Tractable Real-Time Schedulability Analysis for Mode Changes under Temporal Isolation

Nathan Fisher  Masud Ahmed
Department of Computer Science
Wayne State University
fishern@cs.wayne.edu, masud@wayne.edu

Abstract—Real-time multimedia subsystems often require support for switching between different resource and application execution modes. To ensure that timing constraints are not violated during or after a subsystem changes mode, real-time schedulability analysis is required. However, existing time-efficient multi-mode schedulability analysis techniques for application-only mode changes are not appropriate for subsystems that require changes in the resource execution behavior (e.g., processors with dynamic power modes). Furthermore, all existing multi-mode schedulability analysis that handles both resource and application mode changes is highly exponential and not scalable for subsystems with a large number of mode configurations. We address the lack of tractable schedulability analysis for such subsystems by proposing a model for characterizing multiple resource and application modes and by deriving a sufficient schedulability test that has pseudo-polynomial time complexity. Simulation results show that our proposed schedulability test, when compared with previously-proposed approaches, requires significantly less time and is just as precise.

I. INTRODUCTION

The ability to change execution modes at runtime is a fundamental requirement for multimedia systems. For example, an application-level mode change may occur in an adaptive video-streaming application due to changes in decoding requirements. Similarly, a resource-level mode change may occur when a system reduces its computational capabilities (e.g., a reduced-power mode). Additionally, many embedded media systems (e.g., smartphones, digital video recorders, etc.) must support multiple simultaneously-executing subsystems (e.g., multimedia streams) on a shared computational platform. Thus, temporal isolation is desirable to ensure that each subsystem maintains its specified quality-of-service, even in the presence of application and resource-level mode changes. However, for multimedia applications with strict temporal constraints such as hard deadlines, guaranteeing both temporal isolation and schedulability is inherently difficult due to the challenge in predicting the aggregate computational resources and demand of a real-time application over any interval of time due to the mode changes.

Traditional real-time systems research has commonly addressed the issues of schedulability under mode changes and temporal isolation, separately and independently. For instance, traditional research in schedulability analysis of real-time multi-mode systems (e.g., see Tindell et al. [20]) has commonly assumed that the system is executing upon a dedicated processing platform. On the other hand, research on temporal isolation in real-time scheduling (often called server-based or hierarchical scheduling), while permitting the analysis of real-time subsystems that co-execute upon a shared computation platform, has often assumed that the application and resource requirements of each subsystem are fixed during runtime. Only recently have researchers started to address the problem of guaranteeing hard deadlines of temporally-isolated subsystems in multi-modal systems. However, most of this recent research suffers from two fundamental drawbacks: 1) full support for both resource-level mode changes or application-level mode changes does not exist, and/or 2) the proposed algorithms for determining schedulability under mode changes have exponential-time complexity. The lack of support for both resource and application mode changes severely limits the ability of the subsystem to adapt to dynamic internal and external events. The lack of tractable analysis for these mode change protocols are also an impediment to the design of multi-modal systems where a large number of system configurations may have to be explored during hardware-software co-design process and a schedulability test has to be executed a corresponding number of times. In this paper, we address these two fundamental drawbacks by providing a theoretical framework and associated tractable, hard-real-time schedulability analysis for subsystems executing in a temporally-isolated environment (i.e., in a periodic server) under both resource and application-level mode changes.

Contributions & Organization. Our paper\textsuperscript{1} makes the following contributions:

- We propose the discrete resource/application real-time multi-mode model in Section III.
- We derive a sufficient schedulability analysis test\textsuperscript{2} for the setting where the sequence of mode changes is a priori fixed and each mode has its workload scheduled by the earliest-deadline-first (EDF) scheduling algorithm [8] in Section IV. This setting is referred to as concrete mode-change request setting.

\textsuperscript{1}All proofs have been omitted from this paper; we have published an extended version [7] of this paper which contains all omitted proofs.

\textsuperscript{2}A sufficient schedulability test (given the resource and processing specification) returns true, only if the subsystem is guaranteed to meet all deadlines.
• We derive a pseudo-polynomial schedulability analysis test for the setting where mode changes are not a priori fixed (called non-concrete sequences) and each mode is scheduled by EDF in Section V. The pseudocode for our schedulability test is presented and explained in Section VI. The time complexity of our proposed test is a significant improvement over previous tests that require exponential-time in the worst case.

• We show via simulations in Section VII that our proposed test is effective (in terms of schedulability) by comparing with the known exponential-time test. Furthermore, we illustrate the scalability of our test with the previous test as the number of subsystem modes increases.

In the next section, we briefly present relevant previous research on multiple modes and temporal isolation in real-time systems.

II. RELATED WORK

Numerous results exist for ensuring timing guarantees during transitions between modes on both uniprocessor [20], [14], [11] and multiprocessor [10] systems under the assumption of a dedicated processing platform. For this paper, we restrict our attention to the uniprocessor setting. For application-level only mode changes, Tindell et al. [21] introduced a simple protocol where new-mode tasks wait until the processor completes old-mode tasks. This approach is known as a synchronous mode change protocol. Tindell et al. [20] defined a closed-form expression for calculating the waiting time (i.e., offset) after which a new-mode tasks can generate jobs. Pedro et al. [11] and Real et al. [14] explored asynchronous mode change protocols (where old-mode tasks may execute concurrently with new-mode tasks) and determined the effects of introducing an offset during the mode change on the system schedulability.

For ensuring temporal isolation between real-time subsystems co-executing on the same processing platform numerous server-based frameworks have been proposed (e.g., constant-bandwidth server (CBS) [1], sporadic server [17], and periodic resource model [16] are just a few examples). However, most of these frameworks and their associated schedulability analysis assume that the application and resource requirements of the subsystems executing upon the server are fixed prior to runtime. Subsequent work has attempted to remove this assumption. Frameworks such as elastic scheduling [5] and rate-based earliest deadline (RBED) [4] permit a subsystem to change its application or resource requirements adaptively; however, each of these previous results are soft-real-time in that they do not guarantee all deadlines are met under transitions between operating modes. An adaptive hard-real-time extension of CBS, called variable-bandwidth server (VBS) has been developed; however, VBS does not consider resource-level mode changes and does not permit arbitrary mode changes. (Instead an application mode change must pass an admission control test and be deferred until the mode change is safe).

In the past two years, there has been increased research attention on developing hard-real-time frameworks and analysis that support temporal isolation and both resource and application mode changes. Stoimenov et al. [18] developed a real-time calculus (RTC) approach for analyzing the application demand of a subsystem during a single application-level mode change. In a complementary paper, Santinelli et al. [15] developed an RTC characterization of the resource supply of a resource-level mode change. Taken together, these two results can be used to analyze the schedulability of a subsystem under a single mode change; however, the results, in general, do not address a subsystem with multiple successive mode changes. In a later paper, Stoimenov et al. [19] investigated resource-level mode changes in a Time Division Multiple Access (TDMA) server; however, schedulability analysis for application-level mode changes was not investigated. Furthermore, none of the schedulability analysis of the aforementioned results has known tractable time complexity. (The authors do not mention the time complexity of their approaches.) In contrast, our approach addresses multiple successive application and resource mode changes and has pseudo-polynomial time complexity.

One set of recent results has addressed multiple resource and application-level mode changes (without waiting for idle times); Phan et al. [12], [13] proposed a general compositional model and associated analysis techniques for processing multiple bursty/complex event/data streams using state-based models such as timed automaton. However, as their analysis is based on exploring a reachability graph, their approach is highly exponential and does not scale efficiently for an increasing number of modes. In contrast, we consider a more specialized model for resource and application modes. Our analysis permits a more precise and efficient calculation of subsystem schedulability, as shown in Section VII.

III. REAL-TIME MODE-CHANGE MODEL

A system is comprised of multiple subsystems. For this paper, we exclusively focus on the subsystem-level schedulability of a single subsystem \( S \) with \( q \) resource/application modes: \( M_1, \ldots, M_q \). Future work will address system-level schedulability of the temporal-isolation servers that execute the subsystems. With each subsystem mode \( M_i \), we will associate a term which describes the real-time workload (i.e., application execution) generated by \( M_i \) and a term which describes the amount of resource execution guaranteed by \( M_i \). To formally represent these two terms, we will use the characterizations of the sporadic task model and the explicit-deadline periodic resource model, respectively. In the following subsection, we formally give background on these two models. We then present our proposed model for resource and application-level modes and mode-change requests. Throughout this paper, we assume that subsystem and timing parameters are natural numbers. This assumption is not restrictive as all timing parameters may be expressed in terms of the number of CPU clock ticks.
A. Background

§ Sporadic Task Model. The sporadic task model [9] is a popular and general model for recurring real-time work often present in real-time embedded control systems. The application execution of each mode $M_i$ is represented by a sporadic task system $τ^{(i)} = \{τ_1^{(i)}, τ_2^{(i)}, \ldots, τ_n^{(i)}\}$. A sporadic task $τ_k^{(i)} = (e_k^{(i)}, d_k^{(i)}, p_k^{(i)})$ is characterized by a worst-case execution requirement $e_k^{(i)}$, a (relative) deadline $d_k^{(i)}$, and a minimum inter-arrival separation $p_k^{(i)}$ (also called a period). A sporadic task $τ_k^{(i)}$ may produce a (potentially infinite) sequence of jobs, where each job has an execution requirement of $e_k^{(i)}$ time units and must complete within $d_k^{(i)}$ time units after its arrival. The first job of $τ_k^{(i)}$ may arrive at any time after subsystem-start-time; however, successive jobs of $τ_k^{(i)}$ must arrive at least $p_k^{(i)}$ time units apart. We will assume that each task $τ_k^{(i)}$ has constrained deadlines; that is, $d_k^{(i)} ≤ p_k^{(i)}$. A useful metric for a sporadic task $τ_k^{(i)}$ is the task utilization $u_k^{(i)} \equiv e_k^{(i)}/p_k^{(i)}$ and $u^{(i)} \equiv \sum_k u_k^{(i)}$. In the next paragraph, we define a function for quantifying the maximum workload over time.

Definition 1 (Demand-Bound Function): For any $t > 0$ and task $τ_k^{(i)}$, the demand-bound function (dbf) quantifies the maximum cumulative execution requirements of all jobs of $τ_k^{(i)}$ that could have both an arrival time and deadline in any interval of length $t$. Baruah et al. [3] have shown that, for sporadic tasks, dbf can be calculated by $\text{dbf}(τ_k^{(i)}, t) = \max \left(0, \left\lfloor \frac{t - d_k^{(i)}}{p_k^{(i)}} + 1 \right\rfloor \cdot e_k^{(i)} \right)$. A. Background

§ Explicit-Deadline Periodic (EDP) Resource Model. The explicit-deadline periodic (EDP) resource model [6], [16] is a general resource model for characterizing the execution of a subsystem upon a periodically-available, non-continuously-executing resource. The processing resource available to each mode $M_i$ is represented by an EDP resource $Ω^{(i)}$. The resource $Ω^{(i)}$ is characterized by a three-tuple $(Π^{(i)}, Θ^{(i)}, Δ^{(i)})$ where $Π^{(i)}$ is the resource period, $Θ^{(i)}$ is the resource capacity, and $Δ^{(i)}$ is the resource deadline. The interpretation of $Ω^{(i)}$ is that the EDP resource guarantees mode $M_i$ an total execution of at least $Θ^{(i)}$ units over successive $Π^{(i)}$-length intervals within $Δ^{(i)}$ units of time. We will assume that $Δ^{(i)} ≤ Π^{(i)}$. Furthermore, we assume that EDF is used to schedule the workload at any point when the resource is providing execution.

The EDP resource model is general enough to model different types of periodic resource execution behavior. The model permits the capacity $Θ^{(i)}$ to be allocated in any manner within the $Δ^{(i)}$ resource deadline. The results contained in this paper (and for any EDP resource) hold regardless of the specific underlying resource execution allocation employed by the subsystem designer. (E.g., this model may be used to analyze a Time Division Multiple Access (TDMA) server [22] by setting $Π^{(i)}$ to the cycle length and $Θ^{(i)}$ and $Δ^{(i)}$ to the number of slots allocated to a subsystem.)

Definition 2 (Supply-Bound Function): For any $t > 0$, the supply-bound function $\text{sbf}(Ω^{(i)}, t)$ quantifies the minimum execution supply that a mode $M_i^{(i)}$ executed upon periodic resource $Ω^{(i)}$ may receive over any interval of length $t$. It has been shown [6] that $\text{sbf}(Ω^{(i)}, t)$ equals $yΘ^{(i)} + \max \left(0, t-x - yΠ^{(i)} \right)$ when $t ≥ Δ^{(i)} - Θ^{(i)}$ where $y \equiv \left\lceil \frac{t-(Δ^{(i)}-Θ^{(i)})}{Π^{(i)}} \right\rceil$ and $x \equiv (Π^{(i)} + Δ^{(i)} - 2Θ^{(i)})$; otherwise, the function equals zero.

B. Proposed Model

We are now prepared to describe our discrete resource / application real-time multi-mode model. Each mode $M_i$ is specified by a three-tuple $(τ^{(i)}, Ω^{(i)}, N^{(i)})$ which respectively characterizes the real-time workload generated by a sporadic task system, the minimum processor execution guaranteed by an EDP resource, and the minimum duration in terms of “number of resource periods” $N^{(i)}$. The interpretation of $N_i$ is that the subsystem remains in mode $M_i$ for at least $N^{(i)} Π^{(i)}$ time units.

§ Mode-Change Request (MCR) Model. At runtime, the subsystem switches between modes during a sequence $mcr_0, mcr_1, mcr_2, \ldots$ of mode-change requests. The $k$’th mode-change request $mcr_k$ (for $k > 0$) is characterized by a three-tuple $(M_i, M_j, t_k)$ where $t_k$ represents the transition time, $M_i$ is the old mode executing prior to $t_k$, and $M_j$ is the new mode executing after $t_k$ (where $M_i \neq M_j$ and $i, j \in \{1, \ldots, q\}$). We assume that if $i < j$, then $mcr_i$ occurs prior to $mcr_j$ (i.e., $t_i < t_j$); that is, the mode-change requests are indexed in ascending-time order. Mode-change request $mcr_0 \equiv (M_0, 0)$ represents transition from the null-mode $M_0$ to any mode in $\{M_1, \ldots, M_q\}$ at subsystem-start time (assumed to be zero). After $mcr_k = (M_i, M_j, t_k)$ has been issued at time $t_k$, there may be a transition period during which jobs generated by $M_i$ have not completed and $M_j$ has not yet begun to generate jobs.

For any $M_i, M_j \in \{M_1, \ldots, M_q\}$, we denote the length of the transition period (called the offset) by $δ_{ij}$. (Note, if there is no transition period, we set $δ_{ij} = 0$.) During the transition period after $mcr_k$, only (non-aborted) jobs of $τ^{(i)}$ are permitted to execute. At and after time $t_k + δ_{ij}$, task system $τ^{(j)}$ may generate and execute jobs along with any remaining execution of jobs from $τ^{(i)}$. During the transition period after $mcr_k$, the EDP resource model may also change execution behavior. We denote the resource parameters for the transition period between $M_i$ and $M_j$ by $Ω_{ij} \equiv (Π_{ij}, Θ_{ij}, Δ_{ij})$. We assume that the offset $δ_{ij}$ is some multiple of $Π_{ij}$. While some of jobs from $M_i$ may continue to execute in the transition period after $t_k$, the subsystem designer may choose to abort some jobs. We denote by $A^{(ij)}(≤ τ^{(i)})$ the set of tasks of $τ^{(i)}$ which abort non-completed jobs at the transition time (e.g., $t_k$) for any mode change from $M_i$ to $M_j$. The subsystem designer may want some tasks that are common to both mode $M_i$ and $M_j$ to be unaffected by the mode change request $mcr_k$. We denote these unchanged tasks by $τ^{(ij)}(≤ τ^{(i)} \cap τ^{(j)})$. Figure 1 illustrates a possible resource execution pattern between successive mode changes.
Given the above definitions, we may observe four phases with respect to mode-change request mcr\(_k\) = \((M_i, M_j, t_k)\) (and the previous request mcr\(_{k-1}\) = \((M_i, M_k, t_{k-1})\)):

1) jobs of \(\tau^{(i)}\) are generated and executed upon EDP resource \(\Omega^{(i)}\) over the interval \([t_{k-1} + \delta_{hi}, t_k)\);
2) (non-aborted) jobs of \(\tau^{(i)} \setminus A^{(ij)}\) with remaining execution at \(t_k\) may execute over \([t_k, t_k + \delta_{ij})\) on EDP resource \(\Omega^{(i)}\);
3) (non-aborted) jobs of \(\tau^{(i)} \setminus A^{(ij)}\) with remaining execution at \(t_k + \delta_{ij}\) and jobs generated by \(\tau^{(j)}\) may execute over \([t_k + \delta_{ij}, t_{k+1})\) upon EDP resource \(\Omega^{(j)}\); and
4) Unchanged tasks \((\tau^{(ij)}\) will act independent of mode change request over the interval \([t_{k-1} + \delta_{hi}, t_{k+1})\).

The above task classifications follow the taxonomy found in the real-time mode-change survey by Real and Crespo [14].

In a general subsystem, the interval of separation between successive mode-change requests may be determined by upper and lower bounds on the amount of time that a subsystem may execute in a given mode (e.g., the multi-mode abstraction proposed by Phan et al. [13]). In this paper, we restrict the mode-change requests and transition intervals to occur only at period boundaries in the EDP model and drop the specification of an upper bound on the separation of mode-change requests. That is, for any mode-change request mcr\(_k\) = \((M_i, M_j, t_k)\), \(\delta_{ij}\) must be a multiple of \(\Pi_{ij}\). Furthermore, for any two successive mode-change requests mcr\(_{k-1}\) = \((M_i, M_j, t_{k-1})\) and mcr\(_k\) = \((M_i, M_j, t_k)\), we require that

\[
t_k = t_{k-1} + \delta_{hi} + a\Pi^{(i)}
\]

for some \(a \in \mathbb{N}^+\) where \(a \geq N^{(i)}\). For instance, the above constraints are valid in control systems when sensing and actuation occur at strict periodic intervals and any mode change would occur at such a time point. We also assume that a non-aborted job may span no more than one mode-change request; i.e.,

\[
t_{k+1} - t_k \geq \max(d^{(h)}_{\text{max}}, d^{(hi)}_{\text{max}}).
\]

where \(d^{(h)}_{\text{max}} = \max_{\tau^{(i)} \in \tau^{(i)} \cup A^{(ij)}} \{d^{(h)}_{ij}\}\) represents the maximum (non-aborted) job deadline from mode \(M_h\) to \(M_i\) and \(d^{(hi)}_{\text{max}} = \max_{\tau^{(i)} \in \tau^{(i)} \cup A^{(ij)}} \{d^{(hi)}_{ij}\}\) represents the maximum (unchanged) job deadline from \(M_h\) to \(M_i\). To the best of our knowledge, all known real-time mode-change protocols implicitly or explicitly require this constraint.

IV. FOR CONCRETE SEQUENCE OF MCR

In the next section, we obtain schedulability conditions for a concrete sequence of mode change requests where each mode change request mcr\(_k\) is fixed a priori at subsystem design time. While concrete sequences of mode changes are unlikely to be present in most multi-mode and control systems, schedulability analysis developed for such sequences can be extended (as discussed in Section V) to the more practical scenario of non-concrete sequences of mode change requests (i.e., sequences in which the mode-changes requests are not a priori known).

In this section, we consider the following problem:

**Problem 1:** Given modes \(M_1, \ldots, M_q\), resources \(\Omega_{ij}\), offset \(\delta_{ij}\), unchanged tasks \(\tau^{(i)}\), and aborted tasks \(A^{(ij)}\) for all \(i, j \in \{1, \ldots, q\} \ (i \neq j)\), and concrete sequence of mode-change request mcr\(_0\), mcr\(_1\), mcr\(_2\), . . . that satisfies Equations 1 and 2, determine whether all jobs (under all legal job arrival sequence) are EDF-schedulable (i.e., EDF always meets each job’s deadline).

**Definitions.** A major challenge for schedulability analysis of multi-mode subsystems (over uni-mode systems) is dealing with the execution of non-aborted jobs from the old mode while a new mode is executing. If all tasks abort at the mode-change request, then the analysis would be identical to traditional uni-mode schedulability analysis. However, aborting jobs is not always appropriate, especially if aborting jobs may leave the subsystem in an unstable or ill-defined state. Thus, to be able to accurately determine the schedulability of multi-mode subsystems, we must precisely quantify the workload and demand that may carry-in from the old mode to the new mode for a mode-change request mcr\(_k\) = \((M_i, M_j, t_k)\). The following definitions with respect to mcr\(_k\) are useful in quantifying this workload. The definitions are with respect to a concrete sequence of mode-change request mcr\(_0\), mcr\(_1\), . . . , mcr\(_{k-1}\), mcr\(_k\). . . We will make use of an indicator function \(\mu_{\geq 0}(x)\) which is zero if \(x < 0\) and is one otherwise; we will also use the notation \((x)_+ = \max(0, x)\).

**Definition 3 (Carry-In Execution for mcr\(_k\)):** The carry-in execution for mode-change request mcr\(_k\) is the maximum remaining execution of non-aborted jobs from mode \(M_i\) for tasks \(\tau^{(i)} \setminus \{\tau^{(i)} \cup A^{(ij)}\}\) at time \(t_k + \delta_{ij}\) that arrive prior to \(t_k\) and maximum remaining execution of unchanged tasks (i.e., \(\tau^{(ij)}\)) that have arrival before \(t_k + \delta_{ij}\). We denote this value by \(\Theta^{(i)}(mcr\(_k\)).

**Definition 4 (Mode-Change DBF for mcr\(_k\)):** For mcr\(_{k-1}\) = \((M_h, M_i, t_{k-1})\) and mcr\(_k\) = \((M_i, M_j, t_k)\), the mode-change demand function for mode change request mcr\(_k\) and \(x, \phi \in \mathbb{R}_{\geq 0}\) is the maximum total execution demand of jobs of \(\tau^{(i)}\) in the interval \([t_k - x, t_k + \phi]\) (and any carry-in jobs when \(t_k - x\) corresponds to the end of a transition for mcr\(_{k-1}\) i.e., \(t_k - x = t_{k-1} + \delta_{hi}\)). For a task \(\tau^{(i)}_k \in \tau^{(i)}\), we denote its contribution to the total demand by mcdbf(mcr\(_k\), \(\tau^{(i)}_k\), \(x, \phi\)). The total demand of all jobs for the mode change is denoted by mcdbf(mcr\(_k\), \(x, \phi\)).

In other words, to be included in the demand, the jobs of changed or aborted tasks must arrive in the interval \([t_k - x, t_k]\) and have deadline in the interval \([t_k - x, t_k + \phi]\). For unchanged tasks of \(\tau^{(ij)}\), the mode-change demand function includes jobs that have an arrival and deadline in the interval \([t_k - x, t_k + \phi]\).
(i.e., we permit unchanged tasks that arrive after \( t_k \) to be included in the execution demand). The mode-change demand also includes the entire execution requirement of all non-aborted jobs generated in \([t_k - x, t_k]\) that have deadlines prior to \( t_k + \phi \), if \( \tau_k^{(i)} \in \tau^{(i)} \setminus A^{(i)} \); otherwise, the demand includes the execution of non-aborted jobs that arrive and have deadline in \([t_k - x, t_k]\) and only the completed portion of aborted jobs that arrive in \([t_k - x, t_k]\) and have deadlines in \((t_k, t_k + \phi]\), if \( \tau_k^{(i)} \in A^{(i)} \) (see Figure 4). If \( x \) equals \( t_k - t_{k-1} - \delta_{hi} \), then the total demand also includes the carry-in execution for \( \text{mcr}_{k-1} \) (i.e., \( \text{ciDBF}(\text{mcr}_{k-1}) \)). The demand \( \text{mdcbf}(\text{mcr}_{k}, x, \phi) \) may be computed by

\[
\sum_{\tau_k^{(i)} \in \tau^{(i)}} \text{mdcbf}(\text{mcr}_{k}, \tau_k^{(i)}, x, \phi) + \mu_{\geq 0}(x - (t_k - t_{k-1} - \delta_{hi})) \cdot \text{ci}(\text{mcr}_{k-1}).
\]

Definition 5 (Carry-In DBF): The carry-in demand-bound function for mode-change request \( \text{mcr}_k = (M_i, M_j, t_k) \) and \( \phi \in \mathbb{R}_{\geq 0} \) is the maximum remaining execution of jobs of tasks \( \tau_k^{(i)} \setminus A^{(i)} \) that arrive prior to \( t_k \) (or prior to \( t_k + \delta_{ij} \) for \( \tau_k^{(j)} \) tasks) and have deadline in the interval \([t_k, t_k + \phi]\) for any \( \text{mcr}_k = (M_i, M_j, t_k) \). We denote this quantity by \( \text{ciDBF}(\text{mcr}_k, \phi) \).

In the next three definitions, we define the minimum resource-execution supply function for three different scenarios: 1) before an MCR; 2) during the transition; and 3) after the MCR transition.

Definition 6 (Pre-Mode-Change SBF): The mode-change supply-bound function, prior to any mode-change request \( \text{mcr}_k = (M_i, M_j, t_k) \), is the minimum execution guaranteed by \( \Omega^{(i)} \) over the interval \([t_k - x, t_k]\). We denote this service by \( \beta_{\text{prior}}(M_i, x) \).

Definition 7 (Transition Mode-Change SBF): The mode-change supply-bound function, during the transition period mode-change request \( \text{mcr}_k \), is the minimum execution guaranteed by \( \Omega^{(j)} \) and \( \Omega^{(i)} \) over the interval \([t_k, t_k + \phi]\). We denote this service by \( \beta_{\text{trans}}(M_i, M_j, \phi) \).

Definition 8 (Post-Mode-Change SBF): The mode-change supply-bound function, following any mode-change request \( \text{mcr}_k = (M_i, M_j, t_k) \), is the minimum execution guaranteed by \( \Omega^{(j)} \) to carry-in jobs of \( M^{(j)} \) and by \( \Omega^{(i)} \) to \( M^{(i)} \) (and any carry-in jobs) over the interval \([t_k + \delta_{ij} - x, t_k + \delta_{ij} + y]\) (for \( 0 \leq x \leq \delta_{ij} \)). We denote this service by \( \beta_{\text{post}}(M_i, M_j, x, y) \).

A. Deriving MCR Service-Bound Function

In this subsection, we derive lower bounds on the supply functions of Definitions 6, 7, and 8. The first lemma gives a lower bound on the pre-mode-change SBF. Figure 2 illustrates that the worst-case allocation for the interval \([t_k - x, t_k]\) of \( \Omega^{(i)} \) occurs when the capacity is allocated at the beginning of each resource period.

Lemma 1: For \( \text{mcr}_k = (M_i, M_j, t_k) \) and \( x \geq 0 \),

\[
\beta_{\text{prior}}(M_i, x) \geq a\Theta^{(i)} + \min \left( \Theta^{(i)}, x - \left( (a + 1)\Pi^{(i)} - \Theta^{(i)} \right) \right) _{+}.
\]

where \( a \equiv \frac{x}{\Pi^{(i)}(\delta)} \).

\[
\begin{array}{c}
M_i \\
\theta^{(i)}_i \ \\
\theta^{(j)}_i \\
\delta_{ij} \\
\delta_{ij} \\
x \\
k_t \\
t_k \\
\end{array}
\]

Fig. 2: Minimum supply in \( x \) before \( \text{mcr}_k \).

The next two lemmas give bounds for the transition and post mode-change supply-bound functions. Similar allocation arguments to Lemma 1 may be shown for these lemmas.

Lemma 2: For \( \text{mcr}_k = (M_i, M_j, t_k) \), \( y \geq 0 \), and \( x : 0 \leq x \leq \delta_{ij} \),

\[
\beta_{\text{post}}(M_i, M_j, x, y) \geq a\Theta_{ij} + \min \left( \Theta_{ij}, x - \left( (a + 1)\Pi_{ij} - \Theta_{ij} \right) \right) _{+} + f\Theta^{(j)} + \min \left( \Theta^{(j)}, y - \left( f\Pi^{(j)} + \Delta^{(j)} - \Theta^{(j)} \right) \right) _{+}
\]

where \( a \equiv \frac{\Theta^{(i)}}{\Pi^{(j)}} \) and \( f \equiv \frac{\Theta^{(j)}}{\Pi^{(j)}} \).

Lemma 3: For \( \text{mcr}_k = (M_i, M_j, t_k) \) and \( \phi \geq 0 \),

\[
\beta_{\text{trans}}(M_i, M_j, \phi) \geq b\Theta_{ij} + \min \left( \Theta_{ij}, (\min(\phi, \delta_{ij}) - (b\Pi_{ij} + \Delta_{ij} - \Theta_{ij})) \right) _{+} + d\Theta^{(j)} + \min \left( \Theta^{(j)}, ((\phi - \delta_{ij}) + (d\Pi^{(j)} + \Delta^{(j)} - \Theta^{(j)})) \right) _{+}
\]

where \( b \equiv \frac{\min(\phi, \delta_{ij})}{\Pi_{ij}} \) and \( d \equiv \frac{\phi - \delta_{ij} + (d\Pi^{(j)} + \Delta^{(j)} - \Theta^{(j)})}{\Pi^{(j)}} \).

B. Deriving the Mode Change DBF

In this subsection, we derive upper bounds on the demand function of Definition 4 for the different types of tasks present for mode change \( \text{mcr}_k = (M_i, M_j, t_k) \) (i.e., changed, unchanged, and aborted tasks). We first derive an upper bound for the tasks that are changed, but not aborted during the mode change. The bounds obtained in this subsection are similar to general results of Phan et al. [13]; however, our results are more specific to the sporadic task and periodic resource models permitting a more precise analysis of carry-in in later subsections. Furthermore, Phan et al. [13] do not consider aborted jobs in their analysis.

The next two lemmas describe the sequence of job arrivals over \([t_k - x, t_k]\) that maximizes the demand for non-aborted, changed jobs of \( \tau_k^{(i)} \) during a mode change from \( M_i \) to \( M_j \). Figure 3 illustrates this sequence.

Lemma 4: For \( \text{mcr}_k = (M_i, M_j, t_k) \), \( \tau_k^{(i)} \in \tau^{(i)} \setminus \{\tau^{(j)} \cup A^{(i)}\} \), and \( x, \phi \geq 0 \), if \( d_k^{(i)} > \phi \), then \( \text{mdcbf}(\text{mcr}_k, \tau_k^{(i)}, x, \phi) \) is maximized by job arrival sequence where the last job of \( \tau_k^{(i)} \) arrives at \( t_k + \phi - d_k^{(i)} \) and previous jobs arrive as late as legally allowed.

Lemma 5: For \( \text{mcr}_k = (M_i, M_j, t_k) \), \( \tau_k^{(i)} \in \tau^{(i)} \setminus \{\tau^{(j)} \cup A^{(i)}\} \), and \( x, \phi \geq 0 \), if \( d_k^{(i)} \leq \phi \), then \( \text{mdcbf}(\text{mcr}_k, \tau_k^{(i)}, x, \phi) \) is maximized by job arrival sequence where the last job of \( \tau_k^{(i)} \) generated in \([t_k - x, t_k] \).
occurs an arbitrarily small $\epsilon > 0$ prior to $t_k$ and previous jobs arrive as late as legally allowed.

Lemmas 4 and 5 permit the calculation of the mode change carry-in demand for the non-aborted jobs using the following corollary. The corollary follows by simply counting the number of jobs in the sequences described by Lemmas 4 and 5.

**Corollary 1:** For $\text{mcr}_k = (M_i, M_j, t_k), \tau^{(i)}_k \in \tau^{(i)} \setminus \{\tau^{(j)} \cup A^{(i)}\}$, and $x, \phi \geq 0$, 

$$
\text{mcdbf}(\text{mcr}_k, \tau^{(i)}_k, x, \phi) 
\leq \left[ \frac{(x - \lambda^{(i)}_k)}{p^{(i)}_k} \right] \cdot e^{(i)}_k + \mu \geq 0 \left( x - \lambda^{(i)}_k \right) \cdot e^{(i)}_k
$$

(7)

where $\lambda^{(i)}_k \equiv \left( \frac{d^{(i)}_k - \phi}{p^{(i)}_k} \right)$.

We now describe the sequence of job arrivals that maximize demand over $[t_k - x, t_k]$ for aborted jobs. The sequence is depicted in Figure 4.

**Lemma 6:** For $\text{mcr}_k = (M_i, M_j, t_k), \tau^{(i)}_k \in A^{(i)}$, and $x, \phi \geq 0$, 

$$
\text{mcdbf}(\text{mcr}_k, \tau^{(i)}_k, x, \phi) 
\leq \left[ \frac{x}{p^{(i)}_k} \right] \cdot e^{(i)}_k + \mu \geq 0 \left( x - \frac{x}{p^{(i)}_k} \right) \cdot p^{(i)}_k + \phi - d^{(i)}_k
\cdot \min \left( x - \left[ \frac{x}{p^{(i)}_k} \right] p^{(i)}_k, e^{(i)}_k \right).
$$

(8)

Now we consider the last set of tasks $\tau^{(j)}$ which remains unaffected by the mode change request from $M_i$ to $M_j$. As there are no constraints on new job generation immediately after mode change, the mcdbf function for $\tau^{(j)}$ represents the execution of the maximum number of jobs of $\tau^{(j)}$ that can arrive and have deadline within the interval $[t_k - x, t_k + \phi]$. Note that this is the same as the dbf (according to Definition 1) and is summarized in the following lemma.

**Lemma 7:** For $\text{mcr}_k = (M_i, M_j, t_k), \tau^{(j)}_k \in \tau^{(j)}$, and $x, \phi \geq 0$, 

$$
\text{mcdbf}(\text{mcr}_k, \tau^{(j)}_k, x, \phi) = \text{dbf}(\tau^{(j)}_k, x + \phi).
$$

(9)

### C. Deriving Carry-In Demand Function

We are now prepared to obtain an upper bound on the carry-in demand function, as described in the following lemma.

**Lemma 8:** Consider $\phi \geq 0$ and successive mode change requests $\text{mcr}_0, \text{mcr}_1, \ldots, \text{mcr}_{k-1}, \text{mcr}_k$ where $\text{mcr}_{k-1} = (M_h, M_i, t_{k-1})$ and $\text{mcr}_k = (M_i, M_j, t_k)$. If there are no deadline misses prior to $t_{k-1}$, we may obtain an upper-bound on the carry-in demand for mode-change request $\text{mcr}_k$.

$$
\text{cidbf}(\text{mcr}_k, \phi) 
\leq \sup_{0 \leq x \leq t_k - t_{k-1} - \delta_{hi}} \left\{ \sum_{\tau^{(i)}_k \in \tau^{(i)} \setminus \{\tau^{(j)} \cup A^{(i)}\}} \text{mcdbf}(\text{mcr}_k, \tau^{(i)}_k, x, \phi) \right\}
\leq \beta_{\text{post}}(M_h, M_i, x) - \beta_{\text{trans}}(M_h, M_i, \delta_{hi})
\leq \beta_{\text{post}}(M_h, M_i, x)
\leq \beta_{\text{post}}(M_h, M_i, x) - \delta_{hi}
\leq \delta_{hi}
\leq \delta_{hi} + d^{(h)}_{\text{max}}
\leq \delta_{hi} + d^{(h)}_{\text{max}}
\leq \delta_{hi} + p^{(h)}
\leq \beta_{\text{trans}}(M_h, M_i, \delta_{hi})
\leq \beta_{\text{trans}}(M_h, M_i, \delta_{hi})
$$

(10)

such that $\text{ci}(\text{mcr}_{k-1})$ is upper bounded by the maximum of

$$
\sum_{\tau^{(h)} \in \tau^{(h)} \setminus \{\tau^{(i)} \cup A^{(h)}\}} e^{(h)}_k
\sum_{\tau^{(h)} \in \tau^{(h)} \setminus \{\tau^{(i)} \cup A^{(h)}\}} \left[ \delta_{hi} / p^{(h)} \right] + 1
\left[ \delta_{hi} / p^{(h)} \right] + 1
\delta_{hi} + d^{(h)}_{\text{max}}
\delta_{hi} + p^{(h)}
\delta_{hi} + p^{(h)}
\delta_{hi} + p^{(h)}
\delta_{hi} + p^{(h)}
\delta_{hi} + p^{(h)}
$$

(11)

D. A Sufficient Schedulability Condition

For the schedulability analysis, we need to make sure the overall demand over any interval is always less than corresponding supply (i.e., $\beta_{\text{post}}(M_i, M_j, x, y), \beta_{\text{trans}}(M_i, M_j, \phi)$, and $\beta_{\text{post}}(M_i, x)$). We will establish Theorem 1 which will check the schedulability conditions for a sequence of concrete mode change requests.

**Theorem 1:** For a concrete sequence of mode-change requests $\text{mcr}_0, \text{mcr}_1, \ldots$, the subsystem is EDF-schedulable, if for all $k = 0, 1, \ldots$, the following five conditions hold for $\text{mcr}_k = (M_i, M_j, t_k)$:

$$
\sum_{\tau^{(j)} \in \tau^{(j)}} \text{dbf}(\tau^{(j)}_k, t) \leq \text{sbf}(\Omega^{(j)}, t),
\forall t : 0 < t \leq t_{k+1} - t_k - \delta_{ij};
\text{cidbf}(\text{mcr}_k, \phi) \leq \beta_{\text{trans}}(M_i, M_j, \phi), \forall \phi : 0 \leq \phi \leq \delta_{ij};
\text{cidbf}(\text{mcr}_k, \phi) \leq \beta_{\text{post}}(M_i, M_j, x), \forall \phi : 0 \leq \phi \leq \delta_{ij};
\text{cidbf}(\text{mcr}_k, \phi) \leq \beta_{\text{trans}}(M_i, M_j, 0), \forall \phi : 0 < \phi < \delta_{ij};
$$

(13a)

(13b)

(13c)

(13d)

(13e)

The formal proof of this theorem can be found in the extended version [7] of this paper. To improve the readability of this
report, we just provide intuitive explanation. Equation 13a ensures the schedulability of individual mode; that is, the demand over any interval of length $t$ for any mode $M_j$ is always less than the corresponding supply. Equation 13b ensures the schedulability during the transition period while accounting for carry-in demand from the non-aborted jobs and the mode-change supply function derived in Equation 6. Similarly, Equation 13c ensures schedulability after a transition; this condition accounts for the carry-in from all past mode change requests through $\tilde{c_i}$ function. Equation 13d and 13e ensure that an individual mode is schedulable with respect to the demand from unchanged tasks.

V. NON-CONCRETE SEQUENCES OF MCR

In the previous section, we obtained schedulability conditions for any concrete sequence of MCRs. In this section, we remove the assumption that the mode changes are a priori known and consider non-concrete sequences; the following summarizes our problem for this setting:

**Problem 2:** Given modes $M_1, \ldots, M_p$, resources $\Omega_{ij}$, offset $\delta_{ij}$, unchanged tasks $\tau^{(ij)}$, and aborted tasks $A^{(ij)}$ for all $i, j \in \{1, \ldots, q\}$ ($i \neq j$), determine whether all jobs (under all legal job arrival sequences and all possible legal mode-change requests according to Equations 1 and 2) are EDF-schedulable.

A. A Sufficient Schedulability Test

The analysis of a non-concrete sequence will differ from the concrete sequence via the calculation of carry-in for past modes. For a concrete sequence, the maximum carry-in is determined from the fixed sequence of previous MCRs. As the exact time of an MCR is not known a priori for non-concrete sequences, the analysis needs to consider the maximum possible carry-in that could be generated from previous modes. That is, we must obtain an upper bound on any possible mode change from any mode $M_i$ to any other mode $M_j$ over all possible sequences of MCRs, $\text{mc}_{i1}, \text{mc}_{i2}, \ldots$. We now define equivalent carry-in execution and demand for a non-concrete sequence of MCRs.

**Definition 9 (Non-Concrete Carry-In Execution):** The non-concrete carry-in execution from mode $M_i$ to any other mode $M_j$ at time $t_k$ is an upper bound on the maximum possible remaining execution (over any legal sequence of MCRs) of non-aborted jobs from mode $M_i$ for tasks $\tau^{(i)} \setminus \{\tau^{(ij)} \cup A^{(ij)}\}$ at time $t_k + \delta_{ij}$ that arrive prior to $t_k$ and maximum total execution of unchanged tasks (i.e., $\tau^{(ij)}$) that have arrival before $t_k + \delta_{ij}$. We denote this value by $\tilde{c_i}(M_i, M_j)$ over any legal sequence of mode changes prior to $t_k$.

**Definition 10 (Non-Concrete Carry-In Demand DBF):** The non-concrete carry-in demand-bound function for a mode change from $M_i$ to $M_j$ at time $t_k$ and $\phi \in \mathbb{R}_{\geq 0}$ is the maximum remaining execution (over any legal sequence of MCRs prior to $t_k$) of jobs of tasks $\tau^{(i)} \setminus A^{(ij)}$ that arrive prior to $t_k$ (or prior to $t_k + \delta_{ij}$ for $\tau^{(ij)}$ tasks) and have deadline in the interval $[t_k, t_k + \phi]$. We denote this quantity by $\text{cidbf}(M_i, M_j, \phi)$.

The following lemma establishes an upper bound on the non-concrete carry-in demand.

**Lemma 9:** For $\phi \geq 0$ and a mode change from $M_i$ to $M_j$, if all modes executing prior to the mode change did not miss a deadline, then

$$\tilde{\text{cidbf}}(M_i, M_j, \phi) \leq \tilde{\Psi}(M_i, M_j, \phi, \tilde{c_i}(M_i, M_j)) \quad (13)$$

where

$$\tilde{\Psi}(M_i, M_j, \phi, \zeta) \equiv \sup_{\mathbb{R}_{>0}} \left\{ \begin{array}{l}
\sum_{e^{(i)}_t \in \tau^{(i)} \setminus (\tau^{(ij)} \cup A^{(ij)})} \left[ \frac{(x - \lambda^{(i)}_t + \rho^{(i)}_t)}{p^{(i)}_t} \right] + 1 \times e^{(i)}_t \\
\sum_{e^{(i)}_t \in A^{(ij)}} \left[ \frac{x - \rho^{(i)}_t}{p^{(i)}_t} \right] \times e^{(i)}_t \\
\sum_{e^{(i)}_t \in \tau^{(ij)}} \min \left( x - \rho^{(i)}_t, p^{(i)}_t, e^{(i)}_t \right) \\
\sum_{e^{(i)}_t \in \tau^{(ij)}} \text{dbf} \left( \tau^{(ij)}_t, \infty + \phi \right) \\
\beta_{\text{trans}}(M_i, M_j, \zeta) \\
\mu_{\text{trans}} \left( x - N_t^{(i)} \right) \cdot \zeta
\end{array} \right\}$$

and $\tilde{c_i}(M_i, M_j)$ is obtained from the convergence of the sequence $c_i(M_i, M_j)$, $\tilde{c}_1(M_i, M_j)$, $\tilde{c}_2(M_i, M_j)$, \ldots for all $i, j \neq j \in \{1, \ldots, q\}$. For any $g \in \mathbb{N}$, $M_i$, and $M_j$ ($i \neq j$), if $g \equiv 0$, then $\tilde{c}_g(M_i, M_j) = 0$; otherwise, if $g > 0$, then $\tilde{c}_g(M_i, M_j)$ is the minimum of $W_{ij}$ and $f_{ij}(\max_{h=1}^{\min(g-1,M_h)} \left\{ c_{h-1}(M_h, M_i) \right\})$

$$W_{ij} = \sum_{e^{(i)}_t \in \tau^{(i)} \setminus A^{(ij)}} \left[ \frac{\delta_{ij}}{p^{(i)}_t} \right] + 1 \times e^{(i)}_t$$

and $f_{ij}(\zeta)$ equals:

$$f_{ij}(\zeta) = \begin{cases}
\tilde{\Psi}(M_i, M_j, \max(\delta_{ij} + \delta_{ij}, d^{(ij)}_{\max}), \zeta) \\
- \sum_{e^{(i)}_t \in \tau^{(ij)}} \text{dbf} \left( \tau^{(ij)}_t, \max(\delta_{ij} + d^{(ij)}_{\max}, \delta_{ij}) - \delta_{ij} - p^{(i)}_t \right) \\
- \beta_{\text{trans}}(M_i, M_j, \delta_{ij})
\end{cases}$$

The convergence of the above sequence occurs at the smallest $g \in \mathbb{N}$ such that $\forall i, j (i \neq j) \in \{1, \ldots, q\}$, $c_g(M_i, M_j) = c_{g-1}(M_i, M_j)$.

Some remarks on Lemma 9: The function $f_{ij}(\zeta)$ calculates an upper bound on carry-in of a mode change request from $M_i$ (assuming the carry-in into $M_i$ from a previous mode is $\zeta$) to the next mode $M_j$. This function acts in the very similar way to the Equation 12 for a concrete sequence of mode-change requests. Whereas Equation 12 evaluates only a finite number of $x$ values, $f_{ij}(\zeta)$ invokes $\tilde{\Psi}$ which evaluates all possible values of $x(> 0)$ as the potential MCR instants. (This is necessary as the exact MCR instants are not known priori). Although, $f_{ij}(\zeta)$ function is defined in terms of $\tilde{\Psi}$ function, there is no circular dependency between $\tilde{c}_i$ and $\tilde{\text{cidbf}}$ as $\tilde{c}_i$ is calculated iteratively from the sequence $c_i(M_i, M_j), \ldots
Theorem 2: For any possible sequence of mode-change requests, the subsystem is EDF-schedulable, if the following five conditions hold for any two distinct modes $M_i$ and $M_j$,

\[
\sum_{\tau_t^{(j)} \in \tau_t^{(j)}} \text{dbf}(\tau_t^{(j)}, t) \leq \text{sbf}(\tau_t^{(j)}) t, \quad \forall t > 0,
\]

(17a)

\[
\Psi (M_i, M_j, \phi, C_i) \leq \beta_{\text{trans}}(M_i, M_j, \phi)
\]

\forall \phi: 0 < \phi \leq \delta_{ij},

(17b)

\[
\Psi (M_i, M_j, \delta_{ij} + t, C_i) - \beta_{\text{trans}}(M_i, M_j, \delta_{ij}) + \sum_{\tau_t^{(j)} \in \tau_t^{(j)} \setminus \tau_t^{(j)}} \text{dbf}(\tau_t^{(j)}, t) \leq \beta_{\text{post}}(M_i, M_j, 0, t), \quad \forall t > 0
\]

(17c)

\[
\sum_{\tau_t^{(j)} \in \tau_t^{(j)} \setminus \tau_t^{(j)}} \text{dbf}(\tau_t^{(j)}, t) + \sum_{\tau_t^{(j)} \in \tau_t^{(j)}} \text{dbf}(\tau_t^{(j)}, t + s) \leq \beta_{\text{post}}(M_i, M_j, s, t), \quad \forall s, t: 0 < t, 0 < s < \delta_{ij}
\]

(17d)

\[
\sum_{\tau_t^{(j)} \in \tau_t^{(j)} \setminus \tau_t^{(j)}} \text{dbf}(\tau_t^{(j)}, t) \leq \text{sbf}(\Omega(\delta_{ij}), t), \quad \forall t: 0 < t < \delta_{ij},
\]

(17e)

where $C_i \overset{\Delta}{=} \max_{k=1, \ldots, q} \tilde{c}_i(M_k, M_i)$.

B. Reducing the Time Complexity

For finding the worst-case mode-change carry-in demand, Theorem 2, as written, has to evaluate a potentially unbounded number of values of $t$ for Equations 17a, 17c, and 17d. Furthermore, it is also not immediately obvious how to efficiently compute $\Psi$ from Lemma 9, as it requires evaluating an expression over any infinite number of values for $x$ and iteratively computing a converging sequence of values for $\tilde{c}$. In this subsection, we derive more efficient time bounds for our schedulability test. The next section will use the lemmas derived in this section to efficiently implement our schedulability test for non-concrete MCRs.

In the following four lemmas, we obtain upper bounds on the number of times for which the right-hand-side of Equations 14, 17c, 17a and 17d (respectively) need to be evaluated. The results are inspired by similar bounds obtained by Baruah et al. [2] for uni-modal systems. We will abuse notation below and assume that a zero in the denominator of a fraction evaluates to $\infty$. We will also assume that for each $M_i$ the utilization $u^{(i)}$ is at most $\Theta(i)/\Pi(i)$ as this is a necessary condition for schedulability on a periodic resource [16].

Lemma 10: For $\phi \geq 0$ and any MCR $\tilde{M} = (M_i, M_j, t_k)$ in arbitrary sequence of mode-change requests, if $\Psi (M_i, M_j, \phi, \zeta)$ (Equation 14) is at least $\xi \geq 0$, then the $x$ that maximizes the supremum in the right-hand-side of Equation 14 occurs at or before the maximum of $d_{\text{max}}^{(j)}$ and the minimum of $\text{lcm}_{\tau_t^{(j)} \in \tau_t^{(j)}} \{\mathcal{P}_{\tau_t^{(j)}}\}$ + $d_{\text{max}}^{(j)}$

\[
\left[ u^{(i)} + \frac{d_{\text{max}}^{(j)}}{\mathcal{P}_{\tau_t^{(j)}}} + u^{(i)} \phi + \frac{\Theta(i)(\Pi(i) - \Theta(i))}{\Pi(i)} - \xi \right]
\]

\[
\frac{\Theta(i)(\Pi(i) - \Theta(i))}{\Pi(i)} - u^{(i)}
\]

(18)

where $\mathcal{P}_{\tau_t^{(j)}} = \max_{\tau_t^{(j)} \in \tau_t^{(j)}} \{\mathcal{P}_{\tau_t^{(j)}}\}$.

Lemma 11: For any distinct modes $M_i$ and $M_j$, if Equation 17c of Theorem 2 is violated, then the violation must occur for some $t$ at or before the maximum of $d_{\text{max}}^{(j)}$ and the minimum of $\text{lcm}_{\tau_t^{(j)} \in \tau_t^{(j)}} \{\mathcal{P}_{\tau_t^{(j)}}\}$ + $d_{\text{max}}^{(j)}$

\[
\left[ \sum_{\tau_t^{(j)} \in \tau_t^{(j)} \setminus \delta_t^{(j)}} \beta + \delta_{ij} u^{(i)} + \sum_{\tau_t^{(j)} \in \tau_t^{(j)}} \frac{\Theta(i)(\Pi(i) - \Theta(i))}{\Pi(i)} - u^{(i)} \right]
\]

(19)

where $\beta \overset{\Delta}{=} \beta_{\text{trans}}(M_i, M_j, \delta_{ij})$.

Lemma 12: For any distinct modes $M_j$, if Equation 17a of Theorem 2 is violated, then the violation must occur for some $t$ at or before the maximum of $d_{\text{max}}^{(j)}$ and the minimum of $\text{lcm}_{\tau_t^{(j)} \in \tau_t^{(j)}} \{\mathcal{P}_{\tau_t^{(j)}}\}$ + $d_{\text{max}}^{(j)}$

\[
\left[ \sum_{\tau_t^{(j)} \in \tau_t^{(j)} \setminus \delta_t^{(j)}} \beta + \delta_{ij} u^{(i)} + \sum_{\tau_t^{(j)} \in \tau_t^{(j)}} \frac{\Theta(i)(\Pi(i) - \Theta(i))}{\Pi(i)} - u^{(i)} \right]
\]

(20)

Lemma 13: For any distinct mode change from $M_i$ to $M_j$ and integer $s : 0 < s \leq \delta_{ij}$, if Equation 17d of Theorem 2 is violated, then the violation must occur for some $t$ at or before the maximum of $d_{\text{max}}^{(j)}$ and the minimum of $\text{lcm}_{\tau_t^{(j)} \in \tau_t^{(j)}} \{\mathcal{P}_{\tau_t^{(j)}}\}$ + $d_{\text{max}}^{(j)}$

\[
\left[ s \cdot u^{(i)} + u^{(j)} \max_{\tau_t^{(j)} \in \tau_t^{(j)}} \{\mathcal{P}_{\tau_t^{(j)}}\} - d_{\text{max}}^{(j)} \right]
\]

(21)

Finally, the following corollary on the number of iterations for convergence of $\tilde{c}$ follows immediately from Equation 15 of Theorem 2.

Corollary 2: The convergence of the sequence defined in Equation 16 occurs at a $g \leq \sum_{i=1, \ldots, q} \max_{i \neq i} \tilde{c}_{ij}$ where $\tilde{c}_{ij}$ is the value defined in Equation 15.

VI. ALGORITHMS

In this section, we join all the pieces together to develop a schedulability test algorithm for non-concrete MCRs. We present our algorithm called schedulability using bounded iteration (SUBI) for checking the schedulability of any multi-modal subsystem $S$ with non-concrete MCRs. (Please note, that since Theorem 2 covers all possible sequence of MCRs, any subsystem that satisfies Theorem 2 will also satisfy Theorem 1 for any fixed legal concrete sequence of MCRs.)

Schedulability Algorithm. The algorithm SUBI (presented in Algorithm 1) checks for schedulability for non-concrete sequences of MCRs. This algorithm uses MaxCarry, a helper function (pseudocode may be found in [7]), as a subroutine to calculate $\tilde{c}(M_i, M_j)$ for every pair of modes. The algorithm SUBI checks all five conditions of Theorem 2 for schedulability. The for loop at Line 1 uses the condition of Equation 17a, the loop at Line 10 checks the condition of Equation 17b, and
the innermost loop starting at Line 27 checks the condition of Equation 17c. MaxCarry is called prior to checking the conditions of Equations 17b and 17c, so that maximum carry-in can be used from the stored value. Equations 17d and 17e are checked in the second main loop. The algorithm returns true only if all of the above mentioned five conditions do not fail for any interval length of \( t \).

The runtime complexity of each iteration of SUBI depends upon the \( \Psi \) function. The function \( \Psi \) given \( \zeta \) determines carry-in demand using \( O(n \times Q(P, B)) \) steps where \( n \) is the maximum total number of tasks in any mode, \( B \) is the maximum value of Equation 15, and \( Q(\phi, \zeta) \) equals Equation 18 as a function of \( \phi \) and \( \zeta \). Let \( R \) corresponds to the maximum value of Equation 20 of Lemma 12; \( S \) corresponds to the maximum value of Equation 19 of Lemma 11 over all modes; \( V(s) \) corresponds to the maximum value of Equation 21 of Lemma 13 (given a value of \( s \)); and \( \delta_{\text{max}} \equiv \max_{i,j \in \{1, \ldots, q\}} \{\delta_{ij}\} \). Observe that the maximum value for \( s \) in Lemma 13 is \( \delta_{\text{max}} \). The first loop requires \( O(nqR) \) steps; it may be shown that MaxCarry requires \( O(nq^2 BQ(P, B)) \); and the second loop requires \( O(nq^2 (P \times Q(P, B) + P \times V(\delta_{\text{max}}) + S \times Q(P + S, B))) \).

When \( \Theta^{(i)}(\Pi, i, j, \phi, \zeta) = 0 \) for all \( M_i \) is lower bounded by a fixed positive constant, \( Q(\phi, \zeta), V(\delta_{\text{max}}), B, P, S, \) and \( R \) are all pseudo-polynomial function or constants; thus, the total time complexity is pseudo-polynomial.

### VII. Simulations

In this section, we present the performance results for our proposed algorithm. We compare SUBI with exponential-time schedulability analysis using reachability graph (SURG) proposed by Phan et al. [13]. For the simulation, we implemented SURG and SUBI in MATLAB and performed our simulations on a 2.33GHz Intel Core 2 Duo machine with 2.0GB RAM. During the simulation, we have the following parameters and value ranges for the multi-modal subsystem \( S \):

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Properties</th>
<th>Modes</th>
<th>M1</th>
<th>M2</th>
<th>M3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td>y</td>
<td>y</td>
<td>y</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>30</td>
<td>y</td>
<td>y</td>
<td>n</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>40</td>
<td>n</td>
<td>n</td>
<td>y</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>10</td>
<td>y</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>20</td>
<td>n</td>
<td>y</td>
<td>n</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>24</td>
<td>n</td>
<td>n</td>
<td>y</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>20</td>
<td>y</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>10</td>
<td>y</td>
<td>n</td>
<td>n</td>
</tr>
</tbody>
</table>

Table I: Simulation tasks set properties. A ‘y’ in the mode column indicates this task is present in the mode.

1) The number of modes \( (q) \) in the subsystem is 3.
2) The total number of tasks in the multi-modal tasks system is 8. Task properties and distributions are described in Table I.
3) During a mode transition, jobs from task \( \tau_1 \) are considered as aborted jobs. Task \( \tau_5 \) is unchanged between MCRs involving \( M_2/M_3 \).
4) The resource period (\( \Pi \)) and deadline (\( \Delta \)) are set to 10 for all modes.

Algorithm 1 SUBI(S).

```java
1: for \( i = 1 \) to \( q \) do
2:  \( T \) is set by Equation 20.
3:  for \( t = 1 \) to \( T \) do
4:      if \( \text{sbf}(\Pi(i), \Phi(i), t) < \text{dbf}(\tau(i), t) \) then
5:          return false
6:      end if
7:  end for
8:  \( \zeta \leftarrow \text{MaxCarry}(S) \)
9:  for \( i = 1 \) to \( q \) do
10:     for \( j = 1 \) to \( q \) do
11:        if \( \Phi(M_i, M_j, \phi, \zeta_{ij}) > \beta_{\text{trans}}(M_i, M_j, \phi) \) then
12:            return false
13:      end if
14:        if \( \text{dbf}(\tau(i), \phi) \leq \text{dbf}(\Omega(i), \phi) \) then
15:            return false
16:      end if
17:     end for
18:  end for
19:  if \( T \) is set by Equation 21
20:     for \( t = 0 \) to \( T \) do
21:        if Equation 17d is false for \( s = \phi \) then
22:            return false
23:        end if
24:     end for
25:  end for
26:  if \( T \) is set by Equation 19
27:     for \( t = 0 \) to \( T \) do
28:        if \( \text{carry} \leftarrow \Psi(M_i, M_j, i, j, t, \zeta_{ij}) \)
29:            \( -\beta_{\text{trans}}(M_i, M_j, \delta_{ij}) \)
30:       if \( \text{carry} + \text{dbf}(\tau(i), t) > \beta_{\text{post}}(M_i, M_j, 0, t) \) then
31:           return false
32:       end if
33:     end for
34:  end for
35: end for
36: return true
```

5) The offset \( \delta_{ij} \) is set to \( \Pi \) for both simulations. \( N^{(i)} \) is set equal to 2 for all modes \( M_i \).

![Success](image1.png)

Fig. 5: Comparison of SURG and SUBI

In the first simulation, we randomly generate a set of capacities \( (G^{(i)}) \) where the total sum is taken from the range \([1, q\Pi]\). We execute SUBI and SURG for checking schedulability of the subsystem \( S \) with \( R \). The graph at the top of the Figure 5 presents the percentage of ‘YES’ responses out of 200 runs on each distinct summation of capacities (i.e., the value on the \( x \) axis). The dashed line depicts the results for SURG, and solid line is for SUBI. The graph at the bottom in Figure 5 presents
the average elapsed time for deciding the schedulability using over randomly-generated capacities for each given utilization. For this particular subsystem, Figure 5 illustrates that SUBI does as well or better than SURG and is clearly more efficient.

For checking the scalability of SUBI, we perform a second simulation with a higher number of modes (up to 15). In each step, we increase by one mode with four tasks chosen randomly from Table I and perform schedulability test using SURG and SUBI with capacity set to the largest possible value (i.e., II). The result is depicted in Figure 6. The dashed line shows the elapsed time in second for reaching a decision using SURG, whereas the solid line depicts the results for SUBI. SURG is more general, and designed to compute the feasible minimum capacity using a reachability graph; thus, the longer runtime of SURG is due to the exponential-time complexity of traversing a reachability graph.

![Fig. 6: Comparison of SURG and SUBI](image)

### VIII. CONCLUSION

In this report, we proposed a model for resource and application modes in a real-time subsystem. Each mode is characterized by an explicit deadline periodic resource (i.e., resource behavior) and a sporadic task system (i.e., application behavior). For our proposed model, we derived a schedulability test for mode-change request for two settings: concrete and non-concrete mode change requests. For non-concrete subsystems (i.e., the sequence of mode-changes are not known a priori), we obtain a schedulability analysis algorithm that has pseudo-polynomial time complexity. The previous known algorithm which uses a reachability graph requires exponential time complexity. Furthermore, our simulation results validate the effectiveness and efficiency of algorithm and demonstrate that it scales as the number of modes increases. Thus, our proposed approach can be used to quickly verify the schedulability of control systems with a large number of modes.

Currently we are working on developing algorithms for allocating minimum capacities for modes. Our ultimate goal is to restrict the runtime complexity of capacity determination to a pseudo-polynomial number of candidates. (Currently, to find the optimal, we must try all possible combinations and use the schedulability test derived in this paper.) Obtaining the minimum capacity for a multi-modal subsystem opens doors for further fruitful research in designing control systems that minimize energy consumption or minimize the peak temperature of the processing platform. Furthermore, we would like to extend our tractable-analysis techniques to more general task models (e.g., RTC-based models).

### REFERENCES


