Summer 2018

Precise energy efficient scheduling of mixed-criticality tasks & sustainable mixed-criticality scheduling

Sai Sruti

Department:

PRECISE ENERGY EFFICIENT SCHEDULING OF MIXED-CRITICALITY TASKS & SUSTAINABLE MIXED-CRITICALITY SCHEDULING

by

SAI SRUTI

A THESIS

Presented to the Graduate Faculty of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

in

COMPUTER SCIENCE

2018

Approved by:

Zhishan Guo, Advisor
Sanjay Madria
Wei Jiang

ABSTRACT

In this thesis, the imprecise mixed-criticality model (IMC) is extended to precise scheduling of tasks, and integrated with the dynamic voltage and frequency scaling (DVFS) technique to enable energy minimization. The challenge in precise scheduling of MC systems is to simultaneously guarantee the timing correctness for all tasks, HI and LO, under both pessimistic and optimistic (less pessimistic) assumptions. To the best of knowledge this is the first work to address the integration of DVFS energy conserving techniques with precise scheduling of LO-tasks of the MC model.

In this thesis, the utilization based schedulability tests and sufficient conditions for such systems under Earliest Deadline First EDF-VD scheduling policy are presented. Quantitative study in the forms of speedup bound and approximation ratio are also proved for the unified model. Extensive experimental studies are conducted to verify the theoretical results as well as the effectiveness of the proposed algorithm.

In safety- critical systems, it is essential to perform schedulability analysis prior to run-time. Parameters characterizing the run-time workload are generated by pessimistic techniques; hence, adopting conservative estimates may result in systems performing much better than anticipated during run-time. This thesis also addresses the following questions associated to the better performance of the task system: (i) How does parameter change affect the schedulability of a task set(system)? (ii) In the event that a mixed-criticality system design is deemed schedulable and specific part/parts of the system are reassigned to be of low-criticality, is the system still safe to run? (iii) If a system is presumed to be non-schedulable, does it invariably benefit to reduce the criticality of some task?

To answer these questions, in this thesis, we not only study the property of sustainability with regards to criticality levels, but also revisit sustainability of several uniprocessor and multiprocessor scheduling policies with respect to other parameters.

ACKNOWLEDGMENTS

I would like to express my gratitude to all those who have helped me during this research. Dr. Zhishan Guo, my advisor, has been instrumental in my effectiveness as a graduate student. I would like to thank him for giving me the opportunity to work on this project; his valuable insights and suggestions have helped me to overcome many hurdles during this work. I am grateful to him for the advice and guidance he gave me throughout my master's program and for granting me the opportunity to work with truly outstanding researchers in the past two years.

Further, I would like to express my sincerest appreciation to my committee members, Dr. Sanjay Madria and Dr. Wei Jiang, for being part of my thesis committee and for taking time to review this work. I am extremely proud to have spent two years in the Department of Computer Science at Missouri S&T, and appreciate the infinite help from all the faculty members and staff. A special thanks to all my friends in the real-time systems group, from whom I learned a lot. I want to thank my parents and sisters for their constant and unconditional love.

TABLE OF CONTENTS

Page

LIST OF ILLUSTRATIONS

LIST OF TABLES

NOMENCLATURE

AMC	Adaptive Mixed Criticality
ASILs	Automotive Safety and Integrity Levels
CA	Certification Authorities
CM	Criticality Monotonic
COTS	Commercial Off the Shelf
DALs	Design Assurance Levels
DVFS	Dynamic Voltage and Frequency Scaling
EDF	Earliest Deadline First
EDF-VD	Earliest Deadline First with Virtual Deadline
G-EDF	Global Earliest Deadline First
IMC	Imprecise Mixed Criticality
MC	Mixed Criticality
MC-Fluid	Mixed Criticality Fluid
MC2	Mixed Criticality Multi-Core
OCBP	Own Criticality Based Priority
pEDF	Partitioned Earliest Deadline First
RTA	Response Time Analysis

SILs	Safety Integrity Levels
UAV	Unmanned Aerial Vehicles
WCET	Worst Case Execution Time

1. INTRODUCTION

REAL-TIME SYSTEMS: Real-time frameworks are characterized as systems that guarantee temporal correctness. Advanced embedded frameworks broadly associated with physical applications require an output or a response for every input signal within a foreseeable or predictable time-frame. Such real-time applications entail two notions of correctness, (i) logical correctness which ensures that correct results are produced and, (ii) temporal correctness focuses on the generation of results at the right time. In real-time systems, temporal correctness is often achieved by anticipation or predictability of the system's behavior, i.e., in order to guarantee every task's response within the assigned deadline, it is crucial to anticipate the performance of the task's execution time and throughput within a time-frame prior to run-time. Real-time scheduling theory consists of the study of scheduling algorithms constructed for such real-time task models. These algorithms are validated by deriving schedulability tests to guarantee temporal correctness.

In this thesis, we study the scheduling of tasks in mixed-criticality real-time systems while ensuring energy efficiency. This section provides a brief introduction literature for the sections to follow. The motivation will be described in the next section, followed by the problem statement addressed in the thesis, and finally the contribution and organization.

1.1. MIXED CRITICALITY SYSTEMS

There has been an exponential rise in the study of security essential systems with mixed-criticality implementation. In mixed-criticality systems, components attributed with different levels of criticality are facilitated onto a common framework to enable minimization of energy and reduction of resource costs.

Examples of such systems are: UAV (unmanned aerial vehicles), different ASILs (Automotive Safety and Integrity Levels), DALs (Design Assurance Levels or Development Assurance Levels) and SILs (Safety Integrity Levels) are designated in industrial standards such as IEC 61508, DO-178B and DO-178C, DO-254, and ISO 26262) (Esper *et al.*, 2015; Graydon and Bate, 2013; Paulitsch *et al.*, 2015). In case of UAV, the functions are broadly divided into two categories: (i) mission critical which comprises of functions that fall under the surveillance jurisdiction such as capturing or communicating images, and (ii) flight critical which consists of more performance oriented functionalities. The Certification Authorities (CA) impose a mandatory safety requirement validation corresponding to strict conservative safety requirements for flight critical functionalities. The mission critical functions are usually validated by the system designers who do not impose stringent criteria. The difference created in the certification requirement is modeled as varying *Worst-Case Execution Times*. The WCET value thus generated for the flight critical functionalities by the CA is more pessimistic than the WCETs obtained from the system designer.

This model for characterizing such mixed criticality workloads suggested by Vestal (Vestal, 2007) over a decade ago, has been extensively followed by the real-time scheduling network. There have been multiple extensions of this model analyzing the aspects of scheduling and schedulability conditions under various platforms. These scheduling strategies are centered around: (i) guaranteeing resources to all the tasks under less pessimistic behaviors of the system and (ii) protecting HI-criticality tasks under more pessimistic behaviors i.e. in the event of task overrun. On the grounds of a significant rise in interest in mixed-criticality scheduling, it is crucial to investigate the various modulations of the mixed-criticality model and its platforms to segregate the more efficient mixed-criticality scheduling algorithms and their associated schedulability tests on the respective platforms.

CURRENT STATE-OF-THE ART WORK: Most of the current and existing literature focuses on the real-time facet of mixed criticality systems which adopts a popular workload model to specify these systems.

The systems start in the LO-mode where all the tasks are guaranteed execution w.r.t their LO-worst case execution times (WCET). However, in case of a task overrun i.e. if a HI-criticality task exceeds its LO-WCET without signaling completion, the system switches to HI-criticality mode in which the more critical tasks are guaranteed execution dropping the less critical tasks (Baruah *et al.*, 2015; Baruah and Guo, 2014). Sometimes in HI-mode, degraded services are provided to the less critical tasks and the released resources are used to guarantee to meet HI-task deadlines (Burns and Baruah, 2013). In recent times, the imprecise mixed-criticality system (IMC) model is being studied which allows graceful degradation of LO-crtiticality tasks in HI-criticality mode (Baruah *et al.*, 2016; Burns and Baruah, 2013; Liu *et al.*, 2016). It embraces the concept of imprecise computing in which, upon mode-switch: each individual LO-criticality task can execute with inaccuracy in computing which results in relatively short worst-case execution time (WCET), thus saving resources for the more critical tasks.

To date, significant amount of research in mixed criticality systems has focused on the changing speeds of platforms on which MC systems are executed such as unpredictability and varying (deteriorating) speed of Commercial Off-The-Shelf (COTS) processors during runtime (Baruah and Guo, 2014) or intentionally varying the frequency of the processor in order to minimize energy. Progressive analysis is centered on providing heterogeneous temporal order guarantees for tasks of varied criticality levels. This can be achieved by dropping less crucial tasks once crucial tasks overrun. However, with increasing demand of drastically exaggerated computing needs of the typically battery operated nature of platforms on which mixed-criticality systems run, energy reduction for such systems is turning crucial. In fact, this has already been feasible since several modern processors are equipped with the capability of dynamic voltage and frequency scaling (DVFS), where processor frequency is decreased at runtime to save energy.

Huang et.al proposed the integration of dynamic voltage and frequency scaling (DVFS) technique with the earliest deadline first with virtual deadlines (EDF-VD) scheduling scheme for dual-criticality systems (Huang *et al.*, 2014) to enable energy minimization. (Huang *et al.*, 2014) established that increased speeds during overrun conditions are beneficial to minimize expected energy consumption of the system. This model is extended to accommodate multi-core processors, in which a trade-off is determined between both static and dynamic energy consumptions in different operation modes(LO- and HI) (Narayana *et al.*, 2016). Numerous advanced processors are prepared with the capacity of dynamic voltage and frequency scaling (DVFS), where processor frequency can be diminished at runtime to conserve energy as demonstrated by the energy model in (Huang *et al.*, 2014). We adopt an off-line DVFS scheme to diminish energy utilizations in the LO-mode by choosing a minimum speed (≤ 1) for the processor, while protecting mixed-criticality schedulability of the framework.

1.2. SUSTAINABILITY PROPERTY OF SCHEDULING ALGORITHMS

The notion of *sustainability* was introduced (Baruah and Burns, 2006) to characterize and define the likelihood that a system that is schedulable under its worst-case specifications should continue to be schedulable when its actual run-time behavior is better than the worst-case. Given a particular run-time scheduling algorithm and an associated schedulability test, one of the important questions is that *will tighter estimations always lead to better schedulability results?* To better answer such a question, the idea of the *sustainability* property was introduced in 2006 by Baruah and Burns (Baruah and Burns, 2006) for real-time schedulers: "A scheduler is *sustainable* if any task set is schedulable under the most pessimistic specifications under certain schedulability test, it will continue to be schedulable when its performance is improved (e.g., less pessimistic)."

On the grounds of the significant rise in interest in multi-criticality systems, another parameter is considered: criticality levels and consequently the WCETs respective to each criticality level; for instance, if the criticality level of several jobs is changed from HI- to LO-criticaliy, will the schedulability test still hold? Sustainability corresponding to criticality levels is an introspectively intriguing and potentially relevant question, and is examined it in some detail in this thesis.

1.3. CONTRIBUTION AND ORGANIZATION

As discussed above, significant work on MC scheduling has been done which individually considers some level of precision in scheduling LO-criticality tasks in pessimistic conditions and speed scaling in order to minimize energy during run-time. Our work addresses the need to save energy in platforms supporting mixed criticality applications. The idea behind the work is to entertain both accuracy and energy efficiency in real-time system applications. In this thesis, the goal is to integrate the precision model in (Pathan, 2017) to guarantee precise computing to *all* LO-criticality tasks in HI-mode and the varying-speed model in (Huang *et al.*, 2014) where we introduce an energy conserving speed or optimal minimum speed for the processor in LO-mode. The unified model is used to schedule implicit-deadline sporadic tasks by the well known EDF-VD scheduling policy (Baruah *et al.*, 2011b). The main contributions of this thesis are:

- This thesis explores the aggregation of mixed-criticality scheduling with design options of recent computing platforms, i.e. combining precise computing of LO-criticality tasks on varying-speed processors.

- We present conditions to derive the minimum speed for the processor to execute in LO-mode, while correctly scheduling all the tasks in each mode of operation.

- We propose a sufficient test for our precise-energy conserving model under EDF-VD (see Theorem 3 in Section 4.3.1) and prove a quantitative speedup bound and approximation ration on the worst-case performance of EDF-VD.

- We also show that no non-clairvoyant algorithm can guarantee to always meet all deadlines on a processor that is less than 2 times as fast as the processor available to the optimal clairvoyant algorithm, thereby proving that EDF-VD is an optimal non-clairvoyant algorithm from the perspective of this metric.

- We defined another metric named approximation ratio, which compares the minimum possible degraded processor speed without speeding up upon the mode switch (unlike what is done in speedup bounds), and proved the relationship between the approximation ratio of our algorithm and the per-level utilizations of the input task.

- Experimental studies are conducted based on randomly generated synthetic tasks, which verifies the theoretical findings as well as effectivenesses of the proposal algorithm.

- Sustainability analysis is performed upon numerous popular and frequently used uniprocessor and multiprocessor mixed-criticality scheduling policies.

ORGANIZATION: The rest of the thesis is organized as follows: The next section(Section 2) consists of literature of previous works. In Section 3 the adopted model is elaborated in detail comprising of system behavior, varying-speed processors and correctness specification. Section 4 consists of the literature of the EDF-VD scheduling algorithm and the revised schedulability conditions pertaining to the chosen model. In Section 4.3.1 and 4.3.2 of Section 4, the speedup bound and approximation ratio metrics of the modified EDF-VD algorithm proposed are determined which is followed by the performance evaluation with experiments.

Section 5 comprises of detailed sustainability analysis of EDF-VD and several other MC-scheduling policies on both uni-core and multi-core platforms. Section 6 consists of a summary of the thesis.

2. LITERATURE REVIEW

Significant work has been done on various versions of the MC model proposed by Vestal (Vestal, 2007). A thorough review of the various adaptations are reviewed in the survey by Burns et. al (Burns and Davis, 2017). Several of these existing works adopt a stringent approach of dropping the LO-criticality jobs in HI-mode (Baruah *et al.*, 2015; Baruah and Guo, 2014; Easwaran, 2013; Ekberg and Yi, 2014). (Burns and Baruah, 2013) was the first literature to address this issue by assigning time budgets to LO-priority tasks by switching their priority or degrading their services by extending the periods in HI-mode (Huang *et al.*, 2015; Jan *et al.*, 2013). These techniques however have minor setbacks and are not practical (Ernst and Di Natale, 2016). Another approach is presented in (Burns and Baruah, 2013) popularly known as the IMC model, where execution time of LO-tasks is diminished in the event of a mode-switch. The schedulability analysis of the IMC model has been studied for both fixed-priority scheduling and EDF-VD in (Burns and Baruah, 2013) and (Liu *et al.*, 2016) respectively. Minimizing energy utilization has also become a rising concern in mixed-critical applications. (Huang *et al.*, 2014) exploits the DVFS technique to address energy minimization issue in mixed-criticality systems during overrun by speeding up the system but LO-tasks are however penalized in HI-criticality mode. (Huang *et al.*, 2014) also established that increased speeds during overrun conditions are beneficial to minimize expected energy consumption of the system. This model was extended to accommodate multi-core processors, in which a trade-off is determined between both static and dynamic energy consumptions in different operation modes(LO- and HI) (Narayana *et al.*, 2016).

2.1. LIMITATION OF TRADITIONAL MODELS

All of the above works (Baruah *et al.*, 2015; Baruah and Guo, 2014; Huang *et al.*, 2014; Narayana *et al.*, 2016), consider a stringent model in which all the LO-tasks are dropped upon mode switch. Such handling of LO-criticality tasks is argumentative as in HI-mode, all LO-criticality tasks are penalized and can result in failures in timing assumptions in HI-criticality tasks (Burns and Davis, 2017; Ernst and Di Natale, 2016). (Burns and Baruah, 2013) exploits the elastic task model in (Su and Zhu, 2013) where LO-criticality tasks continue to execute with extended time-periods. This model generates accurate but delayed execution results. To ensure sufficient safety and performance features, imprecise computing was introduced in mixed-criticality systems in (Baruah *et al.*, 2016; Burns and Baruah, 2013; Liu *et al.*, 2016) in which each LO-critical task is also guaranteed to some (degraded) service after the system switches to the HI-critical behavior. However, (Ernst and Di Natale, 2016) argues that the period and priority of a task are functional requirements and cannot be altered easily, while degrading services for the execution of LO-criticality tasks can result in performance or service loss. (Pathan, 2017) observed that in case of utilization slack during execution of HI-criticality tasks in HI-mode, all the LO-criticality tasks need not be penalized with degraded service. Considering this model, implicit-deadline IMC sporadic tasks were scheduled, in which some (if not all) LO-criticality tasks were provided full service during the HI-critical behaviors as well.

2.2. SUSTAINABLE SCHEDULING

The formal abstraction and definition of sustainability was introduced by Baruah and Burns (Baruah and Burns, 2006; Burns and Baruah, 2008). Earlier work in this domain centered around sustainability analysis of periodic and sporadic (non-MC) task systems in uniprocessors, and established that several notable schedulability tests in preemptive uniprocessor scheduling are not sustainable.

(Baker and Baruah, 2009) comprises of detailed study if the sustainability attribute of global scheduling algorithms that utilize sporadic task model such as EDF, Earliest-Deadline with Zero-Laxity and fixed priority scheduling. The sustainability property of these scheduling policies are cross-examined against numerous parameters such as depreciated execution time, postponed arrivals, and deadline relaxations.

3. SYSTEM MODEL

In this section, the system or mixed critical real-time workload model is introduced. The considered MC workload comprises of an implicit-deadline sporadic task model where each task set τ includes n tasks that are scheduled on a preemptive uniprocessor. Every task $\tau_i \in \tau$ may generate an unbounded number of MC jobs, with successive jobs being released at least T_i time units apart. Without loss of generality, we assume that all tasks in τ start at time 0.

MC INSTANCE: In this paper, we restrict our attention to dual-criticality task systems where the system has two criticality levels, $\chi_i \in \{\text{LO}, \text{HI}\}$. Each task $\tau_i \in \tau$ is characterized by 5-tuples = $\{T_i, D_i, \chi_i, C_i^L, C_i^H\}$, where T_i represents the minimum inter-arrival time between any two consecutive job releases (by the same task), $C_i^L, C_i^H \in \mathbb{R}_+$ are the WCET estimations, D_i specifies the deadline, and $\chi_i \in \{\text{LO}, \text{HI}\}$ characterizes the criticality level. Due to pessimistic impositions on assurance for HI-criticality tasks, for our model, we assume that $0 < C_i^L \leq C_i^H \leq T_i$. Every task $\tau_i \in \tau$ may generate an unbounded number of MC jobs, with successive jobs being released at least T_i time units apart.

A job prototypes a single piece of code, to be carried out to completion in a consecutive manner upon a processor. An *MC job* J_i is described by a 5-tuple of parameters $(a_i, c_i^L, c_i^H, d_i, \chi_i)$, where

- $a_i \geq 0$ denotes its release time (after which the piece of code may start to execute),
- $c_i^L \leq c_i^H \in \mathbb{R}_+^L$ are per-mode WCET estimations,
- $d_i \geq a_i$ represents the deadline, and
- $\chi_i \in \{\text{LO}, \text{HI}\}$ indicates the criticality level.

The utilizations in all modes of operation of each task τ_i and the whole task set τ are determined as follows:

$$\forall \tau_i \in \tau, u_i^{\text{LO}} = \frac{C_i^{\text{LO}}}{T_i};$$

$$\forall \tau_i \in \tau, u_i^{\text{HI}} = \frac{C_i^{\text{HI}}}{T_i}.$$

The total utilization for each mode of operation is represented as follows:

- The total utilization for all LO-criticality tasks in LO- and HI-modes respectively, we have: $U_{\text{LO}}^{\text{LO}} = \sum_{\forall \tau_i \in \tau_{\text{LO}}} u_i^{\text{LO}}$, $U_{\text{LO}}^{\text{HI}} = \sum_{\forall \tau_i \in \tau_{\text{HI}}} u_i^{\text{LO}}$.
- Similarly for all HI-criticality tasks, the utilization for HI-and LO-criticality tasks is represented as: $U_{\text{HI}}^{\text{LO}} = \sum_{\forall \tau_i \in \tau_{\text{LO}}} u_i^{\text{HI}}$, $U_{\text{HI}}^{\text{HI}} = \sum_{\forall \tau_i \in \tau_{\text{HI}}} u_i^{\text{HI}}$.

Since we do not degrade services for LO-criticality tasks in HI-mode, their utilization in both modes of operation remains the same, i.e. $U_{\text{LO}}^{\text{LO}} = U_{\text{LO}}^{\text{HI}}$.

3.1. VARYING-SPEED PROCESSOR AND DVFS

Dynamic voltage and frequency scaling (DVFS) technique is customarily used to moderate energy by investigating free slacks within the framework to slow the processor down (Benini *et al.*, 1999). Conversely, DVFS technique can also employed to speed up the processor in the event of urgency caused by task overrun, to guarantee that all tasks still meet their deadlines (Huang *et al.*, 2014). DVFS technique by speeding up the processor has been applied in the service degradation model, in which less critical tasks are executed in HI-criticality mode, in order to enhance degraded services for LO-criticality tasks (Huang *et al.*, 2015). Processors today are manufactured with several advanced features, one of them being the capability of DVFS, where processor frequency can be diminished or boosted at runtime to conserve energy (Huang *et al.*, 2014).

We examine off-line DVFS to diminish energy utilizations in the LO-mode by choosing a minimum speed for the processor, while protecting mixed-criticality schedulability of the framework. The processor is characterized by a normal speed s (without loss of generality, $s \leftarrow 1$) and an energy-conserving speed ρ (where $0.5 \leq \rho \leq 1$). During LO-mode, the processor is assumed to exhibit *energy conserving behavior* where its speed remains ρ. In the event of overrun when the system switches to HI-mode, the processor exhibits *normal behavior* where the speed of the processor is maximized (s $\leftarrow$ 1).

3.2. SYSTEM BEHAVIOR

The behavioral semantics of the MC workload is as follows: a job released by task T_i may first execute for its LO-WCET (C_i^L units of time). If all jobs indicate completion at after executing for their LO-WCETs, the system is claimed to perform in LO-mode, else a system-wide mode switch is triggered and the system exhibits HI-criticality behavior. Analogous to the traditional MC task-model behavior, instead of discarding all LO-criticality tasks in HI-mode, our model *guarantees execution time to all tasks* according to their given WCETs during *both* modes of operation. We incorporate the DVFS technique to impose a minimum energy-conserving speed in the LO-mode. Upon mode-switch, the processor continues to perform at normal speed (of 1). Since we do not know statically how long the system will overrun, we can safely assume that the system can recover when the processor is idle, i.e., when all arrived/active workloads are finished.

3.3. SPEEDUP AND APPROXIMATION RATIO METRICS

The theoretical real-time systems community considers speedup as a viable measure to measure the effectiveness of mixed-criticality scheduling algorithms.

(Baruah and Agarwal, 2018) lists the sources of intractability for mixed criticality systems, isolates their impacts and highlights the need for metrics that are able to independently evaluate the approximation-ratio impact (i.e., comparison with MC-optimality) and the competitive-ratio impact (i.e., the sub-optimality emerging from the need of clairvoyance) of EDF-VD in planning recurrent frameworks.

3.3.1. Speedup. A resource augmentation bound called speedup bound is used as a conventional way to characterize the worst-case performance of mixed criticality scheduling algorithms and provide relevant insight about the algorithm's properties.

Definition 1: (Kalyanasundaram and Pruhs, 2000) *An algorithm A is defined to have a speedup bound of* $s \geq 1$ *if any task system* τ *that can be correctly scheduled upon a processor with energy-conserving speed* ρ *and normal speed* 1 *by any hypothetical clairvoyant scheduling algorithm, can be correctly scheduled upon a processor with speed s and energy-conserving speed* $\rho \times s$ *by algorithm A.*

The closer the speedup bound is to 1 (which means the scheduler is optimal), the better.

3.3.2. Approximation Ratio. Approximation ratio provides an insight in the computational tractability of the mixed criticality schedulability analysis. An algorithm A has an approximation ratio of $\alpha \geq 1$ if and only if some non clairvoyant on-line algorithm can guarantee MC correctness with full speed of 1 and a energy conserving speed of ρ, Algorithm A guarantees MC correctness to the same set with a processor of normal speed 1 and energy conserving speed of $\alpha \times \rho$.

3.4. SUSTAINABILITY PROPERTY OF MIXED CRITICALITY SCHEDULING ALGORITHMS

Any algorithm that establishes an MC instance as MC-schedulable is termed as sustainable if and only if the MC instance continues to be schedulable by the algorithm if a single or all of the parameters representing the MC instance is improved.

Correspondingly, a schedulability test for some MC scheduling policy is said termed as sustainable if any MC instance that is established to be MC-schedulable by the schedulability test will continue to be MC-schedulable by the test if a single or all of the parameters representing the MC instance is improved. During run-time the system tends to perform better when the parameters of one or more individual job(s) are changed in any, some, or all of the following ways:

1. Diminishing WCET of one or more individual tasks or jobs (C_i^L and/or C_i^H).
2. Elongating periods for sporadic task systems; prolonging release times(i.e., decreasing a_i) for a single or a set of jobs.
3. Extending relative deadlines of one or more tasks (D_i).
4. Lowering the criticality level assigned (χ_i) to a task/job (from HI to LO in case of dual-criticality systems).

3.5. THESIS PROBLEM STATEMENT

It is widely understood that it should be rare for any task to exhibit HI-criticality behavior; i.e., not signaling completion after executing for a length of LO-WCET. As a result, more attention should be placed on LO-mode in terms of energy consumption. In this paper, *we seek to reduce energy utilizations in the* LO*-mode by minimizing the energy-conserving speed ρ for the processor, while protecting mixed-criticality correctness of the system.*
ASSUMPTION: The relationship between the speed and worst-case execution time (WCET) of the task is considered to be linear, e.g., half speed will lead to double execution time. I.e., a task with LO-WCET of C_i (on a speed-1 processor) would take C_i/ρ time units to finish its execution on a processor of degraded speed ρ such that$(0.5 \leq \rho \leq 1)$.

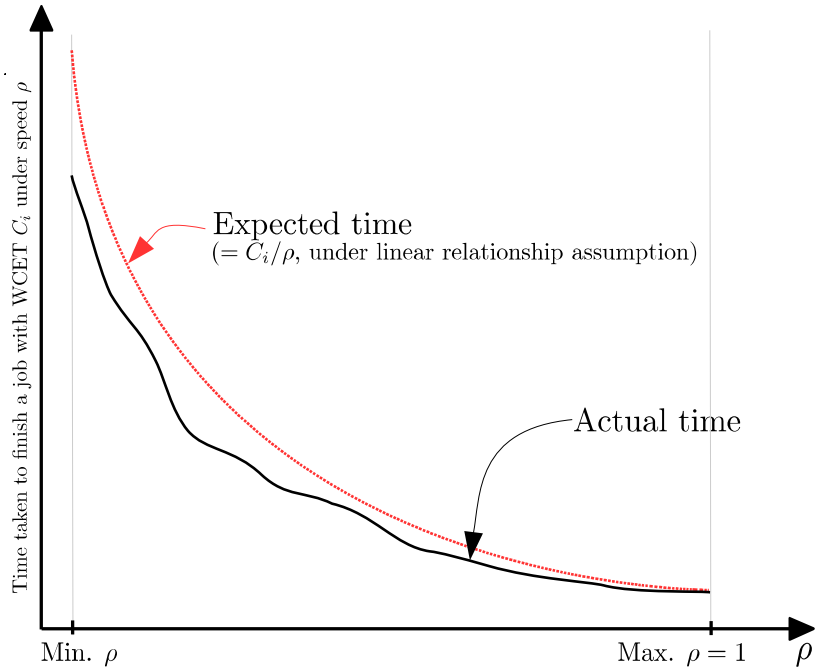


Figure 3.1. Relationship between variable-speed and execution time, where the expected WCET always caps the actual execution time. Here ρ represents the energy conserving speed.

In practice, this assumption should always hold since the speed of cache and memory access, I/O bus etc. are not extensively affected by a decrease in processor speed. We believe it is *safe* in terms of schedulability to consider a linear relationship, as illustrated in Figure 3.1. The actual C_i is a considerably accurate portrayal of the relationship between processor speed and execution time. We are considering a pessimistic upper bound on C_i (expected C_i from Figure 3.1) by assuming a linear relationship.

We present a model that integrates precise scheduling of LO-criticality tasks and energy-minimization using DVFS techniques. The *correctness* of the system is mode based, defined as follows:

- During all LO-criticality behaviors of the system, the processor is down-scaled by the energy-conserving speed ρ. All jobs receive up to their LO-WCET and meet their deadlines.

- In HI-criticality mode, the processor speed increases to 1, where all jobs may receive computation time up to their HI-WCET and meet their deadlines.

Another problem tackled in this thesis comprises of the detailed sustainability analysis of mixed-criticality scheduling algorithms. A scheduling policy and/or schedulability test may be sustainable with respect to some but not all parameters. In Section 5 of the thesis, we will consider several well-known MC schedulers, and examine the sustainability with respect to all four parameters. The algorithms studies are frequently used and posses a variety of schedulability tests. This lets us throw light on fixed priority based, utilization and response time analysis based schedulability tests among several others. Table 3.1 lists the algorithms studies and their sustainability properties with respect to various parameters which are elaborately discussed in Section 5. In the table, summary of sustainability results for some MC scheduling algorithms with respect to various parameters. A 'Y' / 'N', denotes that the scheduler is / is not sustainable with respect to that parameter.(The first four listed algorithms are uniprocessor algorithms; the remaining two, multiprocessor ones.)

Table 3.1. Summary of sustainability results for some MC scheduling algorithms with respect to various parameters.

SCHEDULER	CRIT. LEVEL	WCET	PERIOD	DEADLINE
Crit. Mono.	N	Y	Y	Y
EDF-VD	Y	Y	Y	Y
AMC	Y	Y	Y	Y
OCBP	Y	Y	Y*	Y
MC^2	N	Y	Y	Y
MC Fluid	Y	Y	Y	Y

*Please note that OCBP is for scheduling MC job sets and thus the 'Y' in the period column represents sustainability over release time, while others are for MC task set scheduling.

4. EDF-VD AND ITS CORRECTNESS

As discussed earlier, the motivation of the thesis to integrate the precision model in (Pathan, 2017) with the energy conserving DVFS implemented in (Huang *et al.*, 2014) to guarantee computation time to *all* tasks (HI and LO) in both modes of operation with an energy conserving speed or optimal minimum speed for the processor in LO-criticality mode. The well-known EDF-VD scheduling algorithm (Baruah *et al.*, 2011b) is extended to incorporate the proposed unified model. In this section, we first present how our modified EDF-VD algorithm for the precise energy efficient model (Section 4.2) varies from the EDF-VD algorithm for traditional MC system model in (Baruah *et al.*, 2011b). A description of how the modified EDF-VD algorithm works and then proof of its correctness in both modes of operation is included thereafter with detailed representation of the schedulability tests. Conclusive analysis of the speedup and approximation ratio are presented in Section 4.3.1 and 4.3.2 respectively in this section.

4.1. AN OVERVIEW OF ALGORITHM EDF-VD

Traditional EDF-VD SCHEDULING (Baruah *et al.*, 2011b) is an modification of the Earliest Deadline First (EDF) scheduling algorithm in order to adapt to the dual-criticality implicit-deadline sporadic task systems on a unit-speed processor. In order to guarantee that HI-criticality tasks can still meet their deadlines in situations of overrun, resources have to be retained for those tasks. In the EDF-VD algorithm, the reservation of resources for HI-criticality tasks is accomplished in the LO-mode by artificially scaling down the deadlines of HI-criticality tasks. Such scaled virtual deadline settings enables HI-criticality tasks to budget enough resources to handle overrun while finishing their LO-WCET within the LO-mode. In order to address the resource demands on different levels of criticality:

For a dual-criticality task-set $\tau = \{\tau_1, \tau_2, ..., \tau_n\}$ to be scheduled by energy conserving preemptive processor:

- Scaling factor x is computed to determine virtual deadline of HI-criticality tasks:

$$x \leftarrow \frac{U_{\text{HI}}^{\text{LO}}}{\rho - U_{\text{LO}}^{\text{LO}}}$$

- If $(x-1)U_{\text{LO}}^{\text{LO}} + U_{\text{HI}}^{\text{HI}} + x \leq 1$
  then virtual-deadline $\widehat{T_i} \leftarrow xT_i$ for every HI-criticality task τ_i.

Figure 4.1. Modified EDF-VD schedulability condition and scaling factor x

– In LO-mode, the EDF-VD algorithm adjusts the deadlines of all the HI-criticality tasks by a common factor x. This is done to retain budget for the HI-criticality tasks in HI-mode by triggering an earlier mode-switch. Here, all the deadlines of HI-tasks are scaled down by x to obtain virtual deadlines.

– In HI-mode, the HI-criticality tasks are scheduled according to their original deadlines and all LO-criticality tasks are discarded.

4.2. EDF-VD FOR PRECISE ENERGY-CONSERVING MODEL

In this section, we describe in detail the enhanced EDF-VD algorithm to compromise our precise energy-conserving model. As discussed earlier, we consider the DVFS technique to conserve energy in LO-mode by slowing down the processor to speed ρ. Figure 4.1 represents a *modified* EDF-VD algorithm which determines if the task-set τ is schedulable and then assigns virtual deadline $\widehat{T_i}$ for all HI-criticality tasks of the schedulable task-set. According to the algorithm in Figure 4.1, the scaling factor x is computed and $\widehat{T_i}$ values are assigned to all HI-criticality tasks as $\widehat{T_i} \leftarrow x \times T_i$. Theorem 1 demonstrates how parameter x

is derived. Contradictory to the traditional MC model, instead of discarding all LO-criticality tasks in HI-criticality behaviors, our model schedules both LO- and HI-criticality tasks with their given WCETs at processor speed $s \leftarrow 1$.

4.2.1. Correctness Under LO-Criticality Mode. In this mode, LO-and HI-criticality tasks are guaranteed time budgets equal to their LO-WCET values within their deadlines and virtual deadlines respectively at processor speed ρ. For all tasks being scheduled in LO-mode, we establish the following theorem.

Theorem 1. *The following condition is sufficient for guaranteeing that EDF-VD correctly schedules all the assignments in* LO*-criticality mode:*

$$x \geq \frac{U_{\text{HI}}^{\text{LO}}}{\rho - U_{\text{LO}}^{\text{LO}}} \tag{4.1}$$

where $U_{\text{LO}}^{\text{LO}} < \rho$.

Proof. According to the EDF-VD algorithm, the virtual deadlines of all the HI-criticality tasks for our model are determined where $\widehat{T_i} = xT_i$ prior to runtime. Scaling down the period of each HI-criticality task by parameter x indicates increase in its utilization by x. If all the jobs execute for no more than LO-WCETs (C_i^L), the density bound of EDF for implicit-deadline tasks which is equal to processor capacity (Liu and Layland, 1973) we can therefore conclude that:

$$U_{\text{LO}}^{\text{LO}} + \frac{U_{\text{HI}}^{\text{LO}}}{x} \leq \rho$$

$$\implies x \geq \frac{U_{\text{HI}}^{\text{LO}}}{\rho - U_{\text{LO}}^{\text{LO}}}$$

is the sufficient for guaranteeing that EDF-VD correctly schedules all the assignments in LO-criticality mode. □

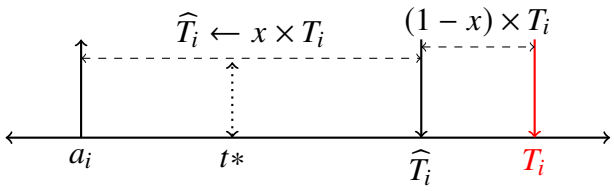


Figure 4.2. Relation between time instants

The smallest value of x such that Theorem 1 is satisfied is assigned by the EDF-VD algorithm:

$$x \leftarrow \frac{U_{\text{HI}}^{\text{LO}}}{\rho - U_{\text{LO}}^{\text{LO}}} \tag{4.2}$$

We now derive a sufficient condition to ensure that EDF-VD meets all deadlines in HI-criticality mode using obtained value of parameter x.

4.2.2. Correctness Under HI-Criticality Mode. Similar to the classical model, if a task τ_i does not signal completion after C_i^L units of execution within its virtual deadline equal to $\widehat{T}_i$, the system exhibits HI-criticality behaviors and triggers a mode-switch. At a specific time instant t* during run-time, in the event that the scheduler identifies a HI-criticality task executing for a duration greater that its C_i^L without indicating completion, mode-switch corresponding to the whole system is activated which advocates a need to perform the following:

- re-assignment of time period T_i of active HI-criticality tasks from $\widehat{T}_i$ ($xT_i = \widehat{T}_i$) to T_i.
- continue to execute LO-criticality tasks and not discard them as in the traditional MC-model.
- the speed of the processor increases from ρ to 1.

Theorem 2. *The following condition is sufficient for guaranteeing that EDF-VD correctly schedules all the assignments in* HI*-criticality mode:*

$$U_{\text{LO}}^{\text{LO}} + \frac{U_{\text{HI}}^{\text{HI}}}{(1-x)} \leq 1 \tag{4.3}$$

Proof. If there is an active HI-criticality task at mode-switch instant t*, the relative deadline of the HI-criticality task is adjusted to $\widehat{T_i} = xT_i$. The actual deadline is $T_i - xT_i = x(1 - T_i)$ time units in the future. The relation is illustrated in Figure 4.2. Thus the utilization of HI-criticality tasks after time instant t* is upper bounded by $\frac{C_i^H}{(1-x)T_i}$.

Summing over all HI- and LO-criticality tasks according to the fact that EDF has a utilization bound equal to the processor capacity (which in our case is 1) we arrive at a conclusion that:

$$\sum_{\tau_i \in \chi = LO} \frac{C_i^{\text{LO}}}{T_i} + \sum_{\tau_i \in \chi = HI} \frac{C_i^{\text{HI}}}{(1-x)T_i} \leq 1$$
$$\Rightarrow U_{\text{LO}}^{\text{LO}} + \frac{U_{\text{HI}}^{\text{HI}}}{(1-x)} \leq 1$$

is a sufficient condition for guaranteeing that EDF-VD correctly schedules all the assignments in HI-criticality mode. □

In order to guarantee schedulability in HI-criticality mode, the upper bound of scaling parameter x is determined below.

$$\Rightarrow (1-x)U_{\text{LO}}^{\text{LO}} + U_{\text{HI}}^{\text{HI}} \leq 1 - x$$
$$\Rightarrow U_{\text{LO}}^{\text{LO}} - xU_{\text{LO}}^{\text{LO}} + U_{\text{HI}}^{\text{HI}} \leq 1 - x$$
$$\Rightarrow x(1 - U_{\text{LO}}^{\text{LO}}) \leq 1 - (U_{\text{HI}}^{\text{HI}} + U_{\text{LO}}^{\text{LO}})$$

The upper bound of scaling parameter x is:

$$x \leq \frac{1 - (U_{\text{HI}}^{\text{HI}} + U_{\text{LO}}^{\text{LO}})}{(1 - U_{\text{LO}}^{\text{LO}})} \tag{4.4}$$

We have thus justified the correctness of the EDF-VD scheduling algorithm. From Theorem 1, the value of x ensures the correctness of all LO-criticality behaviors and Theorem 2 guarantees correctness of all HI-criticality behaviors. We give the following sufficient condition for MC-schedulability by EDF-VD for our precise energy conserving model.

Theorem 3. *If τ satisfies*

$$U_{\text{LO}}^{\text{LO}} + min\left(U_{\text{HI}}^{\text{HI}}, \frac{U_{\text{HI}}^{\text{LO}}}{\left(1 - \frac{U_{\text{HI}}^{\text{HI}}}{1 - U_{\text{LO}}^{\text{LO}}}\right)}\right) \leq \rho \tag{4.5}$$

then it is schedulable by EDF-VD.

Proof. We consider two cases:

Case A: $U_{\text{LO}}^{\text{LO}} + U_{\text{HI}}^{\text{HI}} \leq \rho$, In this case, all the LO- and HI-criticality tasks are considered performing on processor with speed ρ which is worst-case reservation schedulable by EDF (Baruah *et al.*, 2012a). The total utilization of the tasks can be represented as:

$$\frac{U_{\text{LO}}^{\text{LO}}}{\rho} + \frac{U_{\text{HI}}^{\text{HI}}}{\rho} \leq 1$$

The task set can be scheduled by EDF without deadline scaling for HI-criticality tasks at an energy conserving speed.

Case B: $U_{\text{LO}}^{\text{LO}} + U_{\text{HI}}^{\text{HI}} \geq \rho$

For Condition (4.5) to hold, it must be the case that,

$$
\begin{aligned}
& U_{\text{LO}}^{\text{LO}} + \frac{U_{\text{HI}}^{\text{LO}}}{(1 - \frac{U_{\text{HI}}^{\text{HI}}}{1 - U_{\text{LO}}^{\text{LO}}})} \leq \rho \\
\Rightarrow & \frac{U_{\text{HI}}^{\text{LO}}}{\frac{1 - (U_{\text{LO}}^{\text{LO}} + U_{\text{HI}}^{\text{HI}})}{1 - U_{\text{LO}}^{\text{LO}}}} \leq \rho - U_{\text{LO}}^{\text{LO}} \\
\Rightarrow & \frac{U_{\text{HI}}^{\text{LO}}}{\rho - U_{\text{LO}}^{\text{LO}}} \leq \frac{1 - (U_{\text{LO}}^{\text{LO}} + U_{\text{HI}}^{\text{HI}})}{1 - U_{\text{LO}}^{\text{LO}}} \\
\Rightarrow & x \leq \frac{1 - (U_{\text{LO}}^{\text{LO}} + U_{\text{HI}}^{\text{HI}})}{1 - U_{\text{LO}}^{\text{LO}}} \\
\Rightarrow & U_{\text{LO}}^{\text{LO}} + \frac{U_{\text{HI}}^{\text{HI}}}{(1 - x)} \leq 1
\end{aligned}
$$

which is the schedulability condition to correctly schedule all the tasks in HI-mode. □

By combining Theorem 1 and Theorem 2, we prove the following theorem.

Theorem 4. *Given a precise mixed criticality model task set, the minimum value of ρ for the task set to be schedulable by EDF-VD is:*

$$
min\left(U_{\text{LO}}^{\text{LO}} + U_{\text{HI}}^{\text{HI}},\ U_{\text{LO}}^{\text{LO}} + \frac{U_{\text{HI}}^{\text{LO}}(1 - U_{\text{LO}}^{\text{LO}})}{1 - (U_{\text{HI}}^{\text{HI}} + U_{\text{LO}}^{\text{LO}})}\right) \tag{4.6}
$$

only when,

$$
U_{\text{LO}}^{\text{LO}} + \frac{U_{\text{HI}}^{\text{LO}}(1 - U_{\text{LO}}^{\text{LO}})}{1 - (U_{\text{HI}}^{\text{HI}} + U_{\text{LO}}^{\text{LO}})} \leq 1
$$

Proof. On combining Theorem 1 and 2, from Condition (4.1) and Condition (4.4) we can represent the range of the value of parameter x:

$$
\frac{U_{\text{HI}}^{\text{LO}}(\tau)}{\rho - U_{\text{LO}}^{\text{LO}}(\tau)} \leq x \leq \frac{1 - (U_{\text{HI}}^{\text{HI}} + U_{\text{LO}}^{\text{LO}})}{(1 - U_{\text{LO}}^{\text{LO}})}
$$

Thus determining the minimum value of ρ as:

$$\frac{U_{\text{HI}}^{\text{LO}}(\tau)}{\rho - U_{\text{LO}}^{\text{LO}}(\tau)} \leq \frac{1 - (U_{\text{HI}}^{\text{HI}} + U_{\text{LO}}^{\text{LO}})}{(1 - U_{\text{LO}}^{\text{LO}})}$$
$$\Rightarrow U_{\text{LO}}^{\text{LO}} + \frac{U_{\text{HI}}^{\text{LO}}(1 - U_{\text{LO}}^{\text{LO}})}{1 - (U_{\text{HI}}^{\text{HI}} + U_{\text{LO}}^{\text{LO}})} \leq \rho$$
$$\Rightarrow U_{\text{LO}}^{\text{LO}} + \frac{U_{\text{HI}}^{\text{LO}}}{\frac{1 - (U_{\text{HI}}^{\text{HI}} + U_{\text{LO}}^{\text{LO}})}{1 - U_{\text{LO}}^{\text{LO}}}} \leq \rho$$

which is the sufficient condition (refer Theorem 3) to ensure that EDF-VD successfully schedules all the HI-criticality tasks in τ. □

4.3. SPEEDUP AND APPROXIMATION RATIO METRICS

The theoretical real-time systems community considers speedup as a viable measure to measure the effectiveness of mixed-criticality scheduling algorithms. (Baruah and Agarwal, 2018) lists the sources of intractability for mixed criticality systems, isolates their impacts and highlights the need for metrics that are able to independently evaluate the approximation-ratio impact (i.e., comparison with MC-optimality) and the competitive-ratio impact (i.e., the sub-optimality emerging from the need of clairvoyance) of EDF-VD in planning recurrent frameworks.

4.3.1. Speedup Factor of EDF-VD Algorithm. In this section, we prove that EDF-VD for our problem has a speedup bound equal to 2, for any non-clairvoyant algorithm for our chosen workload and platform model with ρ ranging from [0.5, 1].

Theorem 5. *For the optimization problem described in Section 3.5, algorithm EDF-VD (in Section 4.2) has a speedup bound no larger than* 2.

Proof. We will show below that any MC task system τ that can be correctly scheduled by a clairvoyant optimal algorithm on a processor with normal speed b and energy conserving speed $b \times \rho$, is correctly scheduled by EDF-VD on a processor with normal speed 1 and the energy-conserving speed ρ (where $0.5 \leq \rho \leq 1$). This way we prove the theorem, such that a processor that is faster by a factor of b is sufficient for EDF-VD to correctly schedule τ.

Note that any task set τ that is correctly schedulable by a clairvoyant scheduler should necessarily satisfy

$$max\left(\frac{U_{\text{LO}}^{\text{LO}}}{\rho} + \frac{U_{\text{HI}}^{\text{LO}}}{\rho}, U_{\text{LO}}^{\text{LO}} + U_{\text{HI}}^{\text{HI}}\right) \leq b \tag{4.7}$$

since its LO-criticality utilization ($U_{\text{LO}}^{\text{LO}} + U_{\text{HI}}^{\text{LO}}$) must be $\leq b\rho$ and its HI-criticality utilization ($U_{\text{LO}}^{\text{LO}} + U_{\text{HI}}^{\text{HI}}$) must be $\leq b$.

From Theorems 1 and 2, we know that if an x satisfying both theorems exists, sufficient conditions for both LO- and HI-mode schedulability are met and there will be no deadline miss. Since LO-mode schedulability condition Theorem 1 requires that:

$$x \geq \frac{U_{\text{HI}}^{\text{LO}}}{\rho - U_{\text{LO}}^{\text{LO}}}$$

where $U_{\text{LO}}^{\text{LO}} < \rho$.

And HI-mode schedulability condition from Theorem 2 requires that:

$$x \leq \frac{1 - (U_{\text{HI}}^{\text{HI}} + U_{\text{LO}}^{\text{LO}})}{(1 - U_{\text{LO}}^{\text{LO}})}.$$

We can represent the schedulability conditions from both modes of operation as:

$$\frac{U_{\text{HI}}^{\text{LO}}}{\rho - U_{\text{LO}}^{\text{LO}}} \leq \frac{1 - (U_{\text{HI}}^{\text{HI}} + U_{\text{LO}}^{\text{LO}})}{(1 - U_{\text{LO}}^{\text{LO}})}$$

Since $U_{\text{LO}}^{\text{LO}} + U_{\text{HI}}^{\text{LO}} \leq b\rho \implies U_{\text{HI}}^{\text{LO}} \leq b\rho - U_{\text{LO}}^{\text{LO}}$

$$\Rightarrow \frac{(b\rho - U_{\text{LO}}^{\text{LO}})}{\rho - U_{\text{LO}}^{\text{LO}}} \leq \frac{1 - (U_{\text{HI}}^{\text{HI}} + U_{\text{LO}}^{\text{LO}})}{(1 - U_{\text{LO}}^{\text{LO}})}$$

we have necessary condition $U_{\text{LO}}^{\text{LO}} + U_{\text{HI}}^{\text{HI}} \leq b$

$$\begin{aligned}
\Rightarrow \quad & \frac{(b\rho - U_{\text{LO}}^{\text{LO}})}{\rho - U_{\text{LO}}^{\text{LO}}} \leq \frac{1 - b}{(1 - U_{\text{LO}}^{\text{LO}})} \\
\Rightarrow \quad & (b\rho - U_{\text{LO}}^{\text{LO}})(1 - U_{\text{LO}}^{\text{LO}}) \leq (1 - b)(\rho - U_{\text{LO}}^{\text{LO}}) \\
& (U_{\text{LO}}^{\text{LO}})^2 - (b\rho + b)U_{\text{LO}}^{\text{LO}} + 2b\rho - \rho \leq 0 \qquad (4.8)
\end{aligned}$$

Now, if we set b $\leftarrow \frac{1}{2}$, Condition (4.8) becomes:

$$\begin{aligned}
& (U_{\text{LO}}^{\text{LO}})^2 - (\frac{\rho}{2} + \frac{1}{2})U_{\text{LO}}^{\text{LO}} \leq 0 \\
\Rightarrow & U_{\text{LO}}^{\text{LO}} \leq \frac{\rho}{2} + \frac{1}{2}
\end{aligned}$$

which is true for all values of $U_{\text{LO}}^{\text{LO}}$ since $0 \leq U_{\text{LO}}^{\text{LO}} \leq \rho \leq 1$. This is because according to Equation (4.2) in Theorem 1, $U_{\text{LO}}^{\text{LO}} < \rho$. We have thus shown that any task system that is clairvoyant schedulable by an optimal algorithm on a speed $\frac{1}{2}$ processor with energy conserving speed $\rho/2$ is scheduled by EDF-VD to meet all deadlines on a unit-speed processor and energy conserving speed ρ. This establishes the theorem since it shows that a processor that has a speedup bound of 2 is sufficient for EDF-VD to correctly schedule τ. □

We now show that the speedup bound cannot get better than 2 for any non-clairvoyant scheduler by providing a counter-example.

Theorem 6. *No non-clairvoyant algorithm for scheduling dual-criticality implicit-deadline sporadic task systems can have a speedup bound better than* 2.

Proof. We prove this theorem by providing a counter-example. Consider the example task system $\tau = \{\tau_1, \tau_2\}$ running on a processor with energy conserving speed 0.5, with the following parameters:

τ_i	χ	C_i^{LO}	C_i^{HI}	T_i
τ_1	LO	$0.45 + \epsilon$	$0.45 + \epsilon$	2
τ_2	HI	0.05	$0.55 + \epsilon$	2

This system has a utilization which is larger than 0.5 by an arbitrarily small value ϵ. Let us consider the minimum possible energy conserving speed ρ equal to 0.5 [1].

We want to prove that no non-clairvoyant algorithm can have a speedup bound better than 2. In order for that to happen, it is crucial to show that this system is: (i) schedulable by a clairvoyant optimal algorithm on a processor with normal speed b and energy conserving speed $b \times \rho$ (since speedup bound is better than 2, $b > 0.5$. For this theorem we consider $b = 0.5+\epsilon$) and, (ii) is not correctly scheduled by a non-clairvoyant algorithm on a processor with normal speed 1 and the energy-conserving speed ρ.

Clairvoyant Algorithm: The processor executes at a normal speed (in HI-criticality mode) of $0.5+\epsilon$ and energy conserving speed (in LO-criticality mode) equal to $(0.5+\epsilon)\times 0.5 = 0.25 + 0.5\epsilon$. For an MC-instance to be schedulable by clairvoyant EDF algorithm, it is necessary for the utilization to not exceed processor speed. From the example, we observe that the total utilization in LO-criticality mode is $0.25 + 0.5\epsilon$ which is equal to the energy conserving processor speed. Correspondingly, the HI-mode utilization is $0.5 + \epsilon$ which does not exceed the normal speed of the processor. It is safe to conclude that the clairvoyant EDF scheduler will meet all the deadlines in both HI- and LO-criticality behaviors.

Non-clairvoyant Algorithm: We consider the same example task system τ consisting of tasks τ_1 and τ_2. Suppose all tasks were to generate jobs simultaneously (i.e., arrival time of the jobs = 0).

[1] Since the speedup bound obtained in Theorem 5 depends upon the value of ρ chosen ($0.5 \leq \rho \leq 1$), we consider the minimum possible ρ for the model, which is 0.5.

Since the system is non-clairvoyant, it is not revealed prior to the execution of the job, whether the behavior is a LO-criticality or a HI-criticality one.

Consider a case: τ_1's job receives $(0.45 + \epsilon)$ units of execution before τ_2's job and is executed at an energy conserving speed $\rho = 0.5$. From Figure 3.1 we have considered a linear relationship between speed ρ and WCET. In this case, τ_1's job executes for $(0.45 + \epsilon)/0.5 = 0.9 + 2\epsilon$ units. If τ_2's job reveals itself to be a HI-criticality job, there is not enough time remaining for τ_2's job to complete by its deadline at time-instant 2. The MC instance is not schedulable by the non-clairvoyant algorithm on a processor with normal speed 1 and the energy-conserving speed.

□

4.3.2. Approximation Ratio. It has been proven that MC-schedulability for dual-criticality recurrent task systems is NP-hard in the strong sense, thus adopting non-optimal algorithms (EDF-VD) is justified (Baruah and Agarwal, 2018). An instance is declared as MC-schedulable if it is correctly scheduled by any non clairvoyant on-line algorithm.

An algorithm A has an approximation ratio of $\alpha \geq 1$ if and only if some non clairvoyant on-line algorithm can guarantee MC correctness with full speed of 1 and a energy conserving speed of ρ, Algorithm A guarantees MC correctness to the same set with a processor of normal speed 1 and energy conserving speed of $\alpha \times \rho$.

Theorem 7. *For this model, algorithm EDF-VD has an approximation ratio no larger than*

$$1 + \frac{U_{\text{HI}}^{\text{LO}}(1 - U_{\text{LO}}^{\text{LO}})}{U_{\text{LO}}^{\text{LO}}(1 - (U_{\text{HI}}^{\text{HI}} + U_{\text{LO}}^{\text{LO}}))} \tag{4.9}$$

Proof. We observe that any task-set τ that is correctly scheduled by a clairvoyant scheduler upon a processor with normal speed 1 and energy conserving speed $\alpha \times \rho$ must necessarily satisfy:

$$max\left(\frac{U_{\text{LO}}^{\text{LO}}}{\alpha\rho} + \frac{U_{\text{HI}}^{\text{LO}}}{\alpha\rho}, U_{\text{LO}}^{\text{LO}} + U_{\text{HI}}^{\text{HI}}\right) \leq 1 \tag{4.10}$$

Inequality (4.10) only indicates that the speed of the processor is increased by an approximation ratio α in the LO-mode.

It is safe to assume that energy conserving speed is always $\leq$ normal speed. For an on-line algorithm A, to correctly claim that τ is MC-schedulable, we derive a bound (range) for the approximation ratio where $\alpha \times \rho \leq 1$ (for $0.5 \leq \rho \leq 1$). At first glance, it is quite evident that the maximum value of α can be $1/\rho$ for the system to be schedulable (since $\alpha\rho \leq 1$). This however is a loose upper bound on the value of approximation α. To generate a more viable upper bound for approximation ratio, we have the following two cases:

CASE 1: If $\rho \geq U_{\text{LO}}^{\text{LO}} + min\big(U_{\text{HI}}^{\text{HI}}, \frac{U_{\text{HI}}^{\text{LO}}(1 - U_{\text{LO}}^{\text{LO}})}{1 - (U_{\text{HI}}^{\text{HI}} + U_{\text{LO}}^{\text{LO}})}\big)$.

If this condition is true, we can claim that $\rho \geq \rho_{min}$. This is evident from Condition (4.6) of Theorem 4. If $\rho \geq \rho_{min}$, for any value of $\rho \leq 1$ the system will be schedulable. Thus for this case the maximum value of α to guarantee MC-correctness is 1.

CASE 2: If $\rho \leq U_{\text{LO}}^{\text{LO}} + min\big(U_{\text{HI}}^{\text{HI}}, \frac{U_{\text{HI}}^{\text{LO}}(1 - U_{\text{LO}}^{\text{LO}})}{1 - (U_{\text{HI}}^{\text{HI}} + U_{\text{LO}}^{\text{LO}})}\big)$.

The maximum value of α for an on-line algorithm to correctly schedule a system can be given as:

$$\alpha \leq 1 + \frac{U_{\text{HI}}^{\text{LO}}(1 - U_{\text{LO}}^{\text{LO}})}{U_{\text{LO}}^{\text{LO}}(1 - (U_{\text{HI}}^{\text{HI}} + U_{\text{LO}}^{\text{LO}}))}$$

We justify the reason for selecting such a bound below:

From Theorem 4 we have the minimum value of ρ_{min} as,

$$min\left(U_{\text{LO}}^{\text{LO}} + U_{\text{HI}}^{\text{HI}},\ U_{\text{LO}}^{\text{LO}} + \frac{U_{\text{HI}}^{\text{LO}}(1 - U_{\text{LO}}^{\text{LO}})}{1 - (U_{\text{HI}}^{\text{HI}} + U_{\text{LO}}^{\text{LO}})}\right)$$

The schedulability condition is guaranteed if and only if $\rho \geq \rho_{min}$. In this case the value of α should be such that $\alpha\rho$ should satisfy the schedulability condition i.e., $\alpha \geq \rho_{min}/\rho$. Keeping this as a minimum bound required for α, we select a maximum bound for α as $\rho_{min}/U_{\text{LO}}^{\text{LO}}$. Since $0 \leq U_{\text{LO}}^{\text{LO}} < \rho \leq 1$, for this value of α, the schedulability condition will

always hold. Thus we have

$$\alpha \leq 1 + \frac{U_{\text{HI}}^{\text{LO}}(1 - U_{\text{LO}}^{\text{LO}})}{U_{\text{LO}}^{\text{LO}}(1 - (U_{\text{HI}}^{\text{HI}} + U_{\text{LO}}^{\text{LO}}))}$$

The inequality (4.9) may not represent the tightest upper bound, however is significantly better than the loose upper-bound of $1/\rho$. The performance of the algorithm with the derived value of α is demonstrated in Figure 4.5 in Section 4.5. □

4.4. EXPERIMENTAL EVALUATION

We have conducted a progression of schedulability tests to assess the effectiveness of the EDF-VD scheduling technique to guarantee that MC implicit deadlines sporadic task systems are correctly scheduled.

4.4.1. Workload Generation. The experiments were conducted on a randomly generated task-set that were generated according to the workload generation model established by Guan et al. (Guan *et al.*, 2013) with further modifications. The input specifications for our workload generation are as follows:

- U_{bound} is the desired upper bound of utilization of the system: $(U_{\text{LO}}^{\text{LO}}(\tau) + U_{\text{HI}}^{\text{HI}}(\tau))$
- The time period of a task is randomly chosen in the range $[T_{down}, T_{up}]$; $0 \leq T_{down} \leq T_{up}$.
- For each task, a value is randomly selected in the range $[U_{down}, U_{up}]$ and multiplied with task's period, to obtain execution time in the LO-mode; $0 \leq U_{down} \leq U_{up} \leq 1$.
- The ratio of HI-WCET and LO-WCET is drawn from the range $[Z_{down}, Z_{up}]$; $1 \leq Z_{down} \leq Z_{up}$.
- P: Probability that the chosen task is HI-critical; $0 \leq P \leq 1$

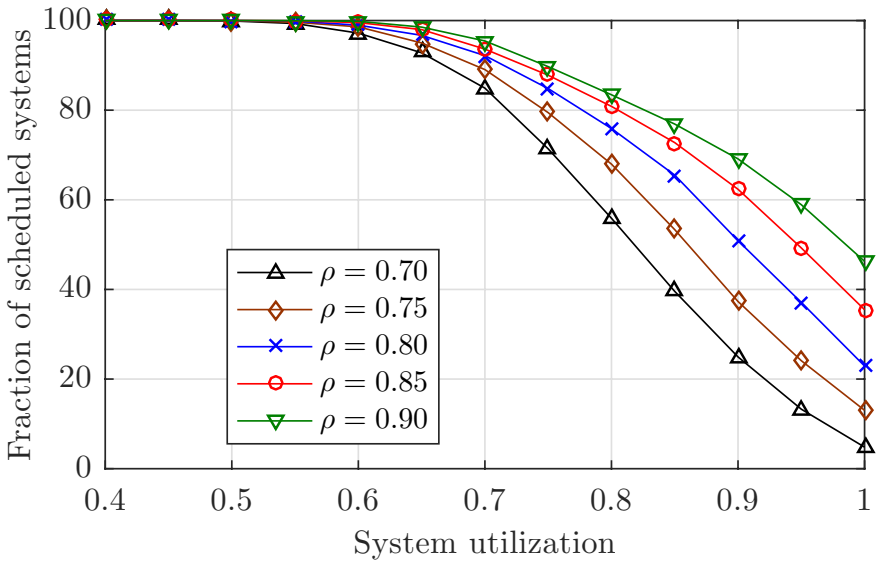


Figure 4.3. Example outcome of schedulability experiments, for parameters $[U_{down}, U_{up}] = [0.02, 0.2]$; $[T_{down}, Tup] = [5, 50]$; $[Z_{down}, Z_{up}] = [1, 4]$; $P = 0.5$ for different values of ρ

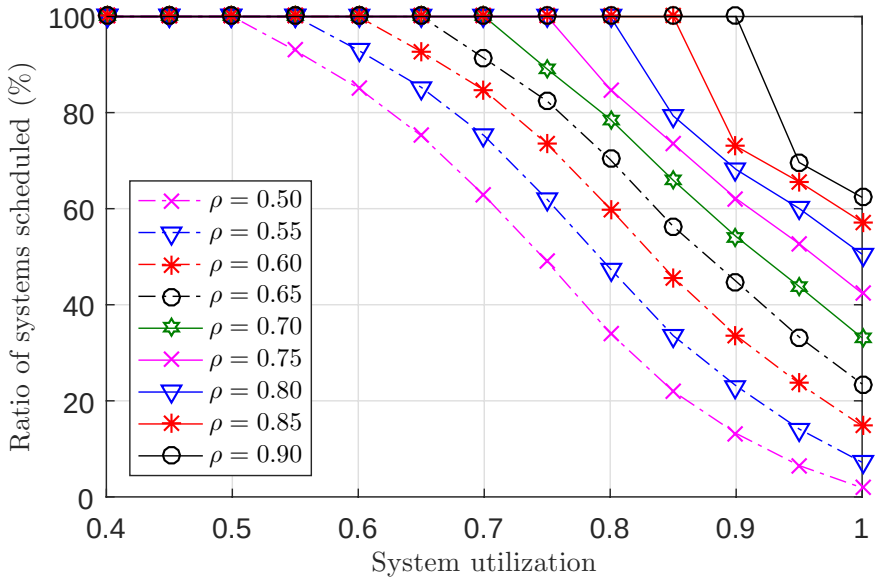


Figure 4.4. Example outcome of schedulability experiments, for parameters $[U_{down}, U_{up}] = [0.02, 0.2]$; $[T_{down}, Tup] = [5, 50]$; $[Z_{down}, Z_{up}] = [1, 8]$; $P = 0.5$ for different values of ρ

For the generation of a MC-workload from the combination of these parameter values, the task generation algorithm iteratively adds tasks to an empty set until the utilization bound is met.

In our experiments, we determine the ratio of systems scheduled correctly against the system utilization U_{bound}. Simulations are carried out for different values of ρ. Although we cannot draw authoritative conclusions from the experiments as the results are influenced by the random workload generator, we do make some interesting observations.

When the average utilization percentage is smaller than 0.5, the task system is always schedulable. This observation from Figure 4.3 matches our speed-up factor computation, since speed up bound is 2 (0.5).

Figure 4.3 clearly demonstrates the ratio of systems scheduled correctly as a function of system utilization. For different values of ρ ranging from [0.5,0.9], the system is completely schedulable for average utilization ≤ 0.5.

Likewise in Figure 4.4, the performance of the EDF-VD algorithm is demonstrated for a workload with different values of ρ and $[Z_{down}, Z_{up}] = [1, 8]$. Figure 4.5 shows the ratio of correctly scheduled task sets with an energy conserving speed of $\alpha \times \rho$ against system utilization.

The performance of the algorithm was determined again with an energy conserving speed of $\alpha\rho$ and normal speed 1, where α is the approximation ratio. The maximum value of α was considered according to Condition (4.9) as proved in Section 4.3.2. It is interesting to observe that the maximum bound chosen for approximation ratio α is sufficient to guarantee MC correctness by an on-line non-clairvoyant algorithm.

4.5. SUMMARY OF ENERGY EFFICIENT PRECISE COMPUTING

The conventional mixed-criticality model, despite its popularity, is controversial for penalizing all LO-criticality tasks in HI-mode. Recent works throw light on overcoming this setback by partially (if not fully) trying to accommodate LO-criticality tasks even under pessimistic behaviors.

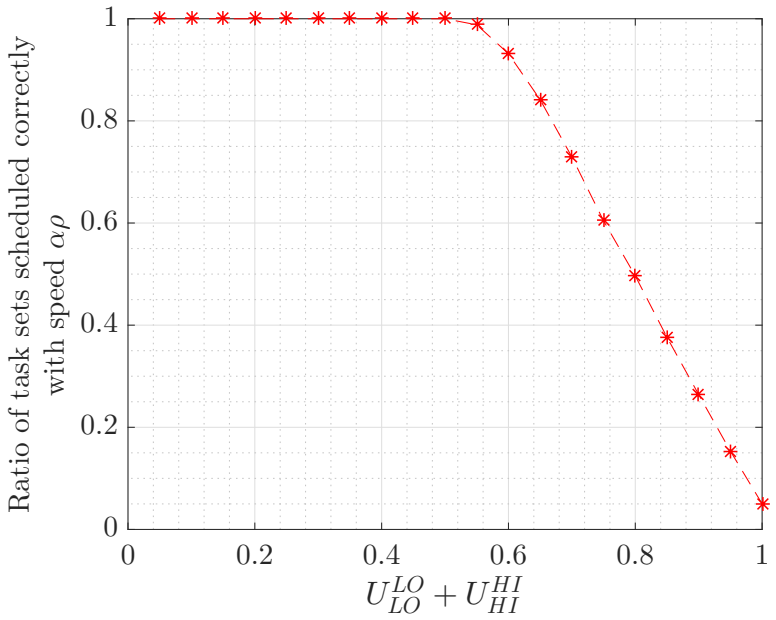


Figure 4.5. Performance of the algorithm under normal speed 1 and energy conserving speed $\alpha\rho$; with α value determined from Equation 4.9

In this work, we develop an integrated model combining precise scheduling of LO-criticality tasks on energy conserving platforms that adopt the DVFS strategy. A sufficient test for this unified model under EDF-VD scheduling algorithm is proposed. The sufficient test is evaluated theoretically with sound proofs and via schedulability experiments on randomly generated workloads. We provide results on calculating both speedup bound and also approximation ratio to satisfy real-time requirements in situation of overrun.

5. SUSTAINABILITY IN MC SCHEDULING

Schedulability tests play an vital part within the confirmation of safety-critical real-time systems. Given the detail of an occurrence comprising the abstraction of workload and the computing framework upon which the workload is to execute, a schedulability test decides whether all timing limitations (frequently indicated by deadlines) are ensured to be met under indicated scheduling policies. For safety-critical frameworks, schedulability investigation must be performed earlier to run-time; in order to do so, parameters characterizing the run-time workload must be evaluated earlier to run-time. Distinctive tools and strategies utilized for making such estimations may be more or less pessimistic (cynical) than each other; consequently, the use of traditional techniques may result in frameworks exhibiting run-time behavior way better than estimated.

An MC scheduling policy is said to be sustainable if any MC instance that is MC-schedulable by the policy remains so if one or more of the parameters characterizing the instance is improved. Analogously, a schedulability test for some MC scheduling policy is said to be sustainable if any MC instance that is deemed MC-schedulable by the schedulability test will continue to be deemed MC-schedulable by the test if one or more of its parameters is improved in one/all of the following ways:

1. Diminishing WCET parameters (C_i^L and/or C_i^H).
2. Elongating periods for sporadic task systems; expediting release times forward (i.e., decreasing a_i) for a set of jobs.
3. Prolonging relative deadlines (D_i).
4. Lowering the criticality level assignment (χ_i) of a task/job (from HI to LO in case of dual-criticality systems).

In this section of the thesis we will visit several existing uniprocessor and multiprocessor algorithms and perform sustainability analysis on each of them. Here, we will consider the traditional behavior where an MC system is assumed to begin execution in LO-mode and if a job has executed for more than its LO-criticality WCET specification without signaling completion, a system-wide mode switch to HI-mode is said to occur. The system returns to LO-mode at the first idle instant after the mode switch (S. Baruah and A. Burns, 2014). In all other scenarios, the system is considered as an erroneous mode, where no correctness guarantees are made and thus is not considered in this work.

A scheduling policy and/or schedulability test may be sustainable with respect to some but not all parameters. In each of the following subsections, we will consider one well-known MC scheduler, and examine the sustainability with respect to all parameters. We limit ourselves in this paper to two criticality levels – although many results are easily extended to more than two levels, we leave filling in the details as future work.

5.1. SUSTAINABILITY IN UNIPROCESSOR SCHEDULING ALGORITHMS

In this section, we study sustainability properties of four uniprocessor MC scheduling algorithms and their associated schedulability tests. The first three – Criticality Monotonic, EDF-VD, and AMC – are task-scheduling algorithms; the fourth, OCBP, schedules collections of jobs.

5.1.1. Criticality Monotonic. Criticality Monotonic (CM) (Vestal, 2007) is a scheduling policy that schedules at each time instant an available job of highest criticality. Hence a task of criticality level ℓ cannot affect the scheduling of tasks of criticality greater than ℓ. In this paper we restrict ourselves with only two criticality levels, LO and HI. We study a sporadic task model where each task is characterized by $\tau_i = \{C_i^L, C_i^H, T_i, D_i, \chi_i\}$.

We assume that the (non MC) mechanism to schedule tasks within each criticality level is sustainable w.r.t. all parameters, and examine the sustainability of CM as a general MC scheduling framework.

SUSTAINABILITY W.R.T. RELATIVE DEADLINE, WCET, AND PERIOD. Changing relative deadline, WCET, and period parameters will not affect the general CM framework since no criticality level is modified. As it is assumed that the scheduler used within each criticality level is sustainable to all parameters, the schedulability conditions will still hold within each criticality level, leading to sustainability of CM.

Now we look into sustainability w.r.t. criticality levels.

Theorem 8. *Criticality Monotonic scheduling algorithm is not sustainable with respect to criticality levels.*

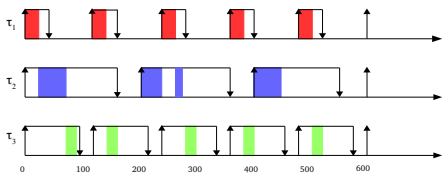
Proof. Consider the task-set shown in Table 5.1, which is CM-schedulable (using deadline monotonic within each criticality level). Figure 5.1(a) illustrates the schedule of the task-set with its respective arrival times and deadlines.

Table 5.1. An MC task-set that is not sustainable under criticality monotonic scheduling policy.

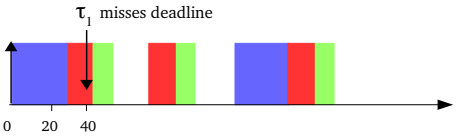
Task	C_i^L	C_i^H	T_i	D_i	Criticality	Priority
τ_1	20	25	120	40	HI $\rightarrow$LO	$1 \rightarrow 3$
τ_2	28	60	200	160	HI	2
τ_3	12	12	120	100	LO	4

We now decrease the criticality level of task τ_1 from HI- to LO- criticality and observe the outcome schedule in Figure 5.1(b), where τ_1 misses its deadline. Thus we conclude that the CM scheduling policy is not sustainable w.r.t. criticality levels.

□



(a) Criticality-Monotonic schedule for tasks in Table 5.1 under LO-criticality mode.



(b) Criticality-Monotonic schedule for tasks in Table 5.1, when criticality level of τ_1 is changed from HI to LO, where τ_1 misses its deadline.

Figure 5.1. Schedule demonstration of the sample task set (shown in Table 5.1) under Criticality-Monotonic before and after the change of criticality level of one of the tasks (τ_1).

5.1.2. Earliest Deadline First with Virtual Deadlines (EDF-VD). The Earliest Deadline First with Virtual Deadline scheduling policy (EDF-VD) (Baruah *et al.*, 2012b) is an adaptation of the Earliest Deadline First (EDF) algorithm to dual-criticality implicit-deadline sporadic task systems. It is proved in (Baruah *et al.*, 2012b) that EDF-VD correctly schedules any dual-criticality task system $\tau = \{\tau_1, \tau_2, \ldots, \tau_n\}$ upon a unit-speed preemptive processor if

$$x\, U_{\text{LO}}^{\text{LO}}(\tau) + U_{\text{HI}}^{\text{HI}}(\tau) \leq 1 \tag{5.1}$$

where x is defined as follows:

$$x \leftarrow U_{\text{HI}}^{\text{LO}}(\tau)/(1 - U_{\text{LO}}^{\text{LO}}(\tau)) \tag{5.2}$$

Condition 5.1, in fact, constitutes a schedulablity test for EDF-VD: EDF-VD computes x according to Equation 5.2 above and determines whether Condition 5.1 is satisfied. In the remainder of this section, we establish that this schedulability test for EDF-VD is sustainable with respect to criticality level, WCETs, and period. (Since this schedulability test is for

implicit-deadline task systems, its sustainability with respect to relative deadlines trivially follows from the observation that EDF-VD does not make use of the relative deadline parameter.)

Recall the various the total utilization for each mode of operation parameters defined in the system model section represented as follows:

- The total utilization for all LO-criticality tasks in LO- and HI-modes respectively, we have: $U_{\text{LO}}^{\text{LO}} = \sum_{\forall \tau_i \in \tau_{\text{LO}}} u_i^{\text{LO}}$, $U_{\text{LO}}^{\text{HI}} = \sum_{\forall \tau_i \in \tau_{\text{LO}}} u_i^{\text{HI}}$.
- Similarly for all HI-criticality tasks, the utilization is represented as: $U_{\text{HI}}^{\text{LO}} = \sum_{\forall \tau_i \in \tau_{\text{HI}}} u_i^{\text{LO}}$, $U_{\text{HI}}^{\text{HI}} = \sum_{\forall \tau_i \in \tau_{\text{HI}}} u_i^{\text{HI}}$.

Let us introduce some simplifying notations:

$$\begin{aligned} u_l &\leftarrow U_{\text{LO}}^{\text{LO}}(\tau) \\ u_h &\leftarrow U_{\text{HI}}^{\text{LO}}(\tau) \\ u'_h &\leftarrow U_{\text{HI}}^{\text{HI}}(\tau) \end{aligned}$$

While executing in LO-criticality mode, the deadlines of the high criticality tasks are determined by scaling down the original period of a HI-criticality task with a factor x (x $\leq$ 1) to obtain a virtual deadline. The scaling factor x is calculated off-line as $x = u_h/(1 - u_l)$.

For the EDF-VD scheduling policy to correctly schedule a dual-criticality implicit deadline task system on a single unit-speed processor, the sufficient conditions for tasks to be scheduled in both LO- and HI-mode respectively are (Baruah *et al.*, 2012b):

$$x \geq \frac{u_h}{1 - u_l}, \tag{5.3}$$

$$x \cdot u_l + u'_h \leq 1. \tag{5.4}$$

We now determine the sustainability of the scheduling policy by making favorable alterations in the parameters and verify if the schedulability condition still persists.

Lemma 1. *EDF-VD is sustainable w.r.t criticality levels; i.e., when changing the criticality of a task from* HI *to* LO*, Conditions* (5.3) *and* (5.4) *will continue to hold if they used to be so.*

Proof. The change to the criticality level of a task from HI to LO will result in an increase of the utilization of LO-criticality tasks (u_l) and decreases in the utilization of HI-criticality tasks (u_h, u'_h), all with the same amount (assumed to be δ) i.e.,

$$u_h = u_h - \delta,$$

$$u'_h = u'_h - \delta,$$

$$u_l = u_l + \delta.$$

Now, on substituting these notations in Equations (5.3) and (5.4), the scaling term x can then be denoted as:

$$\text{x} \leftarrow \frac{u_h}{1 - u_l}.$$

The equation for the HI-criticality schedulability test can be written as:

$$\frac{u_h.u_l}{1 - u_l} + u'_h \leq 1. \tag{5.5}$$

On modifying the utilization values with δ in the Equation (5.5) we get:

$$\frac{(u_h - \delta)(u_l + \delta)}{1 - (u_l + \delta)} + (u'_h - \delta) \leq 1. \tag{5.6}$$

To determine if the schedulability condition in Equation 5.6 still holds, we show the following proof: By demonstrating that the difference between Equations (5.5) and (5.6) is positive, i.e.,

$$\frac{(u_h - \delta)(u_l + \delta)}{1 - (u_l + \delta)} + (u'_h - \delta) \leq \frac{u_h.u_l}{1 - u_l} + u'_h \leq 1. \tag{5.7}$$

$$\begin{aligned}
&\frac{u_h \cdot u_l}{1 - u_l} + u'_h - \frac{(u_h - \delta)(u_l + \delta)}{1 - (u_l + \delta)} + (u'_h - \delta) \geq 0 \\
&\Rightarrow \frac{u_h \cdot u_l}{1 - u_l} - \frac{(u_h - \delta)(u_l + \delta)}{1 - (u_l + \delta)} - \delta \geq 0 \\
&\Rightarrow \frac{u_h.u_l}{1 - u_l} - \frac{(u_h - \delta)(u_l + \delta) - \delta + \delta(u_l + \delta)}{1 - (u_l + \delta)} \geq 0 \\
&\Rightarrow \frac{u_h \cdot u_l}{1 - u_l} - \frac{(u_l + \delta)u_h - \delta}{1 - (u_l + \delta)} \geq 0 \\
&\Rightarrow \frac{u_h \cdot u_l}{1 - u_l} - \frac{u_h.u_l + \delta u_h - \delta}{1 - (u_l + \delta)} \geq 0 \\
&\Rightarrow u_h \cdot u_l(1 - u_l + \delta) - u_h \cdot u_l(1 - u_l) - \delta(u_h - 1)(1 - u_l) \geq 0 \\
&\Rightarrow \delta(1 - u_h)(1 - u_l) - u_h \cdot u_l \delta \geq 0 \\
&\Rightarrow (1 - u_h)(1 - u_l) \geq u_h \cdot u_l \\
&\Rightarrow 1 - u_h - u_l \geq 0 \\
&\Rightarrow u_l + u_h \leq 1.
\end{aligned} \tag{5.8}$$

The solution obtained in Equation (5.8) satisfies the schedulability conditions stated in Equations (5.3) and (5.4), thus establishing that the EDF-VD scheduling policy is sustainable w.r.t. to criticality levels. □

Lemma 2. *EDF-VD is sustainable w.r.t WCETs.*

Proof. According to the definition of sustainability, on decreasing the WCET of a task τ_i (either C_i^L or C_i^H), the schedulability conditions of the whole task system should still hold. To demonstrate the sustainability, we consider a small arbitrary value δ by which we decrease C_i^L or C_i^H values, and check the two sufficient conditions.

(1) Decrease of C_i^H.

Upon decreasing the C_i^H by a menial amount $\delta > 0$, the utilization of the set (u_h) will decrease by a value $(\delta' = \delta/T_i > 0)$. Thus, the corresponding HI-mode schedulability condition for the new task set is:

$$x \cdot u_l + (u'_h - \delta') \leq 1, \tag{5.9}$$

which obviously holds from Condition (5.4) and the fact that $\delta' > 0$. This conveys that in HI mode, decreasing the C_i^H value does not have any adversary effect on the schedulability of the whole system.

Now we will check the schedulability under LO mode; i.e., if the condition in Equation (5.3) holds. Since C_i^H values have nothing to do with the condition for LO mode, it remains true. Thus we conclude that the decrease of C_i^H will not have any adversary effects on the schedulability of the whole system.

(2) Decrease of C_i^L.

Similarly, if C_i^L is diminished in such a way, the u_l value decreases by δ. The following schedulability test (in HI mode) will also hold as $x > 0$ and $\delta > 0$.

$$x \cdot (u_l - \delta) + u'_h \leq 1. \tag{5.10}$$

Now we inspect the schedulability under LO mode; i.e., if the condition in Equation (5.3) complies after substituting the modified value of u_l:

$$\frac{u_h}{1 - (u_l - \delta)} \leq \frac{u_h}{1 - u_l} = x, \tag{5.11}$$

and

$$\frac{u_h - \delta}{1 - u_l} \leq \frac{u_h}{1 - u_l} x. \tag{5.12}$$

This indicates that the condition for LO-mode correctness continues to prevail. □

Lemma 3. *EDF-VD is sustainable w.r.t period.*

Proof. This follows directly from the proof for sustainability over WCETs as an increasing period will lead to a decrease of per-mode utilization. □

Theorem 9. *EDF-VD is sustainable w.r.t all parameters.*

Proof. This follows from Lemmas 1, 2, and 3. □

5.1.3. Adaptive Mixed-Criticality (AMC). The adaptive mixed criticality scheduling policy (AMC) (Baruah *et al.*, 2011a) is a fixed-priority algorithm for scheduling MC sporadic task systems on preemptive uniprocessors. A priority order is achieved by applying Audsley's priority assignment algorithm (Audsley, 2001), and has been demonstrated to be optimal (Baruah *et al.*, 2011a; Vestal, 2007); i.e., whenever a feasible priority order exists, the system will be AMC-schedulable.

Response Time Analysis (RTA) techniques are used to determine the schedulability of AMC scheduling policy. The analysis is done in three phases (Baruah *et al.*, 2011a):

1. Verifying schedulability of LO-criticality mode with:

$$R_i^L = C_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i^L}{T_j} \right\rceil C_j^L, \tag{5.13}$$

where *hp(i)* is the set of all tasks with priority higher than that of task τ_i.

2. Verifying schedulability of HI-criticality mode with

$$R_i^H = C_i + \sum_{j \in hpH(i)} \left\lceil \frac{R_i^H}{T_j} \right\rceil C_j^H, \tag{5.14}$$

where *hpH(i)* is the set of HI-critical tasks with priority higher than, or equal to, that of task τ_i.

3. Verifying schedulability during criticality (mode) change in an iterative manner w.r.t. maximum response time R_i^* until it is stabilized with:

$$R_i^* = C_i^H + \sum_{j \in hpH(i)} \left\lceil \frac{R_i^*}{T_j} \right\rceil C_j^H + \sum_{j \in hpL(i)} \left\lceil \frac{R_i^L}{T_k} \right\rceil C_k^L. \tag{5.15}$$

Theorem 10. *AMC is sustainable w.r.t. to all parameters*

Proof. The proof will contain two parts – one for showing sustainability w.r.t. criticality levels, and the other for the remaining parameters:

Sustainability w.r.t WCETs, periods, and relative deadlines. It has been proved by Baruah and Burns in (Burns and Baruah, 2008) that the response time analysis of fixed priority preemptive task system is sustainable w.r.t parameters such as execution requirements (C_i), relative deadlines (D_i) and periods (T_i). Thus Conditions (5.13) and (5.14) will hold when we adjust the parameters.

With respect to Condition (5.15), although the value of R_i^L is fixed, decreasing C_k^L and increasing T_k will deplete the overall value of response time R_i^*. The modified value of response time can be recursively determined until a value less than the initial response time is obtained. The altered value of R_i^* is acquired from recursive calculations and can be represented as:

$$new(R_i^*) \leq R_i^* \leq D_i$$

The above equation satisfies the schedulability condition for AMC scheduling algorithm. For all tasks $\tau_i \in \tau_{LO,HI}$ the response time R_i^L and R_i^H are no larger than the relative deadline D_i. It is observed that the amount of execution available to a task τ_i over a period [0,t) can only increase if job execution requirements decrease. The similar rationale is applied when job periods increase, i.e., modifying the parameters accordingly only guarantees the execution of the task in $[0, R_i]$.

Consequently, the AMC scheduling model is sustainable with respect to execution-requirements, periods and relative deadlines.

SUSTAINABILITY W.R.T CRITICALITY LEVELS. As mentioned earlier, the AMC scheduling policy employs an optimal priority assignment technique before scheduling the jobs. (Baruah *et al.*, 2011a) states that Audsley's priority assignment algorithm delivers an optimal priority ordering in polynomial time, i.e., Audsley's algorithm is guaranteed to find a priority assignment, if there exists one, which is AMC-schedulable. If we change the criticality level of a task from HI to LO, the priority of the task may remain the same or decrease; if another feasible priority assignment exists, it will be determined by the Audsley's algorithm. In case no other feasible priority order exists, the available order of the task-set before the criticality level modification can be used as the valid priority ordering. Since the order is already AMC-schedulable, we claim that it is sustainable w.r.t. criticality levels.

□

5.1.4. OCBP for MC Job Scheduling. The OCBP (Own Criticality Based Priority) scheduling policy (Baruah *et al.*, 2010) is a priority based MC-job scheduling algorithm.

It derives a valid priority ordering of the jobs prior to run-time in order to guarantee a correct schedule. These priorities are assigned in a recursive manner following Audsley's approach (Audsley, 2001). That is, a job J_i is assigned lowest priority if it meets its deadline, while J_i and all other jobs (of higher priority) execute for a duration not exceeding their WCETs estimated at J_i's own criticality level χ_i. If such a job J_i is found, then it is assigned lowest priority and the process repeated on the remaining (higher-priority) jobs.

Specifically if the candidate job J_i of LO-criticality is assigned lowest priority, the following set of conditions will be checked for any l such that $l \in hp(i)$[1] and $a_l \leq d_i$:

$$C_i^L + \sum_{j \in hp(i) \cap a_j \geq a_l} C_j^L \leq d_i - a_l. \tag{5.16}$$

[1] $hp(i)$ indicates the set of jobs with higher priority assignment than J_i.

While if J_i is of HI-criticality, for any l such that: $l \in hp(i)$ and $a_l \leq d_i$, we check[2]:

$$C_i^H + \sum_{j \in hp(i) \cap \chi_j = \mathbf{HI} \cap a_j \geq a_l} C_j^H \leq d_i - a_l. \tag{5.17}$$

It is relatively straightforward to implement this priority-assignment process in such a manner that the following assumption is satisfied:

ASSUMPTION 2: *Upon changing some parameter of a single job, the priority assigned to this particular job may be different from the original but the relative priority order of other jobs remains the same.*

This assumption can be achieved by restricting the order of the jobs in each iteration while determining a lowest priority job; e.g., in decreasing deadline order or simply following job indices.

Theorem 11. *OCBP is sustainable for all parameters under Assumption 2.*

Proof. Assume that an instance J of dual-criticality jobs is OCBP schedulable, and modify the parameter of a particular job $J_i \in J$ by one of the four actions: decrease its release time by δ, increase its deadline by δ, decrease its LO- or HI-WCET by δ, or change its criticality level from HI to LO ($C_i^L \leq C_i^H$) to obtain a new job set J' (while parameters of other jobs remains unchanged). According to Audsley's priority assignment algorithm and Assumption 2, there are three possible scenarios upon assigning priorities to J': (1) the job J_i is assigned a higher priority than before, (2) the priority order of all jobs does not change, and (3) the job J_i is assigned a lower priority than before.

We first show that Case (1) is not possible. Since OCBP is a fixed priority scheme, the schedulability of J_i is only affected by the higher priority jobs. If J_i is assigned priority p_i at a certain iteration before changing the parameter, we would make the same attempt to assign it the lowest priority at that round, with the same higher priority jobs left (according

[2]Note that none of the existing work stated the math conditions for OCBP to be schedulable – this is part of our contribution in this paper as sustainability proof requires clearly expressed equations.

to Assumption 2). Since it is schedulable before the parameter change, the claim is that J_i will continue be assigned the lowest priority at that round (if not sooner). The reason is that changing J_i's parameters in the given manner will just relax Conditions (5.16) and (5.17) such that the schedulaiblity test on current priority assignment remains a success.

For Case (2), sustainability also holds as Conditions (5.16) and (5.17) will continue to subsist for other jobs. For J_i, again the condition are more relaxed and will continue to hold. As a result, OCBP will return success after change in the parameters.

For Case (3), since parameter changing is leading to relaxation of original conditions, it is possible that J_i can be assigned a priority earlier than before; i.e., a lower priority than the one before such change. Figure 5.2 depicts the priority assignment of the jobs before and after incorporating the criticality level change.

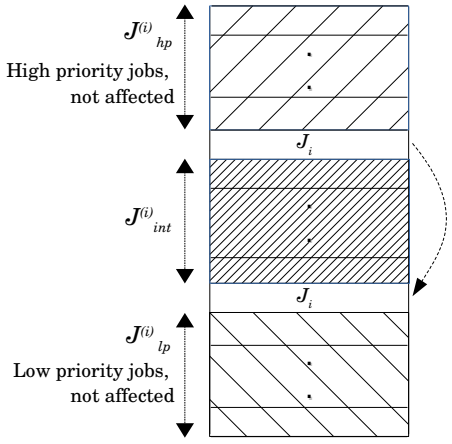


Figure 5.2. Priority assignment before and after the change of job J_i's criticality level from HI to LO.

The first shaded section $J^{(i)}_{hp}$ comprises jobs that initially have a higher priority than job J_i. Similarly, after J_i is assigned a lower priority, the jobs with still lower priority than J_i forms a set denoted by $J^{(i)}_{lp}$.

The densely shaded region in the middle, denoted by $J_{int}^{(i)}$; is the rest of jobs that are originally assigned lower priority than J_i and then higher than J_i after its parameter gets changed.

The entire job set can be represented as:

$$J_{hp}^{(i)} \cap J_i \cap J_{int}^{(i)} \cap J_{lp}^{(i)}.$$

It is assumed that the priority order within each subset does not change (as we restrict OCBP to try the same order each round). We first know that J_i's schedualbility conditions are satisfied.

For the rest of the jobs in three sets:

- The schedulability conditions of jobs in $J_{hp}^{(i)}$ is never affected as lower priority jobs have nothing to do with their priority assignment check.
- The schedulability conditions of jobs in $J_{int}^{(i)}$ will hold as for any job in this set, there is one less higher priority job (J_i) after the change.
- The schedulability conditions of jobs in $J_{lp}^{(i)}$ will hold as well, since for them the higher priority job set remains the same, while one of them, J_i, has less interference than before due to the parameter change.

We can thus claim that the job-set is OCBP-schedulable after the parameter change of job J_i.

□

5.2. SUSTAINABILITY IN MULTIPROCESSOR SCHEDULING ALGORITHMS

We now study the sustainability properties of two multiprocessor MC scheduling algorithms.

5.2.1. MC2. The MC^2 algorithm (Mollison *et al.*, 2010) employs a hierarchical scheduling approach: special tasks called container tasks are scheduled alongside the higher-criticality tasks. LO-criticality tasks will be assigned to containers (i.e., servers) and will use the container's budget to execute only when the container task is scheduled for execution in the platform. Tasks at each criticality level are scheduled by different intra-container schedulers, and thus according to different scheduling policies. Four criticality levels are considered in (Mollison *et al.*, 2010) – A, B, C, and D. Level-A tasks adopt a table-driven approach modeled on a cyclic executive scheduler, with tasks statically assigned to processors and scheduling tables precomputed prior to runtime. Each processor also hosts a level-B container, to which level-B tasks are assigned. Partitioned EDF is used at level B, so each level B container is served by an EDF scheduler. The periods of all level-B tasks are required to be integer multiples of the level-A hyperperiod, and the sum of the utilizations of all level-A and level-B tasks must not exceed 1.0. Both level-A and level-B tasks are guaranteed to meet their deadlines. Level-C tasks are grouped into the Level-C container which is served by all processors and is scheduled using global EDF. Level-C tasks are guaranteed only for soft real-time correctness (i.e., with bounded tardiness). G-EDF is executed on any processor whenever some level-C task is eligible but no higher-criticality tasks (level-A or -B) are eligible. At level D, best effort jobs are scheduled by a server that is invoked whenever a processor would otherwise be idle – no guarantee is made to those.

SUSTAINABILITY W.R.T WCET. As stated earlier, the cyclic executive execution of level A tasks is table driven and the execution order is determined off-line. Therefore on decreasing WCET of a level-A task, a modified schedule is established off-line according to which tasks are dispatched. For an existing MC^2-schedulable task set, on decreasing the WCET of a level B task by a small value $\delta > 0$, the utilization of the task decreases, which results in an easier partitioning problem to obtain a partitioned-EDF schedule. Thus,

the schedulability of the set will be maintained. For level-C tasks, MC^2 only guarantees the tardiness bound instead of hard real-time constraints. We therefore conclude that MC^2 algorithm is schedulable w.r.t to execution time.

SUSTAINABILITY W.R.T RELATIVE DEADLINES AND PERIODS. On increasing the period/deadline of the task by δ, the schedulability conditions should hold in order to establish sustainability. Since tasks of level A are statically scheduled while level B and C adopt partitioned-EDF and global-EDF respectively, on increasing the deadline by a small value δ, the schedulability conditions are not affected adversely and continue to persist.

SUSTAINABILITY W.R.T CRITICALITY LEVEL. We separately consider the cases where a task's criticality level is lowered from A to B, B to C, and C to D.

Level A to B: Levels A and B are criticality-monotonically partitioned; since we have shown (Theorem 8) that criticality monotonic is not sustainable, it follows that MC^2 is not sustainable w.r.t criticality level for level-A tasks.

Level B to C: At level C, tasks are allocated at instants when the processor is available and not consumed by tasks of levels A and B.

Theorem 12. *The MC^2 scheduling algorithm is* not *sustainable with respect to the criticality-level change $B \rightarrow C$.*

Table 5.2. A mixed-criticality task-set which is not sustainable under MC^2 scheduling policy.

Task	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	τ_7	τ_8	τ_9	τ_10	τ_11	τ_12	τ_13	τ_14	τ_15
Crit.	A	A	A	A	A	A	B	B	B	B	B	B	G*	G*	G*
CPU	1	1	2	2	3	3	1	1	2	2	3	3	1	2	3
T_i	5	10	10	5	10	10	10	20	20	10	10	20	20	20	20
C_i^A	3	4	4	3	6	4	-	-	-	-	-	-	-	-	-
C_i^B	1	1	2	2	2	1	6	2	4	3	4	6	-	-	-
C_i^C	1	1	2	1	2	1	6	1	3	2	3	3	1	2	1

G indicates Level-C container which is served by all processors is scheduled using global EDF

Proof of Theorem 12: The MC^2 scheduling algorithm is not *sustainable with respect to criticality levels.*

Proof. Consider the multi-criticality task-set shown in Table 5.2. We first show that the given example is MC^2-schedulable.

Calculating U_i^C for each task:

$$U_1^C = 1 - \left(\frac{1}{5} + \frac{1}{10} + \frac{6}{10} + \frac{1}{20}\right) = \frac{1}{20}$$

$$U_2^C = 1 - \left(\frac{3}{20} + \frac{2}{10} + \frac{2}{10} + \frac{1}{5}\right) = \frac{5}{20}$$

$$U_3^C = 1 - \left(\frac{3}{10} + \frac{3}{20} + \frac{2}{10} + \frac{1}{10}\right) = \frac{5}{20}$$

Substituting in Equation (5.18),

$$\frac{1}{20} + \frac{2}{20} + \frac{1}{20} = \frac{4}{20} < \frac{11}{20}$$

Substituting in Equation (5.19),

$$\frac{11}{20} - 2 \cdot \left(\frac{1}{20}\right) - \left(\frac{1}{20} + \frac{2}{20}\right) > 0$$

The tardiness is thus bounded at level C.

The next step is to decrease the criticality level of task τ_7 from B to C, and check if the schedulability condition (tardiness bounds) still holds. The schedulability conditions are given in Equations (5.18) and (5.19). On substituting the values from the table,

$$U_1^C = 1 - \left(\frac{1}{5} + \frac{1}{10} + \frac{1}{20}\right) = \frac{13}{20}$$

$$U_2^C = 1 - \left(\frac{3}{20} + \frac{2}{10} + \frac{2}{10} + \frac{1}{5}\right) = \frac{5}{20}$$

$$U_3^C = 1 - \left(\frac{3}{10} + \frac{3}{20} + \frac{2}{10} + \frac{1}{10}\right) = \frac{5}{20}$$

Substituting in Equation (5.18),

$$\frac{1}{20} + \frac{2}{20} + \frac{1}{20} + \frac{6}{10} = \frac{16}{20} < \frac{23}{20}$$

Substituting in Equation (5.19),

$$\frac{23}{20} - 2 \cdot \left(\frac{6}{10}\right) - \left(\frac{6}{10} + \frac{2}{20}\right) \not> 0$$

The second tardiness bound Condition (5.19) does not hold. Thus we conclude that the MC^2 scheduling policy is not sustainable w.r.t. criticality levels.

The example above illustrates the non-sustainability of MC^2 upon changes in criticality level – a task's criticality decreasing from level B to level C rendered a schedulable system unschedulable. We now provide some insight into why sustainability failed to hold in our example.

There are two conditions to demonstrate that the tardiness is bounded (Mollison *et al.*, 2010). The first condition is:

$$\sum_{i:\chi_i=C} u_i^C \leq \sum_{k=1}^{m} (1 - u_{(k)}^{AB}), \tag{5.18}$$

where $\tau_{(k)}^{AB}$ denotes the set of tasks on processor k above level C, and $1 - u_{(k)}^{AB}$ is the available utilization on processor k after assigning level-A and level-B tasks.

Thus, when changing the criticality level from B to C, the task is added to the level-C container (serving by all the processors). The increase in utilization available for level C tasks is proportional to the drop in $u_{(k)}^{AB}$ (utilization of tasks in level A and B). Thus the Equation (5.18) is always satisfied.

However, the second condition for the tardiness bound for level-C may not hold as we make such changes, which is originally given by:

$$\sum_{k=1}^{m} (1 - u_{(k)}^{AB}) > (m-1) \cdot \max_{i:\chi_i=C} u_i^C + U_{max(m-1)}^C. \tag{5.19}$$

Here $U^C_{max(m-1)}$ denotes the sum of the m-1 largest u^C_i values of tasks T_i which belong to task system τ. It is possible that the task of interest, whose criticality level is changed from B to C, has a maximum very high utilization value, such that the increase in right hand side of Equation (5.19) is much more significant than the gain on the left hand side (difference by *m-2* times its utilization), leading to a violation of the condition.

□

Level C to D: Since no guarantee is made for level-D task, such change is trivially sustainable (although rather meaningless).

Overall, we conclude that MC^2 is not sustainable w.r.t criticality level in general.

5.2.2. MC-Fluid. In the MC Fluid scheduling algorithm (Lee *et al.*, 2014), scheduling occurs under a fluid scheduling model which allows for schedules in which an individual task may be assigned a fraction of a processor at each time instant.

Each job of each task τ_i is executed at a rate of θ^L_i under LO-criticality mode, and another at a rate of θ^H_i after a mode switch (with $\theta^H_i = 0$ for all LO tasks).

The MC-Fluid schedulability conditions (Lee *et al.*, 2014) for a task set τ and associated LO- and HI-mode execution rates (θ^L_i and θ^H_i) is MC-schedulable under MC-Fluid if and only if the following set of conditions:

$$\forall \tau_i \in \tau, \theta^L_i \geq u^L_i \tag{5.20}$$

$$\forall \tau_i \in \tau_H, \frac{u^L_i}{\theta^L_i} + \frac{u^H_i - u^L_i}{\theta^H_i} \leq 1, \tag{5.21}$$

$$\sum_{\tau_i \in \tau} \theta^L_i \leq m, \tag{5.22}$$

$$\sum_{\tau_i \in \tau_H} \theta^H_i \leq m \tag{5.23}$$

We now establish the sustainability of the MC-Fluid scheduling algorithm with respect to different parameters.

Lemma 4. *MC-Fluid is sustainable w.r.t WCET and period.*

Proof. According to the definition of sustainability, on decreasing the WCET of a task τ_i (either C_i^L or C_i^H) and/or increasing the time period, the schedulability conditions of the whole task system will still hold.

To analyze sustainability w.r.t to execution amounts C_i and time period T_i, for a task $\tau_i \in \tau_L$, we decrease C_i^L by a small arbitrary value and/or increase period (T_i). As a result, all modifications can be modeled as a decrease of LO-utilization (u_i^L) by an amount of $\delta > 0$. We then examine the conditions one by one.

Equation (5.20) can be written as:

$$\forall \tau_i \in \tau, \theta_i^L \geq (u_i^L - \delta)$$

and is true for any value of $\tau_i \in \tau_L$.

Consider Equation (5.21) where $\tau \in \tau_H$, we determine the effect of decreasing C_i^L and C_i^H on Equation (5.21). On decreasing the value of C_i^L by δ, the u_i^L also decreases. Substituting in Equation (5.21) we get:

$$\forall \tau_i \in \tau_H, \frac{u_i^L - \delta}{\theta_i^L} + \frac{u_i^H - (u_i^L - \delta)}{\theta_i^H} \leq 1. \tag{5.24}$$

In order to prove that the condition still holds, we subtract the left hand side of Equation (5.24) from that of Equation (5.21) and establish that it is greater than zero.

$$\frac{u_i^L}{\theta_i^L} + \frac{u_i^H - u_i^L}{\theta_i^H} - \frac{u_i^L - \delta}{\theta_i^L} - \frac{u_i^H - (u_i^L - \delta)}{\theta_i^H} \geq 0$$

$$\Leftrightarrow \frac{\delta}{\theta_i^L} - \frac{\delta}{\theta_i^H} \geq 0$$
$$\Leftrightarrow \delta(\frac{1}{\theta_i^L} - \frac{1}{\theta_i^H}) \geq 0$$

The above equation can be easily validated since Equation (5.21) only considers HI-criticality tasks, where $\theta_i^L \leq \theta_i^H$ holds.

It is obvious that Conditions (5.22) and (5.23) will not get affected by utilization changes.

We can thus conclude that MC-Fluid scheduling is sustainable w.r.t the execution time (C_i) and time period (T_i). □

Lemma 5. *MC-Fluid is sustainable w.r.t criticality levels.*

Proof. To check the sustainability of the scheduling model w.r.t. the criticality levels. i.e., if we change the criticality of a task from HI to LO, then Condition (5.21) no longer needs to be validated for this task. Thus if the original conditions can be satisfied, the new condition is a strict relaxation of it, and so will be the execution rate. Since MC-Fluid is optimal in rate searching; i.e., whether there exist a feasible rate assignment, MC-Fluid will find it, we claim that it is sustainable with respect to criticality levels. □

Theorem 13. *MC-Fluid is sustainable w.r.t all input parameters.*

Proof. This follows from Lemmas 4 and 5. □

Sustainable schedulability tests ensure that a system that has been successfully verified will meet all its deadlines at run-time even if its operating parameters change for the better during system run-time. It has been argued (Baker and Baruah, 2009; Burns and Baruah, 2008) that from an engineering perspective, sufficient and sustainable tests are more useful than exact but non-sustainable tests. Here we have analyzed, for the first time, the sustainability properties of a variety of widely studied mixed-criticality scheduling algorithms. While all are sustainable with respect to the parameters WCET, period, and deadline, which MC models inherit from traditional (i.e., non-MC) models, it turns out that Criticality-Monotonic and MC^2 schedulability analysis are not sustainable with respect to criticality level.

5.3. SUMMARY OF SUSTAINABILITY IN MIXED-CRITICALITY SCHEDULING

Sustainable schedulability tests ensure that a system that has been successfully verified will meet all its deadlines at run-time even if its operating parameters change for the better during system run-time. It has been argued (Baker and Baruah, 2009; Burns and Baruah, 2008) that from an engineering perspective, sufficient and sustainable tests are more useful than exact but non-sustainable tests. Here we have analyzed, for the first time, the sustainability properties of a variety of widely studied mixed-criticality scheduling algorithms. While all are sustainable with respect to the parameters WCET, period, and deadline, which MC models inherit from traditional (i.e., non-MC) models, it turns out that Criticality-Monotonic and MC^2 schedulability analysis are not sustainable with respect to criticality level.

6. CONCLUSION

The main objective of this thesis is to enable efficient and precise scheduling in mixed-criticality systems by integrating the DVFS technique for efficient energy consumption and the precise scheduling model to decrease the penalization of LO-criticality tasks in HI-criticality behaviors. In this thesis we consider the scheduling of an MC-workload comprising of implicit-deadline sporadic tasks upon preemptive uniprocessor with varying speeds. The processor runs at an energy conserving speed during LO-criticality behaviors in order to minimize energy consumption. The popular EDF-VD algorithm was modified to adapt to the precise energy-efficient model. It has been proved that (i) the modified EDF-VD algorithm has a speedup factor of 2 (Theorem 5) (ii) no non-clairvoyant algorithm can have a speedup factor better than 2 (Theorem 6). Another contribution that highlights this work is the approximation ratio presented for the EDF-VD algorithm corresponding to our model. Extensive experiments were conducted to reveal the behavior of EDF-VD on randomly-generated task systems.

The second contribution is this thesis is the sustainability analysis of popular mixed-criticality schedulers on both uniprocessor and multiprocessor platforms [1]. Sustainable schedulability tests establish validation that the system will meet it deadlines during run-time in the event that the parameters change due to better performance. In this thesis, the sustainability properties of a variety of widely studied mixed-criticality scheduling algorithms have been analyzed. While all are sustainable with respect to the parameters WCET, period, and deadline, which MC models inherit from traditional (i.e., non-MC) models, it turns out that Criticality-Monotonic and MC^2 schedulability analysis are not sustainable with respect to criticality level.

[1] This work was published in the Real-Time Systems Symposium (RTSS 2017)(Guo *et al.*, 2017)

Future Direction: For the precise energy efficient scheduling of MC tasks using EDF-VD, we seek to derive and prove a tighter bound for the approximation ratio (if one exists), work on counter examples to show the minimum possible bound, and also explore schedulability conditions under a task-wise mode-switch, contrary to the system-wise mode switch adopted in this work. We also wish to conduct simulation study on actual energy savings with on-board implementations.

The study of sustainability of scheduling algorithms and schedulability tests has been restricted to an off-line analysis, where parameter changes occur prior to run-time. There is another aspect to sustainability, dealing with *dynamic* changes to parameters during run-time. It would be interesting to study sustainability properties of mixed-criticality scheduling algorithms under such a dynamic interpretation.

REFERENCES

Audsley, N. C., 'On priority assignment in fixed priority scheduling,' Information Processing Letters, 2001, **79**(1), pp. 39–44.

Baker, T. P. and Baruah, S. K., 'Sustainable multiprocessor scheduling of sporadic task systems,' in 'Proceedings of the 21st Euromicro Conference on Real-Time Systems (ECRTS),' 2009 pp. 141–150.

Baruah, S. and Agarwal, K., 'Intractability issues in mixed-criticality scheduling,' in 'Proceedings of the 30th EuroMicro Conference on Real-Time Systems (ECRTS), IEEE,' IEEE, 2018 .

Baruah, S., Bonifaci, V., DAngelo, G., Li, H., Marchetti-Spaccamela, A., Van Der Ster, S., and Stougie, L., 'The preemptive uniprocessor scheduling of mixed-criticality implicit-deadline sporadic task systems,' in 'Proceedings of the 24th Euromicro Conference on Real-Time Systems (ECRTS), IEEE,' IEEE, 2012a pp. 145–154.

Baruah, S., Bonifaci, V., D'Angelo, G., Li, H., Marchetti-Spaccamela, A., van der Ster, S., and Stougie, L., 'The preemptive uniprocessor scheduling of mixed-criticality implicit-deadline sporadic task systems,' in 'Proceedings of the 24th Euromicro Conference on Real-Time Systems (ECRTS),' 2012b .

Baruah, S., Bonifaci, V., D'angelo, G., Li, H., Marchetti-Spaccamela, A., Van Der Ster, S., and Stougie, L., 'Preemptive uniprocessor scheduling of mixed-criticality sporadic task systems,' Journal of the ACM (JACM), 2015, **62**(2), p. 14.

Baruah, S. and Burns, A., 'Sustainable scheduling analysis,' in 'Proceedings of the 27th IEEE Real-Time Systems Symposium (RTSS),' 2006 pp. 159–168.

Baruah, S., Burns, A., and Davis, R., 'Response-time analysis for mixed criticality systems,' in 'Proceedings of the 32nd IEEE Real-Time Systems Symposium (RTSS),' 2011a .

Baruah, S., Burns, A., and Guo, Z., 'Scheduling mixed-criticality systems to guarantee some service under all non-erroneous behaviors,' in 'Proceedings of the 28th Euromicro Conference on Real-Time Systems (ECRTS), IEEE,' IEEE, 2016 pp. 131–138.

Baruah, S. and Guo, Z., 'Scheduling mixed-criticality implicit-deadline sporadic task systems upon a varying-speed processor,' in 'Proceedings of the 35th Real-Time Systems Symposium (RTSS), IEEE,' IEEE, 2014 pp. 31–40.

Baruah, S., Li, H., and Stougie, L., 'Towards the design of certifiable mixed-criticality systems,' in 'Proceedings of the 16th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS),' 2010 .

Baruah, S. K., Bonifaci, V., DâĂŹAngelo, G., Marchetti-Spaccamela, A., Van Der Ster, S., and Stougie, L., 'Mixed-criticality scheduling of sporadic task systems,' in 'European Symposium on Algorithms,' Springer, 2011b pp. 555–566.

Benini, L., Bogliolo, A., Paleologo, G. A., and De Micheli, G., 'Policy optimization for dynamic power management,' IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 1999, **18**(6), pp. 813–833.

Burns, A. and Baruah, S., 'Sustainability in real-time scheduling,' Journal of Computing Science and Engineering, 2008, **2**(1), pp. 74–97.

Burns, A. and Baruah, S., 'Towards a more practical model for mixed criticality systems,' in 'Workshop on Mixed-Criticality Systems (colocated with RTSS),' 2013 .

Burns, A. and Davis, R. I., 'A survey of research into mixed criticality systems,' ACM Computing Surveys (CSUR), 2017, **50**(6), p. 82.

Easwaran, A., 'Demand-based scheduling of mixed-criticality sporadic tasks on one processor,' in 'Proceedings of the 34th Real-Time Systems Symposium (RTSS), IEEE,' IEEE, 2013 pp. 78–87.

Ekberg, P. and Yi, W., 'Bounding and shaping the demand of generalized mixed-criticality sporadic task systems,' Real-time systems, 2014, **50**(1), pp. 48–86.

Ernst, R. and Di Natale, M., 'Mixed criticality systems - A history of misconceptions?' IEEE Design & Test, 2016, **33**(5), pp. 65–74.

Esper, A., Nelissen, G., Nélis, V., and Tovar, E., 'How realistic is the mixed-criticality real-time system model?' in 'Proceedings of the 23rd International Conference on Real Time and Networks Systems (RTNS),' 2015 pp. 139–148.

Graydon, P. and Bate, I., 'Safety assurance driven problem formulation for mixed-criticality scheduling,' Proceedings of the Workshop on Mixed Criticality (WMC), 2013, pp. 19–24.

Guan, N., Ekberg, P., Stigge, M., and Yi, W., 'Improving the scheduling of certifiable mixed-criticality sporadic task systems,' Technical Report 2013–008, 2013.

Guo, Z., Sruti, S., Ward, B. C., and Baruah, S., 'Sustainability in mixed-criticality scheduling,' in '2017 IEEE Real-Time Systems Symposium (RTSS),' IEEE, 2017 pp. 24–33.

Huang, P., Kumar, P., Giannopoulou, G., and Thiele, L., 'Energy efficient dvfs scheduling for mixed-criticality systems,' in 'Proceedings of the 14th International Conference on Embedded Software, ACM,' ACM, 2014 p. 11.

Huang, P., Kumar, P., Giannopoulou, G., and Thiele, L., 'Run and be safe: Mixed-criticality scheduling with temporary processor speedup,' in 'Design, Automation & Test in Europe Conference & Exhibition (DATE), 2015,' IEEE, 2015 pp. 1329–1334.

Jan, M., Zaourar, L., and Pitel, M., 'Maximizing the execution rate of low criticality tasks in mixed criticality system,' Proc. WMC, RTSS, 2013, pp. 43–48.

Kalyanasundaram, B. and Pruhs, K., 'Speed is as powerful as clairvoyance,' Journal of the ACM (JACM), 2000, **47**(4), pp. 617–643.

Lee, J., Phan, K.-M., Gu, X., Lee, J., Easwaran, A., Shin, I., and Lee, I., 'Mc-fluid: Fluid model-based mixed-criticality scheduling on multiprocessors,' in 'Proceedings of the 35th IEEE Real-Time Systems Symposium (RTSS),' IEEE, 2014 pp. 41–52.

Liu, C. L. and Layland, J. W., 'Scheduling algorithms for multiprogramming in a hard-real-time environment,' Journal of the ACM (JACM), 1973, **20**(1), pp. 46–61.

Liu, D., Spasic, J., Guan, N., Chen, G., Liu, S., Stefanov, T., and Yi, W., 'Edf-vd scheduling of mixed-criticality systems with degraded quality guarantees,' in 'Proceedings of the 37th Real-Time Systems Symposium (RTSS), 2016 IEEE,' IEEE, 2016 pp. 35–46.

Mollison, M. S., Erickson, J. P., Anderson, J. H., Baruah, S. K., and Scoredos, J. A., 'Mixed-criticality real-time scheduling for multicore systems,' in 'Proceedings of the 10th IEEE International Conference on Computer and Information Technology (CIT),' 2010 pp. 1864–1871.

Narayana, S., Huang, P., Giannopoulou, G., Thiele, L., and Prasad, R. V., 'Exploring energy saving for mixed-criticality systems on multi-cores,' in 'Proceedings of the 22nd Real-Time and Embedded Technology and Applications Symposium (RTAS), IEEE,' IEEE, 2016 pp. 1–12.

Pathan, R. M., 'Improving the quality-of-service for scheduling mixed-criticality systems on multiprocessors,' in 'LIPIcs-Leibniz International Proceedings in Informatics,' volume 76, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017 .

Paulitsch, M., Duarte, O. M., Karray, H., Mueller, K., Muench, D., and Nowotsch, J., 'Mixed-criticality embedded systems–a balance ensuring partitioning and performance,' in 'Proceedings of the 2015 Euromicro Conference on Digital System Design (DSD),' 2015 pp. 453–461.

S. Baruah and A. Burns, 'Towards a more practical model for mixed criticality systems,' in 'Proceedings of the Workshop on Mixed-Criticality Systems (WMC),' 2014 .

Su, H. and Zhu, D., 'An elastic mixed-criticality task model and its scheduling algorithm,' in 'Proceedings of the Conference on Design, Automation and Test in Europe,' EDA Consortium, 2013 pp. 147–152.

Vestal, S., 'Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance,' in 'Proceedings of the 28th IEEE Real-Time Systems Symposium (RTSS),' 2007 .

VITA

The author, Sai Sruti, was born in Odisha, India. She developed a passion for technology early in life and received her bachelors degree in Computer Science from India, in May 2016. After her bachelor's she enrolled in Missouri University of Science and Technology for graduate studies. During her time at Missouri S&T, the author worked as a Graduate Research Assistant under Dr. Zhishan Guo, from August 2016 to May 2018.

In partial fulfillment of the requirements for the Master of Science in Computer Science degree from Missouri University of Science and Technology, this thesis is the culmination of that degree. The author obtained her Master of Science in Computer Science from Missouri University of Science and Technology in July 2018.