Resilient Scheduling of Energy-Variable Weakly-Hard Real-Time Systems

Mahmoud Shirazi
School of Computer Science, Institute for Research in Fundamental Sciences
P.O. Box 19395-5746
Tehran, Iran
m.shirazi@ipm.ir

Mehdi Kargahi
School of Electrical and Computer Engineering, University of Tehran
P.O. Box 14399-57131
Tehran, Iran
kargahi@ut.ac.ir

Lothar Thiele
Computer Engineering and Networks Laboratory, ETHZ
Gloriastrasse 35, 8092 Zurich
Zurich, Switzerland
thiele@ethz.ch

ABSTRACT

Cyber-physical systems and the like have accelerated the growth of demands for long-life energy-limited devices, encouraging the major trend of using energy harvesting from renewable sources like solar and wind. The intermittency of these energies enforces such energy-variable systems to possibly anticipate the changes, and efficiently adapt in face of either predicted or non-predicated changes, i.e. they must be energy-resilient. In this paper, we target weakly-hard real-time energy-variable systems with multiple performance levels. We formalize the concept of energy-resilience in such systems, define and prove some properties of the resilient systems, and give a scheduler with the aim of maximizing the system resilience through adaptive control of the system performance with respect to the energy changes. Also, we propose an online schedulability test that considers the initial available energy in the storage unit and the (m, k)-firm constraints. Furthermore, we give an energy-resilient scheduling algorithm which employs model predictive control. The simulation results show that our proposed method performs very good in approximating the optimal solution.

KEYWORDS

Energy-aware scheduling, Energy harvesting, Resilience, Weakly-hard real-time systems

ACM Reference format:
Mahmoud Shirazi, Mehdi Kargahi, and Lothar Thiele. 2017. Resilient Scheduling of Energy-Variable Weakly-Hard Real-Time Systems. In *Proceedings of International Conference on Real-Time Networks and Systems, Grenoble, France, October 2017 (RTNS 2017),* 10 pages.
DOI:

1 INTRODUCTION

The need to green energies, infrastructure to use traditional energy sources, and the need for device portability are of the reasons to emphasize using renewable sources like solar, wind, vibration, thermal, etc. Energy harvesting is the process of capturing energy from external sources, converting it into electrical energy, and using it immediately or conserving it into storage units for future use [1]. However, in contrast to traditional energy sources which flow when a knob is turned, renewable energy sources like solar and wind produce intermittently as the sun shines or the wind blows; this is a major challenge of using these sources [2].

The intermittency of energy harvesting from renewable energy sources enforces the energy-variable systems to possibly anticipate the changes, and adapt in face of either predicted or non-predicated changes. In fact, such systems must be energy-resilient, i.e. they are to persist performability when facing changes in the harvested energy.

In this paper, we target weakly-hard real-time (WHRT) energy-variable systems with multiple performance levels for individual tasks as well as the system. The rate of harvesting energy is subject to change at intervals with specific granularity, which can be predicted with some level of accuracy. In this paper, we assume that there is a lower bound on the predicted charging rates. The aim is to have an energy-resilient system, namely a system which maximizes the system performance, guarantees the system survivability, and tries to minimize the time taken to recover from the possibly degraded performance.

To the best of our knowledge, we are unaware of any previous study on energy-resilient scheduling of weakly-hard real-time systems. There are some related works in the area of real-time scheduling, including transient fault tolerance [3–6], overload management [7–10] and mixed-criticality systems [11–13]. Also, there exist some works on energy-efficient scheduling [14–16], energy-aware scheduling [17–20], and resilient scheduling [21]. In [21], the authors propose a hard real-time scheduling method for multiprocessor systems which is resilient to core failure via job redundancy. None of these works concentrate on the resilience of real-time systems with the aim of simultaneously minimizing the maximum performance deviation and the automatic recovery time to the target performance.

The contribution of this paper is as follows:

- Proposing a measure of energy-resilience considering the survivability, time to recovery (TTR), and real performance of the system,
- Providing partial-order between performance of (m, k)-firm constraints given an arbitrary (m, k)-firm pattern,
- Defining the system safety, including its performance and energy safety, for some pattern in a WHRT system and proving some theories about that,
- Proposing an energy-aware schedulability condition with low computational complexity, given the initial available energy and the (m, k)-firm constraints, and

RTNS '17, October 4–6, 2017, Grenoble, France

ACM ISBN 978-1-4503-5286-4/17/10…$15.00
https://doi.org/10.1145/3139258.3139282

- Using Model Predictive Control (MPC) in a proposed energy-resilient method to approximate the optimal solution, and proving that the approximation factor of the solution is less than or equal to 2.

The rest of this paper is organized as follows: Section 2 describes the system model including WHRT constraints and its patterns, energy supplier, and energy consumer. Section 3 defines energy-resilience and its metric based on the system performance and the problems considered in this paper. The resilient scheduler might enforce changing the system performance with the aim of maximizing its resilience, considering the possibility of performance and energy anomalies. In Section 4, we give some definitions and properties of the system to avoid such anomalies. Section 5 discusses our proposed online schedulability test. Section 6 describes our proposed energy-resilient method and Section 7 reports the experimental results. Finally, Section 8 concludes the paper.

2 SYSTEM MODEL

In this paper, we consider a single processor energy harvesting WHRT system consisting of an energy harvester (which scavenges energy from a continuous variable energy source), an energy storage unit (e.g. battery or super-capacitor), and an energy consumer (processing) unit consisting of a set of periodic tasks. In the following, we first describe the WHRT system in Subsection 2.1. We then express the energy consumer unit including the task model in Subsection 2.2. Finally, the energy supplier model, including the energy harvester and the energy storage unit, is given in Subsection 2.3.

2.1 Weakly-Hard Real-Time Systems

A WHRT system is a system in which the temporal constraints of periodic tasks can tolerate some well-defined degrees of deadline misses [22]. A type of such constraints for WHRT systems is the (m, k)-firm constraint, i.e. in any window of k consecutive jobs of a task, at least m jobs must meet their deadlines [23]. To satisfy the (m, k)-firm constraint, some jobs are selected as mandatory to meet their deadlines; the other jobs are called optional which can be dropped if necessary [24]. The selection of mandatory jobs can be done according to some patterns, where there exist well known patterns like E-pattern, which evenly distributes mandatory jobs as equally as possible, [25] and R-pattern (Red-only), which selects the first m out of portions with k jobs as mandatory [26]. In order to formally define such patterns, we need to first define (m, k)-pattern as below:

Definition 2.1. ((m, k)-**pattern**) [26]: Suppose τ_i is a task, and τ_{ij} is the j-th job of which. The (m, k)-pattern of task τ_i, denoted by ϕ_i, is a binary string $\phi_i = \{\varphi_{i0}\varphi_{i1}...\varphi_{i(k-1)}\}$ which satisfies the followings: (i) τ_{ij} is a mandatory job if $\varphi_{ij} = 1$ and it is an optional job if $\varphi_{ij} = 0$, and (ii) $\sum_{j=0}^{(k-1)} \varphi_{ij} = m$.

E-pattern [25]: To satisfy the (m, k) constraint of task τ_i according to E-pattern, the mandatory jobs are to be distributed equidistantly as much as possible according to (1) below:

$$\varphi_{ij} = \begin{cases} 1, & if\, j = \left\lfloor \left\lceil \frac{j\times m}{k} \right\rceil \times \frac{k}{m} \right\rfloor \\ 0, & otherwise \end{cases} \qquad j = 0, 1, ..., k-1 \qquad (1)$$

R-pattern [26]: To satisfy the (m, k) constraint of task τ_i according to R-pattern, after partitioning the jobs into sequences of k jobs (except for the last partition), the first m out of k jobs are selected as mandatory. More formally, this pattern satisfies (2):

$$\varphi_{ij} = \begin{cases} 1, & if\, 0 \leq j < m \\ 0, & otherwise \end{cases} \qquad j = 0, 1, ..., k-1 \qquad (2)$$

Scheduling a task with different (m, k) constraints may provide different performances. For example, scheduling with the $(3, 4)$ constraint may lead to a better performance with respect to $(2, 4)$. The relation of the (m, k)-firm constraints in E-pattern is straightforward; a constraint with bigger m/k outperforms another with a smaller one. However, in R-pattern the relation of (m, k) constraints is more complicated; for example, a $(3, 5)$-firm task is not better than $(1, 2)$ in R-pattern, even though $3/5 \geq 1/2$.

Another important point relates to possible anomalies during changes in the system performance level which may be triggered to be resilient with respect to energy changes. For example, the scheduler may decrease the performance of a task which runs according to R-pattern from $(3, 4)$ to $(2, 4)$; in this example, there are some transition windows[1] which are $(2, 5)$-firm, namely some window with performance less than both $(3, 4)$ and $(2, 4)$.

In Section 3, we show the performance relations of the constrains for the patterns, based on which, in Section 4 we discuss the *safety* property as an essential property that a pattern must have to be appropriate for an energy-resilient system.

2.2 Energy Consumer Model

The energy consumer is a set of n real-time fixed priority periodic tasks $\tau_i : (e_i, \pi_i, pow_i, M_i)$, $1 \leq i \leq n$, where e_i is the worst-case execution time (WCET), and π_i denotes the task period (with implicit relative deadline). Further, pow_i is the worst-case power consumption of τ_i, i.e. each instance of τ_i consumes up to $pow_i \times e_i$ units of energy.

The hyperperiod for the periodic tasks is shown as $\Pi = LCM(\pi_i)$. Also, L defines the maximum performance level of the system (discussed in detail in Section 3.1). M_i is a set of tuples with the size of $L + 1$ that shows the pre-defined (m, k)-firm constraints of task τ_i:

$$M_i = \{(m_{il}, k_{il})|\ m_{il} \leq k_{il}\ and\ \ 0 \leq l \leq L\}. \qquad (3)$$

We assume that the predictions are done with the resolution of the hyperperiod, i.e. the harvesting energy is considered constant along with a hyperperiod and the changes occur only at the hyperperiod boundaries. Accordingly, for each task τ_i we have $\frac{\Pi}{\pi_i}\, mod\, k_{il} = 0$, for all $0 \leq l \leq L$. This implies that the performance changes do not occur in the middle of any window of size k_{il}.

We assume that the tasks are ordered according to their priorities where τ_1 is the highest priority task and all of them use the same (m, k) pattern ϕ which can be either E-pattern or R-pattern. Further, we consider the system in the time interval of $[0, T]$ which includes T/Π hyperperiods.

[1] Transition window is a sliding window with jobs from two adjacent windows with different (m,k) constraints.

2.3 Energy Supplier Model

The energy supplier unit consists of an energy harvester and an energy storage unit. The energy harvester scavenges energy from a renewable energy source. There are many exploitable sources of environmental energy such as solar, wind, piezoelectric, radio frequency, etc. However, the yielded energy is not necessarily stable over time. For example, the energy generated by a solar cell depends on the intensity of light which is highly variable because of the day/night cycles and the weather conditions.

In the following, we suppose a solar panel which can charge the storage unit with the rates in the range of $[0, Rate_{max}]$ during the day. During night, however, the charging rate is 0. (This paper is not restricted to this energy source, however, the assumption helps more smooth discussions.) We assume that there is a prediction method which gives a lower bound on the charging rates for a time horizon of H hyperperiods.

The replenishment of the storage unit is performed continuously even during the execution of tasks. The harvested energy in the time interval of $[t_1, t_2]$, that is characterized by an instantaneous charging rate $Rate(t)$, is given as [27–31]:

$$E_P(t_1, t_2) = \int_{t_1}^{t_2} Rate(t)dt \quad (4)$$

As the other part of the energy supplier, we consider an ideal energy storage unit. It has a nominal capacity E_{max}. We assume that the storage unit has neither charging inefficiency, nor energy leakage. Furthermore, there is a safe available energy, E_{min}; if there is E_{min} units of energy in the storage unit, the system can survive during night (i.e. when the charging rate is zero) constantly at the minimum performance level.

3 ENERGY-RESILIENCE

Resilience is fundamentally defined as either "resuming the original shape or position after being bent, compressed, or stretched" or "rising readily again after being depressed" [32]. In a more formal definition, resilience is the persistence of performability when facing changes [33]; a resilient system must survive at some capacity, in order to autonomously recover [34]. In this paper, we consider the changes in the harvested energy that can affect the achievable performance of the system and the aim of this paper is to have an energy-resilient system.

Defined resilience metrics in the literature typically relates to the target performance curve, $Perf_{Target}(t)$, and the real performance curve, $Perf_{Real}(t)$ [35–38]. Target performance is the maximum feasible performance of the system and real performance records performance changes under disruptive events and the system restoration efforts. Resilience can then be quantified as the ratio of the area below $Perf_{Real}(t)$ and that below $Perf_{Target}(t)$ within the same time period of $[0, T]$ [38]:

$$R(T) = \frac{\int_0^T Perf_{Real}(t)dt}{\int_0^T Perf_{Target}(t)dt} \quad (5)$$

Therefore, we first need to define the performance of the system (Subsection 3.1). However, the above simple definition of the resilience measure has some deficiencies (given in the following) with respect to the properties of a resilient system [33, 34], and thus it needs improvement, as done in Subsection 3.2: 1) If the system is scheduled at performance level 0 at a time t, the system is not surviving, even though the value of resilience calculated by (5) can get some value greater than 0, 2) the system might have no trend back to its best performance despite that there is no charging rate reduction and the value of (5) is greater than 0. For the sake of more clarity, we consider an example that the maximum performance level of the system is L. Let's consider system A that schedules the tasks at performance level $l < L$ in the time interval of $[0, T]$ and system B that schedules them at performance level $l - 1$, and then after some time t elevates the performance level to L, which is held to the end of the same interval. If $(l-1)\times t + L\times(T-t) < l\times T$, then A is supposed more resilient than B according to (5), even though it performs nothing to approach the best performance.

3.1 Performance

As mentioned in Subsection 2.2, each task is supposed to have a set of size $L + 1$ of user-defined (m, k)-firm constraints, corresponding to the system performance levels. The following proposition defines the constraint relations.

Proposition 1. *For a periodic task τ_i, using any arbitrary pattern, the (m_{il}, k_{il})-firm constraint outperforms the $(m_{il'}, k_{il'})$-firm constraint, shown as $(m_{il}, k_{il}) \geq (m_{il'}, k_{il'})$, if at least one of these conditions holds:*

(1) $m_{il} = k_{il}$ *and* $m_{il'} = k_{il'}$
(2) $m_{il'} = m_{il} - 1$ *and* $k_{il'} = k_{il}$
(3) $m_{il'} = m_{il}$ *and* $k_{il'} = k_{il} + 1$
(4) $m_{il'} = m_{il} - 1$ *and* $k_{il'} = k_{il} - 1$
(5) $\exists c \in \mathbb{Z}^+, m_{il'} = c \times m_{il}$ *and* $k_{il'} = c \times k_{il}$

Proof. See Appendix A. □

Proposition 1 shows the relation between (m, k) constraints disregarding the pattern, i.e. it works for all patterns including R-pattern and E-pattern. In all conditions of the proposition we have $\frac{m_{il'}}{k_{il'}} \leq \frac{m_{il}}{k_{il}}$. For E-pattern, this relation results in some performance order, however, as also mentioned in the example of Section 2.1, it does not necessarily represent any performance order between (m_{il}, k_{il}) and $(m_{il'}, k_{il'})$ constraints. Thus, Proposition 1 does not represent some total relation (in which, for all (m_{il}, k_{il}) and $(m_{il'}, k_{il'})$, we have either $(m_{il}, k_{il}) \geq (m_{il'}, k_{il'})$ or $(m_{il'}, k_{il'}) \geq (m_{il}, k_{il})$); rather, when there is no restriction on the pattern, thus proposition gives only a partial order.

Regarding the mentioned uncomparability of some constraints for specific patterns, some (m, k) constraints may result in situations that it is not defined whether a desired performance level has been satisfied. To overcome this issue, for each task τ_i, we add a performance level 0 with constraints $(0, \max_{1\leq l\leq L}(k_{il}))$, as the lowest performance level which is valid and comparable to all performance levels, to the performance table. At performance level 0, the scheduler cannot guarantee any user-defined performance level. In this way, when there is no user-defined performance level lower than some (m, k) constraint, the performance level of the constraint is considered 0. In a similar fashion, the performance of task τ_i at time t, when it is (m_i, k_i)-firm, is defined as follows:

$$Perf(\tau_i) = \max(\{l | (m_i, k_i) \geq (m_{il}, k_{il})\}) \tag{6}$$

The performance of the system is then the minimum performance of all tasks:

$$Perf = \min_{1 \leq i \leq n} (Perf(\tau_i)) \tag{7}$$

We assume that the minimum acceptable performance of the system, at which the system still survives, is 1. Also, we study non-overloaded systems with utilization $U = \sum_{i=1}^{n} \frac{e_i}{\pi_i} \times \frac{m_{iL}}{k_{iL}} \leq 1$. Thus, the maximum performance of the system when there is no energy constraint is L.

3.2 Resilience Measure

Resilience of a system depends on the system survivability, its maximum performance disturbance, and the time to recovery to the goal performance. When the system is scheduled at performance level 0, it is not alive, and $R(T) = 0$. Furthermore, the system must have a trend to approach the best performance when there is no negative change in the system (charging rate, here). Additionally, it is expected that a resilient-system maximize the integral of its real performance. Accordingly, we define our measure of energy-resilience in the following.

Suppose that $Rate_h$ and $Perf_h$ (computed by (7)), respectively, are the charging rate and the real performance level of the system during the h^{th} hyperperiod, where $h = 1, 2, ..., T/\Pi$. Also, let's define $Perf_{min} = \min_{1 \leq h \leq T/\Pi} (Perf_h)$ for T/Π hyperperiods of the system. Then, we use the result of $\min(Perf_{min}, 1)$ to consider the survivability of the system, which returns 0, when there is at least one hyperperiod h (out of the T/Π hyperperiods) with performance level 0; otherwise, it returns 1.

Another part of the resilience measure relates to the expectation that the system approaches its best performance while there is no negative change in the charging rates. Suppose $x^+ = \max(0, x)$. If the charging rates are non-decreasing during two consecutive hyperperiods h and $h+1$, the value of $(Rate_h - Rate_{h+1})^+$ is equal to 0; otherwise, it is equal to or greater than 1. Thus, we define:

$$\min(\sum_{h=1}^{H-1} (Rate_h - Rate_{h+1})^+, 1) = \begin{cases} 0, & If\ the\ charging\ rates\ are\ non-decreasing \\ 1, & o.w. \end{cases} \tag{8}$$

If there is no negative change in the performance of the system during two consecutive hyperperiods h and $h+1$, the value of $(Perf_{h+1} - Perf_h + 1)^+$ is equal to or greater than 1; otherwise, it is equal to 0. Thus:

$$\min((Perf_{h+1} - Perf_h + 1)^+, 1) = \begin{cases} 1, & If\ there\ is\ no\ negative\ change\ in\ the\ system\ performance \\ 0, & o.w. \end{cases} \tag{9}$$

Then, we can calculate the maximum value of (8) and (9), and define the TTR coefficient of the h^{th} hyperperiod as:

$$TTR\_CO(h) = \max(\min(\sum_{h=1}^{H-1} (Rate_h - Rate_{h+1})^+, 1), \min((Perf_{h+1} - Perf_h + 1)^+, 1)) \tag{10}$$

The value of $TTR\_CO(h)$ is 0 when the charging rates are non-decreasing and the system performance has negative changes; otherwise, $TTR\_CO(h) = 1$.

The remaining part of our resilience measure relates to the system real and target performances, similar to what is mentioned in (5). Therefore, the resilience of the system in the time interval of $[0, T]$ can be calculated as follows:

$$R(T) = \min(Perf_{min}, 1) \times \min_{1 \leq h \leq T/\Pi} (TTR\_CO(h)) \times \frac{\sum_{h=1}^{T/\Pi} Perf_h}{L \times \frac{T}{\Pi}} \tag{11}$$

where the third part of (11) shows the ratio of the real and target performance of the system.

3.3 Problem Definition

The main problem considered in this paper is the resilience of the system with the model defined in Section 2.

Problem 1. *Suppose the time interval of* $[0, T]$*, with the hyperperiod index of* h*,* $1 \leq h \leq T/\Pi$*, and the prediction horizon of* H *hyperperiods, where the system performance can only be changed at the hyperperiod boundaries. Given* n *periodic task set* $\tau_i, 1 \leq i \leq n$*, and the corresponding performance levels* $(m_{il}, k_{il}), 0 \leq l \leq L$*, determine how we can maximize* $R(T)$*.*

In order to maximizing the system resilience, the scheduler needs to know at what performance level the task set is schedulable, given the current charging rate and the initial available energy in the storage unit as well as the (m, k) constraints. Therefore, we need a schedulability test to give the answer, which is to be addressed before the abovementioned problem.

Problem 2. *Given a periodic task set of tasks* $\tau_i, 1 \leq i \leq n$*, the* (m_{il}, k_{il}) *constraints for each task* τ_i*.* $0 \leq l \leq L$*, the initial available energy in the storage unit, and the constant charging rate* $Rate$*, determine the performance level* $0 \leq l \leq L$ *at which the task set is schedulable in a hyperperiod.*

There are sufficient schedulability test for fixed-priority hard real-time systems with energy harvesting [39]. This schedulability test should be refined in order to solving Problem 1. Because, it has a huge computational complexity that makes it unsuitable for a resilient scheduling algorithm which needs to know at what performance level the task set is schedulable during the runtime.

Furthermore, solving the main problem of this paper (Problem 1), is face to another problem: the safety of performance changes; the scheduler changes the system performance with the aim of maximizing the system resilience. These changes may lead to anomaly. For example, decreasing the performance level of the system from l to $l-1$ may leads to have a transition window with the lower performance level than $l-1$. Furthermore, decreasing the performance may leads to increasing in the energy consumption using some patterns. Therefore, our solutions must be anomaly care. In the next section, we define the safety of the system, including performance and energy safety, and we prove some theories about them.

4 SAFETY

Safety, in the context of this paper, means lack of anomaly, including performance and energy anomalies, when the scheduler reacts at time $t = (h-1)\Pi, h = 1, ..., T/\Pi$ in face of changes.

Definition 4.1. **(Performance Safety)** Suppose that the system changes its performance $Perf$ (see (7)) at an arbitrary time $t \in [0, T)$ and all of the tasks use the same pattern ϕ. Then, the (m, k) constraint of task τ_i, $1 \le i \le n$, changes from (m_i, k_i) to (m'_i, k'_i). ϕ is performance safe if for each transition window of task τ_i which satisfies some (m''_i, k''_i) constraint, we have either $(m_i, k_i) \ge (m''_i, k''_i) \ge (m'_i, k'_i)$ or $(m'_i, k'_i) \ge (m''_i, k''_i) \ge (m_i, k_i)$.

Assume that, $E^l_C(t, T)$ is the energy consumption of the tasks that are scheduled at performance level $0 \le l \le L$ in the time interval of $[t, T]$. When $t = (h-1)\Pi, h = 1, ..., T/\Pi$, it can be computed as:

$$E^l_C(t,T) = \frac{T-t}{\Pi} \sum_{i=1}^{n} \frac{\Pi}{\pi_i} \times \frac{m_{il}}{k_{il}} \times e_i \times pow_i \tag{12}$$

Definition 4.2. **(Energy Safety)** Suppose that the system which uses the pattern ϕ for all its tasks changes its performance at an arbitrary time $t \in [0, T)$ from l to l', where $l > l'$ and $0 \le l, l' \le L$. ϕ is energy safe if it is performance safe and $E^{l'}_C(t, T) \le E^l_C(t, T)$.

Let's discuss the performance safety and energy safety of R-pattern and E-pattern.

Theorem 4.3. *R-pattern is neither performance safe nor energy safe.*

Proof. Let's consider the conditions of Proposition 1. In the third condition, all transition windows are either (m, k) or $(m, k+1)$. In the second condition, there are $k - m$ transition windows that are $(m-1, k+1)$, as can be seen in Figure 1. Furthermore, in the forth condition, there are $k - m$ transition windows that are $(m-1, k)$. As $(m, k) \ge (m-1, k) \ge (m-1, k+1)$ and $(m, k) \ge (m-1, k-1) \ge (m-1, k)$, R-pattern is not performance safe. Consequently, as an energy safe pattern need to be performance safe (see Definition 4.2), R-pattern is not energy safe.

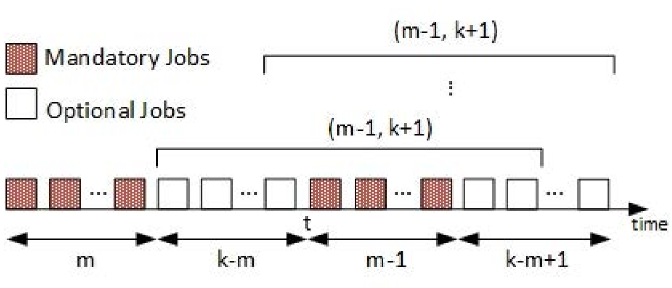

Figure 1: R-pattern is not performance safe considering the second condition of Proposition 1

□

Theorem 4.4. *E-pattern is performance safe.*

Proof. According to Corollary 2 in [24], all transition windows of E-pattern would meet one of the (m, k) constraints between any two windows in transition. Therefore, when the (m, k) constraint of τ_i changes from (m_{il}, k_{il}) to $(m_{il'}, k_{il'})$, all transition windows have the (m, k) constraint between (m_{il}, k_{il}) and $(m_{il'}, k_{il'})$. □

Theorem 4.5. *E-pattern is energy safe.*

Proof. According to the Lemma 4 in [24], The number of mandatory jobs in the time interval of $[t, T)$ in the E-pattern for constraint (m_{il}, k_{il}) is greater than or equal to that in a time interval of the same length for constraint $(m_{il'}, k_{il'})$, when $\frac{m_{il'}}{k_{il'}} \le \frac{m_{il}}{k_{il}}$. Then, we have:

$$E^l_C(t,T) = \frac{T-t}{\Pi} \sum_{i=1}^{n} \frac{\Pi}{\pi_i} \times \frac{m_{il}}{k_{il}} \times Pow_i \times e_i$$
$$\ge \frac{T-t}{\Pi} \sum_{i=1}^{n} \frac{\Pi}{\pi_i} \times \frac{m_{il'}}{k_{il'}} \times Pow_i \times e_i = E^{l'}_C(t,T)$$

□

The above theories shows that using E-pattern ensures that the performance changes does not lead to anomaly; hence, the rest of this paper uses E-pattern.

5 SCHEDULABILITY TEST

Our approach is to give an online schedulability test (Online-ST) for (m, k)-firm constraints subject to the available energy of the storage unit. We use PFP_{ASAP} [28] scheduling algorithm which at any time, selects the job of the highest priority active task and executes the next execution time unit of that job if there is sufficient energy available to do so. Abdeddaim et al. [39] derive the response time upper bound providing sufficient schedulability tests for PFP_{ASAP} when there are consuming[2] and gaining tasks[3].

Two reasons make this schedulabilty test, that we call it Offline-ST [4], unsuitable for an energy-resilient system: 1) Its time complexity is exponential with respect to the number of tasks and the length of the hyperperiod, 2) it don't consider the initial available energy in the storage unit and (m, k)-firm constraints. Our energy-resilient scheduling method needs to know the performance levels at which the task set are schedulable regards to the amount of energy in the storage unit and (m, k)-firm constraint during runtime. Therefore, we propose an online schedulability test according to the following theorem:

Theorem 5.1. *A task set $\tau_i : (e_i, \pi_i, Pow_i, M_i)$, $1 \le i \le n$ is schedulable at performance level $l, 0 \le l \le L$, given the charging rate $Rate$, with the initial available energy IE, if*

$$\forall 1 \le i \le n \qquad \sum_{j=1}^{i} \left\lceil \left\lceil \frac{\pi_i}{\pi_j} \right\rceil \frac{m_{jl}}{k_{jl}} \right\rceil e_j \le \pi_i \tag{13}$$

and

[2] The consuming task is a task with the energy consumption higher than the replenishment rate.
[3] The tasks that have a rate of energy consumption equal or less than the replenishment rate
[4] Offline Schedulability Test

$$\max_{1\leq i\leq n}\left(\begin{array}{l}\sum_{j=1}^{i}\left\lceil\left\lceil\frac{\pi_i}{\pi_j}\right\rceil\frac{m_{jl}}{k_{jl}}\right\rceil\times e_j\times(pow_j-Rate)^{+}+\\(\frac{\Pi}{\pi_i}-1)(\sum_{j=1}^{i}\left\lceil\left\lceil\frac{\pi_i}{\pi_j}\right\rceil\frac{m_{jl}}{k_{jl}}\right\rceil\times e_j\times pow_j\\-Rate\times\pi_i)^{+}\end{array}\right)\leq IE \quad (14)$$

Proof. PFP_{ASAP} executes all higher priority tasks before the lower priority ones. Hence, the periodic task τ_i meets its deadline if within a period of it, i.e. in the intervals of $[(w-1)\times\pi_i, w\times\pi_i]$ for $w\in\{1,2,...,\frac{\Pi}{\pi_i}\}$, we have $\sum_{j=1}^{i}\left\lceil\left\lceil\frac{\pi_i}{\pi_j}\right\rceil\frac{m_{jl}}{k_{jl}}\right\rceil\times e_j\leq\pi_i$. Therefore, for all tasks τ_i, $1\leq i\leq n$ the inequality (13) must hold.

Within a period of τ_i, in a pessimistic condition, the execution of the task τ_j, where $j\leq i$, must be started at time $t_j = w\times\pi_i - \sum_{q=j}^{i}\left\lceil\left\lceil\frac{\pi_i}{\pi_q}\right\rceil\frac{m_{ql}}{k_{ql}}\right\rceil\times e_q$. At that time, the system must have at least $e_j\times(pow_j-Rate)^{+}$ units of energy in the storage unit to execute the task τ_j. Therefore, the storage unit energy at the start of w^{th} period of τ_i, i.e. $E((w-1)\times\pi)$ where $E(t)$ denote the storage unit energy at time t, must be at least $\sum_{j=1}^{i}\left\lceil\left\lceil\frac{\pi_i}{\pi_j}\right\rceil\frac{m_{jl}}{k_{jl}}\right\rceil\times e_j\times(pow_j-Rate)^{+}$. In such condition, the amount of energy in the storage unit at the end of the period is as follows:

$$E(w\times\pi_i) = E((w-1)\times\pi)+Rate\times\pi_i-\sum_{j=1}^{i}\left\lceil\left\lceil\frac{\pi_i}{\pi_j}\right\rceil\frac{m_{jl}}{k_{jl}}\right\rceil\times e_j\times pow_j \quad (15)$$

The difference of the stored energy at the start and the end of the w^{th} period will be $(\sum_{j=1}^{i}\left\lceil\left\lceil\frac{\pi_i}{\pi_j}\right\rceil\frac{m_{jl}}{k_{jl}}\right\rceil\times e_i\times pow_i - Rate\times\pi_i)^{+}$. Therefore, the amount of energy in the storage unit at the start of the hyperperiod in order to executing the task τ_i, IE_i, is as follows:

$$\begin{array}{l}IE_i = \sum_{j=1}^{i}\left\lceil\left\lceil\frac{\pi_i}{\pi_j}\right\rceil\frac{m_{jl}}{k_{jl}}\right\rceil\times e_j\times(pow_j-Rate)^{+}+\\(\frac{\Pi}{\pi_i}-1)(\sum_{j=1}^{i}\left\lceil\left\lceil\frac{\pi_i}{\pi_j}\right\rceil\frac{m_{jl}}{k_{jl}}\right\rceil\times e_j\times pow_j-Rate\times\pi_i)^{+}\end{array} \quad (16)$$

The amount of energy needed to executing all task at the start of the hyperperiod is the maximum value of IE_i, $1\leq i\leq n$. So, inequality (14) and the theorem are proven. □

Online-ST has low computational complexity with respect to Offline-ST. However, its error is more than Offline-ST as will be shown by experiments in Section 7.

6 ENERGY-RESILIENT ALGORITHM

The proposed energy-resilient algorithm determines the performance level of the system at the start of each hyperperiod with the aim of maximizing the resilience in the given time interval $[0,T]$. It is utilized by MPC approach [40] in which at the start of each hypeprperiod, an optimal control problem is solved and the control input is applied for the current hyperperiod. Then, the state of the system is updated and at the start of the next hyperperiod the control problem is solved again.

In this paper, we have two models. The first one is based on the Integer Linear Programming (ILP) which can find the optimal solution[5]. This model can not be used as an online solution due to its huge computational complexity. The second model relaxes the control input in order to reducing the time complexity of the first model. We prove that the approximation ratio of the second model is less or eqaul than 2. The first model is as follows:

$$\begin{array}{l}\min\ -\sum_{h=1}^{H}A\times u_h\\ s.t.\\ E_{h+1}=E_h+B\times u_h+\Pi\times Rate_h\\ 0\leq E_h\leq E_{\max}\\ D\times u_h=1\\ u_h(j)\in\{0,1\}\quad 1\leq j\leq L\\ PR\times\lceil diag(C\times u_h)\rceil\times e\leq\pi\\ ||PR\times\lceil diag(C\times u_h)\rceil\times(e.(pow-Rate_h)^{+})+\\ Q\times(PR\times\lceil diag(C\times u_h)\rceil\times V-Rate_h\times\pi)^{+}||_\infty\leq E_h\\ (1-TTR)\times A\times(u_2-u_1)\geq 0\end{array} \quad (17)$$

where u_h is a binary vector of size L as the control input and E_h is the state of the system corresponds to the amount of energy in the storage unit at the start of the h^{th} hyperperiod. $u_h(j)$ is the j^th element of u_h and $TTR = \min(\sum_{h=1}^{H-1}(Rate_h - Rate_{h+1})^{+}, 1)$ is the time to recovery parameter. The other notations as as follows: A is the performance level coefficient; it is a vector of performance level indices of size L starts from L and end to 1. $B = [-E_C^L(0,\Pi)\ -E_C^{L-1}(0,\Pi)\ ...\ -E_C^1(0,\Pi)]$ is the energy consumption vector and C is the ratio of (m,k) constraints; it is a matrix of size $n\times L$ where the value of the i^{th} row and the j^{th} column of it, i.e. $C(i,j)$ is $m_{i(L-j+1)}/k_{i(L-j+1)}$. $D = [1...1]$ is a vector of size L. $e = [e_1\, e_2\, ...\, e_n]'$ is a vector of size n includes the WCET of the tasks, where x' is the transpose matrix of x. $\pi = [\pi_1\,\pi_2\,...\,\pi_n]'$ and $pow = [pow_1\, pow_2\, ...\, pow_n]'$ are the periods and the power consumption of all tasks, respectively. PR is a $n\times n$ lower triangular matrix contains the ratio of the task periods as follows:

$$PR(i,j) = \begin{cases}\frac{\pi_i}{\pi_j} & if\ i\geq j\\ 0 & o.w.\end{cases}$$

where $PR(i,j)$ is the value of the i^{th} row and the j^{th} column of PR. $Q = (\Pi/pi)-1$, $V = e.pow$, and $\|.\|_\infty$ is the infinity norm which in (17), it is equal to the maximum norm.

Solving the optimization problem (17) have a huge computational complexity. We relax the control parameter and the Online-ST of (17) in the second model which leads to a sub-optimal solution as follows:

$$\begin{array}{l}\min\ -\sum_{h=1}^{H}g(u_h)\\ s.t.\\ E_{h+1}=E_h+G\times u_h+\Pi\times Rate_h\\ 0\leq E_h\leq E_{\max}\\ u_{\min}\leq u_h\leq 1.0\\ PRE\times u_h+SE\leq\pi\\ ||PRE\times(u_h.(Pow-Rate_h)^{+})+e\times(Pow-Rate_h)^{+}+\\ Q\times(PRE\times(u_h.V)+Z-Rate_h\times\pi)^{+}||_\infty\leq E_h\\ (1-TTR)\times(g(u_2)-g(u_1))\geq 0\end{array} \quad (18)$$

[5] With respect to the prediction error and sufficient schedulability test.

where in this model, u_h is a vector of size n. Each element of this vector shows the ratio of m and k in (m, k)-firm model. $g(u_h)$ is the objective function that converts the ratio of m and k to the system performance. As (m, k) constraints are defined by user, the objective function may be different with respect to the defined performance levels. However, it must be a linear function which if for each task τ_i, $\frac{m_{il}}{k_{il}} \leq u_h(i) < \frac{m_{i(l+1)}}{k_{i(l+1)}}$ then $l \leq g(u_h) < l+1$. The other notations of (18) are as follows: $u_{\min} = [m_{11}/k_{11} \;\; m_{21}/k_{21} \;\; .... \;\; m_{n1}/k_{n1}]$ includes the (m, k) ratio of the tasks at performance level 1. Note that this is the minimum performance level of the system at which the system is alive. The value of the i^{th} row and the j^{th} column of PRE, i.e. $PRE(i, j)$, is as follows:

$$PRE(i,j) = \begin{cases} \frac{\pi_i}{\pi_j} e_j & if \;\; i > j \\ 0 & o.w. \end{cases}$$

SE is a vector of size n; the i^{th} element of SE is $\sum_{k=1}^{i} e_k$ and $Z = e' \times pow$.

The optimization problem of (18) is a relaxed version of (17) in which the controller determines the value of $\frac{m_{il}}{k_{il}}$ of each task τ_i at performance level $1 \leq l \leq L$; actually, it uses the facts that for the tasks τ_i and τ_j at performance level l, $\left\lceil \left\lceil \frac{\pi_i}{\pi_j} \right\rceil \times \frac{m_{jl}}{k_{jl}} \right\rceil = \left\lceil \frac{\pi_i}{\pi_j} \times \frac{m_{jl}}{k_{jl}} \right\rceil \leq (\frac{\pi_i}{\pi_j} \times \frac{m_{jl}}{k_{jl}}) + 1$, and $\left\lceil \frac{m_{il}}{k_{il}} \right\rceil = 1$ where $m_{il} > 0$. As (18) is a relaxed version of (17) which ignores some integer constraints, such as ceil function, it can be solved many faster than (17). Appendix B shows the simplification of (13) and (14) that leads to have the schedulability conditions in (18).

At the start of each hyperprtiod, the scheduler solve the relaxed optimization problem and schedules the task set at performance level $\lfloor g(u_1) \rfloor$. Theorem 6.1 shows the approximation ratio of (18).

Theorem 6.1. *The approximation ratio of (18) is less than or equal to 2;*

Proof. The relaxation of the schedulability conditions is correspond to the execution of one more job of each task (see Appendix B). Hence, there is at least one task that may go to the performance level $l + 1$. Note that, there are no performance levels that are quite similar, otherwise, they have the same label. As the performance level of the system is the minimum performance level of all tasks, the difference of the schedulability test, i.e. (13) and (14), and its relaxed version is at most one performance level. Therefore, at the start of each hyperperiod the difference of the selected performance level by (17) and (18) is at most 1. Hence, for the T/Π hyperperiods, the ratio of the optimal solution, i.e. OPT, and the relaxed MPC is as follows:

$$Approximation\,Factor = \frac{OPT}{OPT - T/\Pi} \tag{19}$$

In some situations, the optimal solution may exist at performance level 1 whereas the relaxed MPC may goes to performance level 0. This may leads to have an invalid approximation factor by (19). Therefore, without loss of generality, we may simply assume that the performance levels starts from 1 to $L + 1$ and scheduling the tasks at performance level 1 violates the survivability of the system. By this assumption, if there is a resilient solution, i.e. $R(T) > 0$, we have $OPT \geq 2 \times T/\Pi$ and hence the approximation factor, i.e. (19), is less or equal than 2. □

7 EXPERIMENTAL RESULTS

This section shows the results of our experiments that show the error of the proposed schedulability test and the effectiveness of the proposed energy-resilient method.

7.1 Task Set Generation

The task generation is based on the UUnifast algorithm [41] and the hyperperiod limitation technique [42]. The task parameters are randomly generated as follows: the task utilizations are generated using the UUnifast algorithm; the periods are generated with a hyperperiod limitation technique and the power consumption of the tasks are between 1 and 10 units of energy.

In order to generating the (m, k) constraints of the tasks in our experiments, for each task τ_i, the maximum value in the interval of $[1, 9]$[6], namely κ_i, that satisfies $\frac{\Pi}{\pi_i} \, mod \, \kappa_i = 0$ is computed. The maximum performance level of the system, i.e. L, is the maximum value of κ_i, $1 \leq i \leq n$. For each task τ_i, we set $k_{i1} = k_{i2} = ... = k_{iL} = \kappa_i$. At each performance level $1 \leq l \leq L$, the (m, k) constraint of task τ_i is $(\max(l - L + k_{il}, 1), k_{il})$ and at performance level 0, the constraint is $(0, k_{il})$.

7.2 Tightness of Online-ST

In order to measuring the error of the Online-ST with respect to the performance level that the task set is schedulable using PFP_{ASAP}, we compare the result of Theorem 5.1, i.e. Online-ST, with the real performance level that the task set is schedulable at it. For each task set we construct a table; each rows of this table correspond to a charging rate in the interval of $[0, Rate_{max}]$ and each column is correspond to an initial available energy in the interval of $[0, E_{max}]$. The (i, j)-th cell of the table shows the maximum performance level at which the task set is schedulable while the charging rate is i and the initial available energy is j. The value of each cell is computed using Online-ST; we call this table as $MSPT_{Online-ST}$[7] and Offline-ST, $MSPT_{Offline-ST}$. On the other hand, we fill this table according to PFP_{ASAP}; for the given initial available energy and charging rate, we run PFP_{ASAP} at maximum performance level (L); if there is any deadline miss, we try to schedule the tasks at performance level $L - 1$, otherwise L is the maximum performance level that the task set is schedulable at which using PFP_{ASAP} given the charging rate and initial available energy; it is repeated as long as such maximum performance level is found. We call the table as $MSPT_{PFP\_ASAP}$.

The error of a schedulability test x, which x is either Online-ST or Offline-ST, can computed as follows:

$$Error = \frac{MSPT_{PFP\_ASAP}(Rate, E(0)) - MSPT_x(Rate, E(0))}{MSPT_{PFP\_ASAP}(Pr, E(0))} \tag{20}$$

where $MSPT_{PFP\_ASAP}(Rate, E(0))$ and $MSPT_x(Rate, E(0))$, respectively, are the maximum performance level at which the tasks are schedulable according to PFP_{ASAP} and schedulability test x, given

[6] Simply assume that each task has at most 9 performance levels
[7] Maximum System Performance Table of Online-ST

the charging rate $Rate$ and the initial available energy $E(0)$. Note that the error value is a positive number.

Figure 2 shows the error of Online-ST and Offline-ST. Each point in Figure 2 shows the mean and variance of the error of the schedulability tests for 50 task sets with the given utilization. The number of tasks in each task set is 5 and 10 in Figure 2.a and Figure 2.b, respectively. As can be seen, the error of the proposed Online-ST is negligible with respect to Offline-ST. In terms of time, Online-ST takes about 100 to 250 milliseconds while Offline-ST takes several minutes to compute the schedulability of the tasks set in a Core i7 2.8 GHz CPU.

7.3 Energy-Resilient Scheduling

In this section, we illustrate the effectiveness of our proposed energy-resilient algorithm. For MPC design we have used YALMIP toolbox [43]. At the start of the h^{th} hyperperiod, the controller gets the value of $E_h = E((h-1)\times\Pi)$, $[Rate_h \; Rate_{h+1} \ldots Rate_{h+H-1}]$, and TTR as the parameters. For the experiments, we use solar data from Solar Radiation Lab (SRL) [44]. We assume that the solar panel is $10cm^2$; hence, after the conversion $Rate_{max} = 12$.

According to the generated (m, k) constraints in our experiments (see Subsection 7.1), $g(u_h) = \|F \times u_h + Offset\|_{-\infty}$ where F is a diagonal matrix of size $n \times n$ includes the number of different performance levels for each task and $Offset$ is a vector of size n; the i^{th} element of this vector, corresponds to task τ_i is $L - F(i, i)$.

Figure 3 shows the simulation results of the proposed energy-resilience method over a time period of 5 days (from 1 to 5 July 2017). The initial available energy in the storage unit is equal to E_{min}. The prediction time horizon in this experiment is 144 hyperperiod and the length of each hyperperiod is 10 minutes, i.e. we simply assume that the prediction time horizon is one day. Each point of the figure shows the mean and variance of the resilience/execution time over 50 periodic task set.

Because of the huge computational complexity of (17), we done some experiments on the 250 small task sets ($n = 3$) and prediction time horizon ($H = 10$). The experiments show that the mean and variance of the approximation factor is 1.21 and 0.11, respectively.

As can be seen, the time complexity of the proposed method is very low and it makes a good approximation of the optimal solutions. Hence, it is an effective method to find the performance level of each hyperperiod in order to maximizing the value of resilience.

8 CONCLUSIONS

In this paper, we target the energy-resilience of WHRT systems in which the amount of harvested energy from a renewable energy source such as solar, changes due to the environmental conditions. We assume that the energy changes occur only at hyperperiod boundaries and there is a lower bound on the predicted charging rates.

We propose some conditions to have a partial order over (m, k) constraints in WHRT systems. Based on that, we define the safety property of the system. We prove that E-pattern guarantees either performance or energy safety when the resilient scheduling algorithm changes the performance of the system. Then, we propose an online schedulability test for PFP_{ASAP} while considering the available energy at the start of each hyperperiod and (m, k) constraints. This schedulability test can be used by the proposed energy-resilient algorithm during runtime.

Further, we propose a measure of resilience that considers the survivability and the system trends to back to its best performance. Finally, we propose an energy-resilient scheduling algorithm utilized by MPC approach which with the approximation factor of less than or equal to 2 can provides a suboptimal solution. The experiments show the effectiveness of the proposed energy-resilient method.

A PROOF OF PROPOSITION 1

All constraints of the first condition have hard deadlines. Thus, they are equal. According to the definition of (m_{il}, k_{il})-firm model [23], Conditions 2 and 3 are obviously valid. In Condition 4, suppose we have a sequence S of k_{il} jobs of task τ_i that is (m_{il}, k_{il})-firm and A is the first and B is the last job. Also, suppose that S_1 is a subsequence of consecutive jobs of S from 1 to $k_{il} - 1$ and S_2 is one from 2 to k_{il} (see Figure 4). A and B have four states related to meet or miss their deadlines. As shown in Table 1, in all of these states, S_1 and S_2 are $(m_{il} - 1, k_{il} - 1)$-firm deadline. Note that, if a sequence of consecutive jobs is $(m_{il}, k_{il} - 1)$-firm deadline, according to Condition 2, it is also $(m_{il} - 1, k_{il} - 1)$-firm deadline.

Table 1: All possible states of A and B

A	B	Number of successful jobs in C	S_1	S_2
meet	meet	$m_{il} - 2$	$(m_{il} - 1, k_{il} - 1)$	$(m_{il} - 1, k_{il} - 1)$
meet	miss	$m_{il} - 1$	$(m_{il}, k_{il} - 1)$	$(m_{il} - 1, k_{il} - 1)$
miss	meet	$m_{il} - 1$	$(m_{il} - 1, k_{il} - 1)$	$(m_{il}, k_{il} - 1)$
miss	miss	m_{il}	$(m_{il}, k_{il} - 1)$	$(m_{il}, k_{il} - 1)$

In Condition 5, we need to prove that $(m_{il}, k_{il}) \geq (cm_{il}, ck_{il})$, where $c \in \mathbb{Z}^+$. If task τ_i is (m_{il}, k_{il})-firm constraint according to pattern ϕ, then there are m_{il} mandatory jobs in any non-overlapped consecutive window of size k_{il}, namely $W_1, W_2, \ldots$. Now, consider all non-overlapped consecutive windows of size ck_{il}, namely $W_1', W_2', \ldots$, from the start point of W_1. All of these windows have cm_{il} mandatory jobs according to pattern ϕ (note that, $W_1, W_2, \ldots$ are constructed according to pattern ϕ). Consequently, all sliding windows of size ck_{il} have cm_{il} mandatory jobs.

These conditions provide the partial order constraints. To complete the proof, lets consider the symmetry and transitivity of the relations. For symmetry, we need to prove that if $(m_{il}, k_{il}) \geq (m_{il'}, k_{il'})$ and $(m_{il'}, k_{il'}) \geq (m_{il}, k_{il})$ then $(m_{il}, k_{il}) = (m_{il'}, k_{il'})$. These relations can be held under Condition 1, and Condition 5 when $c = 1$. In Condition 1, (m_{il}, k_{il}) and $(m_{il'}, k_{il'})$ have hard deadline constraints and are equal. In Condition 5, when $c = 1$, we have $m_{il} = m_{il'}$ and $k_{il} = k_{il'}$.

For the transitive property, we need to prove that if $(m_{il}, k_{il}) \geq (m_{il'}, k_{il'})$ and $(m_{il'}, k_{il'}) \geq (m_{il''}, k_{il''})$ then $(m_{il}, k_{il}) \geq (m_{il''}, k_{il''})$. For Conditions 1 to 3, according to the definition of (m, k)-firm model, the proof is immediate. For Condition 4, we can consider a scenario similar to Figure 4, with three subsequences of consecutive jobs S_1, S_2, and S_3, as shown in Figure 5. A, A', B, and B'

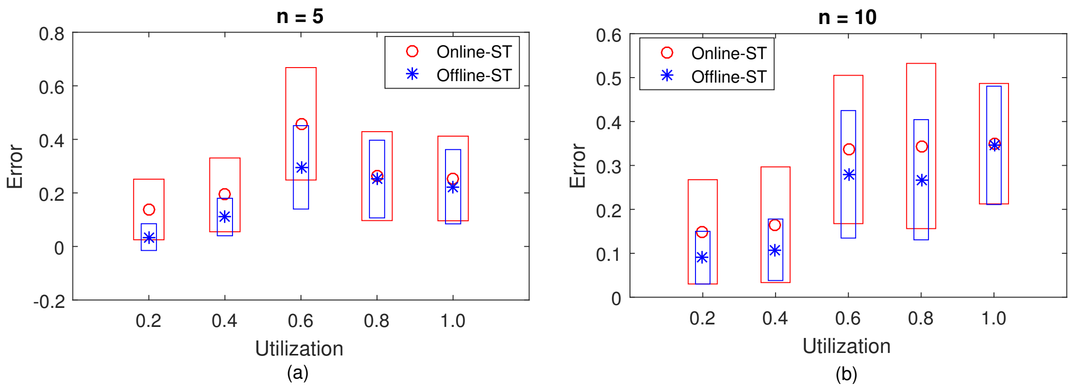

Figure 2: Comparing the mean and variance of the Online-ST and Offline-ST error, a) $n = 5$**, b)** $n = 10$.

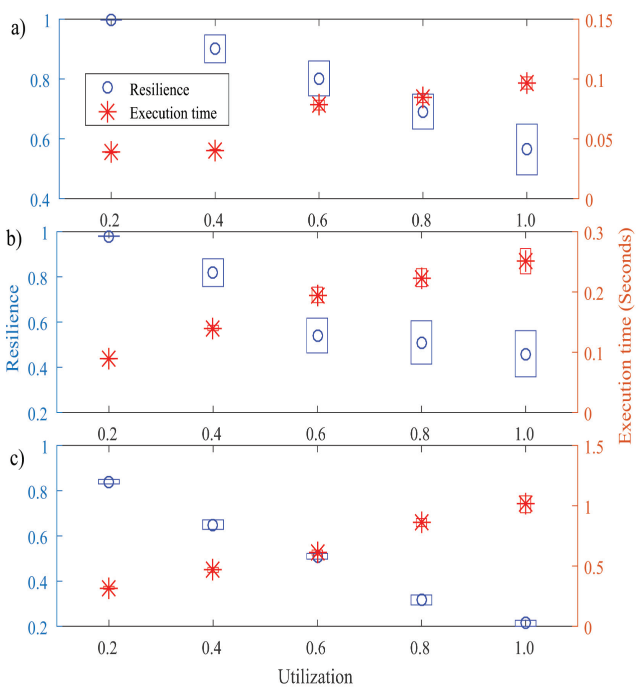

Figure 3: The mean and variance of the resilience and the execution time of the proposed method, a) $n = 5$**, b)** $n = 10$**, c)** $n = 20$

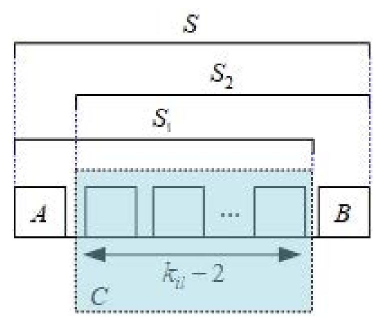

Figure 4: A (m, k)**-firm task is aslo a** $(m-1, k-1)$**-firm deadline.**

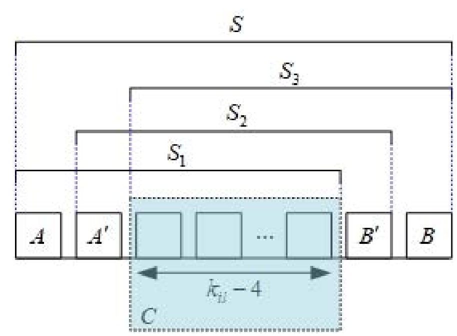

Figure 5: A (m, k)**-firm task is aslo a** $(m-2, k-2)$**-firm deadline.**

have 16 states related to meet or miss their deadlines. In all of these states, S_1, S_2, and S_3 are $(m_{il} - 2, k_{il} - 2)$, $(m_{il} - 1, k_{il} - 2)$, or $(m_{il}, k_{il} - 2)$. According to Condition 2 and its transitive property, $(m_{il}-1, k_{il}-2) \geq (m_{il}-2, k_{il}-2)$ and $(m_{il}, k_{il}-2) \geq (m_{il}-2, k_{il}-2)$.

In Condition 5, suppose that $(m_{il}, k_{il}) \geq (cm_{il}, ck_{il})$ and $(cm_{il}, ck_{il}) \geq (c'(cm_{il}), c'(ck_{il}))$, for all $c, c' \in \mathbb{Z}^+$. As $cc' \in \mathbb{Z}^+$, then $(m_{il}, k_{il}) \geq (cc'm_{il}, cc'k_{il})$ as well.

B RELAXATION OF ONLINE-ST

This appendix simplifies the inequalities (13) and (14).

Let's start with (13).

$$\sum_{j=1}^{i}\left\lceil\left\lceil\frac{\pi_i}{\pi_j}\right\rceil\frac{m_{jl}}{k_{jl}}\right\rceil e_j = \sum_{j=1}^{i-1}\left\lceil\frac{\pi_i}{\pi_j}\times\frac{m_{jl}}{k_{jl}}\right\rceil e_j + e_i$$
$$\leq \sum_{j=1}^{i-1}\left(\frac{\pi_i}{\pi_j}\times\frac{m_{jl}}{k_{jl}} + 1\right)e_j + e_i = \sum_{j=1}^{i-1}\left(\frac{\pi_i}{\pi_j}\times\frac{m_{jl}}{k_{jl}}\right)e_j + \sum_{j=1}^{i} e_i \qquad (21)$$

As a result, if $\sum_{j=1}^{i-1}\left(\frac{\pi_i}{\pi_j}\times\frac{m_{jl}}{k_{jl}}\right)e_j + \sum_{j=1}^{i} e_i \leq \pi_i$ then $\sum_{j=1}^{i}\left\lceil\left\lceil\frac{\pi_i}{\pi_j}\right\rceil\frac{m_{jl}}{k_{jl}}\right\rceil e_j \leq \pi_i$.

Similar to (21), for (14) we have:

$$\begin{aligned}
&\sum_{j=1}^{i}\left\lceil\left\lceil\frac{\pi_i}{\pi_j}\right\rceil\frac{m_{jl}}{k_{jl}}\right\rceil\times e_j\times(pow_j-Rate)^{+}+\\
&(\frac{\Pi}{\pi_i}-1)(\sum_{j=1}^{i}\left\lceil\left\lceil\frac{\pi_i}{\pi_j}\right\rceil\frac{m_{jl}}{k_{jl}}\right\rceil\times e_j\times pow_j-Rate\times\pi_i)^{+}\\
&\leq\sum_{j=1}^{i-1}(\frac{\pi_i}{\pi_j}\times\frac{m_{jl}}{k_{jl}})\times e_j\times(pow_j-Rate)^{+}\\
&+\sum_{j=1}^{i}e_j\times(pow_j-Rate)^{+}\\
&+(\frac{\Pi}{\pi_i}-1)(\sum_{j=1}^{i-1}(\frac{\pi_i}{\pi_j}\times\frac{m_{jl}}{k_{jl}})\times e_j\times pow_j\\
&+\sum_{j=1}^{i}e_j\times pow_j-Rate\times\pi_i)^{+}
\end{aligned}\tag{22}$$

The above equalities leads to have the schedulablity conditions as shown in (18).

REFERENCES

[1] S. Priya and D. J. Inman. 2009. Energy Harvesting Technologies. NewYork, NY, USA: Springer-Verlag.

[2] D. Fridley. 2010. Nine Challenges of Alternative Energy. Post Carbon Institute.

[3] Alan Burns, Robert Davis, and Sasikumar Punnekat. 1996. Feasibility Analysis of Fault-Tolerant Real-Time Task Sets. In Proceedings of the 8th Euromicro Workshop on Real-Time Systems (EUROWRTS), 29-33.

[4] Mahesh H. Pandya and Miroslaw Malek. 1998. Minimum Achievable Utilization for Fault-Tolerant Processing of Periodic Tasks. IEEE Trans. on Computers, 47, 10, 1102-1112.

[5] F. Liberato, R. Melhem, and D. Mosse. 2000. Tolerance to Multiple Transient Faults in Hard Real-Time Systems. IEEE Trans. on Computers, 49, 9, 906-914.

[6] Mohammad A. Haque, Hakan Aydin, and Dakai Zhu. 2014. Real-Time Scheduling under Fault Bursts with Multiple Recovery Strategy. IEEE 20th Real-Time and Embedded Technology and Applications Symposium (RTAS).

[7] Guy Koren and Dennis Shasha. 1995. Skip-over algorithms and complexity for overloaded systems that allow skips. Proceedings of the 16th IEEE Real-Time Systems Symposium (RTSSfi95), 110-117.

[8] M. Chetto. 2015. Graceful Overload Management in Firm Real-Time Systems. Journal of Information Technology and Software Engineering, 5,3.

[9] Gilad Koren and Dennis Shasha. 1995. An Approach To Handling Overloaded Systems That Allow Skips. 16th IEEE Real Time Systems Symposium (RTSS), 110-119.

[10] Maryline Chetto and Audray Marchand. 2007. Dynamic Scheduling of Skippable Periodic Tasks in Weakly-Hard Real-Time Systems. 14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS).

[11] Francois Santy, Laurent George, Philippe Thierry, and Joel Goossens. 2015. Relaxing mixed-criticality scheduling strictness for task sets scheduled with FP, In Proceedings of Euromicro Conference on Real-Time Systems (ECRTS), 155-165.

[12] Tom Fleming and Alan Burns. 2014. Incorporating the notion of importance into mixed criticality systems. In Proceedings of Workshop on Mixed Criticality, IEEE Real-Time Systems Symposium (RTSS), 33-38.

[13] Oliver Gettings, Sophie Quinton and Robert I. Davis. 2015. Mixed Criticality Systems with Weakly-Hard Constraints. In the Proceedings of the 23rd International Conference on Real Time and Networks Systems (RTNS'15), 237-246.

[14] Han S. Yun and Jihong Kim. 2003. On energy-optimal voltage scheduling for fixed priority hard real-time systems. ACM Trans. on Embedded Computing Systems, 2,3, 393-430.

[15] Youngsoo Shin and Kiyoung Choi. 1999. Power conscious fixed priority scheduling for hard real-time systems. In Proceedings of the 36th ACM/IEEE Conference on Design Automation Conference, 134-139.

[16] Saowanee Saewong and Ragunathan Rajkumar. 2003. Practical voltage-scaling for fixed-priority RT-systems. In Proceedings of the 9th IEEE Real-time and Embedded Technology and Applications Symposium (RTASfi03), 106-115.

[17] Hessam Kooti, Nga Dang, Deepak Mishra and Eli Bozorgzadeh. 2012. Energy Budget Management for Energy Harvesting Embedded Systems. IEEE International Conference on Embedded and Real-Time Computing Systems and Applications, 320-329.

[18] Christiane Moser, Davide Brunelli, Lothar Thiele, and Luca Benini. 2006. Real-time scheduling with regenerative energy. Proceedings of the 18th EuroMicro Conference on Real-Time Systems (ECRTS'06).

[19] Hussein El Ghor, Maryline Chetto and Raffic Hage Chehade. 2011. A Real-Time Scheduling Framework for Embedded Systems with Environmental energy Harvesting. In Computers and Electrical Engineering.

[20] Maissa Abdallah, Maryline Chetto and Audray Queudet. 2013. Energy-aware schedulers for Real-Time Energy Harvesting systems with Quality of Service requirements. In 2nd International Conference on Advances in Computational Tools for Engineering Applications (ACTEA), 342-347.

[21] Borislav Nikolic, Konstantinos Bletsas and Stefan M. Petters. 2015. Hard real-time multiprocessor scheduling resilient to core failures. International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA), 122-131.

[22] Guillem Bernat, Alan Burns and Albert Liamosi. 2001. Weakly Hard Real-Time Systems. IEEE Trans. on Computers, 50,4, 308-321.

[23] Mohammad Hamdaoui and Parameswaran Ramanathan. 1995. A Dynamic Priority Assignment Technique for Streams with (m, k)-Firm Deadlines. IEEE Trans. on Computers, 44, 12, 1443-1451, 1995.

[24] Nga Dang, Eli Bozorgzadeh and Moonju Park. 2015. Multi-level QoS Support with Vari- able Window Size in Weakly Hard Real-Time Systems. IEEE 3rd International Conference on Cyber-Physical Systems, Networks, and Applications, 25-30.

[25] Parameswaran Ramanathan. 1999. Overload Management in Real-Time Control Applications Using (m, k)-Firm Guarantee. IEEE Trans. on Parallel and Distributed Systems, 10, 6, 549-559.

[26] Gang Quan and Xiaobo Hu. 2000. Enhanced Fixed-Priority Scheduling with (m, k)-Firm Guarantee. In RTSS10 Proceedings of the 21st conference on Real-time sys- tems, 79-88.

[27] Hussein EL Ghor, Maryline Chetto and Rafic Hage Chehade. 2011. A real-time scheduling framework for embedded systems with environmental energy harvesting. Journal of Computers and Electrical Engineering, 37, 4, 498-510.

[28] Yasmina Abdeddam, Younes Chandarli, and Damien Masson. 2013. The Optimality of PFP_{asap} Algorithm for Fixed-Priority Energy-Harvesting Real-Time Systems. Proceedings of the EuroMicro Conference on Real-Time Systems (ECRTS'13).

[29] Maryline Chetto. 2014. Optimal Scheduling for Real-Time Jobs in Energy Harvesting Computing Systems. IEEE Trans. of Emerging Topics in Computing, 2, 2, 122-133.

[30] Yasmina Abdeddaim, Younes Chandarli, Robert I. Davis and Damien Masson. 2016. Response time analysis for fixed priority real-time systems with energy-harvesting. Journal of Real-Time Systems, Springer, 52, 2, 125-160.

[31] Meng-Lin Ku, Wei Li, Yan Chen and K. J. Ray Liu. 2016. Advances in Energy Harvesting Communications: Past, Present, and Future Challenges. IEEE Communications Surveys and Tutorials, 18, 2, 1384-1412.

[32] John Simpson and Edmund Weiner. 1989. The Oxford English Dictionary. Clarendon Press, Oxford, UK, second edition.

[33] Matt Bishop, Marco Carvalho, Richard Ford and Liam M. Mayron. 2011. Resilience is more than Availability. New Security Paradigms Workshop, 95-104.

[34] J. F. Meyer. 2013. Model-based evaluation of system resilience. Annual IEEE/IFIP Conference on Dependable Systems and Networks, 24-27.

[35] Michel Bruneau, Stephanie E. Chang, Ronald T. Eguchi, George C. Lee, Thomas D. OfiRourke, Andrei M. Reinhorn, et al. 2003. A framework to quantitatively assess and enhance the seismic resilience of communities. Earthquake Spectra, 19, 4, 733-752.

[36] Michel Bruneau and Andrei Reinhorn. 2007. Exploring the concept of seismic resilience for acute care care facilities. Earthquake Spectra, 23, 1, 41-62.

[37] Tomas D. OfiRourke. 2007. Critical infrastructure, interdependencies, and resilience. The BridgeLinking Engineering and Society, 22-29.

[38] Min Quyang and Leonardo Dueas-Osorio. 2014. Multi-dimensional hurricane resilience assessment of electric power systems. Journal of Structural Safety, 48, 15-24.

[39] Yasmina Abdeddam, Younes Chandarli, Robert I. Davis, and Damien Masson. 2014. Approximate Response Time for Fixed Priority Real-Time Systems with Energy-Harvesting. http://hal.archives-ouvertes.fr/hal-00986340/en.

[40] E. F. Camacho and C. Bordons Alba. 2004. Model Predictive Control. Springer.

[41] Enrico Bini and Giorgio C. Buttazzo. 2005. Measuring the Performance of Schedulability Tests. Real-Time Systems, 30, 1, 129-154.

[42] Joel Goossens and Christophe Macq. 2001. Limitation of the hyper-period in real-time periodic task set generation. In Real-time and Embedded Systems, 133-147.

[43] Johan Lofberg. 2004. YALMIP : a toolbox for modeling and optimization in MATLAB. IEEE International Symposium on Computer Aided Control Systems Design.

[44] J. Ramos and A. Andreas. 2011. University of Texas Panamerican (UTPA): Solar Radiation Lab (SRL); Edinburg, Texas (Data); NREL Report No. DA-5500-56514. http://dx.doi.org/10.5439/1052555