Analysis and Scheduling of a Battery-Less Mixed-Criticality System with Energy Uncertainty

Sedigheh Asyaban, DRTS Research Lab, School of ECE, University of Tehran, Tehran, Iran
Mehdi Kargahi, School of ECE, University of Tehran, Tehran, Iran
Lothar Thiele, Swiss Federal Institute of Technology Zurich, Switzerland
Morteza Mohaqeqi, School of ECE, University of Tehran, Tehran, Iran

We consider a battery-less real-time embedded system equipped with an energy harvester. It scavenges energy from an environmental resource according to some stochastic patterns. Because of technical constraints, the harvester device first buffers the harvested energy into a reservoir; then, at regular time intervals, the buffered energy is transferred to a super-capacitor. The success of jobs is threatened in case of energy shortage which might be due to lack of harvested energy, losses originated from the super-capacitor self-discharge, as well as power consumption of executed tasks. The periodic real-time tasks of the system follow a dual-criticality model. In addition, each task has a minimum required success-ratio that needs to be satisfied in steady-state. We analytically evaluate the behavior of such a system in terms of its energy-related success-ratio for a given schedule. Based on these results, we propose a scheduling algorithm that satisfies both, the temporal and success-ratio constraints of the jobs, while respecting task criticalities and corresponding system modes. The accuracy of the analytical method as well as its dependence on the numerical computations and other model assumptions are extensively discussed through comparison with simulation results. Also, the efficacy of the proposed scheduling algorithm is studied through comparison to some existing non-mixed- and mixed-criticality scheduling algorithms.

Additional Key Words and Phrases: Mixed criticality and Real-time scheduling and Stochastic analysis and Energy harvesting

1. INTRODUCTION

One of the biggest challenges for Wireless Sensor Networks (WSNs) and the emerging concept of Internet of Things (IoT) is powering tens of billions of WSN/IoT devices with the expectation of perpetual and long-term autonomous operation [Brown 2014]. Many such devices have limited space for a battery with appropriate size, motivating cordless power provisioning using energy harvesters. Furthermore, repeated battery replacement is almost impossible due to the expected enormous number of such devices. A promising approach to address these limitations is to combine energy harvesting with super-capacitors with very high cycle life [Chai and Zhang 2015].

In order to use energy storages like super-capacitors, however, their pros and cons must be carefully taken into account. For example, potential for quick charge and discharge and very high cycle life are positive characteristics of a super-capacitor, while power dissipation (due to self-discharge) and low power density [Yang and Zhang 2013] are negative properties. Further, as there is a linear charge-voltage relationship for super-capacitors, the usable power spectrum reduces, namely some considerable amount of stored energy will not be available because the super-capacitor voltage drops to a too low level (the cut-off voltage) for a standard voltage-regulator to operate.

Despite the existing limitations, the development of intelligent system-level power management techniques and the use of environmental energy scavenging decreases or even eliminates the dependence on batteries [Brown 2014]. Overall, there is a paradigm shift towards small-scale energy harvesters, especially in miniature devices [Taufik et al. 2012]. As a consequence of this development, it is a major challenge to seamlessly and efficiently maintain the system survivability in the presence of intermittent and unreliable power supplies. As also mentioned in [Piorno et al. 2010], even for relatively stable resources the incoming energy follows stochastic patterns, a point which must be considered in the hardware and software design. For real-time embedded systems using harvested energy, some additional constraints need to be satisfied in order to guarantee that energy uncertainty will not violate the system schedulability.

In this paper, we assume that tasks may fail due to energy or time shortage, where the failure probability of tasks due to lack of energy is required to be bounded. The combination of the following properties create major challenges for a viable system design: (i) the stochastic nature of energy harvesting, (ii) the super-capacitor power dissipation, (iii) the energy consumption of executing tasks, and (iv) the task real-time and success probability requirements. In order to take full advantage of the energy-harvesting technology, we need efficient scheduling algorithms which take the finite capacity of the energy storage into account [Huang and Neely 2013].

This paper considers mixed-criticality real-time systems [Burns and Davis 2015] where tasks are classified according to their criticality levels and may show different worst-case execution time behaviors at run-time. More specifically, to each task there is associated a set of execution time bounds with different levels of pessimism. Based on the observed execution time in comparison to these bounds, the overall system changes its scheduling mode such that temporal guarantees can still be given, but only for well-defined subsets of tasks. The fact that the scheduler switches between two or more criticality modes helps to integrate more functionalities into a device and to better utilize its limited resources.

In this paper, we adopt the widely used dual criticality model, where the tasks are of two criticality levels. Tasks with lower criticality may not be executed at all, stopped or are provided lesser service if high-criticality tasks exceed their low execution time bounds. However, in an energy harvesting scenario, high-criticality tasks exceeding their low execution time bound require extra energy in addition to the needed extra time. Due to the finite capacity of the energy storage and the uncertainty in the harvested energy, the described variation in execution time has a strong effect on the success ratios of subsequent tasks.

As usual, a feasible schedule guarantees that mixed-criticality time constraints of jobs will be satisfied. For jobs belonging to high-criticality tasks, the statistical success-ratio constraints should be always guaranteed. When all jobs execute within their low criticality execution time bound, the success-ratio constraints of the low-criticality tasks must be satisfied in addition. In this case, all jobs that are guaranteed to execute according to the classic mixed-criticality scheduling paradigm need to satisfy their energy-related success probability constraints as well. Certainly, the approach can further be generalized to settings where low-criticality tasks receive a degraded service in such a case, e.g., they may violate their deadline or they may reduce their time and energy demands by switching to a basic functionality.

A typical application example may be a mission-critical node in a WSN/IoT scenario. The node is "zero power" (i.e., it is completely energy autonomous) and harvests all necessary energy from the environment, for example using light, temperature differences, vibration, acceleration or radio waves. We consider five different tasks in a node with different criticalities. Task 1 (high criticality) has to regularly read a sensor, for example from an image sensor in a surveillance scenario. The high-critical image processing Task 2 is subject to different execution time bounds obtained with different levels of pessimism. This processing is done locally to reduce the amount of high-power data transmission to the central station. Occasionally, when some important event has been identified, the highly critical Task 3 needs to transfer the result wirelessly to a central host. This communication may need the repetition of radio packets in case of transmission errors and therefore, results in different execution times. A node uses the low criticality Task 4 to periodically switch on its radio, to maintain synchronization with the host, and to exchange status data via the wireless communication medium.

Finally, the low criticality Task 5 regularly checks system health, relays some packets, and manages the node memory.

This paper contributes to consider the aforementioned aspects, as summarized below:

(1) We present an analytical method for calculating the energy content of the super-capacitor and the job success-ratios for a given schedule, both in steady state, while considering the stochastic availability of energy and the physical characteristics of the super-capacitor.
(2) We derive basic theoretical results which quantify the impact of each modification in a given schedule on the job success-ratios.
(3) We propose a mixed-criticality scheduling algorithm for periodic tasks with time and success-ratio constraints, inspired by the theoretical results, to manage the inherent unpredictability of the system. This algorithm is mainly based on an offline phase which determines static start time of individual jobs depending on the scheduler criticality mode.

In summary, the paper considers energy as an additional resource in comparison to the classic mixed-criticality model. The jobs of a task may not execute successfully due to lack in available energy, but they still need to satisfy given success probability constraints. Therefore, we are faced with a complex interdependence between temporal behavior (computation resource) and success behavior (energy resource): The harvested energy follows some stochastic patterns, and the task behavior changes due to different execution times and mode switches in the mixed-criticality scheduler. To the best of our knowledge, the paper describes the first results on mixed-criticality scheduling under energy harvesting constraints. The authors are also unaware of any similar study which is applicable to non-mixed-criticality real-time systems, namely those which need simultaneous consideration of energy uncertainty and success-ratio constraints.

The structure of the paper is as follows. After presenting some related works in Section 2, we give the system model and define the problem in Section 3. Then, we present an analytical method to determine the super-capacitor energy behavior as well as individual job steady-state success-ratios for given schedules in Section 4. Section 5 contains a scheduling algorithm for a restricted system model that guarantees the required mixed-criticality timing behavior and the success-ratio constraints of tasks. This algorithm is provided on the basis of the analytical method proposed in Section 4. The efficacy of the analysis and the algorithm are then extensively studied for a number of system setups in Section 6. Finally, Section 7 concludes the paper.

2. RELATED WORK

As mentioned in the previous section, some challenges of perpetual devices [Brunelli et al. 2009] are related to energy management as many such systems should work in an energy-autonomous manner after deployment.

Some studies have focused on determining size, type and scheduling of batteries [Jongerden et al. 2009; He et al. 2013]; some authors replace the batteries with super-capacitors [Brown 2014; Chai and Zhang 2015; Brunelli et al. 2009]. Battery/super-capacitor scheduling and their joint management in systems which use both storage principles is also considered as a favorite method [Jin et al. 2014; Krishna 2011; Mirhoseini and Koushanfar 2011]. However, to be able to guarantee some performance level of the overall system, energy prediction is needed. A major example is the Lazy Scheduling Algorithm (LSA) [Moser et al. 2007] which works optimally for a real-time system if the energy prediction is accurate. However, uncertainties due to environmental conditions negatively impact the accuracy of predictions, even for resources like solar [Piorno et al. 2010]. Some studies try to circumvent these non-deterministic properties through scavenging energy from multiple resources [Porcarelli et al. 2012] or through improving efficiency via techniques like Maximum-Power-Point Tracking (MPPT) [Brunelli et al. 2009], diminishing the need to have a large energy storage [Weddell et al. 2013].

One further approach is to use load management, i.e., task and energy scheduling, e.g., through voltage and frequency scaling [Wang et al. 2013], to adapt to the uncertainties. Although many systems need guaranteed performance levels, example studies like [Audet et al. 2011] which schedule periodic tasks for uncertain energy arrival do not provide guarantees, rather they only reduce the task energy violation ratio. Similar to [Audet et al. 2011], the authors of [Huang and Neely 2013] present an online algorithm, however within the scope of non-real-time networks. A non-real-time system with probabilistic load and energy arrivals has been considered in [Liu et al. 2015] as well, where the energy deficiency is supplied from a power grid. The target of the paper is to minimize the average delay while satisfying a constraint on the maximum grid power consumption. In [Piorno et al. 2010], an online task scheduler has been proposed for a system with multiple priority queues which works based on some energy predictions. It does not consider time limitation for the tasks, and delays the task execution if there is not enough energy available in the energy harvesting device.

A stochastic view to energy harvesting and event arrival is taken in [Zhang et al. 2013]. The study follows an analytical approach based on a Markov model and it unifies the energy harvesting and event arrival processes to derive the probability of event loss and average delay. There exist additional studies like [Su et al. 2014] which consider the fact that the energy arrival is intermittent and variable. The approach proposes stochastic energy scheduling for operational cost and power loss reductions. From the aspect of scheduling in stochastic conditions, [Li et al. 2014] employs Dynamic Voltage and Frequency Scaling (DVFS) to schedule independent tasks with normally distributed execution times, and with deadline and energy consumption budgets.

The usual pessimistic view on execution times in real-time systems negatively impacts the amount of resources that are necessary to guarantee the temporal behavior. Therefore, an accepted approach is to have a mixed-criticality view on the system specification [Burns and Davis 2015] that allows to certify the high-critical functionalities under conservative assumptions and to improve the utilization of platform resources under less pessimistic assumptions. Extending the mixed-criticality literature to energy-efficient design using DVFS [Huang et al. 2014a], fault-tolerance with respect to transient hardware faults [Huang et al. 2014b], timing analysis using probabilistic techniques and appropriate hardware/software architectures [Davis et al. 2014], reliability guarantee using combinations of fault management techniques in multi-core systems [Kang et al. 2014], and battery-awareness [Wognsen et al. 2014] has been attended. Different but similar views considering load uncertainty have also been given in [Samadi et al. 2013].

In contrast to previous studies, we allow efficient resource utilization through mixed-criticality scheduling in an energy harvesting setup. The availability of energy can stochastically be characterized and lower bounds on task success probabilities in spite of possible energy shortage can be guaranteed. The main challenges of this problem arise from uncertainty in the provided and required energy, as well as physical properties of the energy storage. There are very few investigations that consider the latter problem, especially in the context of real-time systems [Chai and Zhang 2015]. We analytically investigate the interaction between these three challenges by modeling stochastic energy arrival, mixed-criticality task systems, and energy-dependent supercapacitor power dissipation.

3. SYSTEM MODEL AND PROBLEM DEFINITION

We consider a battery-less dual-criticality system supplied by a super-capacitor which is charged through an energy harvester. The system must respect the mixed-criticality temporal requirements on one hand and the success probability requirements on the other.

In spite of studies like [Davis et al. 2014] which focus on specifying timing behaviors of mixed-criticality systems using probabilistic timing analysis, this paper concentrates on job success-ratios if jobs may fail due to energy shortage. In other words, a job not only needs to execute within its deadline regarding the mixed-criticality requirements, but it also should satisfy its energy-related success-ratio constraint. To satisfy this property, we need to setup the system in a manner that it shows some type of predictable behavior from both aspects. Therefore, a job that fails due to lack in energy cannot simply resume its execution, even if technologies like non-volatile memories are used to store its state at the point of failure. The reasons for restricting this are the following: i) Resumption of a job after failure may interfere with the temporal feasibility of other mixed-criticality jobs. ii) Postponing the execution of a job (or some portion of it) to use energy that will be harvested in the future may negatively impact the success-ratio of other jobs due to energy scarcity.

Thus, to guarantee the success-ratio constraint, appropriate task scheduling is needed to control the available energy in the super-capacitor at different moments in time, while considering the stochastic pattern of energy arrival, the super-capacitor power dissipation, and the mixed-criticality task energy usages. As a consequence, our proposed approach is based on an analytic evaluation of the energy availability and its interplay with mixed-criticality scheduling. The corresponding formal system model is given in the following subsections.

3.1. Task Model

We consider a mixed-criticality system with $L = 2$ criticality levels, denoted as LO and HI. In an abstract view, the system consists of m periodic real-time tasks $\tau_i : (l_i, \pi_i, \vec{\epsilon}_i, Pow_i, \alpha_i)$, $1 \leq i \leq m$, where $l_i =$ LO or HI denotes the task criticality, π_i denotes the task period (with implicit relative deadline), $\vec{\epsilon}_i$ is a vector, where $\vec{\epsilon}_i$[LO] and $\vec{\epsilon}_i$[HI] show the task Worst-Case Execution Times (WCETs) at different criticality levels ($\vec{\epsilon}_i$[LO] $= \vec{\epsilon}_i$[HI] for tasks with $l_i = \text{LO}$), and Pow_i is the task-specific power consumption, which is the same at both criticality levels. Also, $0 \leq \alpha_i \leq 1$ is the success-ratio constraint for individual jobs of the task. The hyperperiod is defined as $\Pi = LCM(\pi_i)$, $1 \leq i \leq m$. The task scheduling is repeated at the boundaries of Π and $\tau_{i,k}$ is the k^{th} instance of τ_i during Π.

We follow the conventional mixed-criticality scheduling techniques that provide dynamic guarantees to all tasks, and which can be specified by a simple mode switch protocol: i) The system starts in the low criticality mode, where all jobs of tasks τ_i are guaranteed to meet their deadlines if they do not exceed their low criticality execution time bound $\vec{\epsilon}_i$[LO]. ii) Whenever a high criticality job exceeds its low criticality execution time bound, then the scheduler transits immediately to the high mode. Hereafter, jobs of tasks τ_i with criticality $l_i = \text{LO}$ may be dropped or their services may be degraded in order to protect the timeliness of all other jobs of tasks τ_i with criticality $l_i = \text{HI}$. For the sake of simplicity, we suppose that all LO jobs are dropped after such a low-to-high scheduler mode change, while more general cases could also be discussed.

Now, the task success-ratio constraint α_i can be interpreted as follows: For each individual job of a HI task τ_i, the probability that it is interrupted due to energy shortage is at most $1 - \alpha_i$, i.e., with probability at least α_i there is enough energy available to successfully execute that job. The probability that each individual job of a LO task

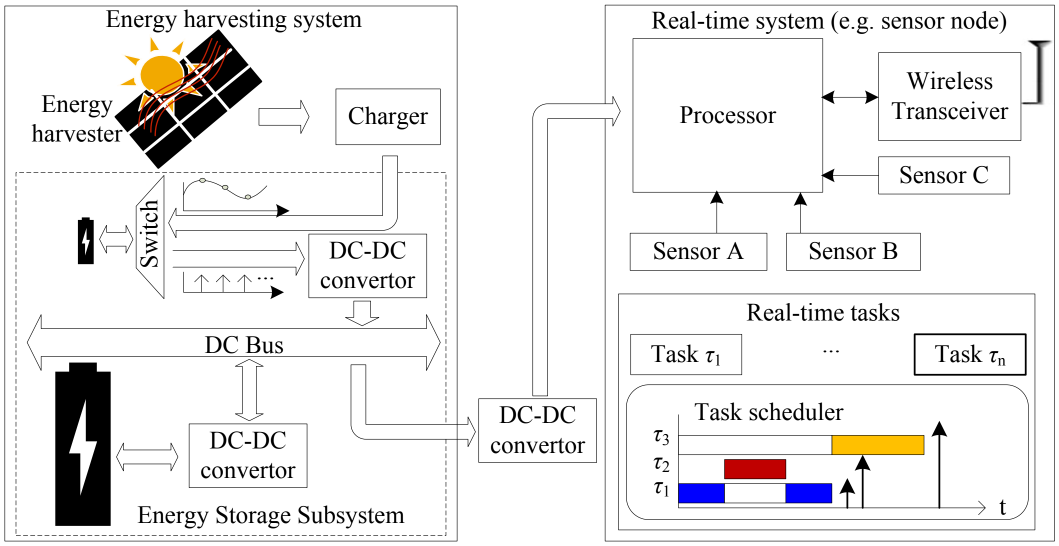

Fig. 1. A schematic view of the system architecture.

τ_i successfully finishes its execution is at least α_i as well, but only provided that the scheduler is in the low mode, i.e., provided that the job is not executed in degraded mode.

3.2. Energy Supplier Model

The energy supplier unit consists of an energy harvester and a corresponding super-capacitor. In an abstract view, we consider that the continuously arriving energy is buffered into an intermediate reservoir, and it is added to the super-capacitor at regular intervals called epochs. This is a common design for an energy harvesting device, where the energy in the intermediate reservoir, for example a capacitor, is converted using a voltage regulator, and then transferred to the super-capacitor and used by the energy consumer (see [Whitaker 2010; Hao and Garcia 2014]). Depending on the source of energy and characteristics of the energy harvester, the capacity of reservoir and the epoch length P are determined. A smaller epoch length better approximates the continuous energy arrival and provides the chance of earlier use of the harvested energy; a larger P, however, reduces the computational complexity of our analysis method (see Section 5.4). We consider that P divides the hyperperiod Π. A schematic view to the system can be seen in Fig. 1.

We can obtain an energy Probability Density Function (PDF) H for the energy harvested during an epoch. $H(e)$, which can be obtained through profiling, describes the relative likelihood for the energy harvested within an epoch of length P to take on the value e (including the dissipated power of the reservoir and the energy transfer overhead), similar to what is obtained for energy resources like solar [Jin et al. 2014], wind [WindPower Program 2015], piezoelectric [Abdeddaïm et al. 2014], and radio frequency [Nishimoto et al. 2010]. For resources like solar which may have noticeable changes at different hours a day, H can change at different epochs/dozens of epochs, e.g., it can be different for every hour during the 24 hours of the day. This property is not in the focus of this paper; it does not change the principles of the forthcoming analytical solution, but makes its presentation more complex and increases the notational overhead.

The super-capacitor, as another part of the energy supplier, is defined by a tuple $(E_{max}, C, E_{CO}, Dis)$, where E_{max} is its maximum stored energy, C its capacity, E_{CO} is the cutoff energy (the minimum stored energy with which the system can still operate), and Dis specifies the dissipated power. Dis consists of super-capacitor self-discharge which occur during frequent charging-discharging cycles of the super-capacitor [Chai and Zhang 2015].

The Energy Iteration Equation model proposes a piecewise linear approximation of the dissipated power Dis [Yang and Zhang 2013; Zhu et al. 2009]. Let $V(t)$ and $E(t)$ denote the super-capacitor voltage and the respective energy at time t. Then, for an interval during which task τ_i is executing with power Pow_i, we have:

$$E(t) = \frac{1}{2}CV^2(t) \tag{1}$$

$$\frac{dE(t)}{dt} = -Pow_i - Dis(E(t)) \tag{2}$$

where the dissipated power $Dis(E(t))$ is defined as follows [Zhu et al. 2009]:

$$Dis(E(t)) = \begin{cases} a_1E(t) + b_1 & E_{R_1} \le E(t) \le E_{R_2} \\ a_2E(t) + b_2 & E_{R_2} < E(t) \le E_{R_3} \\ \cdots & \cdots \\ a_nE(t) + b_n, & E_{R_n} < E(t) \le E_{R_{n+1}} \end{cases} \tag{3}$$

Here $E_{R_1} = 0$, E_{R_2} denotes the cut-off energy with $E_{R_2} = E_{CO}$, and $E_{R_{n+1}} = E_{max}$. We denote the k^{th} interval constraint in $Dis(E(t))$ as

$$Dis_k(E(t)) \equiv E_{R_k} < E(t) \le E_{R_{k+1}}$$

for $1 \le k \le n$. For example, suppose that there is no energy arrival for some time interval starting at t_0 and that the energy $E(t)$ satisfies $Dis_k(E(t))$ for that interval; then the dissipated power is defined by a single linear function $Dis(E(t)) = a_kE(t)+b_k$. As a result, for $t > t_0$ we obtain:

$$E(t) = -A_k(Pow) + (E(t_0) + A_k(Pow))e^{-a_k(t-t_0)} \tag{4}$$

where $A_k(Pow) = \frac{Pow+b_k}{a_k}$ and Pow is the power consumption within that time interval. Accordingly, the Time-To-Constraint-Change (TTCC) starting at time t_0, i.e., the time it takes that the active constraint changes from Dis_k (because $E_{R_k} < E(t) \le E_{R_{k+1}}$) to Dis_{k-1} will be

$$TTCC(E(t_0)) = -\frac{1}{a_k} \ln \frac{E_{R_k} + A_k(Pow)}{E(t_0) + A_k(Pow)} \tag{5}$$

3.3. Problem Definition and the Solution Approach

Considering the mentioned system model and regarding the details of the energy harvester, the super-capacitor, and the tasks, the goal of this paper is broken into solving three problems in the following order.

PROBLEM 1. *While considering the uncertainty in the energy arrival and the physical characteristics of the energy storage, determine the steady-state success ratio of jobs for a given schedule. Given a schedule, we formally define the success-ratio of job $\tau_{i,k}$ as*

$$R_{\tau_{i,k}} \equiv \textit{probability that } \forall t \textit{ at which } \tau_{i,k} \textit{ is scheduled: } E(t) \ge E_{CO} \tag{6}$$

where $E(t)$ and E_{CO} are the super-capacitor energy at time t and the cut-off energy, respectively. □

PROBLEM 2. *Determine a time- and energy-feasible schedule in a standard (i.e., a non-mixed-criticality) setting. The steady state success ratio of each job $\tau_{i,k}$ is required to be greater than or equal to the threshold α_i.* □

Before giving the third problem, we provide the following definition to formally state our notion of feasibility in a mixed-criticality setting:

Definition 3.1 (*Feasible Schedule*). A mixed-criticality schedule for a task set is called time-energy feasible, or simply feasible, if the following two conditions are satisfied:

(1) $\forall i \in [1, m]$ with $l_i = \mathrm{LO}$, $\forall k \in [1, \frac{\Pi}{\pi_i}]$: When the system operates in low criticality mode for enough time that it exhibits its steady-state behavior, then the steady-state success-ratios satisfy $R_{\tau_{i,k}} \geq \alpha_i$ and all jobs $\tau_{i,k}$ meet their real-time constraints.
(2) $\forall i \in [1, m]$ with $l_i = \mathrm{HI}$, $\forall k \in [1, \frac{\Pi}{\pi_i}]$: The steady-state success-ratios satisfy $R_{\tau_{i,k}} \geq \alpha_i$ and all jobs $\tau_{i,k}$ meet their real-time constraints. □

In other words, while the timing behavior of tasks should respect the constraints of a standard mixed-criticality model, from the energy viewpoint, the jobs of low criticality need to satisfy their success-ratio constraints only if the system is in low criticality mode whereas jobs of high-criticality tasks need to satisfy them always.

PROBLEM 3. *According to Definition 3.1 which states conditions for feasibility, even with multiple scheduler mode switches between high criticality and low criticality modes, the steady-state success-ratio of HI jobs must be protected and the corresponding constraint must not be violated. Thus, the last problem that we consider is to determine a feasible schedule in a mixed-criticality setting.* □

For the first problem, we propose an analytical method to obtain the PDF of the super-capacitor energy content at different time instants. According to the analysis, the calculation of the job success-ratio (defined in (6)) and other important quantities is presented.

Based on this analytical method, we provide an algorithm for Problem 2 that allows us to schedule a task set such that both, temporal and steady-state success-ratio constraints are satisfied for a non-mixed-criticality setup, e.g., considering only HI-criticality tasks.

Finally, the third problem, namely feasible scheduling of the mixed-criticality system is addressed. The proposed method is based on the solution to Problem 2 and the constraint that job start times remain unchanged either i) when all jobs exhibit their low criticality behavior or ii) when the system scheduler is in the high criticality mode as at least one HI-criticality task instance has exceeded its low execution time bound. This algorithm is run offline to determine static start times for LO and HI jobs at different scheduler criticality modes. The above mentioned analytical method is used to determine the super-capacitor energy probability distribution at different moments in time, while considering the stochastic energy arrival and the possibility of a sudden scheduler mode switch. In fact, depending on different time instants when the scheduler mode switch may happen, the scheduler reacts in an online manner such that the feasibility of schedule based on Definition 3.1 remains valid. Moreover, we specify the conditions under which the system can safely transit back to the scheduler low mode.

4. ENERGY ANALYSIS

In this section, we consider a system with an architecture as shown in Fig. 1 and some fixed static schedule for the tasks. We propose an analytical method to calculate the super-capacitor energy availability at different time instants. In order to provide guarantees independent of the run-time behavior of jobs, we suppose that execution times equal the WCETs.

At first, we introduce the notion of state for our stochastic analysis. Next, the state energy and the steady-state analysis are described. Finally, we discuss the calculation of the job success-ratio as well as the wasted energy.

4.1. State Space Representation

In the following, we discuss the notion of state for a given static schedule. Such a schedule has a number of scheduling points throughout the hyperperiod Π, determined by the start and end points of jobs (or job segments resulted from preemptive scheduling) on the timeline. Considering the scheduling points and the epoch starting points, the timeline is divided into some intervals, where each is called a *Power Usage Monotonicity Interval* (*PUMI*); a ***PUMI*** is defined as an interval during which the system power requirement does not change and the super-capacitor is not charged. Remember that only at the starting times of an epoch the super-capacitor is charged from the intermediate reservoir, and that the epoch length P divides the hyperperiod Π.

We represent the hyperperiod as a sequence of intervals $PUMI_1$,...,$PUMI_k$, where $PUMI_i = [PUMI_i^s, PUMI_i^e]$, i.e. it starts at $PUMI_i^s$ and ends at $PUMI_i^e$. There exist n constraints for the power dissipation Dis for different levels of the super-capacitor energy content, indexed as Dis_1,..., Dis_n, see also (3).

Due to the energy-dependent dissipated power according to (3), we are faced with a piecewise linear energy loss as well as a constant energy usage within a ***PUMI***. As a result, to calculate the energy stored in the super-capacitor during such an interval, we need two types of information: the current ***PUMI*** with its scheduled task or idle power consumption, and the sequence of active interval constraints Dis in (3). Therefore, we define the notion of a state as a tuple $s_{i,j} = (Dis_i, PUMI_j)$ for $1 \leq i \leq n$, $1 \leq j \leq k$, where Dis_i denotes one of the interval constraints in (3) and $PUMI_j$ denotes one of the power usage monotonicity intervals within the hyperperiod Π.

Transitions between states can be caused by a change in the energy dissipation relation according to (3) or by moving from one ***PUMI*** to the next. More formally, two types of events can result in a transition from $s_{i,j}$ to some other state:

— The active interval constraint Dis_i in (3) changes which leads to a transition to state $s_{i-1,j}$, $1 < i \leq n$.
— The current time reaches the end of the monotonicity interval $PUMI_j^e$. If the end of the interval does not coincide with the end of an epoch, then we observe a transition to the adjacent state $s_{i,(j \mod k)+1}$. Otherwise, the energy that was harvested during the epoch is added to the super-capacitor and the transition from $s_{i,j}$ is not necessarily to an adjacent state, i.e., the destination state can be $s_{i',(j \mod k)+1}$ where $i \leq i' \leq n$. The corresponding active dissipation interval constraint is $Dis_{i'}$ according to (3).

An Illustrative Example: A sample task set is provided in Table I. Fig. 2 shows a corresponding example schedule. The epoch length is $P = 2$. Fig. 3 shows the partial state space of this schedule for the time interval $t \in [0, 4]$ between the start of $PUMI_1$ and the end of $PUMI_3$ for a dissipated power Dis with three interval constraints, namely Dis_1, Dis_2 and Dis_3. Remember that Dis_1 denotes the constraint where the system is below the cutoff energy. Therefore, if the system executes a job and enters a state that is associated to Dis_1, the system stops the job execution and the job execution fails. The system may enter Dis_1 due to either job energy consumption or super-capacitor

Table I. An example to introduce *PUMI* and state.

Task	l_i	π_i	$(\vec{\epsilon}_i[\text{LO}], \vec{\epsilon}_i[\text{HI}])$	α_i	Pow_i
τ_1	-	4	(1, 1)	-	-
τ_2	-	6	(3, 3)	-	-

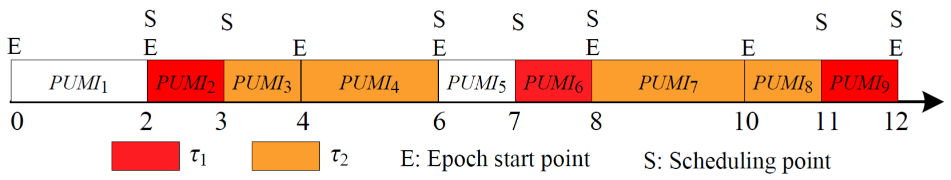

Fig. 2. A schedule for task set of Table I, and the corresponding *PUMI*s.

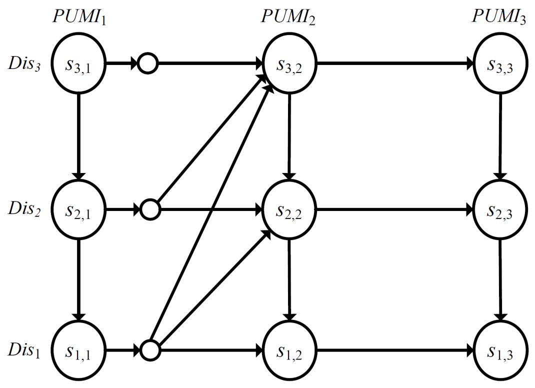

Fig. 3. Partial state space corresponding to interval $[0, 4]$ of Fig. 2.

power dissipation. It will remain in Dis_1 at least until the start of the next epoch at which the super-capacitor is charged.

4.2. State-Based Energy Analysis

In each state, the super-capacitor is discharged due to task power consumption (or idle power) and power dissipation according to (3). On the other hand, at the beginning of an epoch the super-capacitor is charged by a probabilistic amount according to the energy probability density function H. We are interested to know the PDF of the available energy in the super-capacitor at different time instants. This result will be utilized later on for the calculation of job success-ratio and the wasted energy. When talking about the stored energy, we always refer to the available energy in the super-capacitor.

We define the following random variables:

$$Z^{in}_{i,j} \equiv \text{Time of entering state } s_{i,j}, \tag{7}$$

$$E^{in}_{i,j} \equiv \text{The stored energy when entering } s_{i,j}. \tag{8}$$

The entering time to a state and energy level of the super-capacitor upon entering that state are related to the time and energy level when the system leaves the previous state. These quantities are defined as follows:

$$Z^{out}_{i,j} \equiv \text{Time of leaving state} s_{i,j}, \tag{9}$$

$$E^{out}_{i,j} \equiv \text{The stored energy when leaving} s_{i,j}. \tag{10}$$

As the first step in our energy analysis, we are interested in calculating the joint PDF of $Z^{out}_{i,j}$ and $E^{out}_{i,j}$ for an arbitrary state $s_{i,j}$ as a function of the joint PDF of $Z^{in}_{i,j}$ and $E^{in}_{i,j}$. To this end, we first specify $Z^{out}_{i,j}$ and $E^{out}_{i,j}$ as functions of $Z^{in}_{i,j}$ and $E^{in}_{i,j}$:

$$Z^{out}_{i,j} = \begin{cases} Z^{in}_{i,j} + TTCC(E^{in}_{i,j}) & \text{if } Z^{in}_{i,j} + TTCC(E^{in}_{i,j}) < PUMI^{e}_{j} \\ PUMI^{e}_{j} & \text{otherwise} \end{cases} \tag{11}$$

$$E^{out}_{i,j} = \begin{cases} E_{R_i} & \text{if } Z^{in}_{i,j} + TTCC(E^{in}_{i,j}) < PUMI^{e}_{j} \\ E(PUMI^{e}_{j}) & \text{otherwise} \end{cases} \tag{12}$$

where E_{R_i} denotes the lower bound of the relevant interval constraint Dis_i in (3), $TTCC$ is the time to constraint change provided in (5), and the stored energy $E(PUMI^e_j) = -A_i(Pow_j) + (E^{in}_{i,j} + A_i(Pow_j))e^{-a_i(PUMI^e_j - Z^{in}_{i,j})}$ is determined by (4) using the task power consumption Pow_j within $PUMI_j$.

These relations are based on the fact that there are two reasons for a state transition: In the first criteria of (11) and (12), the transition happens due to a change of the active interval constraint to Dis_{i-1}, and in the second criteria due to moving to the next power usage monotonicity interval $PUMI_{j+1}$. In the former case, the value of $E^{out}_{i,j}$ is fixed and known, i.e. it is equal to E_{R_i}, while in the latter case the time of transition $Z^{out}_{i,j}$ is fixed to $PUMI^e_j$.

To calculate the joint PDF $f_{Z^{out}_{i,j},E^{out}_{i,j}}(z^{out}_{i,j}, e^{out}_{i,j})$, we employ the well known method of calculating a joint PDF of functions of two random variables. In our case, these functions (g and h in Lemma 4.1 below) are actually those defined in (11) and (12).

LEMMA 4.1 (JOINT PDF OF FUNCTIONS OF TWO RANDOM VARIABLES). *Suppose X and Y are random variables with a joint PDF of $f_{X,Y}(x,y)$. Moreover, Z and W are random variables which are functions of X and Y, i.e.,*

$$Z = g(X,Y) \tag{13}$$

$$W = h(X,Y) \tag{14}$$

Then, Z and W are random variables with a joint PDF

$$f_{Z,W}(z,w) = \frac{f_{X,Y}(x_1,y_1)}{|J(x_1,y_1)|} + \cdots + \frac{f_{X,Y}(x_n,y_n)}{|J(x_n,y_n)|} \tag{15}$$

in which x_1,...,x_n,y_1, ...,y_n are the roots of equations (13) and (14), and $J(x,y)$ is the Jacobian matrix, calculated as

$$J(x,y) = \begin{vmatrix} \frac{\partial Z}{\partial X} & \frac{\partial Z}{\partial Y} \\ \frac{\partial W}{\partial X} & \frac{\partial W}{\partial Y} \end{vmatrix}_{X=x,Y=y} \tag{16}$$

PROOF. See [Papoulis 1990]. □

We will treat the two cases as shown in (11) and (12) separately and start with the first one. If $E_{i,j}^{out} = E_{R_i}$, then $Z_{i,j}^{out} = Z_{i,j}^{in} + TTCC(E_{i,j}^{in})$, and thus, we have $f_{Z_{i,j}^{out},E_{i,j}^{out}}(z_{i,j}^{out}, e_{i,j}^{out}) = f_{Z_{i,j}^{out},E_{i,j}^{out}}(z_{i,j}^{out}, E_{R_i})$. Note that as $E_{i,j}^{out}$ is constant in this case, the joint PDF has only one free variable $Z_{i,j}^{out}$, while this variable is a function of two random variables (see (11)). In order to use Lemma 4.1, we add the missing random variable $E_{i,j}^{in}$ to this function which results in $f_{Z_{i,j}^{out},E_{i,j}^{out},E_{i,j}^{in}}(z_{i,j}^{out}, E_{R_i}, e_{i,j}^{in})$. As E_{R_i} has a fixed constant value, this new function depends on two random variables now. Therefore, based on Lemma 4.1, we find

$$f_{Z_{i,j}^{out},E_{i,j}^{out},E_{i,j}^{in}}(z_{i,j}^{out}, E_{R_i}, e_{i,j}^{in}) = \frac{f_{Z_{i,j}^{in},E_{i,j}^{in}}(z_{i,j}^{in}, e_{i,j}^{in})}{|J|} \tag{17}$$

where

$$J = \begin{vmatrix} \frac{\partial Z_{i,j}^{out}}{\partial Z_{i,j}^{in}} & \frac{\partial Z_{i,j}^{out}}{\partial E_{i,j}^{in}} \\ \frac{\partial E_{i,j}^{in}}{\partial Z_{i,j}^{in}} & \frac{\partial E_{i,j}^{in}}{\partial E_{i,j}^{in}} \end{vmatrix} = \begin{vmatrix} 1 & \cdots \\ 0 & 1 \end{vmatrix}. \tag{18}$$

In order to calculate $f_{Z_{i,j}^{out},E_{i,j}^{out}}(z_{i,j}^{out}, E_{R_i})$, we integrate (17) over the augmented variable $E_{i,j}^{in}$ and consider that $|J| = 1$. We obtain

$$f_{Z_{i,j}^{out},E_{i,j}^{out}}(z_{i,j}^{out}, E_{R_i}) = \int_{e_{i,j}^{in}} f_{Z_{i,j}^{in},E_{i,j}^{in}}(z_{i,j}^{out} - TTCC(e_{i,j}^{in}), e_{i,j}^{in})\, de_{i,j}^{in}. \tag{19}$$

Now, we study the other case in (11) and (12), i.e., we have $Z_{i,j}^{out} = PUMI_j^e$. Using (4) we find

$$E_{i,j}^{out} = -A_i(Pow_j) + (E_{i,j}^{in} + A_i(Pow_j))e^{-a_i(PUMI_j^e - Z_{i,j}^{in})} \tag{20}$$

Then, similar to the previous case, $f_{Z_{i,j}^{out},E_{i,j}^{out}}(z_{i,j}^{out}, e_{i,j}^{out})$ is augmented with the missing random variable $Z_{i,j}^{in}$ and we obtain

$$f_{Z_{i,j}^{out},E_{i,j}^{out},Z_{i,j}^{in}}(PUMI_j^e, e_{i,j}^{out}, z_{i,j}^{in}) = \frac{f_{Z_{i,j}^{in},E_{i,j}^{in}}(z_{i,j}^{in}, e_{i,j}^{in})}{|J|} \tag{21}$$

where

$$J = \begin{vmatrix} \frac{\partial E_{i,j}^{out}}{\partial Z_{i,j}^{in}} & \frac{\partial E_{i,j}^{out}}{\partial E_{i,j}^{in}} \\ \frac{\partial Z_{i,j}^{in}}{\partial Z_{i,j}^{in}} & \frac{\partial Z_{i,j}^{in}}{\partial E_{i,j}^{in}} \end{vmatrix} = \begin{vmatrix} \cdots & e^{-a_i(PUMI_j^e - z_{i,j}^{in})} \\ 1 & 0 \end{vmatrix} \tag{22}$$

As a result, we can calculate $f_{Z_{i,j}^{out},E_{i,j}^{out}}(PUMI_j^e, e_{i,j}^{out})$ as

$$f_{Z_{i,j}^{out},E_{i,j}^{out}}(PUMI_j^e, e_{i,j}^{out}) = \int_{z_{i,j}^{in}} \frac{f_{Z_{i,j}^{in},E_{i,j}^{in}}(z_{i,j}^{in}, (e_{i,j}^{out} + A_i(Pow_j))e^{a_i(PUMI_j^e - Z_{i,j}^{in})} - A_i(Pow_j))}{|J|}\, dz_{i,j}^{in} \tag{23}$$

where $|J| = e^{-a_i(PUMI_j^e - z_{i,j}^{in})}$.

With (19) and (23), we have determined the joint PDF of $Z_{i,j}^{out}$ and $E_{i,j}^{out}$ for an arbitrary state $s_{i,j}$ as a function of the joint PDF of $Z_{i,j}^{in}$ and $E_{i,j}^{in}$. As the next step, we want to calculate the joint PDF of $Z_{i,j}^{in}$ and $E_{i,j}^{in}$, i.e., $f_{Z_{i,j}^{in},E_{i,j}^{in}}(z_{i,j}^{in}, e_{i,j}^{in})$.

To obtain the joint PDF $f_{Z_{i,j}^{in},E_{i,j}^{in}}(z_{i,j}^{in},e_{i,j}^{in})$ for state $s_{i,j}$, we consider all possible transitions from predecessor states $s_{i',j'}$ to $s_{i,j}$ and add the respective joint PDFs $f_{Z_{i',j'}^{out},E_{i',j'}^{out}}(z_{i',j'}^{out},E_{R_{i'}})$ or $f_{Z_{i',j'}^{out},E_{i',j'}^{out}}(PUMI_{j'}^{e},e_{i',j'}^{out})$, each one multiplied by the corresponding probability of the predecessor state $Prob(s_{i',j'})$. Finally, each resulting joint PDF is normalized such that its integral equals 1.

What is still missing is the probability to be in some state $s_{i,j}$, i.e., $Prob(s_{i,j})$. To this end, we determine the probability that a state is passed during a hyperperiod, which in turn depends on the transition probabilities. Remember that there are two reasons for a transition from $s_{i,j}$, i.e., the super-capacitor energy reaches E_{R_i} or the time reaches $PUMI_j^e$. We need the probability of each type of these transitions for each state. The probability of the former, i.e., the probability of a constraint change $Prob(CC_{i,j})$, can be obtained by integrating $f_{Z_{i,j}^{out},E_{i,j}^{out}}(z_{i,j}^{out},E_{R_i})$ as follows:

$$Prob(CC_{i,j}) = \int_{z_{i,j}^{out}} f_{Z_{i,j}^{out},E_{i,j}^{out}}(z_{i,j}^{out},E_{R_i})\, dz_{i,j}^{out}. \tag{24}$$

Accordingly, the probability of the latter transition can be obtained as $1 - Prob(CC_{i,j})$.

Using these results, the state probability can be calculated by summing the probabilities of all the transitions leading to that state. More precisely, if the state probability of $s_{i,j}$ is denoted as $Prob(s_{i,j})$, and the transition probability from a predecessor state $s_{i',j'}$ to $s_{i,j}$ is denoted as $Prob(s_{i',j'} \to s_{i,j})$, then we find:

$$Prob(s_{i,j}) = \sum_{s_{i',j'} \in Pred(s_{i,j})} Prob(s_{i',j'}) Prob(s_{i',j'} \to s_{i,j}), \tag{25}$$

where $Pred(s_{i,j})$ is the set of predecessor states of $s_{i,j}$.

Finally, we would like to shortly describe the special case when the start of the interval $PUMI_j^s$ coincides with the start of an epoch, and we want to determine $f_{Z_{i,j}^{in},E_{i,j}^{in}}(z_{i,j}^{in},e_{i,j}^{in})$. In this case, for all predecessor states which are left due to the change of the $PUMI$, we first add the normalized version of PDFs $f_{Z_{i',j-1}^{out},E_{i',j-1}^{out}}(PUMI_{j-1}^{e},e_{i',j-1}^{out})$ for $1 \leq i' \leq n$ while considering the state probabilities. Afterwards, we convolve the obtained combined PDF with the energy probability density function H, i.e., with the harvested energy at the start of $PUMI_j$. Here we consider the maximum energy of the super-capacitor by collecting the density of energies higher than E_{max} as a weighted Dirac delta function on point E_{max}. Then, the obtained PDF is partitioned according to the energy interval constraint related to each state, see (3). Finally, for time $PUMI_j^s$ it is specified with what probability each state will be entered. The remaining calculations are straightforward.

4.3. Steady-State Analysis

We described in the previous subsection how the joint PDF of exiting a state can be determined, given the joint PDF of entering that state. This provides a simple approach for the transient analysis of the available energy in the super-capacitor. To this end, suppose an initial energy in the super-capacitor. One can now start with the first states of the hyperperiod, for which the entering time is known to be $t = 0$, and the entering energy is determined according to the harvested energy PDF H while considering the initial energy in the super-capacitor. Then, by performing the analysis for all subsequent states by the method presented above, the PDFs for the states of the first hyperperiod can be determined. Continuing these steps for subsequent hyperperiods until a fixed behavior is arrived yields the required steady-state analysis.

4.4. Calculation of Success Ratios

In this section, we calculate the steady-state success-ratio R_J of some job J as introduced in Section 3 and (6). In other words, we are interested in finding the probabiliy that enough energy is available in the super-capacitor to successfully execute the job J, i.e., the probability that the super-capacitor energy will not drop below the cutoff energy E_{CO} while executing that job. Two cases are considered for the job success-ratio calculation: i) the job execution falls within one *PUMI*, namely it is a single-segment job, ii) the job execution spans several *PUMI*s, i.e., it is multi-segment.

Single-segment jobs: In this case, the successful execution of job J which executes only in $PUMI_j$ requires that the state $s_{1,j}$ is not entered. In the previous section, we determined the steady state probability of $s_{i,j}$ denoted as $Prob(s_{i,j})$. Using the definition of success-ratio of a job according to (6), we can simply derive:

$$R_J = 1 - Prob(s_{1,j}). \tag{26}$$

Multi-segment jobs: In this case, the execution of a job is distributed over two or more *PUMI*s. The reason can be twofold: i) An epoch boundary happens between the start and finish time of a job and thus, the job executes beyond one *PUMI*, or ii) a job is preempted at the end of a *PUMI* and continues its execution at the beginning of some other *PUMI*. Then, in a probabilistic view, the event of successful execution of the job, i.e., the job success-ratio, is determined as the intersection of the probabilistic events of successful execution of all the respective segments. Obviously, if a job segment faced with lack of energy, it will be meaningless to execute the further segments of that job; this event impacts the super-capacitor energy availability. Therefore, in case of multi-segment jobs, the state-based analysis as described in Sections 4.2 and 4.3 needs to be based on an extended state-space.

In particular, suppose that a job is partitioned into a sequence of *PUMI*s and j is one of them, but not the last one. Then, outgoing transitions of the failure state $s_{1,j}$ enter a modified copy of the states graph corresponding to subsequent *PUMI*s. In this copy, the remaining segments of the multi-segment job are not executed. Therefore, there is no energy consumption in the subsequent *PUMI*s of the multi-segment job. The original state graph and its modified copy merge again after the last *PUMI* that contains a segment of the job. As an example, let us assume the two-segment job that spans $PUMI_3$ and $PUMI_4$ in Fig. 2. The resulting partial state graph for the interval $[2, 7]$ is depicted in Fig. 4. In case of several multi-segment jobs, the above construction is applied recursively.

To calculate the job success-ratio, we first need to calculate the steady-state probabilities $Prob(s_{i,j})$ using the state-based analysis as described in Sections 4.2 and 4.3 for the extended state graph. Before describing the general case, let us determine the success ratio of job J for the example shown in Fig. 4, which can be calculated as $R_J = 1 - (Prob(s_{1,3}) + Prob(s_{1,4}))$. This relation is due to the fact that the job fails if the execution passes state $s_{1,3}$ or state $s_{1,4}$, and the corresponding events are mutually exclusive due to the construction of the state graph.

In the general case, let us suppose that a job J is executed in $PUMI_{j_1}$,..., $PUMI_{j_n}$. Then the success ratio of a job J satisfies $R_J = 1 - F_J$, where F_J denotes the failure probability. F_J is given by the sum of steady state probabilites $Prob(s)$ for all states s that are determined using the following rules: (1) Include state s_{1,j_1} in an initially empty set F. (2) Repeat the following rule for all $2 \leq k \leq n$: Include state s_{1,j_k} in F, that s_{1,j_k} is not reachable from states already included in set F.

Due to the 2^{nd} rule, all considered state events are mutually exclusive. Moreover, the fact that we consider all mutually exclusive failure states guarantees that all failure events and the corresponding probabilities are taken into account.

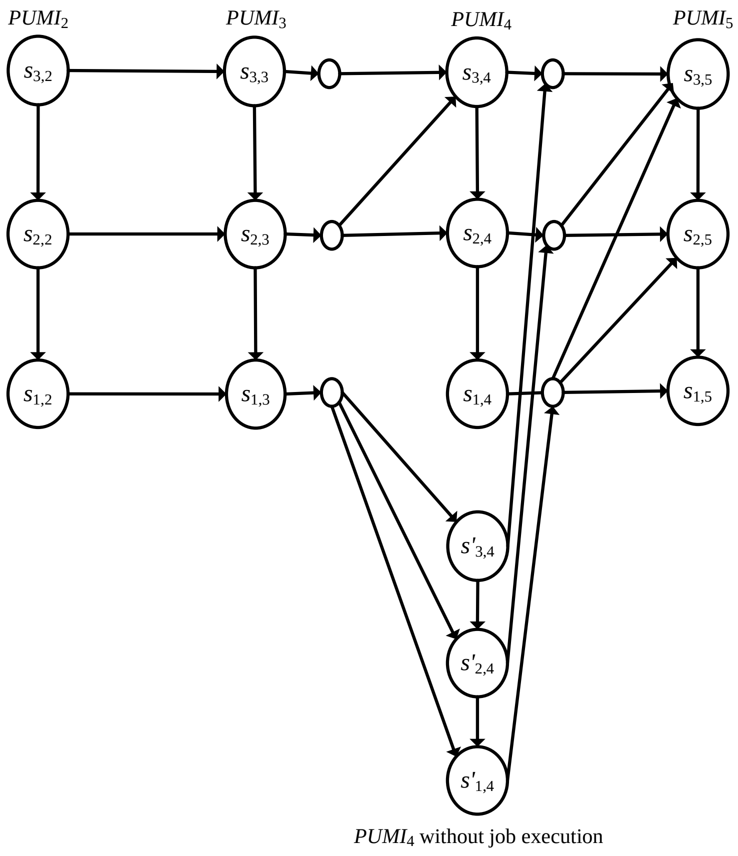

Fig. 4. Partial state space corresponding to interval $[2, 7]$ of Fig. 2 with the two-segment job spanning $PUMI_3$ and $PUMI_4$.

4.5. Calculation of Wasted Energy

An important measure that describes the efficiency of a scheduling policy as well as the dimensioning of the overall system is the wasted energy, which is defined as the amount of energy in one hyperperiod Π that arrives at start of epochs but cannot be stored due to a full super-capacitor.

We are interested to find the PDF of the wasted energy. Suppose an interval $PUMI_j$ whose finishing time ($PUMI_j^e$) coincides with the start of $epoch_k$. Then we find as the PDF of the wasted energy at this time the following expression

$$WE_k(e) = \sum_{i=1}^{n} Prob(s_{i,j}) \cdot \max\left(H(e) * f_{Z_{i,j}^{out}, E_{i,j}^{out}}(PUMI_j^e, e) - E_{max}, 0\right) \tag{27}$$

where n is the number of interval constraints. The PDF of the total wasted energy in one hyperperiod can now be determined by means of convolution

$$WE = WE_1 * WE_2 * ... * WE_{\Pi/P} \tag{28}$$

5. THE SCHEDULING ALGORITHM

In this section, we will present a heuristic scheduling algorithm for the introduced mixed-criticality system in the presence of energy uncertainty. The super-capacitors are supposed to have no power dissipation. We propose a static scheduling algorithm with preemption of jobs, i.e., jobs can be executed as a sequence of segments. The starting times of all segments are determined at compile-time such that the timing as well as the success-ratio constraints in both scheduler modes are satisfied.

— The system starts operation in low mode, where all job segments are started following the computed schedule.
— If a HI job executes longer than its LO execution time, then the scheduler switches to high mode. Job segments of LO tasks are not executed anymore.
— If a certain condition is satisfied, the scheduler switches to its low mode and jobs of LO tasks are being executed again.

At first, we will derive some general properties of the system we are looking at. They form the basis for the construction of a static schedule where feasibility is determined by the analytical method presented in Section 4. Details of the algorithm are then discussed in the following order: i) Scheduling of HI jobs in the high scheduler mode. This can also be considered as a solution to non-mixed-criticality systems, namely Problem 2 of Section 3.3, and ii) accommodating the jobs of LO tasks in appropriate places on the timeline for the low mode. In this way a solution to Problem 3 of Section 3.3 is provided as well.

For the sake of simplicity, we first present properties and an algorithm for single-segment jobs, and then we discuss its extension to the multi-segment case.

5.1. Basic Definitions and Properties

In this section, we restrict ourselves to scheduling of single-segment (non-preemptive) jobs [Jeffay et al. 1991]. We will generalize the discussions to multi-segment jobs later. A scheduling method that guarantees real-time and success-ratio constraints in a mixed-criticality system with energy uncertainty has not been studied previously. As the problem is intractable in general, we will present a heuristic approach which is based on a set of non-trivial results that are described next.

Definition 5.1 (*Safe Points for Scheduler Mode Change*). The proposed scheduling algorithm is expected to work such that, for a feasible schedule, it has the following safe points for a scheduler mode change:

— The scheduler can safely transit from its low mode to its high mode whenever a HI job violates its LO execution-time bound.
— The scheduler can safely transit from its high mode to its low mode whenever (i) a HI job completes by respecting its LO execution-time bound at some time t and (ii) the super-capacitor energy content $E(t)$ is equal to or greater than the maximum super-capacitor energy at t as determined by the steady-state analysis of the system in the low scheduler mode. □

According to Definition 5.1, it must be guaranteed that there is no success-ratio problem for the HI jobs in case of a low-to-high scheduler mode switch, i.e., the mode switch should be safe. Thus, when determining the starting time of LO jobs, one needs to make sure that even in case of a low-to-high scheduler mode switch, the energy PDFs at the start of HI jobs guarantee the success-ratio constraints for all HI tasks.

Given some specific time-feasible schedule, one can use the steady-state analysis of Section 4 to determine the job success-ratios and compare them to the constraints. But the use of a brute-force method to determine a suitable static schedule might be impossible due to combinatorial explosion. Therefore, we follow the strategy of iterative improvement: Starting from an initial solution we add and/or remove jobs, we change the order of jobs and we shift the starting times of jobs. The following properties provide the necessary background for such an efficient search heuristics.

Monotonicity in energy distribution and power consumption:. At first, we define a partial order of energy PDFs that allows us to compare two energy PDFs at the start of a *PUMI*. This way we can determine whether one or the other is better in terms of success-ratios of subsequent jobs.

LEMMA 5.2. *Let us consider the starting time $PUMI_j^s$ of $PUMI_j$ with two corresponding super-capacitor energy PDFs*

$$f_E(e) = \sum_{i=1}^{n} Prob(s_{i,j}) \cdot f_{Z_{i,j}^{in}, E_{i,j}^{in}}(PUMI_j^s, e)$$
$$f'_E(e) = \sum_{i=1}^{n} Prob(s_{i,j}) \cdot f'_{Z_{i,j}^{in}, E_{i,j}^{in}}(PUMI_j^s, e)$$

Let us denote as $F_E(e)$ and $F'_E(e)$ the corresponding Cumulative Distribution Functions (CDFs). If the condition

$$\forall e \geq 0 : F_E(e) \leq F'_E(e), and \tag{29}$$

holds then F'_E is not better than F_E (equivalently, the PDF f'_E is not better than f_E). If in addition

$$\exists e > 0 : F_E(e) < F'_E(e) \tag{30}$$

then F'_E is worse than F_E (or equivalently, the PDF f'_E is worse than f_E).

PROOF. In the context of this paper, a better PDF at the starting time of a job provides a larger success-ratio. Suppose a job J, scheduled at $PUMI_j$, has an energy requirement of E_J. Then, the job success-ratio considering the super-capacitor energy CDF of $F_E(e)$ is

$$R_J = P(E \geq E_J) = 1 - P(E \leq E_J) = 1 - F_E(E_J) \tag{31}$$

Similarly, R'_J can be calculated based on $F'_E(e)$. Then, (29) implies that

$$\forall e \geq 0 : 1 - F_E(e) \geq 1 - F'_E(e) \Rightarrow R_J \geq R'_J \tag{32}$$

If (30) holds as well, then $\exists e > 0$ such that $1 - F_E(e) > 1 - F'_E(e)$, indicating that we have $R_J > R'_J$ in addition if J had an energy requirement of e. □

LEMMA 5.3. *Let $f_E(e)$ and $f'_E(e)$ be two alternative initial PDFs at the start of a hyperperiod, and consider that the $f'_E(e)$ is not better than $f_E(e)$. Then the energy PDFs at the starting times of all PUMIs within the hyperperiod of the two alternative systems have the same relation.*

PROOF. According to the relation between $f_E(e)$ and $f'_E(e)$ at $PUMI_1^s$ (based on Lemma 5.2), and the assumption that both systems have the same schedule, the proof is immediate for $PUMI_2^s$. Taking $PUMI_2^s$ as the new time origin, the same reasoning can be repeated until reaching any further *PUMI*. Thus, the proof is completed inductively. □

COROLLARY 5.4. *Assume that $f_E^k(e)$ and $f_E^{k'}(e)$ denote the PDFs at the start of hyperperiods k and k' with $k < k'$, respectively. If $f_E^{k'}(e)$ is not worse than $f_E^k(e)$, then the super-capacitor energy PDFs $f_E^{k''}(e)$ at the start of hyperperiods k'' are not worse than $f_E^{k'}(e)$ for all $k' < k''$.*

Lemma 5.3 and Corollary 5.4 have important implications to make sure that the low-to-high scheduler mode switch is safe for an obtained schedule, see Subsection 5.2.

LEMMA 5.5. *Suppose a given schedule and the energy PDF $f_E(e)$ at the start of the hyperperiod, which is obtained via a steady-state analysis. Reducing the power consumption Pow of an arbitrary job results in the energy PDF $f'_E(e)$ at start of the hyperperiod. Then, $f_E(e)$ is not better than $f'_E(e)$. The same relation holds for all energy PDFs of $PUMI$s within the hyperperiod before and after the power change.*

PROOF. The steady-state analysis is based on an iterated transient analysis until a fixed-point is reached. Without loss of generality, we consider that the *PUMI* with the reduced power consumption is the last one in the hyperperiod, say $PUMI_k$. The first transient analysis of the modified system starts from some initial $f_E(e)$ and therefore, the PDFs at the start of consecutive *PUMI*s are as before, up to $PUMI_k^s$. However, as the power consumption has been reduced during $PUMI_k$, then $f_E(e)$ is not better than the resulting energy PDF at start of the next hyper-period. Applying standard arguments on fixed-points as well as the monotonicity result of Lemma 5.3 leads to the statement of the Lemma. □

Shifting the starting times of jobs:. As mentioned, we are investigating the impact of different changes in a schedule. For simplicity, in the following, first we consider a fixed job ordering (no reoredering is permitted when changing a schedule) and only investigate moving a job within one epoch or between consecutive epochs, when there are idle times to do so. Later on, we discuss job reorderings.

LEMMA 5.6. *Consider an epoch which contains idle intervals, i.e., time intervals where no job is scheduled to run. Then, moving a job within the epoch (with no job reordering) does not affect the steady-state energy PDFs at the start time of any job.*

PROOF. Consider a job J within an arbitrary epoch with some idle intervals around it. Let f_1 and f_2 denote the super-capacitor energy PDFs at the start of the idle *PUMI* immediately before, and at the end of the idle *PUMI* immediately after that job, respectively (see Fig. 5). Also, assume that the super-capacitor energy PDFs at the start and end of the *PUMI* corresponding to J are denoted as f_s and f_e, respectively. As we consider an ideal super-capacitor with no power dissipation, with moving J the energy available in the super-capacitor does not change from the start of the preceding *PUMI* to the start of J, and also from the end of J to the end of the subsequent *PUMI*. Thus, it is obvious that $f_1 \equiv f_s$ and $f_e \equiv f_2$. □

Now, we investigate the effect of job backwarding, namely moving a job to a former epoch.

LEMMA 5.7. *Consider two consecutive epochs $epoch_j$ and $epoch_{j+1}$. Backwarding the first job, say J, of $epoch_{j+1}$ to be executed as the last job of $epoch_j$ does not worsen the energy PDF at the start of the successor of J in $epoch_{j+1}$. If the super-capacitor is full before adding the non-zero harvested energy at the end of $epoch_j$ then backwarding the job J results in a better super-capacitor energy PDF at the end of $epoch_{j+1}$, while the success-ratio of the backwarded job does not change.*

PROOF. Suppose an initial schedule that the super-capacitor energy at the end of $epoch_j$ is e_1 and the harvested energy at start of the succeeding epoch is e_h (sampled from a distribution with PDF H). Also, suppose e_J denotes the energy needed by the first job J of $epoch_{j+1}$. Then, in the initial schedule, the super-capacitor energy when starting to execute the successor J' of J in $epoch_{j+1}$ is

$$e_2 = Max(Min(e_1 + e_h, E_{max}) - e_J, 0). \tag{33}$$

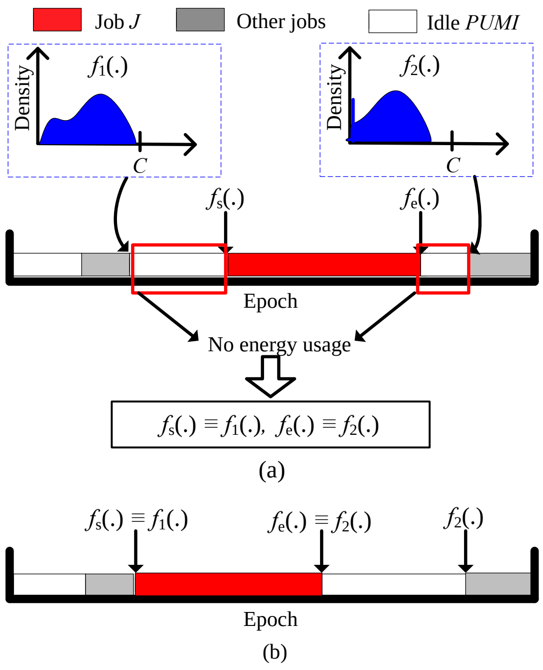

Fig. 5. Moving a job within one epoch describing Lemma 5.6: (a) before movement, (b) after movement.

In the modified schedule where J has been moved to the end of $epoch_j$, the super-capacitor energy at start of J' is

$$e'_2 = Min(Max(e_1 - e_J, 0) + e_h, E_{max}). \tag{34}$$

By some algebraic transformations it can be shown that $e'_2 \geq e_2$ and therefore, there is never less energy in the capacitance for every scenario. As the super-capacitor is full before adding the harvested energy at the end of $epoch_j$, it can be concluded that there is no job running in $epoch_j$. Therefore, the capacitor is full at the start of job J even when backwarded to $epoch_j$. As a result, its success-ratio does not change. From the above relations we find $e'_2 = Min(e_2 + e_h, E_{max})$ for $e_1 = E_{max}$. Therefore, $e'_2 > e_2$ if $e_J > 0$ and $e_h > 0$. □

The effect of job forwarding, namely moving a job to a later epoch, can also be investigated based on the results of job backwarding. For instance, suppose that the last job of $epoch_j$ is forwarded to $epoch_{j+1}$. Then the energy PDFs after movement are equal or worse for subsequent jobs as this change may cause more wastage of the harvested energy due to a full super-capacitor at the end of $epoch_j$.

COROLLARY 5.8. *Suppose a given schedule has been analyzed in its steady-state. According to Lemma 5.7, backwarding a job, i.e., executing it sooner, may decrease the success-ratio of the same job but is neutral or positive for the success-ratio of the others. Inversely, forwarding a job, i.e., executing it later, may improve its success-ratio but is neutral or negative for the success-ratio of all other jobs.*

According to the above corollary, more control is possible through job backwarding, since the movement does not invalidate the possible energy-feasibility of other parts of the schedule.

COROLLARY 5.9. *Consider that we have the steady-state energy PDFs of the super-capacitor at start and end of every job. Following Lemma 5.7 we backward a job. Calculating the success-ratios of all jobs based on the energy PDFs as obtained from the steady-state analysis before the movement gives a pessimistic view to all job success-ratios.*

The justification for Corollary 5.9 is as follows: Suppose all energy PDFs have been obtained based on the steady-state analysis introduced earlier. Then, a job is backwarded and its degraded success-ratio R is calculated based on the given PDFs with no further steady-state analysis, i.e., its initial energy PDF is equal to the energy PDF at the end of its new predecessor job. All following jobs are provided with an equal or better energy PDF which may have a positive impact on their success-ratios, see Lemma 5.7 and Lemma 5.3. According to Corollary 5.4, the fixed point calculations can only lead to equal or better PDFs.

Reordering of jobs:. Finally, we consider job reordering and we determine the corresponding consequences in terms of success-ratios.

LEMMA 5.10. *Let us consider a schedule containing two adjacent jobs that are executed in a single epoch. They are in the Success-Ratio-Constraint Monotonic (SRCM) order, i.e., the job with the lower success-ratio constraint is scheduled first. At least one of the two jobs violates its success-ratio constraint. Then, exchanging the order of the jobs does not make the schedule energy-feasible.*

PROOF. The two jobs J_1 and J_2 have energy requirements E_1 and E_2, and their success-ratio constraints satisfy $\alpha_1 > \alpha_2$. Consider the two different orderings: $J_1; J_2$ and $J_2; J_1$. We show that if the two jobs meet their success-ratio constraints in the second ordering, then they meet their success-ratio constraints in the first ordering too. Let E_s be the initial super-capacitor energy at start of the first job. Then, to successfully execute both jobs in the first ordering, we need to have

$$P(E_0 - E_1 \geq E_{CO}) \geq \alpha_1 \tag{35}$$

$$P(E_0 - E_1 - E_2 \geq E_{CO}) \geq \alpha_2 \tag{36}$$

where E_{CO} denotes the cut-off energy. In order to successfully execute the jobs in the second ordering, we require

$$P(E_0 - E_2 \geq E_{CO}) \geq \alpha_2 \tag{37}$$

$$P(E_0 - E_2 - E_1 \geq E_{CO}) \geq \alpha_1 \tag{38}$$

If (38) holds, then it is obvious that (35) holds as $E_0 - E_1 > E_0 - E_2 - E_1$. As $\alpha_1 > \alpha_2$, (36) also holds. □

Applying the previous Lemma 5.10 to any two adjacent jobs of a given schedule in an epoch indicates that the job with the highest success-ratio constraint should run first (considering no timing violation). With the same reasoning, other jobs should also be scheduled in the decreasing order of their success-ratio constraints.

5.2. The Mixed-Criticality Scheduling Algorithm

In this section, we present a heuristic non-preemptive scheduling algorithm for single-segment jobs with the following properties (suppose $\mathrm{T_{HI}} = \{\tau_i | l_i = \mathrm{HI}\}$ and $\mathrm{T_{LO}} = \{\tau_i | l_i = \mathrm{LO}\}$):

(1) When the scheduler is in its low mode, jobs of all HI- and LO-criticality tasks are executed in a manner that their time and success-ratio constraints are satisfied,

(2) when the system transitions to HI mode, i.e., when at least one HI-criticality job exceeds its LO execution time bound, it is guaranteed that all HI-criticality jobs satisfy their time and success-ratio constraints in steady-state, and
(3) the start times of all jobs in the non-preemptive schedule at both scheduler modes remain intact, resulting in no impact on the jobs which are run in both scheduler modes (namely, $\mathrm{T_{HI}}$); in other words, they even do not experience any transient problem resulted from the scheduler mode change in their steady-state success-ratio.

We use the following algorithms as parts of the overall Mixed-Criticality Success-ratio-Constrained Scheduling algorithm (MC-SCS):

— *No Criticality Change algorithm (NCC)*: This algorithm attempts to find a time- and energy-feasible schedule for a given task set at a fixed scheduler mode. We employ it to schedule $\mathrm{T_{HI}}$ when they exhibit their HI-criticality behavior. NCC may also be employed for the scheduling of non-mixed-criticality systems.
— *Low Scheduler Mode algorithm (LSM)*: This algorithm attempts to add jobs of $\mathrm{T_{LO}}$ tasks to the schedule obtained by NCC as used in the low criticality scheduler mode.

MC-SCS:. We first discuss how the two algorithms are assembled to build MC-SCS using the pseudo-code summarized in Algorithm 1. At first the set $\mathrm{T_{HI}}$, namely the HI tasks which run in both scheduler modes, are scheduled using NCC based on a given schedule $\mathrm{S_{HI}}$ (Line 3). Failure of NCC in finding a feasible schedule means un-schedulability of T using this algorithm (Lines 4-6). Then, the LO tasks are scheduled using LSM which leads to the final schedule S (Line 7).

ALGORITHM 1: MC-SCS

```
Input:  T = $\{\tau_i : (l_i, \pi_i, \vec{e}_i, Pow_i, \alpha_i)\}$ ;   $SC = (E_{max}, E_{CO}, Dis, P, H)$
Output: S: A feasible schedule for T. S = $null$ means T is un-schedulable by this algorithm.
1 $\mathrm{T_{HI}} = \{\tau_i \in T | l_i = \mathrm{HI}\}$;
2 $\mathrm{T_{LO}} = \{\tau_i \in T | l_i = \mathrm{LO}\}$;
3 $\mathrm{S_{HI}}$ = NCC($\mathrm{T_{HI}}, SC$);
4 if $\mathrm{S_{HI}} = null$ then
5     return $null$
6 end
7 S = LSM($\mathrm{T_{HI}}, \mathrm{S_{HI}}, \mathrm{T_{LO}}, SC$);
8 return S
```

NCC:. This algorithm works on top of any given single-segment time-feasible schedule. Based on what is emphasized in Corollary 5.8, we use job backwarding in our proposed idea to have a better control over the changes of the job success-ratios. Thus, we consider that the time-feasible schedule tends to schedule the jobs as late as possible, so that job backwarding is enabled as much as possible.

As a basic scheduling algorithm, we use Earliest Deadline as Late as possible (EDL) [Chetto and Chetto 1989] and modify it for non-preemptive scheduling. Due to the non-optimality of non-preemptive EDL, this approach may result in the failure of NCC at the early stages, even if the task set is time-feasible. On the other hand, EDL tends to schedule the jobs as late as possible and leads to opportunities for job backwarding. The proposed non-preemptive version of EDL first calculates idle times corresponding to each release-time, see [Chetto and Chetto 1989]. After the idle times passed, jobs with the earliest deadline are scheduled.

Table II. A sample task set for the description of the scheduling algorithm.

Task	l_i	π_i	$(\vec{\epsilon}_i[\text{LO}], \vec{\epsilon}_i[\text{HI}])$	α_i	Pow_i
τ_1	HI	6	(1, 2)	0.94	0.33
τ_2	HI	15	(1, 3)	0.98	0.2
τ_3	LO	5	(1, 1)	0.9	0.1
τ_4	LO	30	(4, 4)	0.9	0.12

A pseudo-code for NCC is provided in Algorithm 2. To have a better insight on the way how NCC works, we describe it while applying the algorithm to the sample task set given in Table II. The harvesting system is characterized by $E_{max} = 12$, $P = 10$, and the PDF of the harvested energy H has a uniform distribution in $[1, 2]$.

Step 0 of Fig. 6 relates to Line 2 of Algorithm 2. As can be seen, this gives a time-feasible schedule, thus passing Lines 3-5. After running Line 6 and obtaining the job success-ratios for the schedule, the while loop is executed and the jobs with acceptable success-ratio are added to S_{HI} iteratively (Lines 7-11); in the case that we find a job with unsatisfied success-ratio constraint ($\tau_{2,2}$ in our example, specified by symbol $'\times'$ in Step 0 of Fig. 6), then Lines 12-18 must be followed. The algorithm first tries to solve the problem locally in the same epoch, through SRCM (Line 14), since it has no impact on the success-ratio of jobs in other epochs. The term "temporal commutative" in that line means that no timing problem occurs when exchanging the jobs. If the problem persists, the next choice is to use Lemma 5.7, namely to backward a job that is scheduled ahead of the job with unsatisfied success-ratio. The first applicable choice for our ongoing example to improve the success-ratio of $\tau_{2,2}$ is Line 14. Here, the jobs whose success-ratio might be degraded by the reordering must be returned from S_{HI} to the set of unscheduled jobs S ($\tau_{1,5}$ in our example). Then, Line 17 applies the analysis method to the schedule obtained through merging the scheduled (S_{HI}) and unscheduled (S) jobs on the timeline. Function $\textbf{merge}(\text{S}_{\text{HI}}, \text{S})$ preserves the place of jobs in S_{HI} and S on the timeline. As can be seen in Step 1 of Fig. 6, applying Line 14 results in solving the problem of $\tau_{2,2}$ but making the success-ratio of $\tau_{1,5}$ unacceptable. In this example, one more round is needed to solve the problem of $\tau_{1,5}$ through Line 15, by backwarding $\tau_{1,2}$ (Step 2 of Fig. 6). Backwarding $\tau_{1,4}$ would have invalidated its own success-ratio. The success-ratios are reported in Step 2 of Fig. 6. The reason that the other success-ratios in the example have also been improved through the backwarding relates to what was emphasized in Lemma 5.7 and Corollary 5.8. We continue the algorithm till either S gets empty and a feasible schedule is obtained in S_{HI} (Line 20) or the algorithm fails (Line 16). We see that a feasible schedule for the HI tasks of our example was returned in this way.

LSM:. Suppose that a feasible schedule S_{HI} is given for T_{HI}. We present a heuristic non-preemptive scheduling algorithm which accommodates jobs of tasks belonging to T_{LO} into the time gaps of S_{HI} when the HI tasks show their LO execution time behavior, considering that S_{HI} can be changed if needed. For a simpler presentation, we propose some rules underlying the policies in the heuristic scheduling algorithm:
Rule 1: Let S_{HI} and S_{LO} be the static schedules in the high and low scheduler modes, respectively. Also, consider $f_{\text{S}_{\text{HI}}}$ and $f_{\text{S}_{\text{LO}}}$ as the corresponding steady state supercapacitor energy PDFs at the start of hyper-period when the scheduler is in the high and low modes. If the low-to-high mode change occurs during $PUMI_i$ ($PUMI_i$ certainly belongs to a HI job), we obtain a new schedule $\text{S}_{\text{LO,transient}}$ for that hyper-period constructed from two partitions: It is the same as S_{LO} during interval $[0, PUMI_i^s]$, and the same as S_{HI} during interval $[PUMI_i^s, \Pi]$. Then, for all such ***PUMI***s in which a low-to-high scheduler mode change is possible, at least one of the following conditions must be satisfied to be sure that every low-to-high scheduler mode change is safe:

ALGORITHM 2: NCC

```
Input:    $\mathrm{T} = \{\tau_i : (l_i = \mathrm{HI}, \pi_i, \epsilon_i = \vec{\epsilon}_i(\mathrm{HI}), Pow_i, \alpha_i)\}$ ;    $SC = (E_{max}, E_{CO}, Dis, P, H)$
Output:   $\mathbf{S}_{\mathrm{HI}}$: A feasible schedule for T where
          $\mathbf{S}_{\mathrm{HI}} = null$ means that T is un-schedulable by this algorithm.
1  $\Pi = LCM(\pi_1, ..., \pi_{|T|}, P)$;
2  $\mathbf{S} = \mathbf{EDL}(\mathrm{T}, \Pi)$ ;                   // S must be a possible time-feasible schedule
3  if $\mathbf{S} = null$ then
4      return null
5  end
6  $R_J$ = SS_EnergyAnalysis($\mathbf{S}, SC$) ;                  // For all jobs $J$, based on Section 4
7  $\mathbf{S}_{\mathrm{HI}} = null$ ;          // $\mathbf{S}_{\mathrm{HI}}$ must be a time- and energy-feasible schedule of jobs
8  while $\mathbf{S} \neq null$ do
9      determine the first job of S (i.e., $\tau_{i,k}$) & its epoch ($epoch_j$);
10     if $R_{\tau_{i,k}} \geq \alpha_i$ then
11         move $\tau_{i,k}$ from S to $\mathbf{S}_{\mathrm{HI}}$ ;                  // to the same place on the timeline
12     else
13         case select the first applicable option from the following list do
14             1: find the nearest temporal commutative job $J$ ahead of $\tau_{i,k}$ in $epoch_j$ with $\alpha < \alpha_i$;
                move $J$ and the following jobs from $\mathbf{S}_{\mathrm{HI}}$ to S; reorder $\tau_{i,k}$ and $J$ according to SRCM;
                see Lemma 5.10;
15             2: find the first $epoch_k$ from $epoch_j$ down to $epoch_2$ in $\mathbf{S}_{\mathrm{HI}}$ which the following is
                applicable to: in $epoch_k$, among the jobs ahead of $\tau_{i,k}$, subject to job timing
                constraints, select the first job $J$ of $epoch_k$ and backward it to $epoch_{k-1}$ in $\mathbf{S}_{\mathrm{HI}}$; see
                Lemma 5.7 and Corollary 5.8;
16             3: otherwise return $null$;
17         end
18         $R_J$ = SS_EnergyAnalysis(merge($\mathbf{S}_{\mathrm{HI}}, \mathbf{S}$), $SC$);
19     end
20 end
21 return $\mathbf{S}_{\mathrm{HI}}$
```

— **Condition 1**: Based on Lemma 5.3 and Corollary 5.4, $f_{\mathbf{S}_{\mathrm{LO}}}(PUMI_i^s)$ be not worse than $f_{\mathbf{S}_{\mathrm{HI}}}(PUMI_i^s)$.
— **Condition 2**: This is a more relaxed condition in comparison to Condition 1. The success ratios of all jobs belonging to $\mathbf{S}_{\mathrm{LO,transient}}$ need to be satisfied when carrying out a transient analysis starting from $PUMI_i$ and considering the initial energy PDF $f_{\mathbf{S}_{\mathrm{LO}}}$. Moreover, during a steady-state analysis, the energy PDFs at the start of the hyper-periods need to satisfy the following condition: Let us define as $f^k_{\mathbf{S}_{\mathrm{HI}}}$ with $k \geq 1$ the energy PDF at the start of the kth hyperperiod in the fixed-point iteration, where $f^0_{\mathbf{S}_{\mathrm{HI}}} = f_{\mathbf{S}_{\mathrm{HI}}}$. Now, the success-ratios of all HI jobs must be satisfied until we find a hyper-period with energy PDF $f^k_{\mathbf{S}_{\mathrm{HI}}}$ that is not worse than at least one of the previous energy PDFs $f^i_{\mathbf{S}_{\mathrm{HI}}}$, $0 \leq i < k$.

Rule 2: The jobs of tasks belonging to T_{LO} can be placed in time gaps. These time gaps come from two sources: i) the free times between jobs belonging to T_{HI} considering their HI execution time, and ii) $\vec{\epsilon}_i(\mathrm{HI}) - \vec{\epsilon}_i(\mathrm{LO})$ for each job $\tau_{i,k}$.
Rule 3: If it is required to perform backwarding of a HI job to expand the time gap needed for accommodating a job of T_{LO} or to improve the success-ratio of such a LO job, the timing and the degraded success-ratio of the HI job in the high scheduler mode must be checked to be sure that the change does not invalidate the feasibility of $\mathbf{S}_{\mathrm{HI}}$.

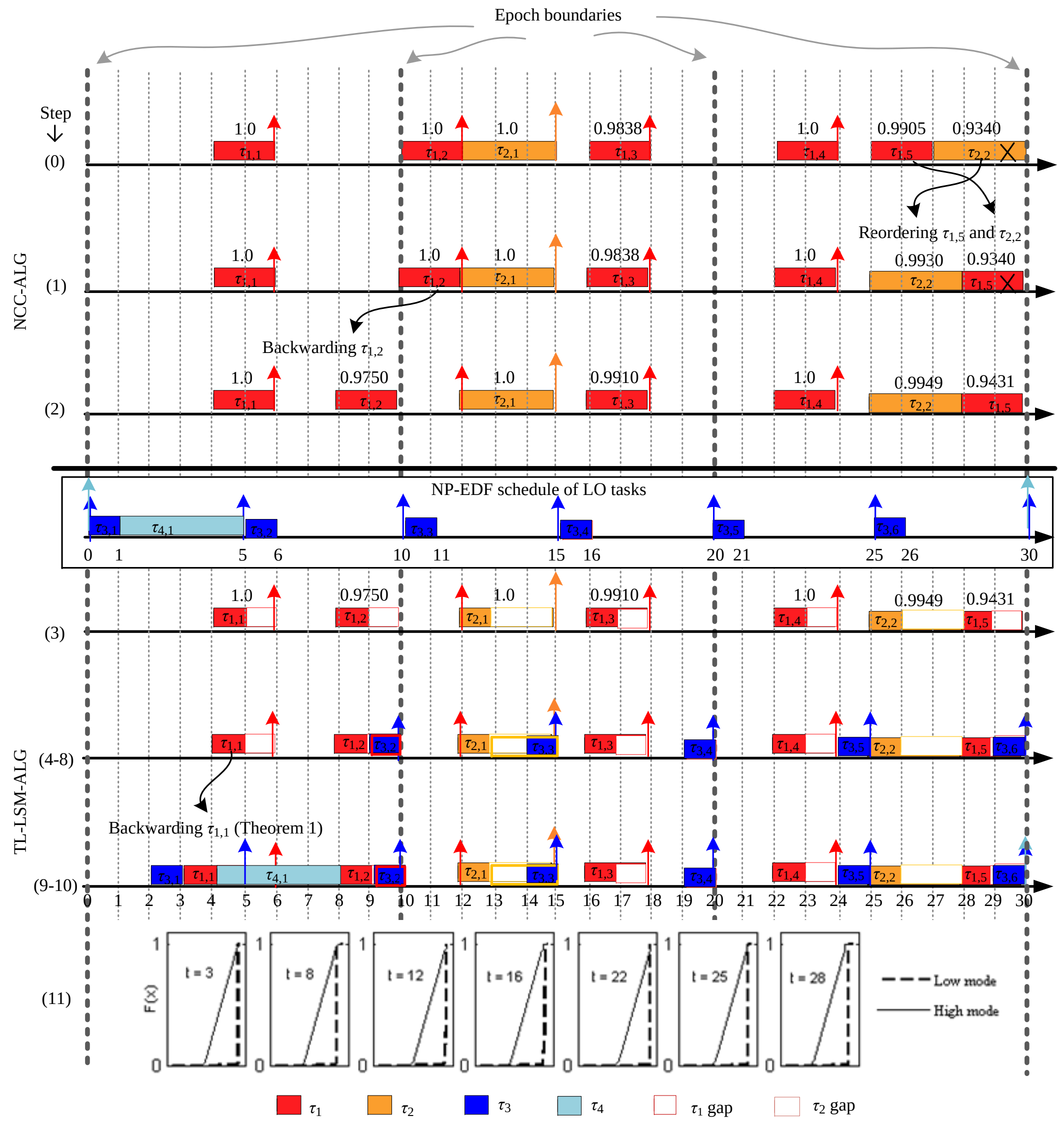

Fig. 6. Applying NCC and LSM on the example of Table II.

Rule 4: Backwarding HI jobs (as well as LO jobs) is done in a parsimony manner to conserve the option of further backwarding which may be needed to address the possible success-ratio problems of next jobs not scheduled yet.

These rules define the major idea of the LSM algorithm. A pseudo-code for LSM is provided in Algorithm 3. The algorithm works in three main steps.

Step 1 (Initialization): Lines 1-2 extend the schedule of the HI scheduler mode to the new hyperperiod expanded by considering T_{LO}. As LSM assumes that the scheduler is in its LO mode, some additional time gaps will be available, as defined in the description of Rule 2. We use NP-EDF (Non-Preemptive EDF) to determine the order of jobs belonging to tasks in T_{LO} (Line 3).

Step 2 (Time-Feasible Job Accommodation): This step tries to move the jobs which have been temporarily scheduled on the ***TS*** timeline to appropriate places on S_{LO} (Lines 4-23). In this step, we find appropriate time gaps (Line 7), backward jobs belonging to T_{HI} if necessary regarding the time requirements of jobs of T_{LO} (Lines 8-21), and we preserve the EDF order from ***TS*** (Lines 5, 13, 14, 22). Each unsuccessful trial is undone and tagged to prevent repeating the same experience by the algorithm (Lines 9, 17-20). To be able to backward LO or HI jobs for solving the possible success-ratio or energy PDF problems (see Rule 1), we start accommodating jobs from the last job, as late as possible, and according to First-Fit in the reverse order. In this way, more maneuver is possible as the time gaps ahead of jobs remain free as much as possible, supporting Rule 4.
Step 3 (Success-Ratio Tuning): As the accommodated jobs may not pass the corresponding success-ratio constraint for jobs of T_{LO} or may invalidate Rule 1, some reordering among the LO jobs, backwarding of LO jobs, or backwarding of HI jobs might help. This is done in Lines 24-28. Again if some trial for change in the schedule has been ineffective or creates new timing/success-ratio problems, the change is undone and other choices are examined. Depending on the outcome, either $null$ or the finalized schedule S will be returned.

To continue the example of Fig. 6, we consider that τ_3 and τ_4 of Table II are to be added by the LSM algorithm, where their corresponding non-preemptive EDF schedule (Line 3 of Algoritm 3) is shown in Fig. 6. The result of applying Line 2 of LSM is shown in Step 3 of Fig. 6. Finding appropriate time gaps for $\tau_{3,6}$, $\tau_{3,5}$, $\tau_{3,4}$, $\tau_{3,3}$, and $\tau_{3,2}$ is done in the first trial of Line 7 for each job (Steps 4-8 in Fig. 6). Also, an appropriate time gap for $\tau_{4,1}$ is simply found by the first trial (in the time gap of $[0-4]$). However, as we have a problem in finding such a gap for $\tau_{3,1}$, we are forced to use backwarding (Lines 9-15). The only option is to backward $\tau_{1,1}$ for 1 time unit. According to Lemma 5.6 backwarding is safe in terms of success ratio constraint. Based on Line 13 of Algorithm 3, $\tau_{4,1}$ is returned to ***TS*** and based on Line 14, $\tau_{4,1}$ is selected again for placement. In this way, $\tau_{4,1}$ is placed in the time gap of $[4, 8]$, and then $\tau_{3,1}$ is scheduled in the time gap of $[2, 3]$ (see Step 9-10 in Fig. 6). The obtained success ratios of all jobs in the low scheduler mode are equal to 1.0, thus they are not reported in the figure. The energy CDFs at the start of *PUMI*s corresponding to the HI jobs in both low and high scheduler modes have been depicted in Step 11 of Fig. 6. As can be observed, all of them satisfy Rule 1.

5.3. Extending the Algorithm to Multi-Segment Jobs

In this section, we extend the proposed single-segment heuristic to multi-segment jobs obtained from preemptive scheduling; it must be emphasized that we use a pessimistic view in this section due to the reasons described in the following.

Suppose that we apply the preemptive versions of EDL and EDF, respectively, to the HI and LO jobs when we initially schedule them at start of the proposed algorithms. Then we have some predetermined segments for each job, where the difference between their LO and HI execution times is interpreted as: i) some job segments have equal LO and HI execution times equal to the segment length, ii) possibly one segment has different non-zero LO and HI execution times (this is the segment in which the scheduler mode switch may occur), and iii) some segments which have LO execution time of zero and HI execution time equal to the segment lengths.

One major problem is that Lemma 5.5 is not valid for multi-segment jobs, and thus many other results presented in Section 5.1 cannot directly be applied to such situations. First, we concentrate on the reason for the invalidity of Lemma 5.5 using a simple example.

ALGORITHM 3: LSM

```
Input:  $\mathrm{T_{HI}} = \{\tau_i : (l_i = \mathrm{HI}, \pi_i, \vec{e}_i, Pow_i, \alpha_i)\}$;   $\mathrm{S_{HI}}$: A feasible schedule for $\mathrm{T_{HI}}$;
        $\mathrm{T_{LO}} = \{\tau_i : (l_i = \mathrm{LO}, \pi_i, \vec{e}_i, Pow_i, \alpha_i)\}$;   $SC = (C, E_{CO}, Dis, P, H)$
Output: $\mathrm{S_{LO}}$: A feasible schedule for $\mathrm{T_{HI}} \cup \mathrm{T_{LO}}$. $\mathrm{S_{LO}} = null$ means that $\mathrm{T_{HI}} \cup \mathrm{T_{LO}}$ is
        un-schedulable by this algorithm.
1  $\Pi = LCM(\pi_1, ..., \pi_{|T|}, P)$ for $\tau_i \in \mathrm{T} = \mathrm{T_{HI}} \cup \mathrm{T_{LO}}$;
2  $\mathrm{S_{LO}}$ = repeated $\mathrm{S_{HI}}$ throughout $\Pi$, according to $\vec{e}_i(\mathrm{LO})$ of tasks;
3  TS = EDF($\mathrm{T_{LO}}$) throughout $\Pi$;
4  while TS $\neq null$ do          // Accommodating $\mathrm{T_{LO}}$ in $\mathrm{S_{LO}}$ in a time-feasible manner
5      select the last job $\tau_{i',k'}$ from the TS timeline;
6      set interval $[R, D]$ based on release-time & deadline of $\tau_{i',k'}$;
7      $\tau_{i,k}$ = First-Fit($\tau_{i',k'}$, $\mathrm{S_{LO}}$, $D$, $R$) ; // searches $\mathrm{S_{LO}}$ from $D$ back to $R$ for the job $\tau_{i,k}$
         after which $\tau_{i',k'}$ can be accommodated or returns the dummy job $\tau_{0,0}$ if the gap is
         at the start of the hyper-period; otherwise it returns $null$
8      if $\tau_{i,k} = null$ then
9          $\tau_{i,k}$ = Find-Closest-Gap($\tau_{i',k'}$, $\mathrm{S_{LO}}$, $D$, $R$) ; // selection is done amongst untagged
             gaps (see Line 18); Rule 4 is also supported by minimalistic backwarding
             (see Line 15)
10         if $\tau_{i,k} = null$ then
11             return $null$
12         end
13         all jobs belonging to $\mathrm{T_{LO}}$ which are scheduled before $\tau_{i,k}$ are moved from $\mathrm{S_{LO}}$ to TS
             which is ordered by EDF;
14         update $\tau_{i',k'}$ and $[R, D]$ ;                      // similar to Lines 5 and 6
15         Backward($\tau_{i,k}$) regarding the extra gap needed for $\tau_{i',k'}$; update $\mathrm{S_{HI}}$;
16         $R_{i,k}$ = SS_EnergyAnalysis($\mathrm{S_{HI}}, SC$) ;             // since $\mathrm{S_{HI}}$ has been changed
17         if $R_{i,k} < \alpha_i$ then                                   // Supporting Rule 3
18             undo all changes performed from Lines 9-15 on $\mathrm{S_{HI}}$, $\mathrm{S_{LO}}$, and TS ;   // all the
                 undone trials are tagged, so they will not be repeated
19             continue from Line 4;
20         end
21     end
22     move $\tau_{i',k'}$ from TS to the latest possible time after $\tau_{i,k}$ in $\mathrm{S_{LO}}$ ;     // Supporting Rule 4
23 end
24 $R_J$ = SS_EnergyAnalysis($\mathrm{S_{LO}}, SC$) ;     // This is the pre-requisite for trying to make
     $\mathrm{T_{LO}}$ jobs energy-feasible
25 case doing steps similar to Lines 7-20 of NCC on $\mathrm{S_{LO}}$, and select the first applicable option
     from the following list for each job of $\mathrm{T_{LO}}$ with unsatisfied success-ratio constraint or each job
     of $\mathrm{T_{HI}}$ for which Rule 1 is not satisfied, to address the problem do
26     1: reordering and backwarding jobs of $\mathrm{T_{LO}}$: try reordering with backwarding the
         nearest possible LO job ahead of the job in $\mathrm{S_{LO}}$ which needs improvement;
27     2: backwarding jobs of $\mathrm{T_{HI}}$: try backwarding the possible nearest HI job ahead of the job
         in $\mathrm{S_{LO}}$ which needs improvement, regarding validity of Rule 3 ($\mathrm{S_{HI}}$ feasibility) and Rule 1
         (for the backwarded HI job in $\mathrm{S_{LO}}$);
28     3: otherwise return $null$
29 end
30 return $\mathrm{S_{LO}}$
```

Suppose a sample task set as shown in Table III and a given multi-segment schedule for the high scheduler mode, as shown in Fig. 7. Here, we consider $E_{max} = 12$, $P = 10$, and the harvested energy PDF H is uniform in $[1, 2]$. In Table IV, which includes the success-ratios calculated based on the analysis, by "normal analysis" we mean that upon failure of the first segment of a job, the job second segment does not run. Looking at job success-ratios under the normal analysis, we see that when the system is in

Table III. A sample task set showing invalidity of Lemma 5.5 for multi-segment jobs.

Task	l_i	π_i	$(\vec{\epsilon}_i[\text{LO}], \vec{\epsilon}_i[\text{HI}])$	α_i[1]	Pow_i
τ_1	HI	30	(10, 11)	-	0.2
τ_2	HI	30	(2, 2)	-	0.6
τ_3	HI	30	(2, 2)	-	0.3
τ_4	HI	30	(4, 4)	-	0.25

[1] This parameter has not been used in this example.

low scheduler mode, in spite of reducing the energy consumption of τ_1 by using its LO (rather than HI) execution-time, the success-ratio of $\tau_{4,1}$ decreases, which contradicts Lemma 5.5. Justification for such a scenario is as follows: When the energy consumption of $\tau_{1,1}$ decreases in reducing its execution time from 11 to 10, the energy PDF at start of $\tau_{3,1}$ gets better. Therefore, the success probability of the first segment of $\tau_{3,1}$ increases, and thus, with a higher probability (with respect to the high scheduler mode) we start executing the second segment of $\tau_{3,1}$ which might consume the energy harvested at start of the third epoch (time 20). This results in a worse energy PDF at start of $\tau_{4,1}$, degrading its success-ratio.

Similarly, we can find an example which shows that backwarding a job, which reduces its own success-ratio and thus reduces the job power consumption, can have a similar impact on the schedule. Therefore, the success-ratio of some jobs in the schedule might decrease with the backwarding which, for multi-segment jobs, contradicts some results which are valid for single-segment jobs.

According to the above discussion, to have a safe success-ratio calculation and to be able to use Lemma 5.5 and other useful results proposed in Section 5.1, we can employ a $pessimistic\ analysis$ by considering that the system consumes the energy of all segments of a job, even if it failed in one of its early segments. The corresponding results for the mentioned example are shown in Table IV. In this table, the maximum energy consumptions of τ_3 and τ_1 with $\vec{\epsilon}_1[\text{LO}]$ are considered in the low scheduler mode; and, the maximum energy consumptions of τ_3 and τ_1 with $\vec{\epsilon}_1[\text{HI}]$ are considered in the high scheduler mode (only jobs of τ_1 and τ_3 are multi-segment in this example). As can be observed, according to the pessimistic view, it is certain that the success-ratios are better in the low scheduler mode. Although it is pessimistic with respect to the normal analysis, since usually the desired success-ratio constraints are high values, this pessimism is not too problematic. For example, if $\alpha_i = 0.99$ for a typical task τ_i, then the second (or later) segment(s) of a job can fail with probability of at most 0.01, and thus, by taking the maximum energy consumption of that segment(s) into account for the calculations, we are at most 1% pessimistic about the energy consumption of that segment of the job.

Then, regarding a few points which must be noticed, the theorems and rules presented for the scheduling of single-segment jobs can be used for individual segments of multi-segment jobs.

First, we focus once more on the calculation method for the success-ratios of multi-segment jobs. As mentioned in Section 4.4, the calculated success-ratio of an individual job is decreasing from one segment to the next. The resulting job success-ratio is equal to what is obtained when the analysis reaches the last job segment (e.g., for the job of Fig. 4, the job success-ratio obtained until end of its first segment is $1 - Prob(s_{1,3})$, and what is obtained until end of the second segment is $1 - Prob(s_{1,3}) - Prob(s_{1,4})$). In this regard, we should take care that at none of the steps of accommodating a job segment into the schedule, a job should experience a calculated success-ratio smaller than its corresponding constraint. For instance, if a four-segment job satisfies a success-ratio constraint of 99%, it is almost definite that the calculated success-ratio has been very close to 1 at its early segments, and in a decreasing order, the job success-ratio at the

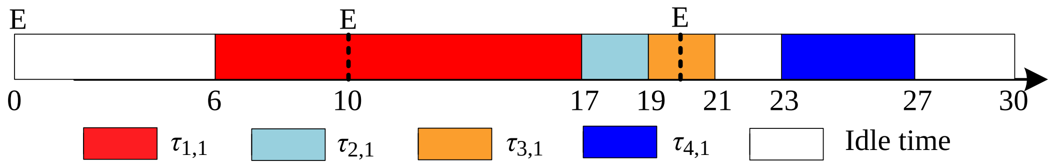

Fig. 7. A multi-segment schedule for the task set of Table III in the high scheduler mode.

Table IV. Success-ratio of jobs in the high and low scheduler modes under normal and pessimistic analyses.

Scheduler mode & Type of analysis	τ_1	τ_2	τ_3	τ_4
$\epsilon_1 = \vec{\epsilon}_1[\text{HI}] = 11$ (Normal)	0.9983	0.6087	0.3734	**0.9714**
$\epsilon_1 = \vec{\epsilon}_1[\text{LO}] = 10$ (Normal)	1.0000	0.7934	0.6078	**0.9694**
$\epsilon_1 = \vec{\epsilon}_1[\text{HI}] = 11$ (Pessimistic)	0.9938	0.4556	0.2538	0.7535
$\epsilon_1 = \vec{\epsilon}_1[\text{LO}] = 10$ (Pessimistic)	1.0000	0.6502	0.4423	0.7997

fourth segment is at least 99%. Based on this property, we propose a pessimistic view to the energy usage of a multi-segment job to have a simpler multi-segment scheduling algorithm. More precisely, although in case of failure of an early segment of a job its further segments are not executed (ignorance of their execution occurs at most in 1% of times for the above example), we pessimistically consider that all the segments are executed and consume energy.

According to the above discussion, when a change in a schedule occurs, more extensive changes occur in the success-ratios, when compared to single-segment jobs. For example, if a job segment is backwarded, the success-ratio of the job it belongs to, calculated until that segment, decreases (see Lemma5.7). Based on the above mentioned decreasing order of calculated success-ratios throughout multiple segments of that job, this affected success-ratio is an upper bound for the overall job success-ratio. Similarly, we can discuss about backwarding and reordering of some job segments, and their positive and negative impacts on several segments, which may be extended to other segments of the respective jobs.

Overall, it is more complex to schedule job segments in such a way that their success-ratio thresholds be satisfied, since upon every action on the schedule, several segments belonging to each affected job must be taken into account. Therefore, we pay more attention to the reordering and backwarding decisions: if we want to improve the success-ratio of a job segment to address its success-ratio problem, we do not perform an action which may have a negative impact on the success-ratio of any previous segment belonging to the same job. Further, in the low scheduler mode, we must take care of energy PDFs according to Rule 1 presented in Section 5.2 for the segment in which the mode switch might occur (containing the end of LO execution time of that job) and the segments before which corresponding to the same HI job. In this way, the low-to-high scheduler mode switch will be safe according to Lemma 5.3 and Corollary 5.4. Following the same pattern, we will find whether the mixed-criticality task set is schedulable by the algorithm or not.

5.4. Computational Complexity

The most time consuming part of the scheduling algorithm is related to the analytical analysis (to calculate the job success ratios) in each step. Thus, the complexity of the algorithm can be stated as the number of times that the analysis is run, which depends on the number of jobs (or job segments) during one hyper-period. In fact, the time complexity of the scheduling algorithm is only implicitly affected by the number of tasks. More details are given below.

Complexity of the analysis: Time complexity of the proposed analytical method depends on:

(1) the number of the generated states throughout one hyper-period (Section 4.1), and
(2) the number of iterations (i.e., hyper-periods) needed to reach steady-state in the analysis (Section 4.3).

The number of the generated states is equal to $n.k$, where n is the number of criteria in the Dis definition and k is the number of ***PUMI***s in one hyper-period. Considering non-preemptive scheduling, the maximum value of k will be $k = (\sum_{\tau_i} \frac{2.\Pi}{\pi_i}) - 1 + (\frac{\Pi}{P} + 1)$, where term $(\sum_{\tau_i} \frac{2.\Pi}{\pi_i}) - 1$ denotes the maximum number of distinct instants which are resulted from the start-time and finish-time of the released jobs during one hyper-period, and term $\frac{\Pi}{P} + 1$ is the number of distinct instants resulted from epochs in one hyper-period. It should be noted that the worst number of released jobs $\sum_{\tau_i} \frac{\Pi}{\pi_i}$ is exponential w.r.t. the number of tasks [Cai and Kong 1996].

In order to calculate the steady-state energy PDF, we assume that the super-capacitor is empty at the start of the first hyper-period. Then we calculate the energy PDF at the start of consecutive hyper-periods iteratively; we stop by observing that the PDF converges (this depends on the accepted precision âĂŞ if better precision is needed, later convergence will be resulted). Therefore, the number of required iterations n_{Π} to reach steady state is at least 1. The maximum number of iterations depends on the accepted precision in calculation, hyper-period length, super-capacitor characteristics, pattern of power consumption and the energy harvester properties. Therefore, the time complexity of the analysis will be of $O(n_{\Pi}.((\sum_{\tau_i} \frac{2.\Pi}{\pi_i}) - 1 + (\frac{\Pi}{P} + 1)))$.
Complexity of the scheduling algorithm: As mentioned above, the most time-consuming part of the scheduling algorithm is the computation of job success ratios using consecutive running of the analysis. Thus, the analysis complexity is the dominant factor.
For the NCC algorithm (Algorithm 2), the major part is the while-loop, in which the analysis method is invoked and the success ratio of one job (the first job in S) is verified (Lines 9-10). If the success ratio constraint of the first job in S is satisfied it is added to S_{HI}; otherwise, it is tried to fulfil the success ratio constraint by reordering and/or backwarding. To satisfy the success ratio of a job, at most one reordering and, if needed, zero or more backwarding are done (Lines 14-15). Thus, to fulfil the success ratio of each job, multiple iterations of the while-loop might be required.

Considering the single-segment case, time complexity of the algorithm (based on the number of times that the analysis method is invoked) is $O(x.(\sum_{\tau_i} \frac{\Pi}{\pi_i}))$, where x is the maximum number of times that the while-loop should be iterated for each job. x depends on some factors like the number of jobs that are completely executed in one epoch, power consumption of the candidate job for backwarding, the number of epochs before the current epoch, and the release-time and deadline constraints of the scheduled jobs (jobs in S_{HI}). Since the number of jobs in one hyper-period is exponential w.r.t. the number of task, time complexity of the scheduling algorithm (based on the number of times the analysis is run) is at least exponential in the number of tasks. It should be noted that we can determine an upper limit for the value of x to have an exact bound on the time complexity of the algorithm while the performance of the algorithm is preserved. This is because of the fact that when we want to improve the success-ratio of a job by backwarding a preceding job, the backwarding losses its impact when the distance between the two jobs gets larger, so we can bound the distance someway.

Complexity of the LSM algorithm, similar to NCC algorithm, is at least exponential in the number of tasks as well.

6. NUMERICAL EVALUATION

This section focuses on verifying the proposed analytical method by comparing it to simulation while considering different sources of inaccuracy and imprecision. Also, we investigate the performance of the proposed scheduling algorithm for both non-mixed and mixed setups in comparison to classic scheduling algorithms.

6.1. Simulation vs. Analytical Results

This section considers the case of non-ideal super-capacitors which have non-zero power dissipation and it discusses the accuracy and precision of the proposed analytical method by comparing it to simulation. The precision is investigated by assuming the same system model for both, analytical method and simulation.

In the simulations, monitoring and fault detection is implemented as follows:

— Single-segment jobs: The simulator, which knows the power consumption of each job, the WCET of that job, and the available energy in the super-capacitor, checks to see whether the job can be successfully completed. If yes, the job is executed and the simulator time and the super-capacitor energy are updated based on the mentioned information. If not, the job is encountered as failed, the scheduler simulation time is updated to the initial finishing time of that job, and the super-capacitor energy content is updated by considering the extra energy leaked after passing the super-capacitor cut-off energy.

— Multi-segment jobs: Here, we follow a method similar to what is mentioned above, but at the granularity of job segments, namely the job failure is encountered when one segment of that job fails. Then, the next segments of a failed job do not run in the simulation (please see Section 4.4 and Fig. 4 for details of the extended state graph).

In addition, there exist some inaccuracies in the model that will be investigated through extensive simulations: i) The power consumption of job segments are varying in time due to the dynamics of task activities [Mohaqeqi et al. 2014], while in the analytical method we consider a fixed average value of power consumption. We investigate how much inaccuracy is introduced by this assumption. ii) We consider an abstract model for the analytical method by assuming that the intermediate reservoir energy can be instantaneously transferred to the super-capacitor, while in fact this transfer takes some time depending on the reservoir capacity and characteristics of the corresponding circuits. To investigate the inaccuracy enforced by this abstraction, we compare the results with some simulation setups considering two connected reservoirs. While one of them is discharging either to the super-capacitor or to the real-time tasks, the other one is charged by energy harvesting. At each epoch boundary, the two reservoirs switch their role. There exist more sources of inaccuracy like the estimation of the dissipated power or the energy loss in the voltage regulators which will not be considered further in this paper.

For the aforementioned comparisons, we consider two medium-scale and small-scale Equivalent Series Resistance (ESR) pulse super-capacitors with nominal capacities of $C = 10$F and $C = 50$mF. Details of these setups are given in Table V. As the power dissipation of the small super-capacitor is insignificant ($Setup_2$), we consider an artificial setup with increases the power dissipation ($Setup_3$) in order to more extensively discuss the precision and accuracy of the analytical method. The energy harvester is considered to scavenge solar energy.

In the experimental arrangement considered in this section, the system scheduler is considered to be in a single criticality mode with a single execution time ϵ_i. Task sets are selected for different system utilizations $U = \sum_{i=1}^{n} u_i$ (where $u_i = \epsilon_i/\pi_i$), each with $n = 3$ tasks (see Table VI). The periods are integer dividers of Π (the hyperperiod) in

Table V. The system parameters for different setups.

System Component	Parameter	***Setup*$_1$**	***Setup*$_2$ & *Setup*$_3$**
Super-capacitor(SC)	C	10F	50mF
	V_r (rated voltage)	2.7V	3.6V
	$C = \frac{1}{2}CV_r^2$	36450mJ	324mJ
	$Dis(E)$: see (3)	Given from [Yang 2013]	***Setup*$_2$**: Scaled $Dis(E)$ of $Setup_1$; ***Setup*$_3$**: Artificially increased $Dis(E)$ w.r.t $Setup_2$
Reservoir(R)	C^R	200mF	1mF
	V_r (rated voltage)	2.7V	3.6V
	$C^R = \frac{1}{2}C^RV_r^2$	729mJ	6.48mJ
	Discharge time $= 5R^RC^R$	1000ms ($R^R = 1\Omega$)	12ms ($R^R = 2.4\Omega$)
Energy Harvester	P	$1000ms$	$12ms$
	$H(.)$	Uniform: $U[300-700]$mJ or Triangular: $\Delta[200, 500, 700]$mJ	Uniform: $U[2,3]$mJ or Triangular: $\Delta[2, 2.5, 3]$mJ
Consumer device (i.e.,tasks)	Pow_{max}	1200mW	616mW
	$[\pi_{min}, \pi max]$	Uniform: $U[200-60000]$ms	$U[60-6000]$ms
	Π	120000ms	12000ms

the interval of $[\pi_{min}, \pi_{max}]$. Power consumptions of tasks are calculated such that the following equation holds

$$\sum_{i=1}^{n} \frac{\Pi}{\pi_i}(\epsilon_i \cdot Pow_i) = \frac{\Pi}{P} \cdot \bar{E}, \tag{39}$$

where $\bar{E}$ is the expected harvested energy in one epoch and $\frac{\bar{E}}{P}$ with epoch length P is the average harvested power. Further, $\epsilon_i \cdot \frac{Pow_i}{\pi_i}$ on the left side of (39) is the average power consumption of each job of task τ_i. Then, we can name the ratio between the latter and the former as the job energy utilization

$$u_i^e = \frac{\epsilon_i \cdot Pow_i/\pi_i}{\bar{E}/P}$$

where $U^e = \sum_{i=1}^{n} u_i^e$. In other words, (39) states that the energy utilization U^e for the task set is 1; $U^e > 1$ leads to the situation where the super-capacitor low energy in steady-state, and $U^e < 1$ leads to the reverse, namely an almost full super-capacitor in average.

The generation of task sets is done as follows: **Step 1.** The UUniFast algorithm [Bini and Buttazzo 2005] is used to generate n utilizations $u_1, u_2, ..., u_n$ for a task set with total utilization equal to U. **Step 2.** The period π_i of each task τ_i is an integer number generated from the range $[\pi_{min}, \pi_{max}]$ with a hyper-period limitation technique [Goossens and Macq 2001] so that the hyper-period will be equal to Π. **Step 3.** The WCET of task τ_i is set to $\epsilon_i = u_i \cdot \pi_i$. **Step 4.** Similar to [Abdeddaïm et al. 2014], the UUniFast algorithm is used to generate n energy utilizations $u_1^e, u_2^e, ..., u_n^e$ of the task set with total energy utilization equal to 1, i.e., $U^e = 1$. **Step 5**. The power consumption of task τ_i is set to $Pow_i = \frac{u_i^e}{u_i} \cdot \frac{\bar{E}}{P}$, considering the maximum power constraint Pow_{max} in Table V.

Table VI. Sample task sets used for comparison between analysis and simulation.

Task set		π_i(ms)	ϵ_i(ms)	Pow_i(mW)	Utilisation	System Setup
$\mathbf{A}^1$ $\mathbf{A}^2$	τ_1	6000	840	454	0.5	$\textbf{\textit{Setup}}_2(\mathbf{A}^1)$ $\textbf{\textit{Setup}}_3(\mathbf{A}^1)$
	τ_2	600	66	211		
	τ_3	96	24	486		
B	τ_1	500	190	234	0.6	$\textbf{\textit{Setup}}_2$
	τ_2	160	6	22		
	τ_3	1200	219	609		
C	τ_1	1500	515	439	0.7	$\textbf{\textit{Setup}}_2$
	τ_2	600	34	594		
	τ_3	800	240	80		
D	τ_1	480	198	260	0.8	$\textbf{\textit{Setup}}_2$
	τ_2	1200	241	260		
	τ_3	1500	280	260		
E	τ_1	300	43	998	0.45	$\textbf{\textit{Setup}}_1$
	τ_2	2000	580	1185		
	τ_3	40000	667	799		
F	τ_1	12000	1116	817	0.7	$\textbf{\textit{Setup}}_1$
	τ_2	7500	254	667		
	τ_3	8000	2985	1076		

For the task sets of Table VI we use the above method and select an individual task set for each configuration. To compute the convolutions needed for the analytical method, we use the Fast Fourier Transform (FFT) while integrating with the same granularities for time and energy discretization selected from the list: $\frac{1}{64}, \frac{1}{128}, \frac{1}{256}, \frac{1}{512}, \frac{1}{1024}$. The algorithm is implemented in Java as proposed in [Press et al. 2002]. Fig. 8 shows the absolute error (with respect to simulation) of the expected super-capacitor energy at the start of *PUMI*s throughout the hyperperiod for task set $\mathbf{A}^1$ as given in Table VI, scheduled according to EDF, with different discretization granularities. Fig. 9 gives job success-ratios and their absolute errors throughout the hyperperiod, along with the expected super-capacitor energy of the respective *PUMI*s. The errors are reported for the two extreme granularities of $\frac{1}{64}$ and $\frac{1}{1024}$. As can be observed, the maximum numerical errors occur when the super-capacitor is almost empty.

According to Figs. 8 and 9, from here to the end of paper, we use the granularity of $\frac{1}{1024}$ for computing the analytical results, which gives an acceptable maximum absolute error. To have a better insight into the accuracy of the analytical method, we consider all task sets given in Table VI and report their task success-ratios, defined as the minimum success-ratio among the jobs of that task during a hyperperiod (see Table VII). For the simulation results, we have reported the average of results obtained from 100 simulation runs of length 100000 hyperperiods for each task set, with a confidence level of 0.99 within 0.005 for the success-ratios. For one task set of the small-scale super-capacitor setup (Task sets $\mathbf{A}^1$ and $\mathbf{A}^2$) and one task set of the medium-scale setup (Task set **F**), we have done more diverse experiments by considering EDF scheduling algorithm and different distributions of harvested energy (uniform and triangular) with the same mean value.

The comparison between analytical and simulation results are based on the system model considered in this paper with constant power consumption for each segment are reported in the column of *basic assumptions* in Table VII. As can be observed, the results are almost identical and the errors are due to discretization in the numerical computations. Throughout the analytical method we find that the super-capacitor energy PDF at the start of a hyperperiod converges towards a stationary distribution. The formal proof of this behavior is still an open issue. As an example, the steady-state energy PDF for the task set $\mathbf{A}^1$ (EDF scheduling and uniform energy distribution of harvested energy) is determined via simulation and analysis, see Fig. 10.

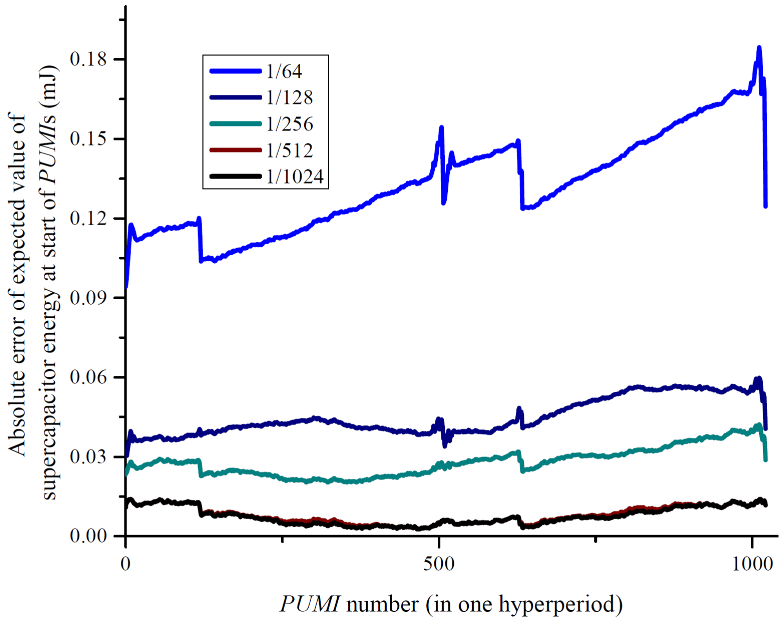

Fig. 8. Absolute error of expected super-capacitor energy at start of *PUMI*s for different discretization granularities.

Next, we discuss the accuracy of the model w.r.t. two assumptions: i) Constant power consumption during job execution, and ii) instantaneous energy transfer from the reservoir to the super-capacitor. To check the impact of the former assumption, we use SimpleScalar [Burger and Austin 1997] and Wattch [Brooks et al. 2000] for the Alpha 21264 processor [Gieseke et al. 1997] to get the power trace of Qsort from the MiBench benchmark suite [Guthaus et al. 2001], which is one of the most power varying programs with 55 percent power jitter and 17 percent variance in the power. The obtained power trace is scaled so that the average power consumption of the job remains equivalent to the constant power considered in the model. In the analysis we still use the average power consumption of each job segment as a constant value corresponding to the average of the power trace for that segment. We feed the simulator with that power trace. The column of *variable task power* in Table VII shows the results which can now be compared to the model results (*basic assumptions*) in Table VII. The difference is less than 0.0018.

Now, we discuss the second assumption in the model, namely the instantaneous energy transfer between the reservoir and the super-capacitor. In fact, this energy transfer takes some time which depends on the resistance of the corresponding RC circuit model. We simulate this non-ideal energy transfer by considering a pessimistic linear energy transfer which exaggerates the errors (the constant rate is considered equal to the ratio between the maximum energy that can be stored in the reservoir and the epoch size) and compare the results with respect to our model with basic assumptions. The column of *non-ideal energy transfer* in Table VII shows the results. We encounter two main reasons for the considerable errors with respect to the model with basic assumptions: i) If the energy of super-capacitor is near cutoff, the assumption of an instantaneous energy transfer makes the success-ratios obtained from analysis optimistic because it creates the illusion of having that amount of energy at the start of epoch, while in fact the energy can only gradually be transferred to the system. ii)

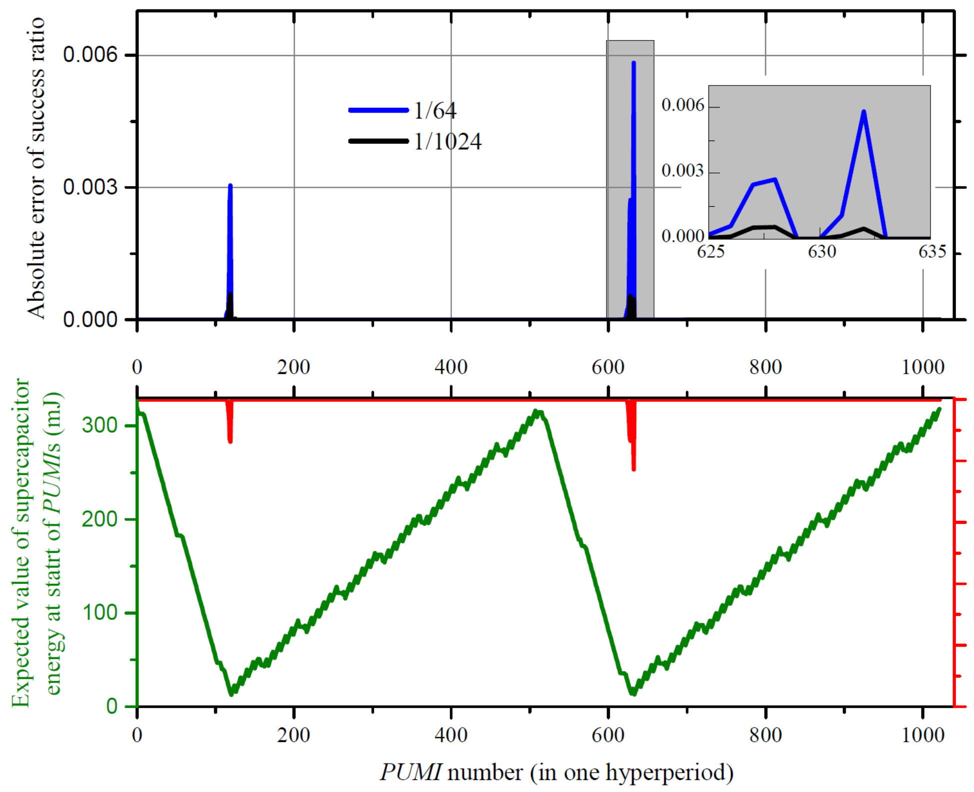

Fig. 9. Absolute errors of success-ratio along with actual success-ratios as well as expected super-capacitor energy at start of *PUMI*s during the hyperperiod.

On the other hand, if the super-capacitor is almost full, the reservoir plays the role of an additional container to conserve energy, while in the analysis the reservoir energy is wasted in the situation of a full super-capacitor. Therefore, in such a situation, the analysis gives pessimistic success-ratios.

6.2. Investigating the Performance of the Proposed Scheduling Algorithm

In this section, we compare the performance of NCC with a number of well-known non-mixed-criticality algorithms to show its performance for tasks with success-ratio constraints in an environment with uncertain energy arrival. Also, we compare MC-SCS with some known mixed-criticality algorithms to experimentally investigate its efficiency for systems with two levels of task criticality.

After validating the analytical method by means of simulation in Section 6.1, we only employ the analytical method in this section. The analytical method gets a schedule as input and returns the (single-segment or multi-segment) job success-ratios as its output, based on the results in Section 4.4. As described in Section 5.3, to have safe calculation of success-ratios despite the variations in the execution times in the multi-segment case of the proposed scheduling algorithm, we use a pessimistic analysis by assuming that all jobs use their maximum energy according to their WCET. For each schedule generated in the steps of the proposed scheduling algorithm, after the analytical calculation of the pessimistic success-ratio of individual jobs, if the success-ratio

Table VII. Comparison between analytical and simulation success-ratio results with different assumptions. U and Δ denote uniform and triangular distribution, respectively.

Task set		Basic assumptions		Variable task power	Non-ideal energy transfer	Alg.	H(.)
		Sim.	Analy-sis	Sim.	Sim.		
A^1	τ_1	0.8619	0.8623	0.8612	0.8573	EDF	$U[2,3]$
	τ_2	1.0000	1.0000	1.0000	1.0000		
	τ_3	0.7560	0.7558	0.7542	0.8089		
A^1	τ_1	0.9126	0.9126	0.9126	0.9052	EDF	$\Delta[2,2.5,3]$
	τ_2	1.0000	1.0000	1.0000	1.0000		
	τ_3	0.7636	0.7635	0.7641	0.8143		
A^2	τ_1	0.0000	0.0000	0.0000	0.0000	EDF	$U[2,3]$
	τ_2	1.0000	1.0000	1.0000	1.0000		
	τ_3	0.1935	0.1935	0.1933	0.2575		
A^2	τ_1	0.0000	0.0000	0.0000	0.0000	EDF	$\Delta[2,2.5,3]$
	τ_2	1.0000	1.0000	1.0000	1.0000		
	τ_3	0.1127	0.1127	0.1133	0.1805		
B	τ_1	1.0000	1.0000	0.9999	1.0000	EDF	$U[2,3]$
	τ_2	0.9999	0.9999	0.9999	0.9999		
	τ_3	0.9861	0.9862	0.9824	0.9862		
C	τ_1	0.9961	0.9961	0.9960	0.9940	EDF	$U[2,3]$
	τ_2	0.9911	0.9912	0.9911	0.9883		
	τ_3	0.0.9942	0.9942	0.9943	0.9987		
D	τ_1	0.9963	0.9962	0.9960	0.9966	EDF	$U[2,3]$
	τ_2	1.0000	1.0000	1.0000	1.0000		
	τ_3	0.9999	0.9999	0.998	0.9998		
E	τ_1	0.9915	0.9914	0.9915	0.9956	EDF	$U[300,700]$
	τ_2	0.9929	0.9929	0.9929	0.9923		
	τ_3	0.9955	0.9955	0.9954	0.9955		
F	τ_1	0.9970	0.9970	0.9968	0.9972	EDF	$U[300,700]$
	τ_2	9921	0.9921	0.9921	0.9944		
	τ_3	0.9902	0.9902	0.9900	0.9908		
F	τ_1	0.9984	0.9984	0.9985	0.9987	EDF	$\Delta[300,500,700]$
	τ_2	0.9930	0.9929	0.9933	0.9944		
	τ_3	0.9918	0.9918	0.9920	0.9919		

is greater than the corresponding success-ratio constraint for all jobs, the schedule is counted as feasible.

We consider an ideal $5m$F super-capacitor, i.e., with no power dissipation, a 0.27mF capacitor as the reservoir, an epoch size of $P = 12$ms, and $H = U(0.5, 1)$mJ.

First, we focus on NCC and compare it with some offline computed static schedules obtained for one hyperperiod using preemptive and non-preemptive versions of EDF, EDL, and rate-monotonic (RM) scheduling. It should be emphasized that the contribution of this study relates to the job success-ratio constraints (not the timing constraints) of real-time tasks, and thus, we only select task systems which are time-feasible by the mentioned scheduling algorithms and report the proportion of schedules which are (energy-) feasible as well. However, to have a better insight into the performance of the proposed algorithm, we compare it to the *best performance* of the six mentioned algorithms (Best-of-Others), namely preemptive and non-preemptive versions of EDF, EDL, and RM; by best performance we mean that in case that each of these algorithms results in a feasible schedule, we count it for Best-of-Others.

The task sets with $m = 3$ tasks are generated for total utilizations of 0.2 to 0.9, with steps of 0.1. We consider that the hyperperiod is $\Pi = 600$ms, $[\pi_{min}, \pi_{max}] = [20, 300]$ms, $Pow_{max} = 616$mW, and success ratio constraints (α_i) are randomly selected from $\{0.95, 0, 96, 0.97, 0.98, 0.99, 1.0\}$. Then, Steps 1 to 5 mentioned in Section 6.1 are used

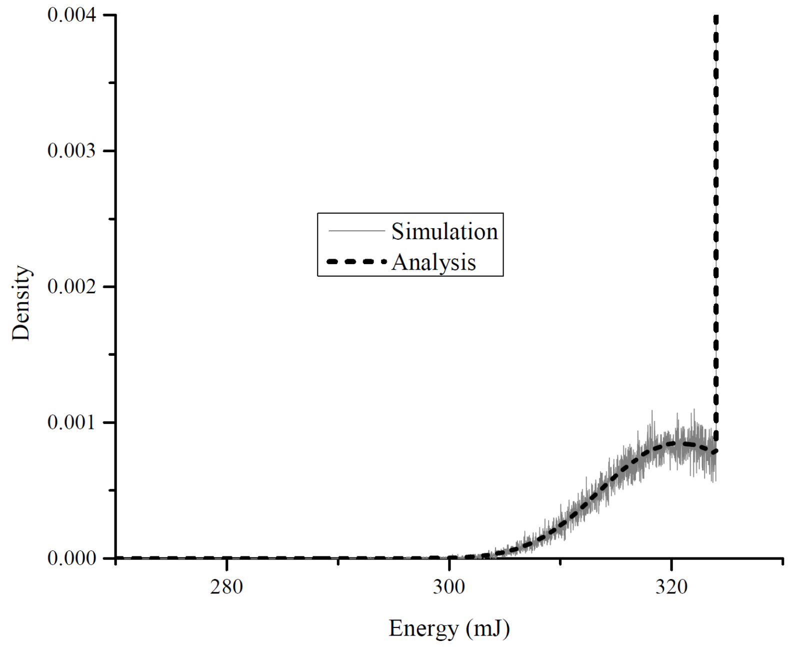

Fig. 10. Comparison between analytical and simulation results of super-capacitor energy PDF at start of hyperperiod for Task set $\mathbf{A}^1$ (EDF and uniform).

to generate the tasks, and the results are averaged over 50 task sets generated for each utilization.

The results are shown in Fig. 11, where UB shows an upper bound of NCC and Best-of-Others in terms of the feasibility ratio, namely when either of them results in a feasible schedule, we count it for UB. As indicated before, we generate the tasks in a way that (39) holds. An outcome of this relation is that, as the harvesting power follows the same pattern, the power consumption of individual tasks gets larger when utilization decreases. Therefore, in the case of low utilization, the time slacks are sufficiently large to easily satisfy timing constraints. However, very small changes in the schedule of jobs can have considerable impact on the energy behavior of the schedule. As a result, when the utilization is low, algorithms like EDF, EDL, and RM which do not necessarily distribute the load equally throughout the hyper-period, may experience some intervals with very high energy need. This means that in some intervals the super-capacitor is almost empty and in other intervals, the harvested energy is wasted. In such a scenario, NCC can work much better, because it distributes the jobs within the available degree of freedom and increases the possibility of a more uniform energy usage pattern during the hyper-period. When the utilization increases, the power usage has a smoother distribution over time during a hyper-period since jobs have longer execution-times and lower power consumptions. Therefore, algorithms like EDF, EDL, and RM might work better, since because of higher utilization, less maneuvers are possible for NCC, and the resulting schedule might remain more similar to the initial EDL from which the maneuvers of NCC starts. The reason that at utilization of 0.9 Best-of-Others works better is that NCC in very high utilization is almost the same as EDL, while other algorithms like EDF and RM might perform better for some task sets.

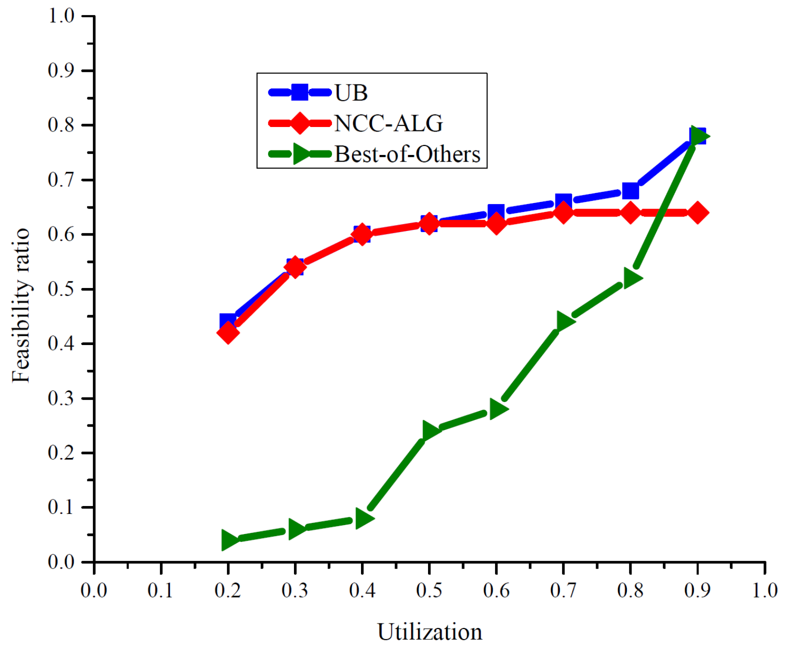

Fig. 11. Comparison between feasibility ratios of NCC and Best-of-Others (among preemptive and non-preemptive versions of EDF, EDL, and RM) for the non-mixed-criticality setups considering $m = 3$.

In the experiments related to the mixed-criticality setup, we focus on MC-SCS and compare it with some offline computed static schedules obtained for one hyperperiod using the well-known mixed-criticality algorithms Adaptive Mixed Criticality (AMC) [Baruah et al. 2011] and EDF-Virtual Deadline (EDF-VD) [Baruah et al. 2012]. AMC is based on fixed-priority scheduling; from the two version of AMC as proposed in [Baruah et al. 2011] we use AMC-rtb for our experiments. EDF-VD introduces the concept of a virtual deadline for HI tasks in the low scheduler mode to ensure schedulability during and after a mode change.

Again it should be emphasized that the contribution of this study relates to meeting the job success-ratio constraints in addition to their timing constraints, and thus, we only select the mixed-criticality task sets which are time-feasible by both, AMC and EDF-VD. Then, we report what proportion of the task sets is energy-feasible by either of the two algorithms (we call it Best-of-Others, with the aforementioned meaning of best performance) and what fraction of the same task sets is feasible after applying MC-SCS (see Fig. 12).

To this end, we still need to decide when a task set scheduled by AMC or EDF-VD is counted as energy-infeasible. First, we recall our previous assumption in this paper, according to which a HI task can only have two execution times $\vec{\epsilon}_i(\mathrm{LO})$ and $\vec{\epsilon}_i(\mathrm{HI})$. Thus, the switching between the low and high scheduler modes can only happen when a HI job just overruns $\vec{\epsilon}_i(\mathrm{LO})$. In this regard, either of the algorithms is counted as having failed to provide a feasible schedule if at least one of the following conditions is not valid: i) The schedule in the high scheduler mode is energy-feasible, ii) the schedule in the low scheduler mode is energy-feasible, iii) the mode switch of the scheduler at all possible switching instants satisfies Rule 1 as provided in Section 5.2.

The considered task sets have $m = 5$ tasks (three HI tasks and two LO ones), which are generated for total utilizations of 0.2 to 0.8 with steps of 0.1. The utilizations more

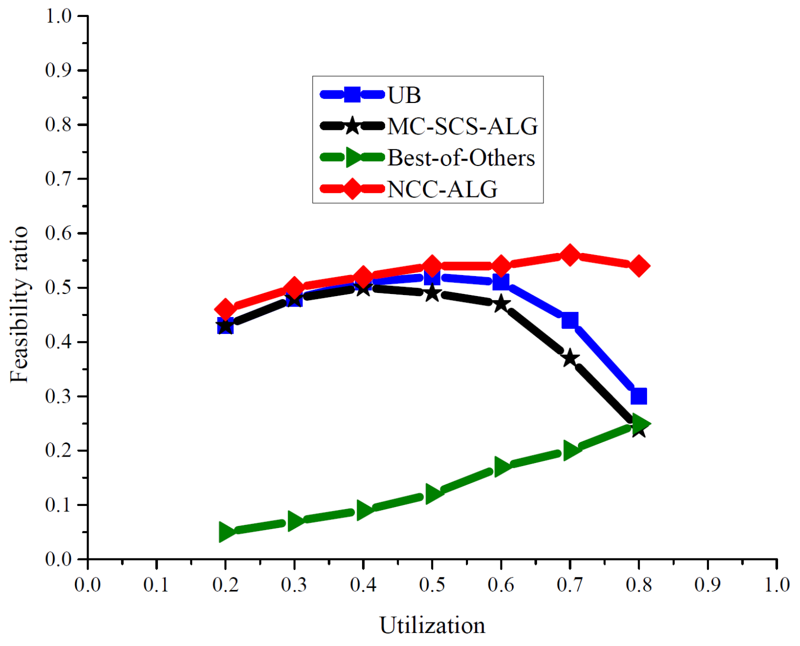

Fig. 12. Comparison between feasibility ratios of MC-SCS and Best-of-Others (between AMC and EDF-VD) for the mixed-criticality setups considering $m = 5$ (three HI tasks and two LO ones).

than 0.8 are not considered because the time-feasibility of AMC and EDF-VD is too low at these utilizations.

Some basic parts of the mixed-criticality setup are the same as in the non-mixed-criticality setup. However, we consider that the utilization of tasks in the low and high scheduler modes are the same (namely, $\mathrm{U_{HI}} = \mathrm{U_{LO}}$, where $\mathrm{U_{HI}} = \sum_{\tau_i|l_i=HI} \frac{\vec{\epsilon}_i(\mathrm{HI})}{\pi_i}$ and $\mathrm{U_{LO}} = \sum_{\tau_i} \frac{\vec{\epsilon}_i(\mathrm{LO})}{\pi_i}$). We define a parameter $\beta = \frac{\mathrm{U_{HI}}}{\mathrm{U_{LO}^{HI}}}$, $\mathrm{U_{LO}^{HI}} = \sum_{\tau_i|l_i=HI} \frac{\vec{\epsilon}_i(\mathrm{LO})}{\pi_i}$, which is used to determine the relation between $\vec{\epsilon}_i(\mathrm{HI})$ and $\vec{\epsilon}_i(\mathrm{LO})$ for HI tasks as $\vec{\epsilon}_i(\mathrm{HI}) = \beta\vec{\epsilon}_i(\mathrm{LO})$. We consider $\beta \geq 1.3$ in the generated task sets and choose $U^e = 1$ for the HI scheduler mode, and $U^e = 0.9$ in the LO scheduler mode.

The results are shown in Fig. 12, where UB shows an upper bound of MC-SCS and Best-of-Others for the feasibility ratios, namely when either of them gives a feasible schedule, we count it for UB. As it can be seen, the feasibility ratio of MC-SCS with respect to the other considered algorithms follows a behavior similar to what is occurred in the non-mixed-criticality setup, which confirms the applicability of the proposed algorithm for mixed-criticality setups.

As the final discussion, we concentrate on the relative behavior of the two phases of MC-SCS, namely, NCC and LSM. Fig. 12 shows the feasibility ratio of NCC for the HI jobs in the high scheduler mode. As can be observed, when LSM tries to add LO jobs to the output schedule of NCC, it decreases the feasibility ratio of NCC for the mixed-criticality setup. As we know, only subsets of task sets which have been feasible by NCC can remain feasible when the LO tasks are added. However, in case of low utilization this degradation of the feasibility ratio is almost ignorable. When the utilization increases, the unsuccessfulness of LSM increases; the reason relates to the size of the slack after job placement by NCC, which restricts the possible maneuvers for LO jobs when the utilization increases.

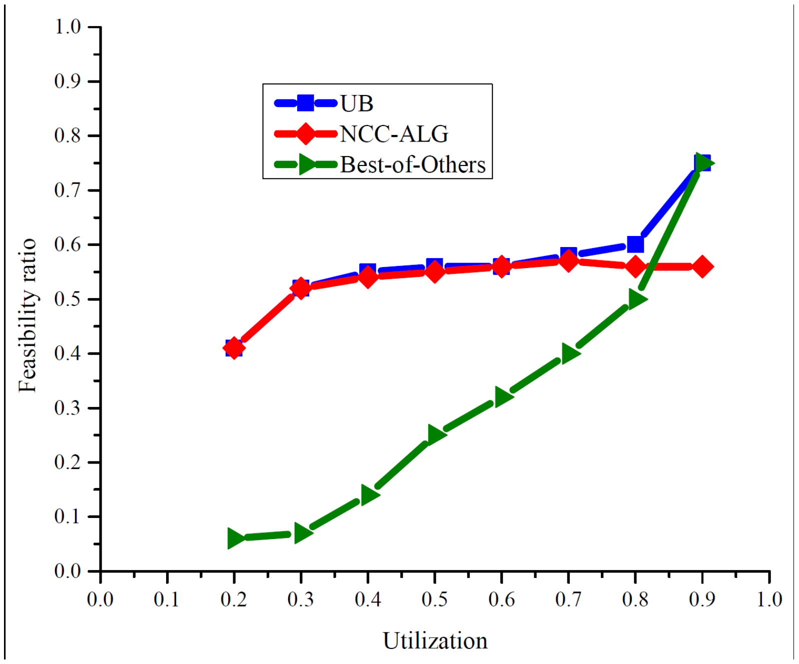

Fig. 13. Comparison between feasibility ratios of NCC and Best-of-Others (among preemptive and non-preemptive versions of EDF, EDL, and RM) for the non-mixed-criticality setups considering $m = 10$.

We repeated the two abovementioned experiments with the same setting except for the number of tasks. More precisely, we have considered $m = 10$ for the non-mixed-criticality setting and $m = 13$ (ten HI tasks and three LO tasks) for the mixed-criticality setting. The obtained results are provided in Figs. 13 and 14. As can be seen in the figures in comparison to Figs. 11 and 12, it is seen that the behavior of the algorithms has no significant change by the modification in the number of tasks.

7. CONCLUSION AND FUTURE WORK

In this paper, we consider mixed-criticality systems with energy harvesters. Because of two sources of uncertainty, namely the mixed-criticality behavior of tasks and uncertainty in the availability of energy, the problem of scheduling tasks is inherently intractable. We first present an analytical method which determines the steady-state success probability of individual jobs from the energy viewpoint. Then, based on the analytical method, we propose a scheduling algorithm which guarantees lower bounds on the job success probabilities. The scheduling algorithm is based on a set of non-trivial insights and properties of real-time systems with success-ratio constraints under energy harvesting. The proposed scheduling algorithm can further be extended to handle non-mixed-criticality as well as mixed-criticality systems with more advanced loss models.

REFERENCES

Yasmina Abdeddaïm, Younès Chandarli, Robert I Davis, and Damien Masson. 2014. Schedulability Analysis for Fixed Priority Real-Time Systems with Energy-Harvesting. In *Proceedings of the 22nd International Conference on Real-Time Networks and Systems*. ACM, 311.

David Audet, Leandro Collares De Oliveira, Neil MacMillan, Dimitri Marinakis, and Kui Wu. 2011. Scheduling recurring tasks in energy harvesting sensors. In *Computer Communications Workshops (INFOCOM WKSHPS), 2011 IEEE Conference on*. IEEE, 277–282.

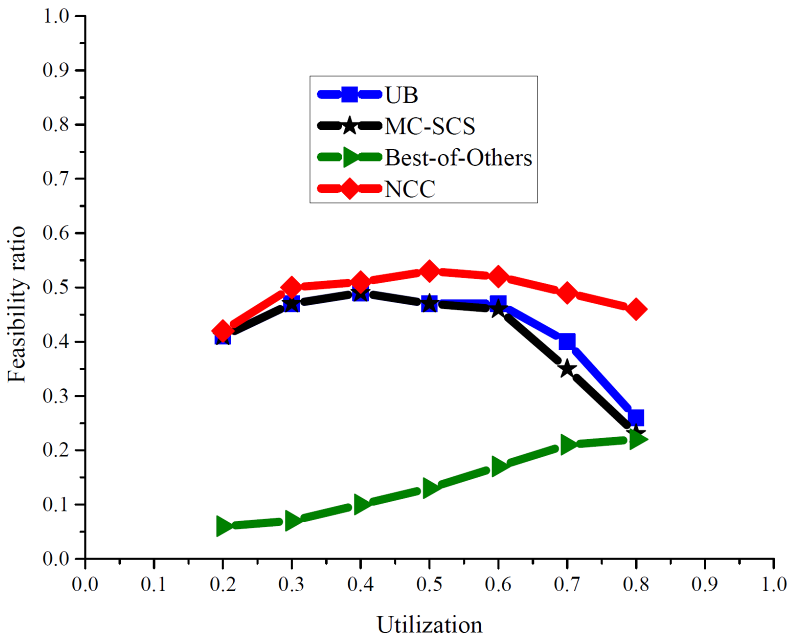

Fig. 14. Comparison between feasibility ratios of MC-SCS and Best-of-Others (between AMC and EDF-VD) for the mixed-criticality setups considering $m = 13$ (ten HI tasks and three LO ones).

Sanjoy Baruah, Vincenzo Bonifaci, Gianlorenzo D'Angelo, Haohan Li, Alberto Marchetti-Spaccamela, Suzanne Van Der Ster, and Leen Stougie. 2012. The preemptive uniprocessor scheduling of mixed-criticality implicit-deadline sporadic task systems. In *Real-Time Systems (ECRTS), 2012 24th Euromicro Conference on*. IEEE, 145–154.

Sanjoy K Baruah, Alan Burns, and Robert I Davis. 2011. Response-time analysis for mixed criticality systems. In *Real-Time Systems Symposium (RTSS), 2011 IEEE 32nd*. IEEE, 34–43.

Enrico Bini and Giorgio C Buttazzo. 2005. Measuring the performance of schedulability tests. *Real-Time Systems* 30, 1-2 (2005), 129–154.

David Brooks, Vivek Tiwari, and Margaret Martonosi. 2000. Wattch: a framework for architectural-level power analysis and optimizations. In *Computer Architecture, 2000. Proceedings of the 27th International Symposium on*. IEEE, 83–94.

Alan S Brown. 2014. POWERING THE INTERNET OF THINGS. *Mechanical Engineering* 136, 3 (2014), 19.

Davide Brunelli, Clemens Moser, Lothar Thiele, and Luca Benini. 2009. Design of a solar-harvesting circuit for batteryless embedded systems. *Circuits and Systems I: Regular Papers, IEEE Transactions on* 56, 11 (2009), 2519–2528.

Doug Burger and Todd M Austin. 1997. The SimpleScalar tool set, version 2.0. *ACM SIGARCH Computer Architecture News* 25, 3 (1997), 13–25.

Alan Burns and Robert Davis. 2015. Mixed criticality systems-a review. *Department of Computer Science, University of York, Tech. Rep* (2015).

Yang Cai and MC Kong. 1996. Nonpreemptive scheduling of periodic tasks in uni-and multiprocessor systems. *Algorithmica* 15, 6 (1996), 572–599.

Ruizhi Chai and Ye Zhang. 2015. A Practical Supercapacitor Model for Power Management in Wireless Sensor Nodes. *Power Electronics, IEEE Transactions on* 30, 12 (2015), 6720–6730.

H Chetto and M Chetto. 1989. Some Results of the Earliest Deadline Scheduling Algorithm. *IEEE Transactions on Software Engineering* 15, 10 (1989), 1261.

Robert I Davis, Tullio Vardanega, Jan Andersson, Francis Vatrinet, Mark Pearce, Ian Broster, Mikel Azkarate-Askasua, Franck Wartel, Liliana Cucu-Grosjean, Mathieu Patte, and others. 2014. PROXIMA: A Probabilistic Approach to the Timing Behaviour of Mixed-Criticality Systems. *Ada User Journal* 35, 2 (2014).

Bruce A Gieseke, Randy L Allmon, Daniel W Bailey, Bradley J Benschneider, Sharon M Britton, John D Clouser, Harry R Fair III, James A Farrell, Michael K Gowan, Christopher L Houghton, and others. 1997. A 600 MHz superscalar RISC microprocessor with out-of-order execution. In *Solid-State Circuits Conference, 1997. Digest of Technical Papers. 43rd ISSCC., 1997 IEEE International*. IEEE, 176–177.

Joel Goossens and Christophe Macq. 2001. Limitation of the hyper-period in real-time periodic task set generation. In *In Proceedings of the RTS Embedded System (RTSâĂŽ01*. Citeseer.

Matthew R Guthaus, Jeffrey S Ringenberg, Dan Ernst, Todd M Austin, Trevor Mudge, and Richard B Brown. 2001. MiBench: A free, commercially representative embedded benchmark suite. In *Workload Characterization, 2001. WWC-4. 2001 IEEE International Workshop on*. IEEE, 3–14.

Wang Song Hao and Ronald Garcia. 2014. Development of a Digital and Battery-Free Smart Flowmeter. *Energies* 7, 6 (2014), 3695–3709.

Liang He, Lipeng Gu, Linghe Kong, Yu Gu, Cong Liu, and Tian He. 2013. Exploring adaptive reconfiguration to optimize energy efficiency in large-scale battery systems. In *Real-Time Systems Symposium (RTSS), 2013 IEEE 34th*. IEEE, 118–127.

Longbo Huang and Michael J Neely. 2013. Utility optimal scheduling in energy-harvesting networks. *IEEE/ACM Transactions on Networking (TON)* 21, 4 (2013), 1117–1130.

Pengcheng Huang, Pranaw Kumar, Georgia Giannopoulou, and Lothar Thiele. 2014a. Energy efficient dvfs scheduling for mixed-criticality systems. In *Embedded Software (EMSOFT), 2014 International Conference on*. IEEE, 1–10.

Pengcheng Huang, Hoeseok Yang, and Lothar Thiele. 2014b. On the scheduling of fault-tolerant mixed-criticality systems. In *Design Automation Conference (DAC), 2014 51st ACM/EDAC/IEEE*. IEEE, 1–6.

K Jeffay, D Stanat, and C Martel. 1991. On non-preemptive scheduling of periodic and sporadic tasks. 12th IEEE Real-Time Systems Symposium. *San Antonio, TX* (1991).

Xi Jin, Fanxin Kong, Peng Zeng, Qingxu Deng, and Huiting Xu. 2014. Joint management of energy harvesting, storage, and usage for green wireless sensor networks. *International Journal of Distributed Sensor Networks* 2014 (2014).

Marijn Jongerden, Boudewijn Haverkort, Henrik Bohnenkamp, and Joost-Pieter Katoen. 2009. Maximizing system lifetime by battery scheduling. In *Dependable Systems & Networks, 2009. DSN'09. IEEE/IFIP International Conference on*. IEEE, 63–72.

Shin-Haeng Kang, Hoeseok Yang, Sungchan Kim, Iuliana Bacivarov, Soonhoi Ha, and Lothar Thiele. 2014. Reliability-aware mapping optimization of multi-core systems with mixed-criticality. In *Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014*. IEEE, 1–4.

CM Krishna. 2011. Managing battery and supercapacitor resources for real-time sporadic workloads. *Embedded Systems Letters, IEEE* 3, 1 (2011), 32–36.

Kenli Li, Xiaoyong Tang, and Keqin Li. 2014. Energy-efficient stochastic task scheduling on heterogeneous computing systems. *Parallel and Distributed Systems, IEEE Transactions on* 25, 11 (2014), 2867–2876.

Juan Liu, Huaiyu Dai, and Wei Chen. 2015. Delay optimal scheduling for energy harvesting based communications. *Selected Areas in Communications, IEEE Journal on* 33, 3 (2015), 452–466.

Azalia Mirhoseini and Farinaz Koushanfar. 2011. HypoEnergy. hybrid supercapacitor-battery power-supply optimization for energy efficiency. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2011*. IEEE, 1–4.

Morteza Mohaqeqi, Mehdi Kargahi, and Ali Movaghar. 2014. Analytical leakage-aware thermal modeling of a real-time system. *Computers, IEEE Transactions on* 63, 6 (2014), 1378–1392.

Clemens Moser, Davide Brunelli, Lothar Thiele, and Luca Benini. 2007. Real-time scheduling for energy harvesting sensor nodes. *Real-Time Systems* 37, 3 (2007), 233–260.

Hiroshi Nishimoto, Yoshihiro Kawahara, and Tohru Asami. 2010. Prototype implementation of ambient RF energy harvesting wireless sensor networks. In *Sensors, 2010 IEEE*. IEEE, 1282–1287.

Athanasios Papoulis. 1990. *Probability & statistics*. Vol. 2. Prentice-Hall Englewood Cliffs.

Joaquin Recas Piorno, Carlo Bergonzini, David Atienza, and Tajana Simunic Rosing. 2010. HOLLOWS: A power-aware task scheduler for energy harvesting sensor nodes. *Journal of Intelligent Material Systems and Structures* 21, 13 (2010), 1317–1335.

Danilo Porcarelli, Davide Brunelli, Michele Magno, and Luca Benini. 2012. A multi-harvester architecture with hybrid storage devices and smart capabilities for low power systems. In *Power electronics, electrical drives, automation and motion (SPEEDAM), 2012 international symposium on*. IEEE, 946–951.

WH Press, SA Teukolsky, WT Vetterling, and BP Flannery. 2002. Numerical recipes in C++: the art of scientific computing by William H. *Press. xxviii* (2002).

Pedram Samadi, Hamed Mohsenian-Rad, Vincent WS Wong, and Robert Schober. 2013. Tackling the load uncertainty challenges for energy consumption scheduling in smart grid. *Smart Grid, IEEE Transactions on* 4, 2 (2013), 1007–1016.

Wencong Su, Jianhui Wang, and Jaehyung Roh. 2014. Stochastic energy scheduling in microgrids with intermittent renewable energy resources. *Smart Grid, IEEE Transactions on* 5, 4 (2014), 1876–1883.

Taufik Taufik, Jameson Thornton, and Mohammad Taufik. 2012. Small-scale wind energy harvesting using piezoelectric converter. In *Power Engineering and Renewable Energy (ICPERE), 2012 International Conference on*. IEEE, 1–5.

Yi Wang, Renhai Chen, Zili Shao, and Tao Li. 2013. SolarTune: Real-time scheduling with load tuning for solar energy powered multicore systems. In *Embedded and Real-Time Computing Systems and Applications (RTCSA), 2013 IEEE 19th International Conference on*. IEEE, 101–110.

Alex S Weddell, Michele Magno, Geoff V Merrett, Davide Brunelli, Bashir M Al-Hashimi, and Luca Benini. 2013. A survey of multi-source energy harvesting systems. In *Proceedings of the Conference on Design, Automation and Test in Europe*. EDA Consortium, 905–908.

Michael Whitaker. 2010. Energy Harvester Produces Power from Local Environment, Eliminating Batteries in Wireless Sensors. *J. Analog Innov* 20 (2010), 1–36.

WindPower Program 2015. Wind statistics and the Weibull distribution. http://www.wind-power-program.com/wind_statistics.htm. (2015). Accessed: 2015-01-01.

Erik Ramsgaard Wognsen, René Rydhof Hansen, and Kim Guldstrand Larsen. 2014. Battery-aware scheduling of mixed criticality systems. In *Leveraging Applications of Formal Methods, Verification and Validation. Specialized Techniques and Applications*. Springer, 208–222.

Hengzhao Yang. 2013. *Task scheduling in supercapacitor based environmentally powered wireless sensor nodes*. Ph.D. Dissertation. Georgia Institute of Technology.

Hengzhao Yang and Ying Zhang. 2013. Analysis of supercapacitor energy loss for power management in environmentally powered wireless sensor nodes. *Power Electronics, IEEE Transactions on* 28, 11 (2013), 5391–5403.

Shenqiu Zhang, Alireza Seyedi, and Biplab Sikdar. 2013. An analytical approach to the design of energy harvesting wireless sensor nodes. *Wireless Communications, IEEE Transactions on* 12, 8 (2013), 4010–4024.

Ting Zhu, Ziguo Zhong, Yu Gu, Tian He, and Zhi-Li Zhang. 2009. Leakage-aware energy synchronization for wireless sensor networks. In *Proceedings of the 7th international conference on Mobile systems, applications, and services*. ACM, 319–332.