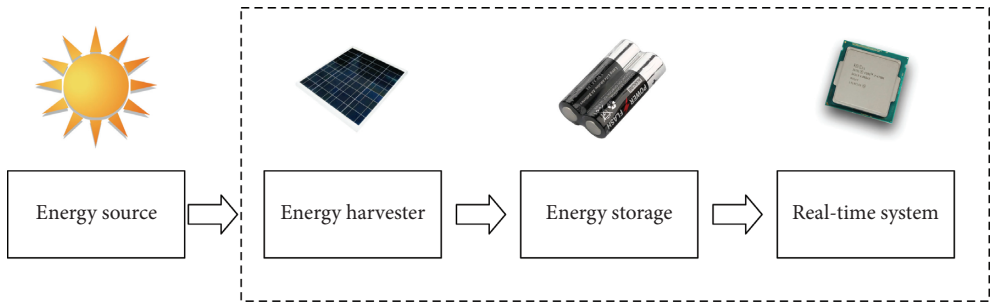

Figure 1: Energy-harvesting embedded system architecture.

that when the system carries out a hyperperiod task, the energy consumed is less than the energy stored in the battery plus the harvested energy, and the system will not stop because of the lack of energy; however, in this hyperperiod, the task may be executed very frequently during a certain period of time; at this time, the energy consumption is relatively large, and the harvested energy and the energy stored in the battery are insufficient to support this consumption, causing the system to stop running. This is very contradictory to the above situation. Through our research, we found that setting the proper initial energy level of the battery can effectively eliminate the ED issue. Setting the battery's initial energy can effectively eliminate the lack of energy in the process of task execution. We proposed an energy level judgment algorithm and two battery initial level calculation algorithms to solve this issue.

The contributions of this paper are as follows:

(1) In this work, we conduct the system schedulability analysis and find the necessary condition for the viability of the EHES, designing a task scheduling pre-judgment method based on task set attributes and energy production power for EHES, which can filter out those task sets that are not globally feasible. Among the 250 task sets, we filtered 2.8% of them as infeasible task sets.

(2) An analysis of the initial battery energy level issue of the battery is given in this work. The battery must have initial energy for some tasks to be schedulable. Based on the worst-case response time (WCRT) model, the WCRT-IELB algorithm is proposed, and the schedulability of the scheduling algorithm is effectively improved by setting the initial energy level. Simulations show that the introduction of the WCRT-IELB algorithm for ALAP and ASAP is better than that without the introduction of the WCRT-IELB algorithm, and when the battery capacity is set to 400, the success rate is increased by 17.2% and 26.8%, respectively.

(3) An online AC-IELB algorithm is proposed, which is more accurate than the initial energy calculation method based on WCRT. Experiments show that the success rate of the AC-IELB algorithm based on ALAP and ASAP is increased by 28%, and 26.8%, respectively, compared with the original algorithm when the battery capacity is set to 400.

(4) The experimental results show that the AC-IELB algorithm has higher successful rate than the WCRT-IELB algorithm. When the battery capacity is set to 400, the success rates of ALAP are increased by 0.8%.

The remainder of this paper is organized as follows. The related works are summarized in Section 2. In Section 3, we review the general model. Section 4 explains the research motivation, and we conduct system schedulability analysis in Section 5. Section 6 presents three algorithms and describes their rules in detail. Simulation results and discussions follow in Section 7. Finally, we conclude this work and provide some directions for future works in Section 8.

2. Related Work

In the past two decades, researchers began to address issues of minimizing energy consumption in scheduling. For EHES, many studies have focused on reducing energy consumption or optimizing battery storage efficiency under the premise of ideal battery capacity while ignoring the influence of battery capacity on the scheduling algorithm. In general, researchers focus on the following three points. There are many power management technologies; for instance, dynamic power management (DPM) [10–12] and dynamic voltage and frequency scaling (DVFS) [13–15] techniques are currently two well-known technologies. DPM selectively shuts down idle components in the process of system operation to achieve the purpose of energy savings. DVFS saves energy by reducing the CPU frequency and extending task execution times. When there is not enough energy to execute the task, these two technologies cannot be used, and they are difficult to use in energy-harvesting systems. The approach proposed by Balsamo et al. [16] has successfully solved this problem.

The strategy of battery energy storage and consumption is designed to extend battery life and achieve the purpose of extending system life. Most of the works use ideal battery and/or supercapacitor models; the current works consider more accurate battery and/or supercapacitor models. At the same time, it causes some very complex issues about prediction of harvestable energy and the battery and/or supercapacitor status. According to the current state of the system, the Highest/Lowest-Power-First (HLPF) real-time task scheduling algorithm proposed by Hasanlooa et al. will store electrical energy in the system as much as possible to avoid waste, thereby increasing the life of the system. And

they proposed hybrid energy storage system (HESS) component scheduling [17] which has a similar function. Furthermore, in [18], Kwak et al. researched the impact of task scheduling on battery aging, and the main principle of minimizing battery aging was proposed based on results.

The feasible scheduling algorithm is designed to extend the system running time. Allavena and Mossé [19] first focused on embedded systems with battery charging and deadline constraints. They proposed a simple and effective task scheduling method in a frame-based system of maximum and minimum energy constraints. However, this method needs to be carried out under a very strict task model in which all tasks have the same period and implicit period. Then, Moser et al. [20] proposed an algorithm called the Lazy Scheduling Algorithm (LSA), which relies on the energy consumption of the task to change the CPU frequency, thereby adjusting the WCRT. However, the result of this work heavily relies on assumptions and energy consumption directly related to WCRT, which is unrealistic for embedded systems [21]. Abdeddaïm et al. adopted the energy harvesting to address the energy and time constraint in the operation of embedded systems. They proposed two classic scheduling strategies, ASAP [22] and ALAP [23]. The ALAP delays the execution time of the task as much as possible and compresses the slack time as much as possible, enabling the systems to supplement the battery energy to the greatest extent. The ASAP algorithm judges whether the current energy level is sufficient to execute a time unit and if it can be executed, it will execute immediately. Otherwise, the systems will suspend a time unit to replenish energy and then judge. Abdeddaïm et al. [22] proved that the ASAP algorithm is optimal on a nonconcrete task set (a nonconcrete task set is a set of real-time tasks whose offset is only known at runtime). However, the ASAP algorithm will cause frequent switching of battery charge and discharge modes in order to perform tasks as much as possible, which will reduce battery life and thereby reduce system life. This is unrealistic. Afterward, Abdeddaïm et al. [24] extended the ASAP algorithm by incorporating the idea of clairvoyance, proposing an algorithm called FPCASAP. The purpose is to find the optimal algorithm for the concrete task set. However, so far, the algorithm has not been proven to be optimal for the concrete task set. Moreover, the LSA considers the battery capacity as an ideal situation when designing the battery model and sets an initial battery capacity that is always equal to the maximum battery capacity, which is unrealistic [25]. Abdeddaïm et al. [22, 26] evaluated two upper limits of the battery capacity of the fixed-priority scheduling algorithm by two tests. In general, it is difficult to calculate the minimum battery capacity performed by the system due to the consideration of environmental factors and the scheduling algorithm. In the first test, they considered that the ASAP algorithm can accurately obtain this value, since the energy replenished each time during the operation of the ASAP algorithm only needs to be equal to the energy consumed by the single time unit. In this case, the minimum battery capacity that is feasible to maintain the task set is the maximum energy consumption, that is, the maximum instantaneous energy consumption.

It is only necessary to ensure that the maximum battery capacity is not less than maximum instantaneous energy consumption to ensure the task set feasibility. For the second test, they consider that the maximum capacity of the battery is at least equal to the energy consumed by all tasks in the longest busy period of the priority n task. Such maximum battery capacity is equivalent to having unlimited battery capacity in terms of task execution; however, whether considering ASAP, ALAP, or FPCASAP, their battery capacity is unlimited, which is not realistic. Designing the proper battery capacity and initial battery capacity will increase the schedulability of the scheduling algorithm. Ghadaksaz et al. [27] first proposed the calculation method for the battery capacity of the EDF-ASAP algorithm, and simulation results verify that the proper battery capacity is an important issue affecting system task scheduling. To the best of our knowledge, there is no previous work in this era that gives computation methods for battery initial level. In this work, we propose two methods to compute battery initial levels (WCRT-IELB and AC-IELB).

3. Model

The EHES generally consists of two parts: the energy system and the real-time system. Correspondingly, we assume that our model also has two parts: the energy model and the task model. In this work, we consider every time interval as one time unit, while the energy unit depends on the real situation such as the type of energy, the rate of energy conversion, and the rate of energy harvesting.

3.1. Energy Model. In this work, the energy model of EHES consists of the energy production model and energy storage model. The available energy of EHES consists of harvesting energy and battery storage energy.

3.1.1. Energy Production Model. We suppose that ambient energy can be collected by a harvesting model to produce energy and convert it into electrical power with an instantaneous charging rate, denoted as $R_p(t)$. $R_p(t)$ is a function of time. The energy harvested during the time interval $[t_1, t_2)$ is denoted as $E_p(t_1, t_2) = \int_{t_1}^{t_2} R_p(t)\mathrm{d}t$. Based on [20, 22, 23], we make $R_p(t)$ to be a constant function and denote it as R. The energy harvesting during the time interval $[t_1, t_2)$ is denoted as $E_p(t_1, t_2) = (t_2 - t_1) \times R$ in the following. In addition, since the method proposed in this work is a general method, we only consider the consumption of electrical energy after energy conversion. Therefore, if the new energy is applied to the method proposed in this work, only this energy production model needs to be replaced.

3.1.2. Energy Storage Model. A battery is generally used as an energy storage device in a real-time embedded system. We suppose storage energy cannot be more than battery maximum capacity $C_{\max}$and use an ideal storage model that stores as much energy as is harvested, ignoring all losses. The energy charge of the energy storage unit at time t is expressed

as $E_s(t)$, and then $C_{\max} \geq E_s(t) \geq 0$ at any time t, where $E_s(0)$ is the initial energy level. The energy of the energy storage unit in the time interval $[t_1, t_2)$ is denoted as $E_s(t_1, t_2) = E_s(t_2) - E_s(t_1)$. When $E_s(t_1, t_2)$ is positive, it indicates that the energy storage unit is in the charging mode during the time interval $[t_1, t_2)$. In contrast, when $E_s(t_1, t_2)$ is negative, it indicates that the energy storage unit is in the discharge mode during the time interval $[t_1, t_2)$.

3.2. Task Model. In this work, the models and analysis methods used in this article are all oriented to fixed real-time embedded systems. In general, to ensure real-time performance, this system will not add new tasks; therefore, we consider a real-time system $P = \{\tau_1, \tau_2, \ldots, \tau_n\}$ of n independent tasks. Tasks in their task sets are periodic tasks. The task is a 5-tuple $(P_i, C_i, D_i, T_i, E_i)$, in which P_i is the task priority (in this work, P_1 is expressed as the maximum priority), C_i is the worst-case execution time, D_i is the relative task deadline, T_i is the task period, and E_i is the worst-case energy consumption (WCEEC). A periodic task τ_i generates an infinite number of real-time jobs, and each job consumes E_i energy units while executing C_i. The deadline of each period's task is constrained or implicit (i.e.$D_i \leq T_i$). The periodic task set is priority-ordered, the task τ_1 being the task with the highest priority task. In the time interval $[t_1, t_2]$, task consumption is denoted as $E_w(t_1, t_2)$. If the task can be scheduled in the time interval $[t_1, t_2]$, the task energy consumption $E_w(t_1, t_2)$ satisfies the following formula:

$$E_w(t_1, t_2) \leq E_s(t_1) + E_p(t_1, t_2). \quad (1)$$

4. Research Motivation

In this work, we focus on the ED issue of scheduling for EHES. When traditional time-constrained fixed-priority pre-emptive scheduling is directly leveraged by EHES, it may cause the originally feasible task set to become infeasible. We consider that after the system completes a hyperperiod task, the energy level variation has the following two cases:

(i) The replenished energy is lower than the total energy consumed. In this case, every time a hyperperiod passes, the stored energy will decrease until the task sequence is not feasible, causing the system to stop running.

(ii) The replenished energy is greater than or equal to the total energy consumed. In this case, every time a hyperperiod passes, storage energy will increase until reaching the maximum storage value of the storage unit.

However, in some situations, an ED issue may occur. For instance, we assume a task set includes two tasks $\tau_1(P_1 = 1, C_1 = 1, D_1 = 4, T_1 = 4, E_1 = 2)$ and $\tau_2(P_2 = 2, C_2 = 2, D_2 = 8, T_2 = 8, E_2 = 4)$, which are executed in a hyperperiod as shown in Figure 2. The task consumption power is $C_{pi} = E_i/C_i$ (where $C_{p1} = 2/1 = 2$, $C_{p2} = 4/2 = 2$), and the system power consumption is ignored in the idle state,

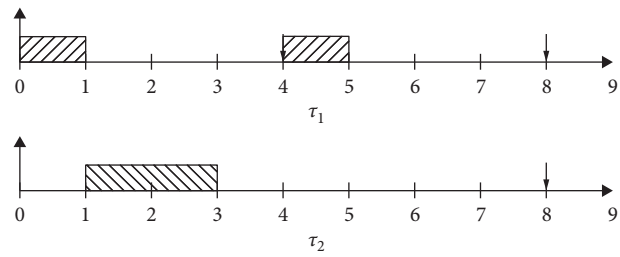

Figure 2: Task τ_1 and task τ_2 executed in a hyperperiod.

such that the energy production power $R = 1$. We can calculate that the total consumed energy (one hyperperiod) $E_t = 2 + 4 + 2 = 8$, and the production energy $E_p(0, 8) = (8 - 0) \times 1 = 8$. We can see that $E_t = E_p$, which appears as if the system will not stop due to insufficient energy. However, as shown in Figure 3, when we set the battery initial value $E_s(0) = 1$, at the time of $t = 2$, the system stops running because energy is exhausted. When we set the battery initial value $E_s(0) = 3$, the system can perform a complete hyperperiod task. Conclusively, although the total energy consumption is equal to the total production energy in some cases, it may still be insufficient energy due to overly frequent task executions in a busy period, or the energy consumption rate of a task is much greater than the energy generation rate. When the above problems occur, the system will stop running.

We found that setting the initial energy of the battery can effectively solve the above problems. Therefore, we propose HEE, WCRT-IELB, and AC-IELB algorithms in Section 6. The HEE algorithm is used to filter task sets that are not feasible under global energy. WCRT-IELB and AC-IELB algorithms are used to calculate the initial energy level of the battery to solve the task scheduling failure caused by the local energy shortage.

5. System Schedulability Analysis

The aim of this section is to characterize the system schedulability. Each task is mapped to a task process when implementing the scheduling. We assumed $\psi(i)$ is a mapping of the task τ_i. This mapping task process is offline and maintains constant during running. A scheduling implementation can be defined as a 2-tuple $I = (\Pi, \Psi)$ for a given system $S = \{\tau_1, \tau_2, \ldots, \tau_n\}$, where

(i) Π: $S \longrightarrow [1, \ldots, n]$ is a priority assignment for the tasks.

(ii) Ψ: $S \longrightarrow [\psi(1), \ldots, \psi(n)]$ is a mapping of tasks into processes.

A scheduling implementation I for a system S is feasible if the WCRT R_i for all tasks under the implementation I is no more than their deadlines D_i. The schedulability is given by equation (2) [28]. Then, a system S is said to be schedulable if there is a feasible implementation for it. In EHES, the WCRT is determined by the worst time requirement and the worst energy requirement together. The calculation is shown below.

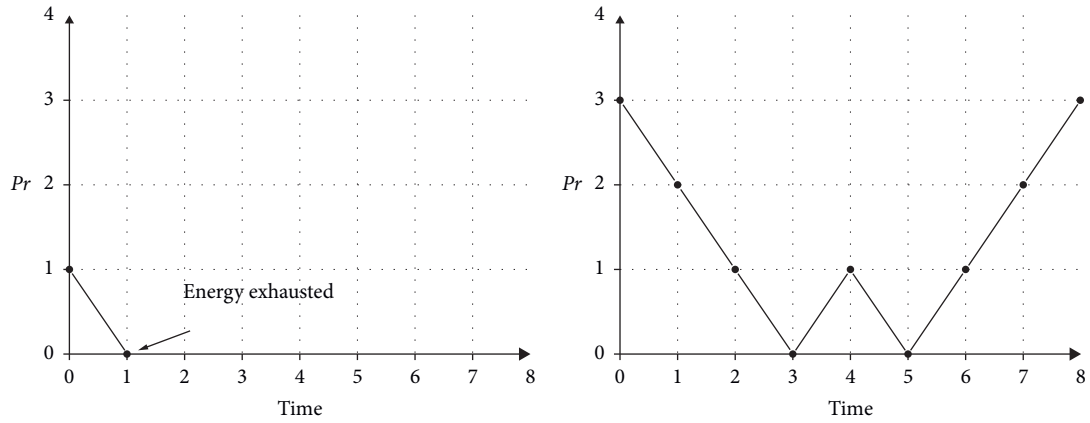

FIGURE 3: Energy consumption of task τ_1 and task τ_2 executed in a hyperperiod.

$$\text{Feasible}(I,P) \stackrel{\text{def}}{=\!=} (\forall_i \in [1,\ldots,n]) R_i(I) \le D_i. \tag{2}$$

Definition 1. The time demand of the task τ_i in the time interval $[0,t]$ is denoted as $wp_i(t)$ in the worst case, which is the execution time of τ_i and the execution time of all tasks whose priority is higher than τ_i. It can be obtained by the following formula [22]:

$$wp_i(t) = \sum_{j \le i} \lceil \frac{t}{T_j} \rceil \times C_j. \tag{3}$$

Definition 2. The energy demand of the task τ_i in the time interval $[0,t]$ is denoted as $we_i(t)$ in the worst case, which is the energy of executing τ_i and the energy of executing all tasks whose priority is higher than τ_i. It can be obtained by the following formula:

$$we_i(t) = \sum_{1 \le j \le i} \lceil \frac{t}{T_j} \rceil \times E_j. \tag{4}$$

Definition 3. The WCRT of the task τ_i in the time interval $[0,t]$ on EHES is determined together by the time demand and energy demand of the task, and the biggest demand between them is the WCRT denoted as $w_i(t)$. It can be obtained by the following formula:

$$w_i(t) = R_i(I) = \max\left(wp_i(t), \lceil \frac{we_i(t)}{R} \rceil \right). \tag{5}$$

Theorem 1. *If after setting the initial battery level and each task of the task set P meets the demand $w_i(t) \le D_i$ in the worst case, the task set is feasible, then the system is schedulable.*

Proof of Theorem 1. We consider the initial battery level $E_s(0) \le C_{\max}$; then, energy demand in the worst case is

$$we_i(t) = \sum_{1 \le j \le i} \lceil \frac{t}{T_j} \rceil \times E_j - E_s(0). \tag{6}$$

According to Definition 3, we know that

$$w_i(t) = \max\left(wp_i(t), \lceil \frac{we_i(t)}{R} \rceil \right), \tag{7}$$

and $\lceil we_i(t)/R \rceil \ge wp_i(t)$ because in our model, $E_s(0) \ge 0, E_i \ge C_i \times R$. This reveals the fact that in our model, we must have replenishment periods that increase task response time (only aim at scheduling algorithms of considering energy, while WCRT of the traditional fixed-priority scheduling algorithm only considers time).

Assume $E_s(0) = 0$. Then,

$$t_s = w_i(t) = \lceil \frac{\sum_{1 \le j \le i} \lceil t/T_j \rceil \times E_j}{R} \rceil. \tag{8}$$

Similarly, assume $E_s(0) > 0$. Then,

$$t_{s1} = w_i(t) = \lceil \frac{\sum_{1 \le j \le i} \lceil t/T_j \rceil \times E_j - E_s(0)}{R} \rceil. \tag{9}$$

We obtain

$$t_s \ge t_{s1}. \tag{10}$$

While $t_s > D_i$ indicating that the task missed the deadline, that is, the system is unschedulable, otherwise, the system is schedulable ($t_s \le D_i$). Therefore, $t_s \ge t_{s1} \ge D_i$ indicates that system is unschedulable while $t_s \ge D_i \ge t_{s1}$ indicates that the system is schedulable. □

Theorem 2. *In the worst case, the energy demand by the EHES is greater than or equal to 0 which is a necessary and insufficient condition for the system to be schedulable. It can be expressed by the following formula :*

$$E(n) = n \times \left(hp \times R - \sum_{1 \le j \le i} \lceil \frac{hp}{T_j} \rceil \times E_j \right) + E(0), \tag{11}$$

where hp denotes the hyperperiod and n denotes the number of the hyperperiod.

Proof of Theorem 2

(1) Necessary condition: according to the above description, if the system is schedulable and all tasks must be completed before the deadline in the worst case, the system will not miss the deadline due to insufficient energy, that is, $E(n) \geq 0$, and the necessity is proved.

(2) Insufficient condition: when the production energy is less than the total consumption energy, it is not difficult to see that $E(n)$ is a monotonically decreasing function and that the battery level reduces as n increases. EHES has sufficient energy in the first few hyperperiods; however, as n increases, more tasks miss the deadline due to insufficient energy. Furthermore, even when $E(n) \geq 0$, the ED issue mentioned in the research motivation will stop the system, and the system is unschedulable. The insufficient condition is proved. □

6. Algorithms

In this section, we propose three algorithms to address the ED issue and improve the success rate of the task sets scheduled. The HEE algorithm aims at filtering the first case mentioned in the research motivation, while the WCRT-IELB and AC-IELB algorithms aim at adopting the initial energy value to eliminate the second case mentioned in the research motivation, namely, the ED issue.

6.1. HEE Algorithm. HEE, which is shown in Algorithm 1, is a general judgment that can make a preliminary judgment on the task set, which the main relies on equation 11 to calculate. Lines 4–9 show the total energy consumption accumulated over a hyperperiod. Line 10 compares the total energy consumped and the total energy producted; if it returns true, then it indicates the total energy producted is equal or lower than the total energy consumed. This case does not completely guarantee that the task set has enough energy. However, if it returns false, it indicates that the set of tasks is infeasible.

Because HEE is a preliminary judgment, we require WCRT-IELB to judge further. Although HEE cannot accurately determine whether task sets can be scheduled, it can exclude the majority of cases that cannot be scheduled and improve the operation efficiency of the following two algorithms.

6.2. WCRT-IELB Algorithm. WCRT-IELB, which is shown in Algorithm 2, first calculates the WCRT of the set task by formula (5) [22] online 3 and then calculates the execution times of each task during the WCRT and the total energy consumption of each task (lines 5–10). Finally, the total energy consumption of each task is accumulated. The total energy consumption and production energy are compared in line 11; if the production energy is lower than the total consumption energy, the absolute value of the difference between production and consumption is returned. This absolute value is the initial value required by the battery, and this value is not more than the battery maximum capacity.

However, this value is still not the most appropriate in some extreme cases. We assume that a task set includes two tasks $\tau_1(P_1 = 1, C_1 = 1, D_1 = 4, T_1 = 4, E_1 = 3)$ and $\tau_2(P_2 = 2, C_2 = 2, D_2 = 8, T_2 = 8, E_2 = 4)$ and energy production power $P_r = 1$. We can calculate that the WCRT is 3, where τ_1 and τ_2 are executed once, and during this interval of time, the total energy consumption is $3 + 4 = 7$ and the energy production is $1 \times 3 = 3$. Therefore, the initial value calculated by WCRT-IELB is $|3 - 7| = 4$. However, when this initial value is set, τ_1 will still stop running due to insufficient energy when it is executed in the second period (available energy $E_a = 4 + 5 = 9$ is less than consumption energy $E_c = 3 + 4 + 3 = 10$). Therefore, we propose a more accurate AC-IELB algorithm.

6.3. AC-IELB Algorithm. AC-IELB, which is shown in Algorithm 3, determines how to accurately calculate the initial value of the battery when a set of tasks is ready to run. AC-IELB first chooses the highest priority task and then calculates the task energy consumption at the time unit (E_i/C_i) and compares it with the current energy level $(E(t))$ plus production energy (R) at the time unit.

The AC-IELB can be divided into three cases. In the first case (lines 8–10), the available energy of the system is greater than energy consumption, and the system can perform tasks. In the second case (lines 11–14), the available energy is lower than the energy consumption, and the system does not have enough energy to perform tasks. AC-IELB will calculate the difference between the available energy and energy consumption and accumulate this difference to the initial battery level value, and AC-IELB will reset the initial battery level value and run again. In the third case (lines 15–18), the energy consumption at the time unit is equal to the energy production rate, and the current energy level is 0. At the next moment, we cannot guarantee that the system first consumes energy, produces energy, or both; therefore, we accumulate an additional single unit of energy to ensure the normal operation of the system. Then, the first task in the task set is deleted and the execution is repeated until all tasks in the task set have been executed. AC-IELB can address the cases where the initial value calculated by WCRT-IELB is not appropriate.

We consider a task set as shown in Table 1. In time intervals $[0, 55]$, the scheduling result is shown in Figure 4. In this example, we set the energy production power $R = 15$, the battery initial value $E(0) = 20$, and the battery maximum capacity $C_M = 300$. Using the ALAP scheduling algorithm, the scheduling result is shown in Figure 4(a). At time $t = 9$, there is a shortage of available energy; therefore, the task τ_4 stops executing. The initial value of the battery $E(0) = 12$ (the WCRT is 20) is calculated by WCRT-IELB, and the initial value of the battery is reset to run again. The

```
Input: A ⟵ set of n active tasks at time t
Output: true or false
(1)  function GLOBAL CALCULATION (A)
(2)    hp ⟵ calculating hyperperiod of the task set A
(3)    sum ⟵ 0
(4)    for i = 1; i ≤ n; i++do
(5)      take the i^th task of A as τ_i
(6)      E_i ⟵ energy cost of τ_i
(7)      T_i ⟵ period of τ_i
(8)      sum ⟵ E_i×⌈hp/T_i⌉ + sum
(9)    end for
(10)   if (sum−hp×R) < 0 then
(11)     return false
(12)   else
(13)     return true
(14)   end if
(15) end function
```

Algorithm 1: Holistic energy evaluation.

```
Input: A ⟵ set of n active tasks at time t
Output: true or false
(1)  function WORSTCASECALCUATION (A)
(2)    τ_l ⟵ the lowest priority task of A
(3)    wt ⟵ WorstCaseResponseTime (τ_l)
(4)    sum ⟵ 0
(5)    for i = 1; i < n; i++ do
(6)      task the i^th task of A as τ_i
(7)      Ei ⟵ energy cost of τ_i
(8)      T_i ⟵ period of τ_i
(9)      sum ⟵ E_i×⌈wt/T_i⌉ + sum
(10)   end for
(11)   if (wt×R−sum) < 0 then
(12)     if |wt × R − sum| ≥ C_max then
(13)       return C_max
(14)     else
(15)       return |wt × R − sum|
(16)     end if
(17)   end if
(18) end function
(19) function SCHEDULABILITYJUDGMENT
(20)   iv_i ⟵ WorstCaseCalcuation (A)
(21)   set iv_i and task set A and execute schedule algorithm
(22)   if Scheduling algorithm is schedulable then
(23)     return true
(24)   else
(25)     return false
(26)   end if
(27) end function
```

Algorithm 2: WCRT-based initial energy level of battery.

scheduling result is shown in Figure 4(b). When the system runs on time $t = 8$, the task τ_1 stops executing due to a shortage of available energy. Until the most accurate AC-IELB is used to calculate the battery initial value $E(0) = 158$, the task set can be scheduled in the time interval $[0, 55]$. The scheduling result is shown in Figure 4(c).

```
Input: A ⟵ set of active tasks at time t, sum ⟵ 0
Output: ture of false
(1)  function CalculateInitialValue(A; sum)
(2)    t ⟵ 0
(3)    loop
(4)      τ_i ⟵ the first task of A
(5)      E_i ⟵ remaining energy cost of the τ_i at time t
(6)      C_i ⟵ remaining execution time of the τ_i at time t
(7)      if A ≠ ϕ then
(8)        if E_i/C_i < E(t) + R then
(9)          t ⟵ t + 1
(10)       end if
(11)       if E_i/C_i > E(t) + R then
(12)         sum ⟵ |(E_i/C_i) − E(t) − R|+sum
(13)         CalculateInitialValue(A, sum)
(14)       end if
(15)     if E_i/C_i = R & E(t) = 0 then
(16)         sum ⟵ sum + 1
(17)         CalculateInitialValue(A, sum)
(18)       end if
(19)     end if
(20)     A ⟵ remove the first task of A
(21)   end loop
(22)   if result ≥ C_max then
(23)     return C_max
(24)   else
(25)     return result
(26)   end if
(27) end function
(28) function SCHEDULABILITYJUDGMENT
(29)   iv_i ⟵ CalculateInitialValue(A, sum)
(30)   set the iv_i and task set A, and execute schedule algorithm
(31)   if Scheduling algorithm is schedulable then
(32)     return true
(33)   else
(34)     return false
(35)   end if
(36) end function
```

Algorithm 3: Accurate cycle-initial energy level of battery.

Table 1: Task set τ_t.

-	C_i	E_i	T_i	D_i	P_i
τ_1	3	150	36	36	1
τ_2	1	100	10	10	2
τ_3	2	20	24	24	3
τ_4	1	18	30	30	4

7. Simulation and Evaluation

In this section, we describe the design and implementation of the experiment from the simulation tool, input data, simulation duration, evaluation metrics, and result analysis.

7.1. Simulation Tool. In this work, to evaluate the effectiveness of the battery initial value for scheduling algorithms, we randomly generated a large number of periodic task sets and verified them with the ALAP and ASAP. We used YARTISS [29, 30] as the simulation experiment environment and conducted secondary development on it. It provides a simulation framework and implements many scheduling algorithms that can simulate different task sets under different energy parameters for EHES.

7.2. Input Data. YARTISS uses an adapted version of the UUniFast-Discard algorithm [31] coupled with a limitation of the hyperperiod technique to generate task sets. We use this function to generate 250 task sets. Each task set contains 4 tasks, and the range of [0, 200] is selected for each attribute of the task. Task sets are time feasible.

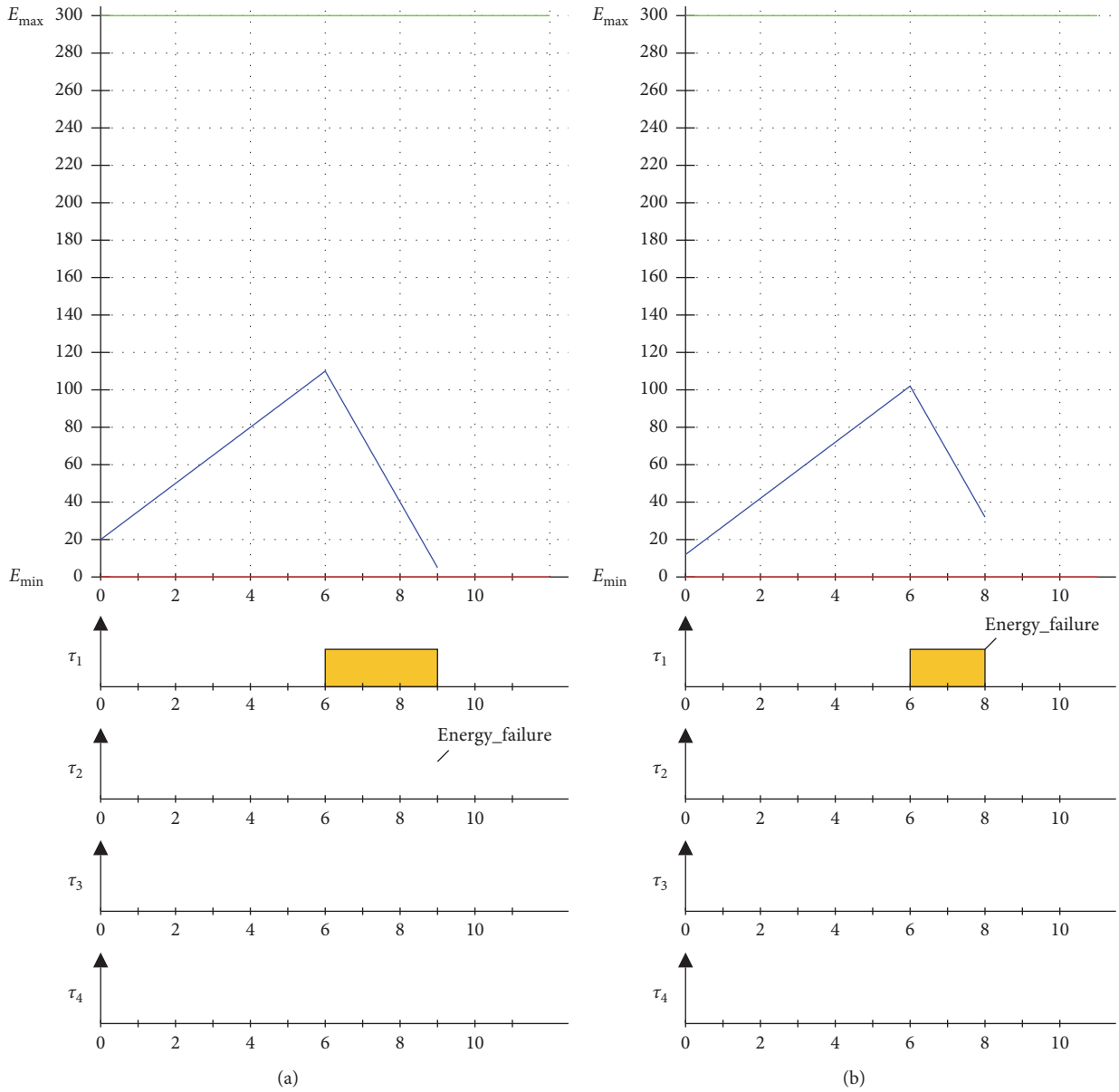

Figure 4: Continued.

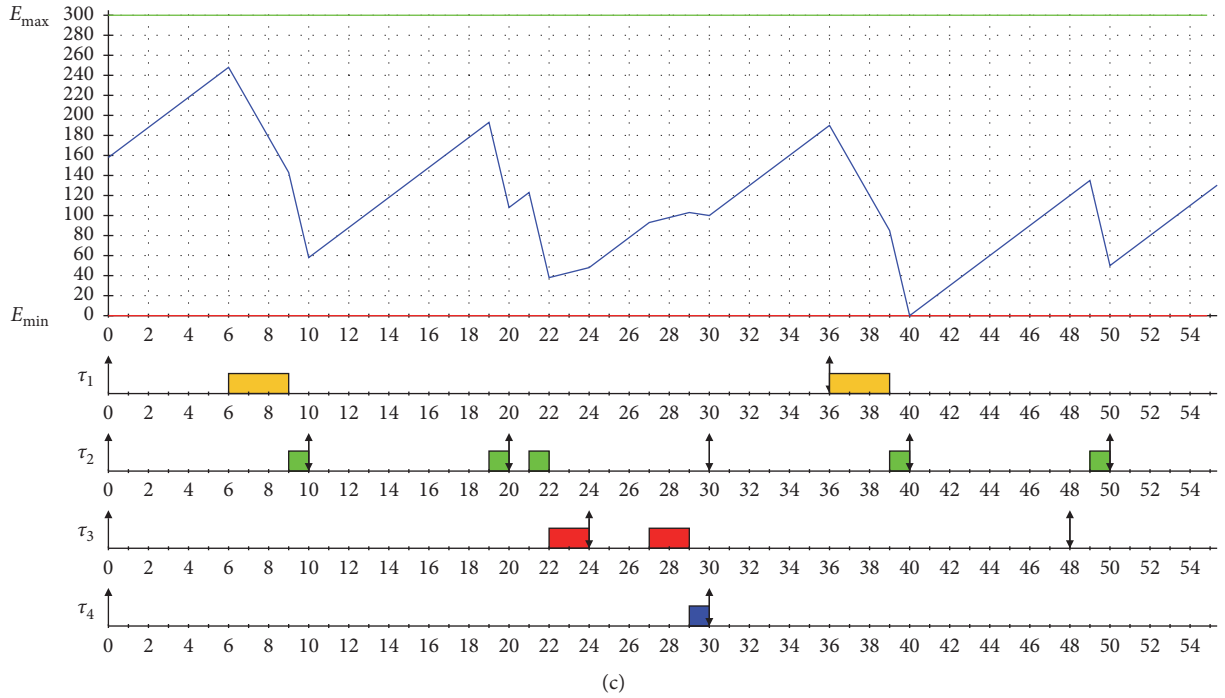

FIGURE 4: Running status of task set τ_T with different initial energies. (a) Fixed initial energy (20). (b) WCRT-IELB algorithm calculating initial energy. (c) AC-IELB algorithm calculating initial energy.

7.3. Parameter Setting. In ALAP and ASAP, the influence of the battery initial value setting is compared. First, we vary the battery capacity to analyse the impact on the success rate. Second, we conducted 250 groups of experiments to observe the success rate of each algorithm under different battery capacities.

We set the same common parameters to ensure the correctness of the simulations. These parameters are set as follows: energy production power $R = 15$, the battery storage minimum energy $E_{\min} = 0$, the battery maximum capacity $C_M = \{50, 100, 200, 300, 400\}$, and the simulation execution time Duration = 2560. We perform three types of simulations with different initial energy levels on ALAP and ASAP.

(1) Sitting a fixed initial energy level of the battery ($E(0) = 20$).

(2) Setting the battery's initial energy level based on WCRT-IELB.

(3) Setting the battery's initial energy level based on AC-IELB.

7.4. Evaluation Metrics

7.4.1. Average Success Rate. We define the average success rate SR_a shown in equation (12) to evaluate the three algorithms, where T_f denotes the number of feasible task sets and T_a denotes the number of all task sets. We conducted 250 groups of experiments and divided them into 5 parts on average and calculated the success rate of each group (SRG_a), evaluating the average by formula (13), where T_{f_i} denotes the $i-th$ group of the number of the feasible task sets and T_{a_i} denotes the $i-th$ group of the number of all task sets.

$$SR_a = \frac{T_f}{T_a}, \tag{12}$$

$$SRG_a = \frac{1}{n}\sum_{i=0}^{n}\frac{T_{f\_i}}{T_{a\_i}}. \tag{13}$$

7.4.2. Average Energy Level. The average energy level is the average energy percentage of the battery or capacitor during the simulation. The higher the average energy level, the lower the energy limit of the system.

7.4.3. Average Overhead. It is the average time taken to execute a scheduled event during the simulation. The greater the average overhead is, the more likely the task will miss the task deadline.

7.5. Result Analysis. We use the WCRT-IELB and AC-IELB algorithms to calculate the initial battery capacity of 250 task sets under ALAP and ASAP. Take ALAP as an example here, as shown in Figure 5. The maximum battery capacity is 50, 100, 200, 300, and 400. As depicted, since the calculation method of WCRT-IELB determines the initial battery level based on the size of the busy period, most of the initial battery levels have reached the maximum battery capacity; although the initial battery level calculated by the AC-IELB algorithm also accounts for a large part of the maximum

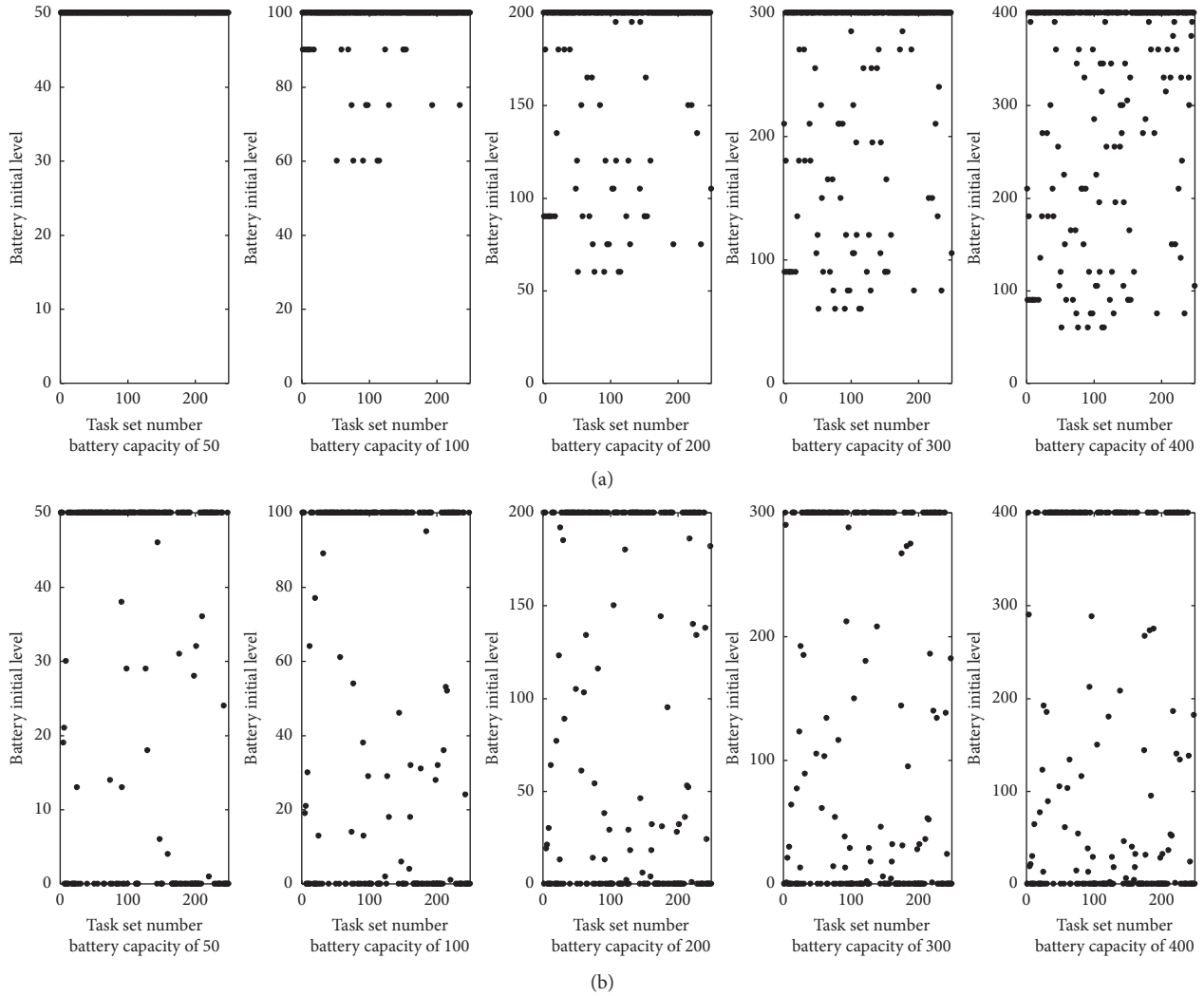

Figure 5: The ALAP scheduling algorithm calculating the initial battery level based on the WCRT-IELB and AC-IELB algorithms. (a) WCRT-IELB algorithm. (b) AC-IELB algorithm.

battery capacity, with the increase of the maximum battery capacity, this situation has eased. And the situation where the initial battery level is 0 is gradually increasing. This is because the ALAP scheduling algorithm is less affected by the initial battery level and is more affected by the maximum battery capacity. When the maximum capacity of the battery increases, the schedulability of the ALAP scheduling algorithm is gradually reduced by the initial level of the battery. Moreover, we found that when the battery capacity is 50, 100, and 200, the battery initial level calculated by the WCRT-IELB and AC-IELB algorithms has reached the maximum battery capacity in most cases, until the battery capacity is increased to 300 and 400. This situation began to ease. This is because according to the task set, it is calculated that the actual required battery initial level is greater than the maximum battery capacity. We have verified this in subsequent experiments. When the maximum battery capacity is 50, 100, and 200, the success rate of the task set is very low. It was not until the maximum battery capacity was increased to 300 and 400 that the success rate increased significantly. Most task sets did not achieve the proper initial battery level.

7.5.1. Average Success Rate. The 250 task sets were tested with the ALAP and ASAP in the following two scenarios.

Scenario 1: make 250 task sets run as a group with battery capacities of 50, 100, 200, 300, and 400.

Figure 6 shows the success rate of three different battery initial level-setting methods (fixed, WCRT-IELB algorithm calculation, and AC-IELB algorithm calculation) for ALAP and ASAP. The black line represents the scheduling algorithm that is set to run at a fixed initial battery level of 20, the red line represents the scheduling algorithm that uses WCRT-IELB to set the initial battery level, and the blue line represents the scheduling algorithm that uses AC-IELB to set the initial battery level.

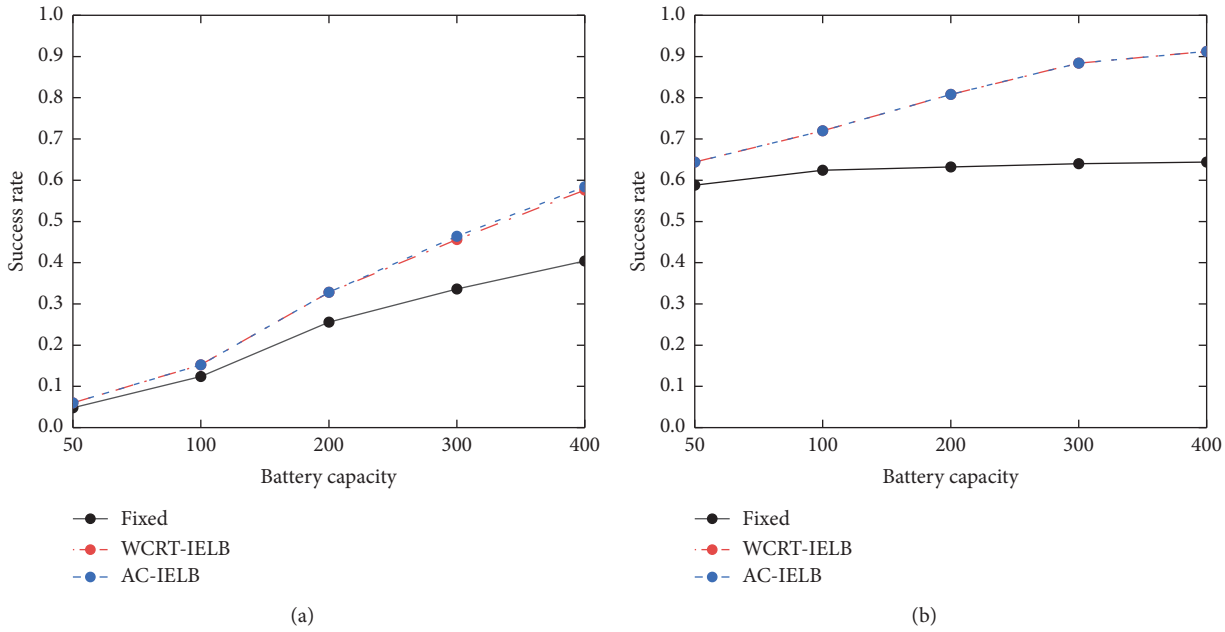

Figure 6: The success rate of 250 experiments. (a) ALAP scheduling algorithm. (b) ASAP scheduling algorithm.

As depicted, we propose AC-IELB and WCRT-IELB algorithm to applicate the ALAP and ASAP that performance is better than schedulling algorithms based on fixed model settings. This is expected; as Section 4 describes, the scheduling algorithms on their busy period have the most energy consumption; however, using this method to calculate the initial battery level in some very extreme cases is not precise, in which case we adopt the AC-IELB for every task to calculate.

On the other hand, the ALAP scheduling algorithm is less affected by the initial battery level and is more affected by the maximum battery capacity, and with the increase of maximum battery capacity, the success rate increases. It has a big rise tendency from a battery capacity of 100 to a battery capacity of 400. Compared with the ALAP scheduling algorithm based on fixed model settings, the success rate of the WCRT-IELB algorithm under battery capacity of 400 increased by 17.2%, while the AC-IELB algorithm increased by 18%. For the ASLP scheduling algorithm, the scheduling algorithm based on fixed model settings remain stable success rate form battery capacity of 50 to 400, suffering rarely fluence from battery capacity, ALAP scheduling algorithm based on AC-IELB and WCRT-IELB algorithm under form battery capacity of 50 to 200, its success rate has a big rise. In addition, by a battery capacity of 300 and a battery capacity of 400, its success rate is basically the same. Compared with the ASAP scheduling algorithm based on fixed model settings, the success rate of the AC-IELB and WCRT-IELB algorithms under battery capacity of 400 increased by 26.8%.

Scenario 2: divide 250 task sets into five groups and run with battery capacities of 50, 100, 200, 300, and 400.

Figure 7 compares the success rates of ALAP and ASAP using three different methods (fixed, WCRT-IELB algorithm calculation, and AC-IELB algorithm calculation) to obtain the initial battery capacity under five different maximum battery capacities. The black line represents the scheduling algorithm running at a fixed initial battery level of 20, the red line represents the scheduling algorithm that adopts the WCRT-IELB algorithm to set the initial battery level, and the blue line represents the scheduling algorithm that adopts the AC-IELB algorithm to set the initial battery level.

To begin with, for the ALAP scheduling algorithm, as depicted, the maximum capacity of the battery has a significant impact on the ALAP scheduling algorithm (the success rate of the ALAP scheduling algorithm with initial battery level increases as the maximum battery capacity increases). When the maximum battery capacity is 50, the success rate is low. Setting the battery's initial level has little significance. This is because the ALAP scheduling algorithm is limited by the battery capacity. Through calculation, most of the initial battery levels that we obtain are more than 50. As the maximum battery capacity increases, the success rate gradually increases, and the effect of using the WCRT-IELB and AC-IELB algorithms to set the initial level continues to improve. Overall, the increase in the number of tasks has little effect on the success rate of the ALAP scheduling algorithm, fluctuating between 5% and 18%.

On the other hand, for the ASAP scheduling algorithm, considering the ASAP algorithm with the fixed initial level, the maximum battery capacity has little effect on it. This is due to the unique scheduling strategy of the ASAP algorithm, which causes the battery energy level to remain relatively low. When the battery capacity is 50 or 100, the success rate of the ASAP scheduling algorithm changes in the same way. When the battery capacity is 200, 300, and 400, compared with the ASAP scheduling algorithm based

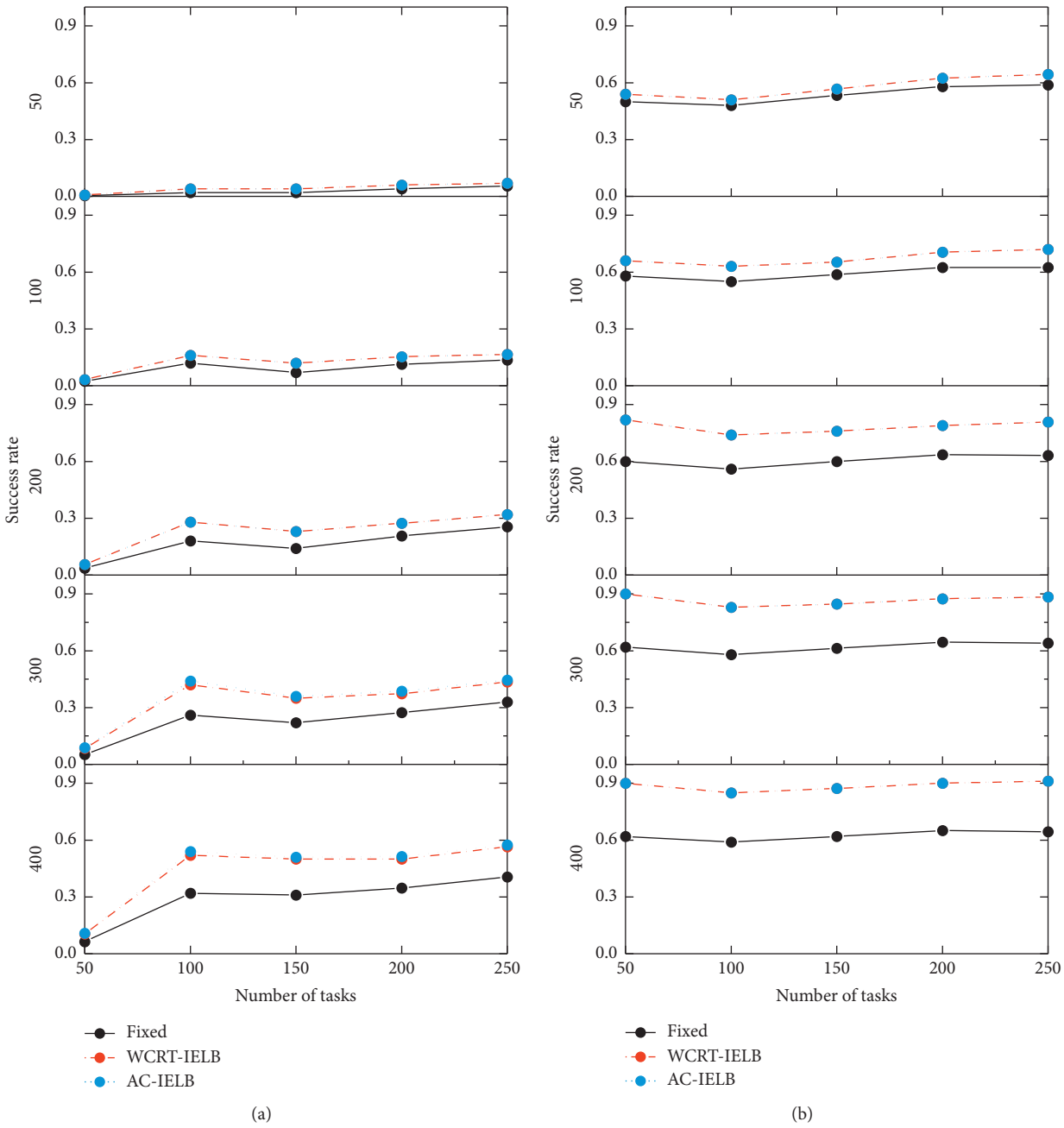

Figure 7: Success rate with a battery capacity of 50, 100, 200, 300, and 400. (a) ALAP scheduling algorithm. (b) ASAP scheduling algorithm.

on fixed model settings, the success rate of the ASAP scheduling algorithm based on the WCRT-IELB and AC-IELB algorithms to calculate the initial level is greatly improved, the success rate increases with the increase in number of task sets, and the task changes are relatively stable. Through experiments, we found that the overall energy level during the operation of the ASAP scheduling algorithm is low. If the busy period consumes large energy, there is not enough energy to run the task before the deadline, which requires a relatively large initial battery level. Therefore, when the battery capacity is high, the performance of the ASAP scheduling algorithm based on WCRT-IELB and AC-IELB is better than the ASAP scheduling algorithm with a fixed initial value.

7.5.2. Average Energy Level. As shown in Figure 8, we observe that the average energy level with adopting the WCRT-IELB and AC-IELB algorithm is higher than with fixed method settings on the ALAP (Figure 8(a)) and ASAP (Figure 8(b)), the primary reason is that we setting an initial battery level. Moreover, the increase of average energy level with the battery capacity increase. The average energy level of adopting AC-IELB algorithm is lower than

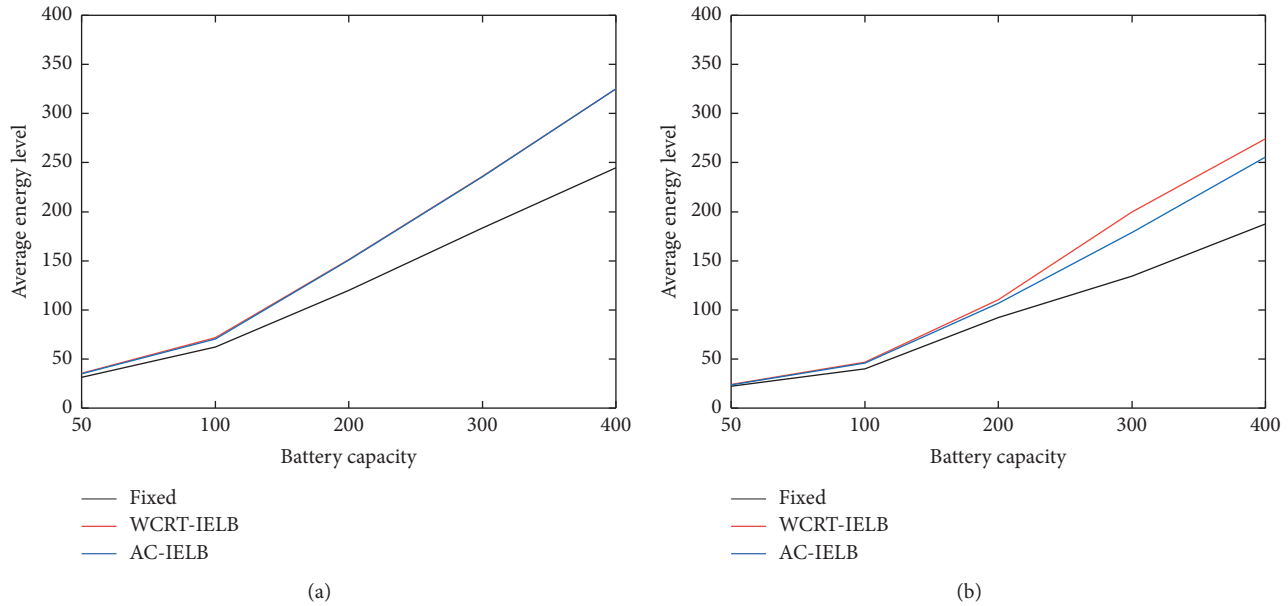

Figure 8: Average energy level.

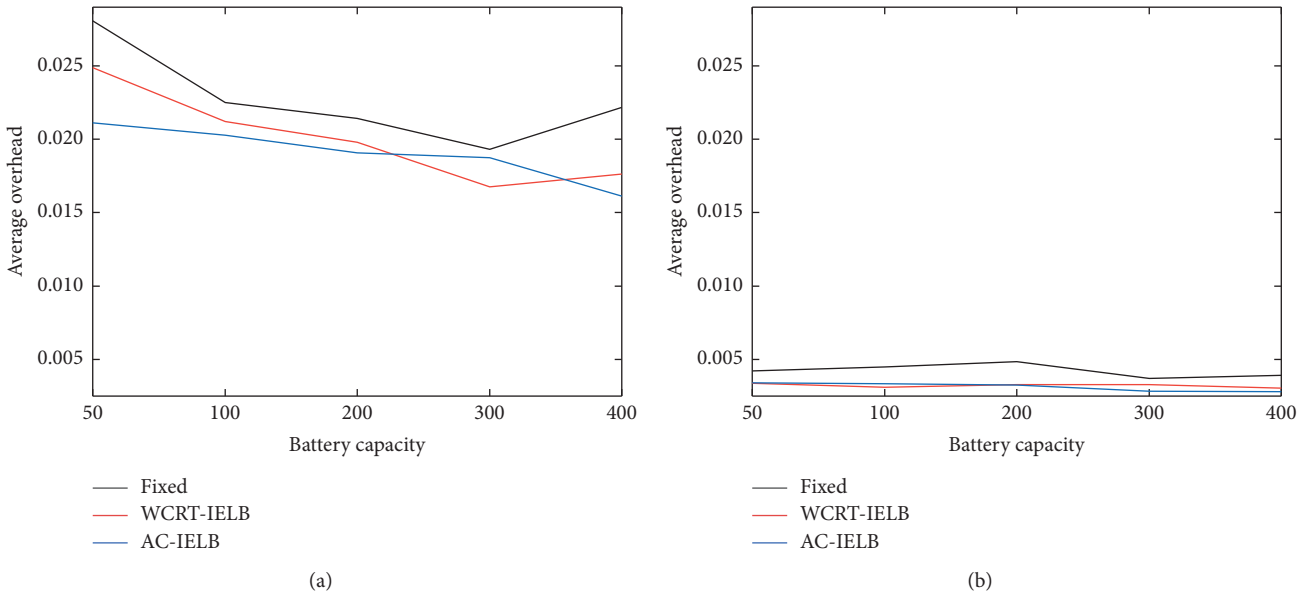

Figure 9: Average overhead.

adopting WCRT-IELB algorithm on the ASAP scheduling algorithm. Since compared with WCRT-IELB algorithm, AC-IELB algorithm calculate the initial battery level is more precise.

7.5.3. Average Overhead. As shown in Figure 9, we observe that the average overhead of adopting WCRT-IELB and AC-IELB algorithms is lower than that use initial battery level on ALAP (Figure (9a)) and ASAP (Figure 9b)). Also, these two different methods have the same tendency, which reduces with the increase of battery capacity.

8. Conclusions and Future Works

In this work, we proposed a filter algorithm named HEE that aimed to remove the infeasible task set of EHES, and we proposed two algorithms named WCRT-IELB and AC-IELB that aimed to improve the success rate of the scheduling algorithm to use the battery initial level to solve the ED problem. From the experiment, we can see that the best performance of the three scheduling algorithms without a proper battery initial level is achieved by employing the ASAP scheduling algorithm, which has a success rate of 60% at the maximum battery capacity, while the success rate

reached 97.2% after introducing the WCRT-IELB and AC-IELB algorithms. We ascribe this case to two problems: ED and the limitation of the maximum battery capacity. As a result, we found that the proper battery initial level and maximum battery capacity could improve the success rate of scheduling algorithms of EHES; however, the effect of this improvement depends on the strategy of scheduling algorithms.

In future work, a valuable endeavour is to calculate a suitable maximum capacity of the battery by implementing the algorithm and combining it with the battery's initial energy level to further improve the success of the scheduling algorithm. We will also try to build a real-world platform to collect real data and try to test the practicality of our proposed algorithms. Furthermore, the EHES computing unit based on multitask scheduling discussed in this work is a discrete computer system; thence, we will consider to research and discuss algorithms of this work on the continuous system.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Jiayuan Wei's contribution in the experiment is equivalent to Xingyu Miao.

Acknowledgments

This study was supported in part by the Young Scholar in Western China of Chinese Academy of Sciences under grant no. XAB2018AW12, in part by the Ningxia Key Research and Development Projects under grant no. 2018BEB04020, and in part by the National Natural Science Foundation of China under grant no. 61862049.

References

[1] R. Norouzi, A. Kosari, and M. H. Sabour, "Real time estimation of impaired aircraft flight envelope using feedforward neural networks," *Aerospace Science and Technology*, vol. 90, pp. 434–451, 2019.

[2] S. Kato, S. Tokunaga, Y. Maruyama et al., "Autoware on board: enabling autonomous vehicles with embedded systems," in *Proceedings of the 2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*, pp. 287–296, IEEE, Porto, Portugal, April 2018.

[3] A. Sharma and P. Sharma, *Energy Harvesting Technology for IoT Edge Applications, in Smart Manufacturing-When Artificial Intelligence Meets the Internet of Things*, IntechOpen, London, UK, 2021.

[4] C. Psomas and I. Krikidis, "Wireless powered mobile edge computing: offloading or local computation?" *IEEE Communications Letters*, vol. 24, no. 11, pp. 2642–2646, 2020.

[5] P. Cong, J. Zhou, L. Li, K. Cao, T. Wei, and K. Li, "A survey of hierarchical energy optimization for mobile edge computing," *ACM Computing Surveys*, vol. 53, no. 2, pp. 1–44, 2020.

[6] Y. Hao, J. Cao, Q. Wang, and J. Du, "Energy-aware scheduling in edge computing with a clustering method," *Future Generation Computer Systems*, vol. 117, pp. 259–272.

[7] M. Malewski, D. M. Cowell, and S. Freear, "Review of battery powered embedded systems design for mission-critical low-power applications," *International Journal of Electronics*, vol. 105, pp. 893–909, 2018.

[8] S. Tzilis, P. Trancoso, and I. Sourdis, "Energy-Efficient runtime management of heterogeneous multicores using online projection," *ACM Transactions on Architecture and Code Optimization*, vol. 15, no. 4, pp. 1–26, 2019.

[9] M. Chetto and H. E. Ghor, "Scheduling and power management in energy harvesting computing systems with real-time constraints," *Journal of Systems Architecture*, vol. 98, pp. 243–248, 2019.

[10] S.-H. Lim, S. W. Lee, M. Sohn, and B.-H. Lee, "Queueing analysis of dynamic power management schemes for mobile devices," *IEEE Access*, vol. 8, pp. 97632–97642, 2020.

[11] N. Chawla, A. Singh, H. Kumar, M. Kar, and S. Mukhopadhyay, "Securing IoT devices using dynamic power management: machine learning approach," *IEEE Internet of Things Journal*, p. 1, 2020.

[12] D. Hosahalli and K. G. Srinivas, "Enhanced reinforcement learning assisted dynamic power management model for internet-of-things centric wireless sensor network," *IET Communications*, vol. 14, no. 12, pp. 3748–3760, 2020.

[13] G. L. Stavrinides and H. D. Karatza, "An energy-efficient, QoS-aware and cost-effective scheduling approach for real-time workflow applications in cloud computing systems utilizing DVFS and approximate computations," *Future Generation Computer Systems*, vol. 96, pp. 216–226, 2019.

[14] A. Toor, S. u. Islam, N. Sohail et al., "Energy and performance aware fog computing: a case of DVFS and green renewable energy," *Future Generation Computer Systems*, vol. 101, pp. 1112–1121, 2019.

[15] C. Zhuo, S. Luo, H. Gan, J. Hu, and Z. Shi, "Noise-Aware DVFS for efficient transitions on battery-powered IoT devices," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 7, pp. 1498–1510, 2020.

[16] D. Balsamo, B. J. Fletcher, A. S. Weddell, G. Karatziolas, B. M. Al-Hashimi, and G. V. Merrett, "Momentum: power-neutral performance scaling with intrinsic MPPT for energy harvesting computing systems," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 17, 2019.

[17] M. Hasanloo and M. Kargahi, "Harvesting-aware charge management in embedded systems equipped with a hybrid electrical energy storage," *Computers & Electrical Engineering*, vol. 69, pp. 98–114, 2018.

[18] J. Kwak, K. Lee, T. Kim, J. Lee, and I. Shin, "Battery aging deceleration for power-consuming real-time systems," in *Proceedings of the 2019 IEEE Real-Time Systems Symposium (RTSS)*, pp. 353–365, IEEE, Houston, TX, USA, May 2019.

[19] A. Allavena and D. Mossé, "Scheduling of frame-based embedded systems with rechargeable batteries, in workshop on power management for real-time and embedded systems," (in Conjunction with RTAS 2001) http://igm.univ-mlv.fr/masson/pdfANDps/allavena_mosse_01.pdf, 2001.

[20] C. Moser, D. Brunelli, L. Thiele, and L. Benini, "Lazy scheduling for energy harvesting sensor nodes," in *Proceedings of the IFIP Working Conference on Distributed and*

Parallel Embedded Systems, pp. 125–134, Springer, Milano, Italy, September 2006.

[21] R. Jayaseelan, T. Mitra, and X. Li, "Estimating the worst-case energy consumption of embedded software," in *Proceedings of the 12th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'06)*, pp. 81–90, IEEE, San Jose, CA, USA, April 2006.

[22] Y. Abdeddaïm, Y. Chandarli, and D. Masson, "The optimality of PFPasap algorithm for fixed-priority energy-harvesting real-time systems," in *Proceedings of the 2013 25th Euromicro Conference on Real-Time Systems*, pp. 47–56, IEEE, Los Alamitos, CA, USA, July 2013.

[23] Y. Chandarli, Y. Abdeddaïm, and D. Masson, "The fixed priority scheduling problem for energy harvesting real-time systems," in *Proceedings of the 2012 IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, pp. 415–418, IEEE, Seoul, South Korea, August 2012.

[24] Y. Abdeddaïm, Y. Chandarli, and D. Masson, "Toward an optimal fixed-priority algorithm for energy-harvesting real-time systems," in *Proceedings of the RTAS 2013 WiP*, pp. 45–48, Montreal, QC, Canada, January 2013.

[25] T. Kim, "Application-driven low-power techniques using dynamic voltage scaling," in *Proceedings of the 12th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA'06)*, pp. 199–206, IEEE, Piscataway, NJ, USA, August 2006.

[26] Y. Abdeddaïm, Y. Chandarli, R. I. Davis, and D. Masson, "Schedulability analysis for fixed priority real-time systems with energy-harvesting," in *Proceedings of the 22nd International Conference on Real-Time Networks and Systems*, pp. 311–320, Versaille, France, October 2014.

[27] E. Ghadaksaz and S. Safari, "Storage capacity for EDF-ASAP algorithm in energy-harvesting systems with periodic implicit deadline hard real-time tasks," *Journal of Systems Architecture*, vol. 89, pp. 10–17, 2018.

[28] Y. Ge, Y. Dong, and H. Zhao, "Energy-efficient task scheduling and task energy consumption analysis for real-time embedded systems," in *Proceedings of the 2014 Theoretical Aspects of Software Engineering Conference*, pp. 135–138, IEEE, Changsha, China, September 2014.

[29] Y. Chandarli, M. Qamhieh, F. Fauberteau, and D. Masson, "Yartiss: a generic, modular and energy-aware scheduling simulator for real-time multiprocessor systems," 2014, https://hal.archives-ouvertes.fr/hal-01076022/file/journal.pdf.

[30] Y. Chandarli, F. Fauberteau, D. Masson, S. Midonnet, and M. Qamhieh, "Yartiss: a tool to visualize, test, compare and evaluate real-time scheduling algorithms," 2012, https://hal-upec-upem.archives-ouvertes.fr/hal-00691985/document.

[31] P. Emberson, R. Stafford, and R. I. Davis, "Techniques for the synthesis of multiprocessor tasksets," in *Proceedings 1st International Workshop on Analysis Tools and Methodologies for Embedded and Real-Time Systems (WATERS 2010)*, pp. 6–11, Brussels, Belgium, June 2010.