Necessary Feasibility Analysis for Mixed-Criticality Task Systems on Uniprocessor

Hoon Sung Chwa
Information and Communication Engineering
DGIST
Republic of Korea
chwahs@dgist.ac.kr

Hyeongboo Baek
Computer Science and Engineering
Incheon National University (INU)
Republic of Korea
hbbaek@inu.ac.kr

Jinkyu Lee[†]
Computer Science and Engineering
Sungkyunkwan University (SKKU)
Republic of Korea
jinkyu.lee@skku.edu

***Abstract*—While feasibility of timing guarantees has been extensively studied for single-criticality (SC) task systems, the same cannot be said true for mixed-criticality (MC) task systems. In particular, there exist only a few studies that address necessary feasibility conditions for MC task systems, and all of them have derived trivial results from existing SC studies that rely on simple demand-supply comparison. In this paper, we develop necessary feasibility tests for MC task systems on a uniprocessor platform, which is the first study that yields non-trivial results for MC necessary feasibility. To this end, we investigate characteristics of MC necessary feasibility conditions. Due to the existence of the mode change and consequences thereof, the characteristics pose new challenges that cannot be resolved by existing techniques for SC task systems, including how to calculate demand when the mode change occurs, how to determine the target sub-intervals for demand-supply comparison, how to derive an infeasibility condition from demand-supply comparisons with different possible mode change instants, how to select a scenario to specify the mode change instant without the target scheduling algorithm, and how to find infeasible task sets with reasonable time-complexity. By addressing those challenges, we develop a new necessary feasibility test and its simplified version. The simulation results demonstrate that the proposed tests find a number of additional infeasible task sets which have been proven neither feasible nor infeasible by any existing studies.**

I. Introduction

The most fundamental issue for hard real-time systems is to provide timing guarantees, and many studies sought "feasibility" of timing guarantees—whether every instance of real-time tasks finishes its execution within its deadline under a given setting (e.g., a computing resource such as uni- and multi-processors and a task model such as the Liu and Layland task model [1]). We may classify the studies into two: (i) developing scheduling algorithms and their schedulability analysis to expand a set of real-time task sets proven schedulable by at least a scheduling algorithm (i.e., addressing sufficient feasibility), and (ii) deriving conditions of task sets that are never schedulable by any scheduling algorithm to reduce a set of task sets that are potentially schedulable but have not been proven schedulable so far (i.e., addressing necessary feasibility).

Studies for both (i) and (ii) have matured for SC (single-criticality) task systems in which all tasks belong to a single category in terms of criticality, and some of their results have been successfully adapted to MC (mixed-criticality) task systems [2] in which tasks have different criticality levels. However, most studies for MC task systems have focused on (i), including MC-specific scheduling algorithms, such as PLRS [3], EDF-VD [4], GREEDY [5], and ECDF [6], with their schedulability analysis; only a few studies have addressed (ii), but all of them have presented trivial results [7], [8], [9]. We illustrate the state-of-the-art of (ii) for MC task systems on a uniprocessor platform in Fig. 1. In the figure, all task sets between 0.75 by 0.75 and 1.0 by 1.0 have not been proven infeasible (i.e., timing-guarantees of all tasks are impossible by any schedule), and most of them have not been also proven feasible (i.e., timing-guarantees of all tasks are possible by at least a schedule); the details will be explained in Section VII.

The goal of this paper is to reduce a set of MC task sets whose feasibility is unknown by existing studies. In particular, we aim at developing *necessary feasibility* tests that prove infeasibility of some of such MC task sets on a uniprocessor, which is the first attempt in MC task systems (except the trivial results in [7], [8], [9]). Addressing necessary feasiblity for MC task systems is beneficial both from the theoretical and the practical point of view. On the theoretical side, tight necessary feasibility analysis eliminates unnecessary efforts for researchers to try to make task sets schedulable by developing a new scheduling algorithm, if the task sets are proven infeasible by the necessary feasibility analysis. On the practical side, when system designers set the configuration for scheduling algorithms, task parameters and computing resources, tight necessary feasibility results reduce burden of tuning parameters for the configuration by excluding some infeasible choices of the configuration.

Before we explain the development of necessary feasibilitiy tests for MC task systems, we present the MC feasibility requirement as follows. A typical MC task model [2] is to assume more pessimistic worst-case execution times (WCETs) for higher criticality levels for each task. In systems with two levels of criticality (high and low), the system can be seen as exhibiting two different behaviors at runtime: the low-criticality behavior as long as each job signals its completion without exceeding its low-criticality WCET, and the high-criticality behavior thereafter if any high-criticality job does not signal its completion after executing for its low-criticality WCET. A typical feasibility requirement for MC systems is that 1) all high-criticality tasks always meet their deadlines and 2) all low-criticality tasks meet their deadlines during the low-criticality system behavior (the notion of MC-feasibility is formally specified in Section II).

To develop necessary feasibility tests in accordance with the above MC feasibility definition, we first offer our own interpretation of the existing necessary feasibility tests for SC task systems on a uniprocessor (in Section III-A). We next

[†]Jinkyu Lee is the corresponding author.

2576-3172/19/$31.00 ©2019 IEEE
DOI 10.1109/RTSS46320.2019.00046

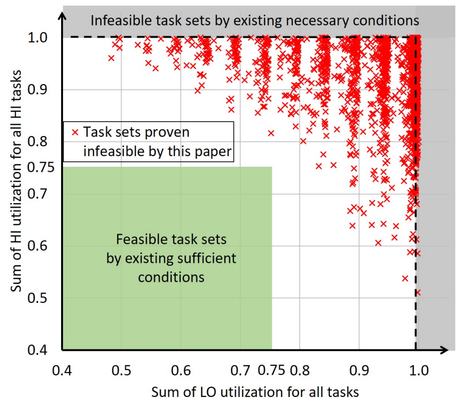

Fig. 1. All task sets in the region between X=0.75 by Y=0.75 and X=1.0 by Y=1.0 have not been proven infeasible by any existing studies (while only some of them have been proven feasible). The necessary feasibility test proposed in this paper proves infeasibility of the task sets marked as red cross. Section VII will explain details.

show that a straightforward extension of such interpretation towards MC task systems is limited to covering only partial cases of MC behaviors, yielding a still huge gap between the region covered by task sets proven feasible and that proven infeasible (in Section III-B) as shown in Fig 1. Utilizing the background, we identify unique issues/challenges specific to MC task systems for developing necessary feasibility tests, based on investigation of their characteristics (in Section III-C), which can be summarized briefly as follows.

C1. According to the MC-feasibility requirement, (i) a set of jobs whose execution requirement should be guaranteed (i.e., all vs. high-criticality jobs) and (ii) high-criticality jobs' amount of execution requirement that should be guaranteed (i.e., no more than vs. more than low-criticality WCET) vary depending on the system behavior.

C2. It is impossible to know beforehand which system behavior will be shown and when the system switches from low-criticality to high-criticality behavior (referred to as *mode change*) during runtime.

Such unique characteristics of MC task systems pose new challenges that cannot be resolved by existing techniques for SC task systems.

Q1. How to characterize and calculate the sum of every job's execution requirement that should be performed in an interval of interest to avoid its deadline miss (called *demand*) that changes depending on the system behavior?

Q2. How to identify in which conditions the mode change cannot occur without missing any deadline?

Q3. How to specify all possible mode change instants that are not known a priori without targeting a scheduling algorithm?

Q4. How to derive a necessary feasibility condition by considering every possible mode change instant?

Q5. How to efficiently find infeasible task sets with reasonable time-complexity?

We develop our necessary feasibility tests by answering those questions, in Section IV, V and VI, which can be summarized briefly as follows. To answer Q1, we first define and target a *scenario* tailored to MC task systems, which specifies an interval of interest, a job release pattern, and each job's execution requirement relevant to the mode change instant. Then, considering the mode change instant is a criterion that determines the demand of low- and high-criticality jobs, we divide the interval of interest into two sub-intervals based on the mode change instant. We next calculate the execution contribution of each of low- and high-criticality jobs to the demand in the sub-intervals separately considering the relationship between the mode change instant and each job's execution window. A key point for the calculation is that we collectively upper-bound the demand in the two sub-intervals for some high-criticality jobs that undergo the mode change with the consideration on the dependency between the demands of each sub-interval.

To answer Q2, we derive a necessary condition for the feasibility of a mode change instant without missing any job deadline by comparing the demand with the supply for a target scenario. To answer Q3, for a given scenario, we define a *crucial* job that can restrict the range of the mode change instant without targeting a scheduling algorithm. To answer Q4 and Q5, we present an efficient algorithm that can find infeasible task sets by considering every possible mode change instant with the complexity of pseudo-polynomial time. In addition, we also explore a tradeoff between time-complexity and capability in finding infeasible task sets, by developing a necessary feasibility test as well as its simplified version that has the same time complexity as in the SC task system case at the expense of sacrificing the capability.

We demonstrate effectiveness of the proposed necessary feasibility tests in finding infeasible task sets via simulations. As shown in Fig. 1, the proposed tests are able to newly cover many infeasible task sets (marked as red cross) which have not been proven neither feasible nor infeasible. In particular, if we focus on the region between X=0.95 by Y=0.95 and X=1.0 by Y=1.0, the proposed necessary feasibility tests find 325 additional infeasible task sets among 1,000 task sets, which have not been proven infeasible by any existing studies.

This paper makes the following contributions:

- Developing necessary feasibility tests for MC task systems, which is the first study that yields non-trivial results for MC necessary feasibility,
- Exploring a number of unique issues pertaining to developing necessary feasibility tests specialized for MC task systems,
- Establishing foundations of necessary feasibility tests for MC task systems, by addressing the unique issues one by one,
- Exploring a tradeoff between time-complexity and capability in finding infeasible task sets, by developing a necessary feasibility test and its simplified version, and
- Demonstrating effectiveness of the proposed tests in finding infeasible task sets.

II. System Model, Assumptions and Notations

We consider the problem of scheduling a dual-criticality (high and low, namely HI and LO) task set τ of n sporadic MC tasks on a uniprocessor platform.

MC tasks. Each MC task $\tau_i \in \tau$ is characterized by a tuple $(T_i, \chi_i, C_i^{\mathtt{LO}}, C_i^{\mathtt{HI}}, D_i)$, where T_i is the minimum separation (or *period*) between successive job releases, $\chi_i \in \{\mathtt{LO}, \mathtt{HI}\}$ is the *criticality level*, $C_i^{\mathtt{LO}}$ is the LO *worst-case execution time* (WCET), $C_i^{\mathtt{HI}}$ is the HI WCET, and D_i is the *relative deadline*. A task τ_i is said to be a LO and HI task, if χ_i is LO and HI, respectively; let $\tau^{\mathtt{LO}}$ and $\tau^{\mathtt{HI}}$ denote a set of LO and HI tasks in τ, respectively. We assume that $C_i^{\mathtt{LO}} \le C_i^{\mathtt{HI}}$ and $C_i^{\mathtt{LO}} = C_i^{\mathtt{HI}}$ hold for every $\tau_i \in \tau^{\mathtt{HI}}$ and every $\tau_i \in \tau^{\mathtt{LO}}$, respectively. We target implicit- and constrained-deadline task systems, respectively, in which $D_i = T_i$ and $D_i \le T_i$ hold for every $\tau_i \in \tau$.

MC jobs and scenarios. Task τ_i generates a potentially infinite sequence of jobs: $J_i^1, J_i^2, J_i^3, \ldots$. The q^{th} job of τ_i (denoted by J_i^q) is characterized by two parameters: $J_i^q = (r_i^q, \gamma_i^q)$, where r_i^q is the *release time* of the job, and $\gamma_i^q \in (0, C_i^{\mathtt{HI}}]$ is the *execution requirement* of the job; J_i^q has completed its execution if it executes for γ_i^q. The *absolute deadline* of job J_i^q is $d_i^q \overset{\text{def}}{=} r_i^q + D_i$, and we call $[r_i^q, d_i^q]$ the *execution window* of J_i^q. We target sporadic task systems, in which successive jobs are released at least T_i time units apart. A job of τ_i is said to be a LO and HI job, if χ_i is LO and HI, respectively. It is important to notice that neither the release times nor the execution requirements are known in advance. Let a *scenario* for a given task set τ mean a collection of release times and execution requirements of jobs of interest invoked by tasks in τ; there exist infinitely many scenarios for a given task set.

System behavior and requirement. Job J_i^q is released at time r_i^q and needs to complete γ_i^q units of work before its absolute deadline of d_i^q. The value of γ_i^q is not known beforehand, but only becomes revealed by actually executing the job until it signals its completion. The values of $\{\gamma_i^q\}$ for a given scenario of a given task set τ define system behavior and its corresponding real-time requirements as follows.

- As long as no job executes for more than its LO WCET for all jobs, the system is regarded as exhibiting the LO *behavior*, and all jobs are required to be completed before their deadlines.
- If any HI job does not signal its completion after executing for its LO WCET at some time instant t^* referred to as *mode change*, the system is regarded as exhibiting the HI *behavior*, and only HI jobs are required to be completed before their deadlines after the mode change; every LO job whose deadline is later than t^* can be discarded in the HI system behavior.

MC-feasibility. Based on the above two real-time requirements, we define MC-feasibility as follows.

Definition 1 (MC-feasible): A scenario is said to be *feasible* if there exists a schedule that satisfies i) every job J_i^q receives execution time γ_i^q during its execution window $[r_i^q, d_i^q]$ when the system exhibits the LO behavior and ii) every HI job J_j^p receives execution time γ_j^p during its execution window $[r_j^p, d_j^p]$ when the system exhibits the HI behavior. A task set τ is said to be *MC-feasible* if *every* scenario is feasible.

According to Definition 1, a task set τ is said to be *MC-infeasible* if there exists *at least one* scenario that is infeasible (i.e., not feasible). Note that determining MC-feasibility for collections of independent dual-criticality periodic and sporadic tasks is known to be NP-hard in the strong sense [10].

Assumption and notation. We assume a quantum-based time; let one time unit a quantum length without loss of generality. We also assume that if the system has switched to high-criticality behavior, it will never switch back to low-criticality. Some recent studies have considered returning the system to low-criticality mode (see [11] for a survey), but this is not relevant to MC-feasibility because it does not change the definition of MC-feasibility. Let LHS and RHS denote left-hand-side and right-hand-side, respectively.

III. Challenges

In this section, we pose the following problem: what are unique issues of developing necessary feasibility tests for MC task systems (that do not matter for SC task systems)? To answer the question, we first present our own interpretation of the existing necessary feasibility test for SC task systems. We next present trivial existing results for necessary feasibility of MC task systems. Based on the background, we observe characteristics specialized for MC task systems, and identify challenges for developing necessary feasibility tests for MC task systems.

A. Existing necessary feasibility test for SC task systems

In this subsection, we offer our own *interpretation* of the existing necessary feasibility test for SC task systems [12].

A typical way to develop a necessary feasibility test for SC task systems is to focus on a scenario (associated with a given interval of interest, each task's job release pattern, and each job's execution requirement) and to compare the sum of every job's *minimum* execution requirement that should be performed in the interval of interest to avoid its deadline miss (called *demand*), with the time duration in which the computing platform allows jobs to execute within the interval (called *supply*). If the demand is larger than the supply, at least one job in the scenario inevitably misses its deadline, yielding infeasibility of the task set that invokes the scenario. While the demand depends on the scenario's interval of interest, each task's job release pattern, and each job's execution time requirement, the existing study for SC task systems focuses on the following scenario S1, S2 and S3' with given interval length $t^{\text{end}} > 0$.

S1. Target $[0, t^{\text{end}}]$ as an interval of interest, for given interval length $t^{\text{end}} > 0$.

S2. Generate jobs according to the *synchronous* periodic job release pattern from $t = 0$ as follows. The first job of every task is released at 0, and the following jobs of every task are released strictly periodically until each job's absolute deadline is no later than t^{end}.

S3'. Determine the execution requirement of every job as its WCET.

For the scenario of S1, S2 and S3' with given $t^{\text{end}} > 0$, the existing study calculates demand and supply in the interval of interest, and deems the scenario (and therefore the corresponding task set) infeasible if the demand is strictly larger than the supply. While it is straightforward that the supply under S1 with given $t^{\text{end}} > 0$ amounts to t^{end}, the demand of a SC task τ_i with its WCET of C_i under the scenario of S1, S2 and S3' with given $t^{\text{end}} > 0$ is calculated by $\mathtt{DBF}_i(t^{\text{end}})$ where

$$\mathtt{DBF}_i(t) = \left(\left\lfloor \frac{t - D_i}{T_i} \right\rfloor + 1 \right) \cdot C_i, \tag{1}$$

which yields the following lemma.

Lemma 1 (From [12]): A SC task set τ is infeasible, if Eq. (2) is violated for given $t^{\text{end}} > 0$:

$$\sum_{\tau_i \in \tau} \mathtt{DBF}_i(t^{\text{end}}) \leq t^{\text{end}}. \tag{2}$$

By Lemma 1, it is possible to find an infeasible task set (if Eq. (2) is violated for given $t^{\text{end}} > 0$), meaning that a necessary feasibility test is successfully derived.

The next issue is how to systematically check the lemma for the scenario of S1, S2 and S3' with as many $t^{\text{end}} > 0$ as possible with reasonable time-complexity. The existing study derives an upper bound of t^{end} such that checking Lemma 1 for the scenario of S1, S2 and S3' with every $t^{\text{end}} > 0$ less than the upper bound is equivalent to checking that with every $t^{\text{end}} > 0$ without the upper bound (up to infinity). Using the upper-bound, the study develops the following collective necessary feasibility test as follows.

- A SC task set τ is infeasible, if there exists at least one $t^{\text{end}} > 0$ that violates Eq. (2) while we repeat to check Lemma 1 with every $t^{\text{end}} > 0$ less than the upper bound.

Note that it has been proven that the collective necessary feasibility test exhibits pseudo-polynomial time-complexity in the task parameters. Also, note that the test is known to be a necessary *and* sufficient feasibility test for SC task systems [12].

One may misunderstand that the collective necessary feasibility test *should check* Lemma 1 for *every* $t^{\text{end}} > 0$ less the upper bound. Different from the corresponding sufficient feasibility test, the necessary feasibility test can check Lemma 1 for any number of candidates for $t^{\text{end}} > 0$, which can affect capability in finding infeasible task sets, but cannot compromise the correctness of whether task sets deemed infeasible by the test is actually infeasible.

B. Trivial results for necessary feasibility of MC task systems

Considering the most prominent difference between SC and MC task systems is existence of the mode change and consequences thereof, we may classify scenarios of MC task systems, based on relationship between the mode change instant and the interval of interest. That is, the mode change instant t^* exists after, before, and within the interval of interest, denoted by Cases A, B, and C, respectively.

Since all jobs in the interval of interest *exclusively* experiences the LO and HI system behavior in Cases A and B, respectively, each scenario of Case A and B can be equivalent to a scenario of a SC task system. The following two lemmas correspond Lemma 1 for Cases A and B, respectively (similar conditions were presented in [7], [8], [9] with a different form).

Lemma 2: A MC task set τ is infeasible, if Eq. (3) is violated for given $t^{\text{end}} > 0$:

$$\sum_{\tau_i \in \tau} \mathtt{DBF}_i^{\mathtt{LO}}(t^{\text{end}}) \leq t^{\text{end}} \text{ where } \mathtt{DBF}_i^{\mathtt{LO}}(t) = \Big(\Big\lfloor \frac{t - D_i}{T_i} \Big\rfloor + 1\Big) \cdot C_i^{\mathtt{LO}}. \tag{3}$$

Proof: Consider the scenario of S1, S2 and S3' with given $t^{\text{end}} > 0$ when the mode change occurs at $t = \infty$. Then, the scenario is the same as Lemma 1 with replacing C_i with $C_i^{\mathtt{LO}}$ for every $\tau_i \in \tau$. ■

Lemma 3: A MC task set τ is infeasible, if Eq. (4) is violated for given $t^{\text{end}} > 0$:

$$\sum_{\tau_i \in \tau^{\mathtt{HI}}} \mathtt{DBF}_i^{\mathtt{HI}}(t^{\text{end}}) \leq t^{\text{end}} \text{ where } \mathtt{DBF}_i^{\mathtt{HI}}(t) = \Big(\Big\lfloor \frac{t - D_i}{T_i} \Big\rfloor + 1\Big) \cdot C_i^{\mathtt{HI}}. \tag{4}$$

Proof: Consider the scenario of S1, S2 and S3' with given $t^{\text{end}} > 0$ when the mode change occurs at $t = -\infty$. Since there is no demand of LO tasks in $[0, t^{\text{end}}]$, the scenario is the same as Lemma 1 with replacing C_i with $C_i^{\mathtt{HI}}$ for every $\tau_i \in \tau^{\mathtt{HI}}$ and C_i with 0 for every $\tau_i \in \tau^{\mathtt{LO}}$. ■

To find as many infeasible task sets as possible, the collective necessary feasibility test can be developed by applying Lemmas 2 and 3 for every $t^{\text{end}} > 0$ less than its upper-bound (derived by the same technique for SC task systems).[1]

Although Lemmas 2 and 3 successfully address Cases A and B, there is a still huge gap between the region covered by task sets proven feasible and that proven infeasible, as shown in Fig. 1; note that Lemmas 2 and 3 with every $t^{\text{end}} > 0$ prove infeasibility of task sets beyond the region X=1.0 by Y=1.0 (colored by grey) in the figure. While tight necessary conditions for feasibility require to consider Case C effectively, deriving the conditions from Case C entails many challenges to be discussed in the next subsection.

Note that although existing demand-based schedulability tests [3], [5], [6] considered Case C using the notion of demand, all of them are unable to detect infeasible task sets because they are designed to find feasible task sets.

C. Challenges for developing necessary feasibility tests for MC task systems

In this subsection, we identify challenges specialized for MC tasks systems to develop necessary feasibility tests, by considering Case C in which the mode change instant t^* exists in the middle of the interval of interest.

We may observe two characteristics of SC task systems to derive necessary conditions for feasibility in Section III-A. First, the demand in the interval of interest is fixed with the scenario of S1–S3' with given $t^{\text{end}} > 0$. Second, if the demand is larger than the supply, the scenario (and therefore the corresponding task set) is actually infeasible. The two characteristics, however, do not hold for MC task systems, due to existence of the mode change. We present the following example showing characteristics of MC task systems that affect derivation of necessary conditions for feasibility. Note that the characteristics to be explained are mostly derived from existence of the mode change, which switches (i) a set of jobs whose execution requirement should be guaranteed (i.e., from HI and LO jobs, to HI jobs only) and (ii) HI jobs' amount of execution requirement that should be guaranteed (i.e., from no more than the LO WCET, to more than the LO WCET).

Example 1: Consider a task set τ with the following three tasks: $\tau_1(T_1 = 12, \chi_1 = \mathtt{HI}, C_1^{\mathtt{LO}} = 3, C_1^{\mathtt{HI}} = 6, D_1 = 12)$,

[1]The collective necessary feasibility test is reduced to checking violation of either $\sum_{\tau_i \in \tau} C_i^{\mathtt{LO}}/T_i \leq 1.0$ or $\sum_{\tau_i \in \tau^{\mathtt{HI}}} C_i^{\mathtt{HI}}/T_i \leq 1.0$, for implicit-deadline task systems [7], [8], [9].

$\tau_2 = \tau_1$, and $\tau_3 = (4, \texttt{LO}, 1, 1, 4)$. Consider the following scenario: (i) the interval of interest is $[0, 12]$; (ii) the synchronous job release pattern from $t = 0$ is applied to all tasks in $[0, 12]$, meaning that the release time and deadline of J_1^1 and J_2^1 are 0 and 12, respectively, and those of J_3^1, J_3^2, and J_3^3 are 0 and 4, 4 and 8, and 8 and 12, respectively; and (iii) the execution requirement of every LO and HI job is its LO and HI WCET, respectively. We now consider two cases with different choices of the mode change instant: $t^* = 3$ and 4. In both cases, the following properties hold: (a) one of J_1^1 or J_2^1 should execute for exactly 3 time units (i.e., its LO WCET) in $[0, t^*]$ (i.e., before the mode change) to trigger the mode change, and therefore (b) the job that triggers the mode change should execute for exactly 3 time units (i.e., its HI WCET minus LO WCET) in $[t^*, 12]$ (i.e., after the mode change).

Suppose the mode change instant occurs at $t^* = 3$. The demand of jobs of τ_3 in $[0, 12]$ is 0 because no job of τ_3 has its deadline no later than the mode change instant. Also, one of J_1^1 and J_2^1 that does not trigger the mode change cannot execute before the mode change (i.e., in $[0, 3]$) because of (a), and hence the job should execute for 6 time units (i.e., its HI WCET) after the mode change (i.e., in $[3, 12]$). In this case, since there is no demand of τ_3, the total demand in $[0, 12]$ is $C_1^{\texttt{HI}} + C_2^{\texttt{HI}} = 12$, which is no larger than the supply in $[0, 12]$.

Suppose the mode change instant occurs at $t^* = 4$. Then, the demand of jobs of τ_3 in $[0, 12]$ is 1 because J_3^1 has its deadline no later than the mode change instant (while other jobs of τ_3 do not). Also, one of J_1^1 and J_2^1 that does not trigger the mode change executes for at most 1 time unit before the mode change (i.e., in $[0, 4]$) because of (a). Considering the sum of execution of the job in $[0, 4]$ and that of $[4, 12]$ (therefore the sum of the demand in both intervals) should be 6 (its HI WCET), the job should execute for 5 or 6 time units after the mode change (i.e., in $[4, 12]$). In this case, since the demand of τ_3 is 1, the total demand in $[0, 12]$ is $C_3^{\texttt{LO}} + C_1^{\texttt{HI}} + C_2^{\texttt{HI}} = 13$, which is larger than the supply in $[0, 12]$.

In fact, the scenario is feasible if J_1^1 and J_2^1 are executed in $[0, 6]$ and $[6, 12]$, respectively. This is because, this schedule yields the mode change at $t^* = 3$, which is before the deadline of all LO jobs (i.e., $t = 4$, 8 or 12); therefore, all LO jobs do not have any demand in $[0, 12]$.

We summarize the following observations from Example 1.

O1. The contribution of each LO job to the demand varies with the mode change instant. For example, the demand of LO jobs amounts to 0 and 1 with $t^* = 3$ and $t^* = 4$, respectively.

O2. The constraints for the amount of the contribution of each HI job to the demand varies with the mode change instant. For example, one of J_1^1 and J_2^1 that does not trigger the mode change executes for 0 and at most 1 time unit before the mode change, respectively in case of $t^* = 3$ and $t^* = 4$.

O3. By O1 and O2, it is impossible (or at least very difficult) to calculate demand without specifying the mode change instant.

O4. The demand strictly larger than the supply in a case does not necessarily yield infeasibility of the scenario. For example, the scenario is feasible although the demand is larger than supply with $t^* = 4$.

O5. Without a concrete schedule determined by the target scheduling algorithm, we may not calculate the exact values for the demand of a HI job in one sub-interval and that in another sub-interval; however, there exists a relationship between the demand in those intervals. For example, the sum of the demand of J_1^1 (or J_2^1) in $[0, t^*]$ and that in $[t^*, t^{\text{end}}]$ is 6.

We now present another example that shows necessity of investigating sub-intervals of the interval of interest for higher capability in finding infeasible task sets.

Example 2: Consider a task set τ with the following three tasks: $\tau_1(T_1 = 12, \chi_1 = \texttt{HI}, C_1^{\texttt{LO}} = 3, C_1^{\texttt{HI}} = 6, D_1 = 12)$, $\tau_2 = (12, \texttt{HI}, 3, 5, 12)$, and $\tau_3 = (2, \texttt{LO}, 1, 1, 2)$. Consider the scenario same as Example 1.

We now present a case with the mode change instant $t^* = 3$, in which one of J_1^1 or J_2^1 should execute for exactly 3 time units (i.e., its LO WCET) in $[0, 3]$. The demand of jobs of τ_3 in $[0, 3]$ is 1 (i.e., $C_3^{\texttt{LO}}$), because J_3^1 is the only job whose deadline is no later than $t = 3$. Therefore, the total demand[2] in $[0, 3]$ is $3 + 1 = 4$, which is larger than the supply in $[0, 3]$; this judges that the mode change cannot occur at $t^* = 3$ without violating i) of Definition 1. However, the same cannot be judged if we focus on the entire interval of $[0, 12]$; this is because the total demand in $[0, 12]$ is $C_3^{\texttt{LO}} + C_1^{\texttt{HI}} + C_2^{\texttt{HI}} = 12$.

In fact (after investigating all possible schedules), the scenario is deemed infeasible because there is no schedule that satisfies i) and ii) in Definition 1. Note that both Lemmas 2 and 3 cannot deem the task set infeasible, while our proposed necessary feasibility test to be developed in this paper (i.e., Theorem 2) can.

From Example 2, we have the following observation.

O6. The inequality of the demand larger than the supply holds in a subset of the interval of interest, while the same does not hold in the entire interval of interest. For example, the inequality holds in $[0, 3]$, but does not hold in $[0, 12]$.

Considering the unique observations O1–O6, we need to address the following challenges to develop necessary feasibility tests for MC task systems.

I1. For a given scenario, how can we characterize and calculate the demand in an interval that changes depending on the mode change instant? (from O1–O3)

I2. For a given scenario, what is the meaning of the demand larger than the supply in an interval when the mode change instant is given? (from O4)

I3. For a given scenario, how can we derive infeasibility of the scenario from the answer of I2 without assuming the mode change instant is given? (from O3–O4)

I4. For a given scenario, what are good choices of sub-intervals to be targeted for I1–I3? (from O6) How can we utilize the relationship between demand of those sub-intervals? (from O5)

Since I1–I4 need a given scenario, we have the following challenge.

[2]In this case, the notion of demand implies the sum of every job's minimum execution requirement that should be performed in the interval of interest not only to satisfy MC-feasility in Definition 1, but also to trigger the mode change at $t^* = 3$.

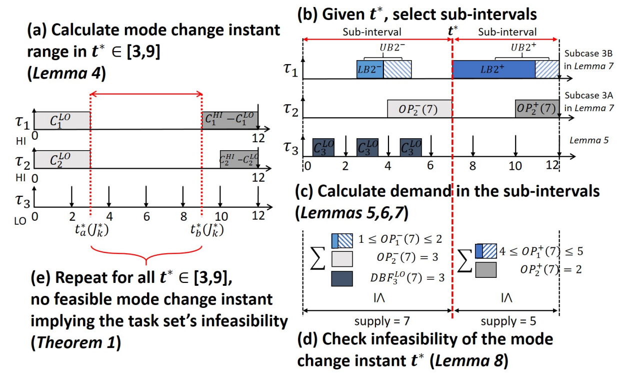

Fig. 2. Overview of our proposed necessary feasibility test for a task set described in Examples 2 and 3

I5. What are additional scenario components for MC task systems other than S1, S2 and S3', which make it possible to address I1–I4? In particular, how can we make the components specify the mode change instant without targeting a scheduling algorithm? (from O3)

In addition, we have the following challenge for higher capability in finding infeasible task sets.

I6. Once we develop a necessary feasibility test for a given scenario by addressing I1–I5, how can we efficiently check the test with as many scenarios as possible with reasonable time-complexity?

Sections IV will present how to develop necessary feasibility tests for MC task systems by addressing I1–I5, while Sections V and VI will address I6 in a different manner.

IV. Development of Necessary Feasibility Test

In this section, we develop a necessary feasibility test for MC task systems, by addressing I1–I5 in Section III-C. An overview of our approach is illustrated in Fig. 2. We first tailor the scenario of S1, S2, and S3' in Section III-A to MC task systems and specify a range of the mode change instant (addressing I5, Fig. 2(a)). We then determine target sub-intervals (addressing I4, Fig. 2(b)) and explain how to calculate the demand in the target sub-intervals (addressing I1, Fig. 2(c)). Comparing the demand with the supply in the target sub-intervals, we derive a necessary condition for feasibility of the mode change instant t^* (addressing I2, Fig. 2(d)). By associating the necessary conditions with different mode change instants, we develop a necessary feasibility test for MC task systems (addressing I3, Fig. 2(e)).

Since a given scenario is assumed in I1–I4, we first determine additional scenario components for MC task systems in order to address I5. Among the scenario of S1, S2 and S3' for SC task systems, S1 and S2 can be applied as they are, while S3' should be adapted because each HI job has LO and HI WCETs. Considering the execution requirement of each HI job is relevant to the mode change instant, we need to adapt S3' for HI jobs so as to (i) determine each HI job's execution requirement as its LO or HI WCET, and (ii) specify the mode change instant without the target scheduling algorithm.

The main idea to address (i) and (ii) is to include a scenario component of J_k^*, which indicates the job with the earliest release time among all HI jobs whose execution requirement is strictly larger than its LO WCET. By the proposed definition, J_k^* should observe a mode change (triggered by itself or another job) within its execution window; otherwise, the job cannot execute for more than its LO WCET, yielding its deadline miss. Therefore, by targeting given J_k^*, we can restrict the range of the mode change instant for each scenario, to at most the execution window of J_k^*. Also, by targeting given J_k^*, we can determine each HI job's execution requirement based on its release time. Then, we consider S3 and S4 by adapting/detailing S3' as follows.

S3. Target a given job J_k^* among all jobs generated by S1 and S2 with given $t^{\text{end}} > 0$, where J_k^* is the job which has the earliest release time among all HI jobs whose execution requirement is strictly larger than LO WCET.

S4. Determine the execution requirement of every LO job as its LO WCET; determine the execution requirement of every HI job whose release time is earlier and no earlier than r_k^* (i.e., J_k^*'s release time) as its LO and HI WCET, respectively.

In the rest of this paper, we target the scenario of S1 and S2 defined in Section III-A, and S3 and S4 defined in this section, with given $t^{\text{end}} > 0$ and J_k^*. Then, the following lemma calculates a more refined range of the mode change instant t^* associated with given J_k^* as illustrated in Fig. 2(a).

Lemma 4: The scenario of S1–S4 with given $t^{\text{end}} > 0$ and J_k^* is feasible, only if the mode change occurs within $[t_a^*(J_k^*), t_b^*(J_k^*)]$, where $t_a^*(J_k^*)$ and $t_b^*(J_k^*)$ are respectively the earliest $(r_i^p + C_i^{\text{LO}})$ and the earliest $(d_i^p - C_i^{\text{HI}} + C_i^{\text{LO}})$ among every HI job J_i^p whose release time is no earlier than r_k^* (i.e., the release time of J_k^*).

Proof: Recall S4; every HI job J_i^p whose release time is no earlier than r_k^* has the execution requirement for its HI WCET.

Suppose that the mode change occurs at $t^* < t_a^*(J_k^*)$. By S4 and the definition of $t_a^*(J_k^*)$, it is impossible for every HI job whose execution requirement equals to its HI WCET to perform its executing for as much as its LO WCET before $t_a^*(J_k^*)$. This contradicts the supposition.

Suppose that a mode change occurs at $t^* > t_b^*(J_k^*)$. By S4 and the definition of $t_b^*(J_k^*)$, there exists at least one job J_i^p whose execution requirement equals to its HI WCET, but whose actual execution time performed until $(d_i^p - C_i^{\text{HI}} + C_i^{\text{LO}})$ is less than C_i^{LO}. If there exists no such job, the mode change occurs no later than $(d_i^p - C_i^{\text{HI}} + C_i^{\text{LO}})$. The former contradicts feasibility, and the latter contradicts supposition. ■

Next, we explain how to choose sub-intervals to be investigated (i.e,. addressing the first part of I4), which should precede addressing I1–I3. As shown in O1 and O2 in Section III-C, the mode change instant t^* determines not only whether each LO job contributes to the demand or not, but also how much execution each HI job can contribute to the demand before and after the mode change. Since the mode change instant t^* is a criterion that determines the demand (or its constraints) of LO and HI jobs, we not only target a situation with given t^*, but also divide the interval of interest into two sub-intervals based on the mode change instant t^* (i.e., $[0, t^*]$ and $[t^*, t^{\text{end}}]$), to be targeted for deriving necessary feasibility conditions as illustrated in Fig. 2(b). As shown in O6, demand-supply comparison with the interval split yields higher capability in finding infeasible task sets, compared to that without the interval split.

Once we determine the target sub-intervals, we are ready to address I1, which is to calculate the demand in the target sub-intervals when the mode change instant t^* is given. One may wonder whether it is possible to use the techniques of demand calculation in existing demand-based schedulability tests for MC task systems [3], [5], [6]; this is impossible because they were designed for a target scheduling algorithm to derive sufficient feasibility conditions, while this paper derives necessary feasibility conditions meaning that no target scheduling algorithm is assumed.

We first investigate the demand of a LO task. Considering a LO job can contribute the demand only if the job's deadline is no later than t^*, the demand of jobs of a LO task in $[0, t^*]$ can be calculated (and that in $[t^*, t^{\text{end}}]$ is zero), as stated in the following lemma.

Lemma 5: Target the scenario of S1–S4 with given $t^{\text{end}} > 0$ and J_k^*, and target a single mode change instant t^* belonging to $[t_a^*(J_k^*), t_b^*(J_k^*)]$ (defined in Lemma 4). Then, the demand of jobs of a LO task $\tau_i \in \tau$ in $[0, t^*]$ and $[t^*, t^{\text{end}}]$ can be calculated as follows.

- The demand of jobs of $\tau_i \in \tau^{\text{LO}}$ in $[0, t^*]$ is $\texttt{DBF}_i^{\text{LO}}(t^*)$.
- The demand of jobs of $\tau_i \in \tau^{\text{LO}}$ τ_i in $[t^*, t^{\text{end}}]$ is 0.

Proof: If the mode change occurs before the LO job's deadline, we do not need to execute the job, meaning that the demand for the job is zero. Therefore, the latter directly holds. Considering the definition of $\texttt{DBF}_i^{\text{LO}}(t)$ in Eq. (3), the former also holds. ■

The next issue is to calculate the demand of a HI task in the target sub-intervals. Considering the relationship between the mode change instant t^* and each job's execution window, we may classify all jobs of a HI task $\tau_i \in \tau^{\text{HI}}$ into three categories:

CG1. jobs J_i^p whose deadline is no later than t^* (i.e., $d_i^p \leq t^*$),
CG2. at most one job J_i^p whose release time is earlier than t^* but whose deadline is later than t^* (i.e., $r_i^p < t^* < d_i^p$),
CG3. jobs J_i^p whose release time is no earlier than the mode change instant (i.e., $t^* \leq r_i^p$).

While CG1 and CG3 exclusively contribute to the demand in $[0, t^*]$ and that in $[t^*, t^{\text{end}}]$, respectively, CG2 may contribute to both. Let $\texttt{OP}_i^-(t^*)$ and $\texttt{OP}_i^+(t^*)$ denote the demand of a job of $\tau_i \in \tau^{\text{HI}}$ belonging to CG2 in $[0, t^*]$ and that in $[t^*, t^{\text{end}}]$, respectively, under the scenario of S1–S4 with given $t^{\text{end}} > 0$ and J_k^*, and given $t^* \in [t_a^*(J_k^*), t_b^*(J_k^*)]$. Note that $\texttt{OP}_i^-(t^*)$ and $\texttt{OP}_i^+(t^*)$ are 0, if no job of $\tau_i \in \tau^{\text{HI}}$ belongs to CG2.

Then, we can calculate/express the demand of jobs of a HI task in $[0, t^*]$ and that in $[t^*, t^{\text{end}}]$ by CG1, CG2, and CG3 separately, as stated in the following lemma.

Lemma 6: Target the scenario of S1–S4 with given $t^{\text{end}} > 0$ and J_k^*, and target a single mode change instant t^* belonging to $[t_a^*(J_k^*), t_b^*(J_k^*)]$ (defined in Lemma 4). Then, the demand of jobs of a HI task τ_i in $[0, t^*]$ and $[t^*, t^{\text{end}}]$ can be calculated/expressed as follows.

- The demand of jobs of $\tau_i \in \tau^{\text{HI}}$ belonging to CG1 in $[0, t^*]$ amounts to $\texttt{DBF}_i^{\text{LO}}(t^*)$.
- The demand of jobs of $\tau_i \in \tau^{\text{HI}}$ belonging to CG1 in $[t^*, t^{\text{end}}]$ amounts to 0.
- The demand of at most one job of $\tau_i \in \tau^{\text{HI}}$ belonging to CG2 in $[0, t^*]$ amounts to $\texttt{OP}_i^-(t^*)$.
- The demand of at most one job of $\tau_i \in \tau^{\text{HI}}$ belonging to CG2 in $[t^*, t^{\text{end}}]$ amounts to $\texttt{OP}_i^+(t^*)$.
- The demand of jobs of $\tau_i \in \tau^{\text{HI}}$ belonging to CG3 in $[0, t^*]$ amounts to 0.
- The demand of jobs of $\tau_i \in \tau^{\text{HI}}$ belonging to CG3 in $[t^*, t^{\text{end}}]$ amounts to $\texttt{DBF}_i^{\text{HI}}\big(t^{\text{end}} - \lceil t^*/T_i \rceil \cdot T_i\big)$.

Proof: Since every HI job of τ_i belonging to CG1 executes for its LO WCET, the first statement holds. The second and fifth statements hold by the definition of jobs in CG1 and CG3. The third and fourth statements are the definition of $\texttt{OP}_i^-(t^*)$ and $\texttt{OP}_i^+(t^*)$. Among jobs of τ_i belonging to CG3, the earliest release time is $(\lceil t^*/T_i \rceil \cdot T_i)$; considering the definition of $\texttt{DBF}_i^{\text{HI}}(t)$ in Eq. (4), the sixth statement holds. ■

While the first, second, fifth and sixth statements in Lemma 6 calculate exact values for the corresponding demand, the third and fourth statements do not. Considering the sum of $\texttt{OP}_i^-(t^*)$ and $\texttt{OP}_i^+(t^*)$ equals to the execution requirement of the job of τ_i belonging to CG2 (addressing the second part of I4), we can derive the constraints of the contribution of $\texttt{OP}_i^-(t^*)$ and $\texttt{OP}_i^+(t^*)$ to the demand in the target sub-intervals, as shown in the following example.

Example 3: Recall the task set and the scenario in Example 2 in Section III-C, and target the mode change instant $t^* = 7$. We now explain the constraints of $\texttt{OP}_1^-(7)$ and $\texttt{OP}_1^+(7)$ for J_1^1. If J_1^1 triggers the mode change, the job should execute for exactly 3 time units (i.e., C_1^{LO}) in $[0, 7]$ and 3 time units (i.e., $C_1^{\text{HI}} - C_1^{\text{LO}}$) in $[7, 12]$, implying $\texttt{OP}_1^-(7) = 3$ and $\texttt{OP}_1^+(7) = 3$. We discuss the case where J_1^1 does not trigger the mode change, from now on, which is illustrated in Fig. 2(c).

Let $\texttt{LB2}^-$ and $\texttt{UB2}^-$ denote lower and upper bounds for $\texttt{OP}_1^-(7)$, and $\texttt{LB2}^+$ and $\texttt{UB2}^+$ denote lower and upper bounds for $\texttt{OP}_1^+(7)$. Then, J_1^1's execution cannot be larger than its target sub-interval lengths, $\texttt{UB2}^- \leq 7$ and $\texttt{UB2}^+ \leq 5$. Also, for J_1^1 not to trigger the mode change, J_1^1's execution in $[0, 7]$ should be strictly less than its LO WCET, yielding $\texttt{UB2}^- = \min(7, C_1^{\text{LO}} - 1 = 2) = 2$. Also, J_1^1's execution in $[7, 12]$ should not be larger than its execution requirement, yielding $\texttt{UB2}^+ = \min(5, C_1^{\text{HI}} = 6) = 5$. Considering the sum of $\texttt{OP}_1^-(7)$ and $\texttt{OP}_1^+(7)$ equals to J_1^1's execution requirement, we can calculate $\texttt{LB2}^- = C_1^{\text{HI}} - \texttt{UB2}^+ = 6 - 5 = 1$ and $\texttt{LB2}^+ = C_1^{\text{HI}} - \texttt{UB2}^- = 6 - 2 = 4$.

Inspired by the example, the following lemma formalizes the constraints of the contribution of $\texttt{OP}_i^-(t^*)$ and $\texttt{OP}_i^+(t^*)$ to the demand in the target sub-intervals.

Lemma 7: Target the scenario of S1–S4 with given $t^{\text{end}} > 0$ and J_k^*, and target a single mode change instant t^* belonging to $[t_a^*(J_k^*), t_b^*(J_k^*)]$ (defined in Lemma 4). Consider a job of $\tau_i \in \tau^{\text{HI}}$ potentially belonging to CG2, J_i^q, whose release time and deadline are $r_i^q = \lfloor \frac{t^*}{T_i} \rfloor \cdot T_i$ and $d_i^q = r_i^q + D_i$, respectively. Considering J_i^q, we have three cases for calculating $\texttt{OP}_i^-(t^*)$ and $\texttt{OP}_i^+(t^*)$. In Case 1, there is no job of τ_i in CG2; J_i^q does not belong to CG2. In Cases 2 and 3, J_i^q is the job of τ_i that belongs to CG2, but the job's execution requirement amounts to its $\texttt{LO}$ and $\texttt{HI}$ WCET, respectively. Case 3 consists of Subcases 3A and 3B, in which J_i^q does trigger and does not trigger the mode change, respectively. Then, $\texttt{OP}_i^-(t^*)$ and $\texttt{OP}_i^+(t^*)$ for every $\tau_i \in \tau^{\texttt{HI}}$ should satisfy the following constraints.

- Case 1: If $r_i^q = t^*$, $d_i^q \leq t^*$ or $d_i^q > t^{\text{end}}$, $\texttt{OP}_i^-(t^*) = \texttt{OP}_i^+(t^*) = 0$ holds.
- Case 2: Otherwise, if $r_i^q < r_k^*$ (recall r_k^* is J_k^*'s release time),
(i) $\texttt{LB}^- \leq \texttt{OP}_i^-(t^*) \leq \texttt{UB}^-$,
(ii) $\texttt{LB}^+ \leq \texttt{OP}_i^+(t^*) \leq \texttt{UB}^+$, and
(iii) $\texttt{OP}_i^+(t^*) + \texttt{OP}_i^-(t^*) = C_i^{\texttt{LO}}$ hold, where
$\texttt{UB}^- = \min(t^* - r_i^q, C_i^{\texttt{LO}})$, $\texttt{UB}^+ = \min(d_i^q - t^*, C_i^{\texttt{LO}})$,
$\texttt{LB}^- = C_i^{\texttt{LO}} - \texttt{UB}^+$ and $\texttt{LB}^+ = C_i^{\texttt{LO}} - \texttt{UB}^-$.
- Case 3: Otherwise (i.e., $r_i^q \geq r_k^*$),
(Subcase 3A) if $t^* - r_i^q \geq C_i^{\texttt{LO}}$ and $d_i^q - t^* \geq C_i^{\texttt{HI}} - C_i^{\texttt{LO}}$ hold and the job of τ_i triggers the mode change,
(i) $\texttt{OP}_i^-(t^*) = C_i^{\texttt{LO}}$ and
(ii) $\texttt{OP}_i^+(t^*) = C_i^{\texttt{HI}} - C_i^{\texttt{LO}}$ hold;
(Subcase 3B) otherwise (i.e., the job of τ_i does not trigger the mode change),
(i) $\texttt{LB2}^- \leq \texttt{OP}_i^-(t^*) \leq \texttt{UB2}^-$,
(ii) $\texttt{LB2}^+ \leq \texttt{OP}_i^+(t^*) \leq \texttt{UB2}^+$, and
(iii) $\texttt{OP}_i^+(t^*) + \texttt{OP}_i^-(t^*) = C_i^{\texttt{HI}}$ hold, where
$\texttt{UB2}^- = \min(t^* - r_i^q, C_i^{\texttt{LO}} - 1)$, $\texttt{UB2}^+ = \min(d_i^q - t^*, C_i^{\texttt{HI}})$,
$\texttt{LB2}^- = C_i^{\texttt{HI}} - \texttt{UB2}^+$ and $\texttt{LB2}^+ = C_i^{\texttt{HI}} - \texttt{UB2}^-$.

Proof: (Case 1) If $r_i^q = t^*$ (or $d_i^q \leq t^*$), J_i^q belongs to CG3 (or CG1) and there is no job of τ_i in CG2. Also, if $d_i^q > t^{\text{end}}$, the scenario of S2 does not generate J_i^q.

(Case 2) If $r_i^q < r_k^*$ (i.e., the release time of J_i^q less than that of J_k^*), J_i^q's execution requirement is $C_i^{\texttt{LO}}$ by the definition of J_k^* and S4, yielding $\texttt{OP}_i^-(t^*) + \texttt{OP}_i^+(t^*) = C_i^{\texttt{LO}}$. J_i^q can be executed in $[r_i^q, t^*]$ and $[t^*, d_i^q]$ (i.e., before and after the mode change), for at most the interval length, i.e., $(t^* - r_i^q)$ and $(d_i^q - t^*)$ time units, respectively, yielding upper bounds of $\texttt{OP}_i^-(t^*)$ and $\texttt{OP}_i^+(t^*)$. Considering J_i^q's execution requirement equals to $C_i^{\texttt{LO}}$, we can derive upper bounds of $\texttt{OP}_i^-(t^*)$ and $\texttt{OP}_i^+(t^*)$ as $\texttt{UB}^-$ and $\texttt{UB}^+$. If we use $\texttt{OP}_i^-(t^*) + \texttt{OP}_i^+(t^*) = C_i^{\texttt{LO}}$, we can derive lower bounds of $\texttt{OP}_i^-(t^*)$ and $\texttt{OP}_i^+(t^*)$ as $\texttt{LB}^-$ and $\texttt{LB}^+$, from $\texttt{UB}^+$ and $\texttt{UB}^-$.

(Case 3) This case implies that J_i^q's execution requirement is $C_i^{\texttt{HI}}$ (by the definition of J_k^* and S4), yielding $\texttt{OP}_i^-(t^*) + \texttt{OP}_i^+(t^*) = C_i^{\texttt{HI}}$.

(Subcase 3A) This subcase implies that J_i^q triggers the mode change, which requires J_i^q to execute for $C_i^{\texttt{LO}}$ in $[r_i^q, t^*]$ and for $(C_i^{\texttt{HI}} - C_i^{\texttt{LO}})$ in $[t^*, d_i^q]$. Therefore, the conditions (i) and (ii) for Subcase 3A hold.

(Subcase 3B) This subcase implies that J_i^q does not trigger the mode change. Therefore, there is a limit for the maximum of $\texttt{OP}_i^-(t^*)$ and $\texttt{OP}_i^+(t^*)$, which is $(C_i^{\texttt{LO}} - 1)$ and $C_i^{\texttt{HI}}$, respectively. While the latter is straightforward, the former holds because executing for $C_i^{\texttt{LO}}$ before the mode change implies that J_i^q triggers the mode change, which contradicts the supposition of Subcase 3B. Applying the same idea as Case 2, we can derive upper bounds of $\texttt{OP}_i^-(t^*)$ and $\texttt{OP}_i^+(t^*)$ as $\texttt{UB2}^-$ and $\texttt{UB2}^+$, and then derive lower bounds of $\texttt{OP}_i^-(t^*)$ and $\texttt{OP}_i^+(t^*)$ as $\texttt{LB2}^-$ and $\texttt{LB2}^+$, from $\texttt{UB2}^+$ and $\texttt{UB2}^-$. ■

Combining Lemmas 6 and 7, we can calculate the demand of all $\texttt{HI}$ tasks in $[0, t^*]$ and $[t^*, t^{\text{end}}]$, by adding the demand of jobs of each $\texttt{HI}$ task in CG1, CG2 and CG3. By combining the demand of all $\texttt{HI}$ tasks and that of all $\texttt{LO}$ tasks in Lemma 5, we can calculate the total demand in $[0, t^*]$ and $[t^*, t^{\text{end}}]$. If the total demand is larger than the total supply in $[0, t^*]$ or the same holds in $[t^*, t^{\text{end}}]$, it is impossible for the mode change to occur at t^* without any job deadline miss, which addresses I2. In other words, comparing the total demand with the total supply in $[0, t^*]$ and $[t^*, t^{\text{end}}]$, we can judge the feasibility of the mode change at t^* without missing any job deadline, as stated in the following lemma, which is illustrated in Fig. 2(d).

Lemma 8: Target the scenario of S1–S4 with given $t^{\text{end}} > 0$ and J_k^*, and target a single mode change instant t^* belonging to $[t_a^*(J_k^*), t_b^*(J_k^*)]$ (defined in Lemma 4). The mode change instant t^* is infeasible without any job deadline miss, if it is impossible to satisfy both Eqs. (5) and (6) subject to Lemma 7 and the following constraint.

- There exists exactly one $\tau_j \in \tau^{\texttt{HI}}$ belonging to Subcase 3A of Lemma 7.

$$\sum_{\tau_i \in \tau^{\texttt{LO}}} \texttt{DBF}_i^{\texttt{LO}}(t^*) + \sum_{\tau_i \in \tau^{\texttt{HI}}} \Big(\texttt{DBF}_i^{\texttt{LO}}(t^*) + \texttt{OP}_i^-(t^*) \Big) \leq t^*. \quad (5)$$

$$\sum_{\tau_i \in \tau^{\texttt{HI}}} \Big(\texttt{DBF}_i^{\texttt{HI}}\big(t^{\text{end}} - \Big\lceil \frac{t^*}{T_i} \Big\rceil \cdot T_i\big) + \texttt{OP}_i^+(t^*) \Big) \leq t^{\text{end}} - t^*. \quad (6)$$

Proof: By Lemma 7 and the fact that the mode change cannot be triggered by more than one job, the two constraints (i.e., the "subject to" part) hold.

By Lemmas 5 and 6, the LHS of Eq. (5) and that of Eq. (6) calculate the total demand of τ in $[0, t^*]$ and $[t^*, t^{\text{end}}]$, respectively under the scenario of S1–S4 with given $t^{\text{end}} > 0$ and J_k^*. Considering the supply amounts to the interval length, violating either Eq. (5) or (6) implies that the mode change cannot occur at t^* or at least one job misses its deadline. ■

Once we repeat Lemma 8 with every $t^* \in [t_a^*(J_k^*), t_b^*(J_k^*)]$, we know whether there exists at least one mode change instant that does not yield any job deadline miss. If it does not exist, it is impossible for the mode change instance to exist without any job deadline miss, which yields infeasibility of the scenario by Lemma 4. This addresses I3, and is recorded by the following theorem (as also illustrated in Fig. 2(e)).

Theorem 1: A MC task set τ is infeasible, if every mode change instant $t^* \in [t_a^*(J_k^*), t_b^*(J_k^*)]$ associated with the scenario of S1–S4 with given $t^{\text{end}} > 0$ and J_k^* makes it impossible to satisfy both Eqs. (5) and (6) subject to Lemma 7 and the constraint in Lemma 8.

Proof: By Lemma 8 and the range of $t^* \in [t_a^*(J_k^*), t_b^*(J_k^*)]$, the impossibility to satisfy both Eqs. (5) and (6) subject to Lemma 7 and the constraint in Lemma 8 implies that the scenario of S1–S4 with given $t^{\text{end}} > 0$ and J_k^* yields no existence of the mode change instant without any job deadline miss. According to Lemma 4, existence of the mode change instant in $[t_a^*(J_k^*), t_b^*(J_k^*)]$ is a necessary feasibility condition for the scenario. Therefore, the theorem holds. ■

Algorithm 1 Efficient test for Lemma 8

```
1: for $\tau_k \in \tau^{\mathtt{HI}}$ do
2:   if $\tau_k$ satisfies (i) and (ii) of Subcase 3A in Lemma 7, and every $\tau_i \in \tau^{\mathtt{HI}} - \{\tau_k\}$ belonging to Case 3 can satisfy (i)–(iii) of Subcase 3B then
3:     $\mathtt{SumOP}^- \leftarrow C_k^{\mathtt{LO}} + \sum_{\tau_i \in \tau^{\mathtt{HI}} \setminus \{\tau_k\}} \max \mathtt{OP}_i^-(t^*)$
4:     $\mathtt{SumOP}^+ \leftarrow C_k^{\mathtt{HI}} - C_k^{\mathtt{LO}} + \sum_{\tau_i \in \tau^{\mathtt{HI}} \setminus \{\tau_k\}} \max \mathtt{OP}_i^+(t^*)$
5:     $\mathtt{DiffOP} \leftarrow \sum_{\tau_i \in \tau^{\mathtt{HI}} \setminus \{\tau_k\}} \max \mathtt{OP}_i^-(t^*) - \sum_{\tau_i \in \tau^{\mathtt{HI}} \setminus \{\tau_k\}} \min \mathtt{OP}_i^-(t^*)$
6:     $\mathtt{DiffLO} \leftarrow \max\left(0, \sum_{\tau_i \in \tau} \mathtt{DBF}_i^{\mathtt{LO}}(t^*) + \mathtt{SumOP}^- - t^*\right)$
7:     $\mathtt{DiffHI} \leftarrow \max\left(0, \sum_{\tau_i \in \tau^{\mathtt{HI}}} \mathtt{DBF}_i^{\mathtt{HI}}(t^{\mathrm{end}} - \lceil t^*/T_i \rceil \cdot T_i) + \mathtt{SumOP}^+ - (t^{\mathrm{end}} - t^*)\right)$
8:     if $\mathtt{DiffLO} + \mathtt{DiffHI} \leq \mathtt{DiffOP}$ then
9:       Return TRUE
10:    end if
11:  end if
12: end for
13: Return FALSE
```

V. Checking the Necessary Feasibility Test Time-Efficiently

In this section, we address I6 in Section III-C—how to check the proposed test with as many scenarios as possible with less time-complexity. To this end, we develop (i) how to test Lemma 8 (and therefore Theorem 1) with low time-complexity and (ii) how to apply the necessary feasibility test in Theorem 1 to the scenario of S1–S4 with every pair of $t^{\mathrm{end}} > 0$ and J_k^* with reasonable time-complexity.

While we successfully developed a necessary feasibility test in Theorem 1 using Lemma 8, we did not discuss how to check the "if" statement of the lemma. A simple way is to check all possible combinations of $\mathtt{OP}_i^-(t^*)$ and $\mathtt{OP}_i^+(t^*)$ for every $\tau_i \in \tau^{\mathtt{HI}}$. Since there are multiple options of a $\mathtt{HI}$ job's contribution to the demand in $[0, t^*]$ and that in $[t^*, t^{\mathrm{end}}]$, the number of combinations of all $\mathtt{HI}$ jobs' contribution to those demands can be an exponential function of the number of $\mathtt{HI}$ jobs. This entails to develop an algorithm that tests Lemma 8 in a time-efficient manner, without investigating all possible combinations. We develop Alg. 1, to be presented now.

In Alg. 1, we repeat the following steps in Lines 2–11 for every $\tau_k \in \tau^{\mathtt{HI}}$ (Line 1). We first check whether it is possible for the job of τ_k to trigger the mode change by checking (a) τ_k satisfies the conditions (i) and (ii) of Subcase 3A in Lemma 7, and (b) every $\tau_i \in \tau^{\mathtt{HI}} - \{\tau_k\}$ belonging to Case 3 can satisfy the conditions (i)–(iii) of Subcase 3B (by having at least one possible value for both $\mathtt{OP}_i^-(t^*)$ and $\mathtt{OP}_i^+(t^*)$) (Line 2). If so (meaning that the constraint in Lemma 8 holds), we calculate the sum of the upper bound of $\mathtt{OP}_i^-(t^*)$ and $\mathtt{OP}_i^+(t^*)$ for every $\tau_i \in \tau^{\mathtt{HI}} \setminus \{\tau_k\}$, called $\mathtt{SumOP}^-$ and $\mathtt{SumOP}^+$, respectively (Lines 3 and 4). We also calculate the difference between the sum of the upper bound of $\mathtt{OP}_i^-(t^*)$ and that of the lower bound of $\mathtt{OP}_i^-(t^*)$, called $\mathtt{DiffOP}$ (Lines 5). Note that $\mathtt{DiffOP}$ is the same as the sum of the followings for every $\tau_i \in \tau^{\mathtt{HI}} \setminus \{\tau_k\}$: the sum of the difference between $\max \mathtt{OP}_i^-(t^*)$ and the actual assignment of $\mathtt{OP}_i^-(t^*)$, and the difference between $\max \mathtt{OP}_i^+(t^*)$ and the actual assignment of $\mathtt{OP}_i^+(t^*)$. See the proof of Lemma 9 about why $\mathtt{DiffOP}$ is the same as the above value. Considering that we select a job of τ_k to trigger the mode change and jobs of other tasks not to trigger the mode change as mentioned Lines 1 and 2, if $\tau_i \in \tau^{\mathtt{HI}} \setminus \{\tau_k\}$ belongs to Case 3 of Lemma 7, we apply Subcase 3B to $\max \mathtt{OP}_i^-(t^*)$, $\min \mathtt{OP}_i^-(t^*)$ and $\max \mathtt{OP}_i^+(t^*)$ in Lines 3–5 (i.e., $\mathtt{UB2}^-$, $\mathtt{LB2}^-$, and $\mathtt{UB2}^+$, respectively).

Then, we calculate $\mathtt{DiffLO}$, the LHS minus the RHS of Eq. (5) assuming the maximum (upper bound) of $\mathtt{OP}_i^-(t^*)$ for every $\tau_i \in \tau^{\mathtt{HI}} \setminus \{\tau_k\}$ (Line 6); in other words, $\mathtt{DiffLO}$ means the minimum amount to be reduced to satisfy Eq. (5) when $\mathtt{OP}_i^-(t^*)$ for every $\tau_i \in \tau^{\mathtt{HI}} \setminus \{\tau_k\}$ has the maximum. Similarly, we calculate $\mathtt{DiffHI}$, the LHS minus RHS of Eq. (6) assuming the maximum (upper bound) of $\mathtt{OP}_i^+(t^*)$ for every $\tau_i \in \tau^{\mathtt{HI}} \setminus \{\tau_k\}$ (Line 7); in other words, $\mathtt{DiffHI}$ means the minimum amount to be reduced to satisfy Eq. (6) when $\mathtt{OP}_i^+(t^*)$ for every $\tau_i \in \tau^{\mathtt{HI}} \setminus \{\tau_k\}$ has the maximum. Then, to satisfy Eqs. (5) and (6), we need to reduce the sum of $\mathtt{OP}_i^-(t^*)$ by as much as $\mathtt{DiffLO}$ and the sum of $\mathtt{OP}_i^+(t^*)$ by as much as $\mathtt{DiffHI}$, while the total budget we can reduce the former or latter sum is as much as $\mathtt{DiffOP}$. Therefore, if $\mathtt{DiffLO} + \mathtt{DiffHI} \leq \mathtt{DiffOP}$ holds, we return $\mathtt{TRUE}$ (Lines 8 and 9), meaning that it may be possible for the mode change to occur at t^* without missing any job deadline. Finally, we return $\mathtt{FALSE}$ in Line 13, if every $\tau_k \in \tau^{\mathtt{HI}}$ cannot yield $\mathtt{TRUE}$, meaning that it is impossible for the mode change to occur at t^* without missing any job deadline.

Alg. 1 can replace Lemma 8, as in the following lemma.

Lemma 9: If Alg. 1 returns $\mathtt{FALSE}$, Theorem 1 (and Lemma 8) judges that for given t^* it is impossible to satisfy both Eqs. (5) and (6) subject to Lemma 7 and the constraint in Lemma 8.

Proof: Suppose that Alg. 1 returns $\mathtt{FALSE}$ but Lemma 8 cannot judge that a mode change cannot occur at given $t^* \in [t_a^*(J_k^*), t_b^*(J_k^*)]$ without any deadline miss of jobs invoked by τ in $[0, t^{\mathrm{end}}]$. We now derive contradiction.

The supposition implies that there is no task $\tau_k \in \tau^{\mathtt{HI}}$ that satisfies the inequality of $\mathtt{DiffLO} + \mathtt{DiffHI} \leq \mathtt{DiffOP}$ in Alg. 1. The supposition also implies that there exists at least $t^* \in [t_a^*(J_k^*), t_b^*(J_k^*)]$ that satisfies Eqs. (5) and (6) subject to Lemma 7 and the constraint in Lemma 8; let t' denote such t^*, and τ_k denote the task that satisfies the constraint (i.e., the task whose job triggers the mode change). Then, $\mathtt{OP}_k^-(t') = C_k^{\mathtt{LO}}$ and $\mathtt{OP}_k^+(t') = C_k^{\mathtt{HI}} - C_k^{\mathtt{LO}}$ holds by Subcase 3A of Lemma 7. Let δ_i^- and δ_i^+ for $\tau_i \in \tau^{\mathtt{HI}} \setminus \{\tau_k\}$ denote the difference between $\max \mathtt{OP}_i^-(t')$ and the actual $\mathtt{OP}_i^-(t')$, and the difference between $\max \mathtt{OP}_i^+(t')$ and the actual $\mathtt{OP}_i^+(t')$, respectively. Then, for $\tau_i \in \tau^{\mathtt{HI}} \setminus \{\tau_k\}$ belonging to Subcase 3B of Lemma 7, the following holds: $\delta_i^- + \delta_i^+ = \mathtt{UB2}^- - \mathtt{OP}_i^-(t') + \mathtt{UB2}^+ - \mathtt{OP}_i^+(t') = \mathtt{UB2}^- + \mathtt{UB2}^+ - C_i^{\mathtt{HI}} = \mathtt{UB2}^- + C_i^{\mathtt{HI}} - \mathtt{LB2}^- - C_i^{\mathtt{HI}}$, which is $\left(\max \mathtt{OP}_i^-(t') - \min \mathtt{OP}_i^-(t')\right)$. The same holds for $\tau_i \in \tau^{\mathtt{HI}} \setminus \{\tau_k\}$ belonging to Case 2 of Lemma 7.

Then, the following inequality holds from Eq. (5):

$$\sum_{\tau_i \in \tau} \mathtt{DBF}_i^{\mathtt{LO}}(t') + \mathtt{SumOP}^- - \sum_{\tau_i \in \tau^{\mathtt{HI}} \setminus \{\tau_k\}} \delta_i^- \leq t'$$
$$\Rightarrow \sum_{\tau_i \in \tau} \mathtt{DBF}_i^{\mathtt{LO}}(t') + \mathtt{SumOP}^- - t' \leq \sum_{\tau_i \in \tau^{\mathtt{HI}} \setminus \{\tau_k\}} \delta_i^-.$$

Similarly, we can derive the following inequality holds from Eq. (6):

$$\sum_{\tau_i \in \tau^{\mathtt{HI}}} \mathtt{DBF}_i^{\mathtt{HI}}(t^{\mathrm{end}} - \lceil t'/T_i \rceil \cdot T_i) + \mathtt{SumOP}^+ - (t^{\mathrm{end}} - t') \leq \sum_{\tau_i \in \tau^{\mathtt{HI}} \setminus \{\tau_k\}} \delta_i^+.$$

Note that the LHSes of the above two inequalites are the same as $\mathtt{DiffLO}$ and $\mathtt{DiffHI}$ without the max operation. Considering δ_i^- and δ_i^+ are non-negative for every $\tau_i \in$

$\tau^{\texttt{HI}} \setminus \{\tau_k\}$ by their definitions, if we combine the above two inequalities, the LHS is $\texttt{DiffLO} + \texttt{DiffHI}$, and the RHS is $\sum_{\tau_i \in \tau^{\texttt{HI}} \setminus \{\tau_k\}} (\delta_i^- + \delta_i^+) = \sum_{\tau_i \in \tau^{\texttt{HI}} \setminus \{\tau_k\}} \big(\max \texttt{OP}_i^-(t') - \min \texttt{OP}_i^-(t') \big)$, which is equal to $\texttt{DiffOP}$. Therefore, the supposition of non-existence of $\tau_k \in \tau^{\texttt{HI}}$ that satisfies $\texttt{DiffLO} + \texttt{DiffHI} \leq \texttt{DiffOP}$ contradicts. ■

The time-complexity of Alg. 1 is $O(n^2)$, and there are at most $\max_{\tau_i \in \tau^{\texttt{HI}}} D_i$ choices of t^* for any given J_k^* in Theorem 1. Therefore, if we test Theorem 1 using Alg. 1, the time-complexity of Theorem 1 is $O(n^2 \cdot \max_{\tau_i \in \tau^{\texttt{HI}}} D_i)$.

While Theorem 1 focuses on the scenario of S1–S4 with given $t^{\text{end}} > 0$ and J_k^*, we can find more infeasible task sets if we apply every pair of $t^{\text{end}} > 0$ and J_k^*. The following lemma calculates an upper-bound of $t^{\text{end}} > 0$ as well as t^* associated with J_k^*.

Lemma 10: If Eq. (5) is violated with any t^*, it is also violated with some t^* which is smaller than $\big(\sum_{\tau_i \in \tau}(T_i - D_i) \cdot C_i^{\text{LO}}/T_i + \sum_{\tau_i \in \tau^{\texttt{HI}}} C_i^{\text{LO}}\big)/\big(1 - \sum_{\tau_i \in \tau} C_i^{\text{LO}}/T_i\big)$. Also, if Eq. (6) is violated with any $(t^{\text{end}} - t^*)$, it is also violated with some $(t^{\text{end}} - t^*)$ which is smaller than $\big(\sum_{\tau_i \in \tau^{\texttt{HI}}}(T_i - D_i) \cdot C_i^{\text{HI}}/T_i + \sum_{\tau_i \in \tau^{\texttt{HI}}} C_i^{\text{HI}}\big)/\big(1 - \sum_{\tau_i \in \tau^{\texttt{HI}}} C_i^{\text{HI}}/T_i\big)$.

Proof: Simply applying the inequality in [12], [13] that upper-bounds the demand bound function for SC task systems, we can derive the following inequality from Eq. (3): $\sum_{\tau_i \in \tau} \texttt{DBF}_i^{\text{LO}}(t) \leq t \cdot \sum_{\tau_i \in \tau} C_i^{\text{LO}}/T_i + \sum_{\tau_i \in \tau}(T_i - D_i) \cdot C_i^{\text{LO}}/T_i$. Also, we use $\texttt{OP}_i^-(t^*) \leq C_i^{\text{LO}}$ and $\texttt{OP}_i^+(t^*) \leq C_i^{\text{HI}}$.

Using the above inequalities, if Eq. (5) is violated, the following holds:

$$\begin{aligned}
t^* &< \sum_{\tau_i \in \tau} \texttt{DBF}_i^{\text{LO}}(t^*) + \sum_{\tau_i \in \tau^{\texttt{HI}}} \texttt{OP}_i^-(t^*) \\
&\leq t^* \cdot \sum_{\tau_i \in \tau} C_i^{\text{LO}}/T_i + \sum_{\tau_i \in \tau}(T_i - D_i) \cdot C_i^{\text{LO}}/T_i + \sum_{\tau_i \in \tau^{\texttt{HI}}} C_i^{\text{LO}} \\
\Rightarrow t^* \cdot \big(1 - \sum_{\tau_i \in \tau} C_i^{\text{LO}}/T_i\big) &< \sum_{\tau_i \in \tau}(T_i - D_i) \cdot C_i^{\text{LO}}/T_i + \sum_{\tau_i \in \tau^{\texttt{HI}}} C_i^{\text{LO}} \\
\Rightarrow t^* &< \frac{\sum_{\tau_i \in \tau}(T_i - D_i) \cdot C_i^{\text{LO}}/T_i + \sum_{\tau_i \in \tau^{\texttt{HI}}} C_i^{\text{LO}}}{1 - \sum_{\tau_i \in \tau} C_i^{\text{LO}}/T_i}.
\end{aligned}$$

Using the same technique, we can derive an upper bound for $(t^{\text{end}} - t^*)$. ■

Applying Lemma 10, we can test the scenario of S1–S4 with all possible pairs of $t^{\text{end}} > 0$ and J_k^* with reasonable time-complexity, yielding the following collective necessary feasibility test.

Theorem 2: A MC task set τ is infeasible, if the following collective necessary feasibility test finds at least one pair of $t^{\text{end}} > 0$ and J_k^* that makes it impossible to satisfy both Eqs. (5) and (6) subject to Lemma 7 and the constraint in Lemma 8.

- Repeat Theorem 1 using Alg. 1 for every pair of $t^{\text{end}} > 0$ and J_k^* that satisfies (i) t^{end} is less than the sum of upper bounds of t^* and $(t^{\text{end}} - t^*)$ in Lemma 10 and (ii) J_k^*'s release time is earlier than the upper-bound of t^* in the lemma.

Proof: By Alg. 1 and Theorem 1, the theorem holds. ■

One may wonder how to "Repeat Theorem 1 using Alg. 1 for every pair of $t^{\text{end}} > 0$ and J_k^*" in Theorem 2. First, by adding two upper-bounds for t^* and $(t^{\text{end}} - t^*)$ in Lemma 10, we have an upper-bound for t^{end}. Second, we target every job J_k^* whose execution window overlaps with $[0, t^{\text{end}})$. For each job J_k^*, we calculate $[t_a^*(J_k^*), t_b^*(J_k^*)]$ using Lemma 4, and then check all $t^* \in [t_a^*(J_k^*), t_b^*(J_k^*)]$ using Alg. 1.

For a given $t^{\text{end}} > 0$ and J_k^*, Theorem 1 conducts Alg. 1 at most $\max_{\tau_i \in \tau^{\texttt{HI}}} D_i$ times, resulting in $O(n^2 \cdot \max_{\tau_i \in \tau^{\texttt{HI}}} D_i)$ as explained. As proved in Lemma 10, the upper-bounds on t^{end} and $(t^{\text{end}} - t^*)$ are a pseudo-polynomial in the size of task parameters (if the following condition holds: $\sum_{\tau_i \in \tau} C_i^{\text{LO}}/T_i$ and $\sum_{\tau_i \in \tau^{\texttt{HI}}} C_i^{\text{HI}}/T_i$ are upper-bounded by some constant strictly less than 1.0). Therefore, the overall time complexity of the above collective necessary feasibility test in Theorem 2 is also pseudo-polynomial in the size of task parameters (if the same condition holds).

We would like to emphasize that the collective necessary feasibility test in Theorem 2 can check Theorem 1 for any number of pairs of $t^{\text{end}} > 0$ and J_k^*, which can affect capability in finding infeasible task sets, but cannot compromise the correctness of whether task sets deemed infeasible by the test is actually infeasible. Therefore, if time-complexity matters, we can limit the number of pairs to be checked, at the expense of sacrificing capability in finding infeasible task sets.

VI. Simplified Necessary Feasibility Test

Although we successfully developed the necessary feasibility test in conjunction with how to check the test with all possible pairs of $t^{\text{end}} > 0$ and J_k^* with reasonable time-complexity, one may demand another necessary feasibility test such that (i) the time-complexity is comparable to Lemmas 2 and 3, and (ii) the capability in finding infeasible task sets is easily compared with the lemmas. We now derive such a test from the proposed necessary feasibility test in Lemma 8, which addresses I6 in Section III-C in a different perspective.

Once we combine Eqs. (5) and (6) under the scenario of S1–S4 with given $t^{\text{end}} > 0$ and J_k^*, the following combined inequality holds:

$$\sum_{\tau_i \in \tau^{\text{LO}}} \texttt{DBF}_i^{\text{LO}}(t^*) + \sum_{\tau_i \in \tau^{\texttt{HI}}} (\text{execution requirement of all jobs of } \tau_i) \leq t^{\text{end}}.$$

Since the $\texttt{DBF}_i^{\text{LO}}(t^*)$ term in the combined inequality is the only term depending on t^*, the LHS of the combined inequality non-decreases as t^* increases. Therefore, if the combined inequality is violated with $t^* = t_a^*(J_k^*)$, it is also violated with every t^* in $[t_a^*(J_k^*), t_b^*(J_k^*)]$. This implies that the scenario is infeasible if the combined inequality is violated with $t^* = t_a^*(J_k^*)$.

In addition, we restrict to target J_k^* as the first job of any $\texttt{HI}$ task, meaning that the execution requirement of every $\texttt{HI}$ job amounts to its $\texttt{HI}$ WCET, yielding the following combined inequality from Eqs. (5) and (6):

$$\sum_{\tau_i \in \tau^{\text{LO}}} \texttt{DBF}_i^{\text{LO}}(t_a^*) + \sum_{\tau_i \in \tau^{\texttt{HI}}} \texttt{DBF}_i^{\text{HI}}(t^{\text{end}}) \leq t^{\text{end}}, \qquad (7)$$

where $t_a^* = \min_{\tau_i \in \tau^{\texttt{HI}}} C_i^{\text{LO}}$. Then, the following theorem holds.

Theorem 3: A MC task set τ is infeasible, if Eq. (7) is violated for given $t^{\text{end}} \geq t_a^* = \min_{\tau_i \in \tau^{\texttt{HI}}} C_i^{\text{LO}}$.

Proof: In the target scenario, it is impossible for any HI job to trigger the mode change (by executing for its LO WCET) before t^*_a. Therefore, LO jobs should perform their execution for as much as at least the first term of Eq. (7) in $[0, t^{\text{end}}]$ with $t^{\text{end}} \geq t^*_a$; otherwise, there exists a LO job's deadline miss at no later than t^*_a. Also, HI jobs should perform their execution for as much as the second term of Eq. (7) in $[0, t^{\text{end}}]$ with $t^{\text{end}} > 0$; otherwise, there exists a HI job deadline miss. Therefore, the total execution requirement that should be performed in $[0, t^{\text{end}}]$ is at least as much as the LHS of Eq. (7) for $t^{\text{end}} \geq t^*_a$. ■

It is trivial that the time-complexity of checking Theorem 3 with all possible $t^{\text{end}} \geq t^*_a$ is comparable to that of checking Lemma 3 (or Lemma 2) with all possible $t^{\text{end}} > 0$. Also, it is easily observed that Theorem 3 exhibits higher capability in finding infeasible task sets, than Lemma 3 (because the LHS of Eq. (7) is no smaller than that of Eq. (4)).

Theorem 3 does not consider the constraints for the amount of the contribution of HI jobs (whose execution window includes the mode change instant) to the demand before and after the mode change instant; in addition Theorem 3 tests only one job for J^*_k. Therefore, the necessary feasibility test in Theorem 3 has lower capability than that in Theorem 2 in terms of finding infeasible task sets, to be evaluated in the next section.

VII. Evaluation

We demonstrate the capability of the proposed feasibility tests in covering a broader range of infeasible MC task sets.

Generation of task sets. We generate a synthetic task set similarly in [14], [15], [16], which can be summarized as follows. We have five input parameters: (i) the number of tasks $n \in \{4, 6, 8, 10\}$, (ii) the probability (CP) of each task τ_i having $\chi_i = \text{HI} \in \{0.3, 0.5, 0.7\}$, (iii) the maximum ratio (CF) of each HI task τ_i's HI WCET to LO WCET, i.e., $\mathsf{CF} \overset{\text{def}}{=} C^{\text{HI}}_i / C^{\text{LO}}_i \in \{2, 3, 4\}$, (iv) LO total utilization $U^{\text{LO}} \overset{\text{def}}{=} \sum_{\tau_i \in \tau} C^{\text{LO}}_i / T_i$, and (v) HI total utilization $U^{\text{HI}} \overset{\text{def}}{=} \sum_{\tau_i \in \tau^{\text{HI}}} C^{\text{HI}}_i / T_i$. We choose U^{LO} and U^{HI} from 0.45 to 1.00 with an incremental step of 0.05 (resulting in 12 choices), respectively.

Given a 5-tuple $(n, \mathsf{CP}, \mathsf{CF}, U^{\text{LO}}, U^{\text{HI}})$ for a task set, each task parameter is determined as follows: T_i is uniformly chosen in $[1, 1000]$; χ_i is selected as HI with probability CP (and as LO with probability $(1.0 - \mathsf{CP})$); C^{LO}_i is determined by using the UUniFast algorithm [17]; C^{HI}_i is uniformly chosen in $[C^{\text{LO}}_i + 1, \mathsf{CF} \cdot C^{\text{LO}}_i + 1]$, if $\chi_i = \text{HI}$ (and set to C^{LO}_i, otherwise); and D_i is set to T_i for implicit-deadline task sets. Using the above task parameters, we first generate 1,000 implicit-deadline task sets whose LO and HI total utilizations are in $[U^{\text{LO}} - 0.05, U^{\text{LO}}]$ and $[U^{\text{HI}} - 0.05, U^{\text{HI}}]$ for given U^{LO} and U^{HI}, respectively, resulting in a total of 144,000 task sets for given n, CP, and CF.

With the generated task sets, we compare the following two proposed necessary feasibility tests:

- MC-NFT: the collective necessary feasibility test in Theorem 2, and
- MC-NFT-S: the simplified version of MC-NFT in Theorem 3.

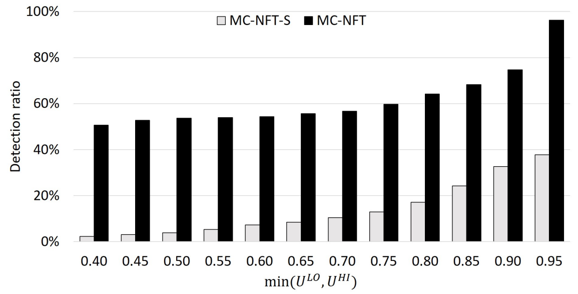

Fig. 3. Detection ratio of MC-NFT and MC-NFT-S for constrained-deadline task sets with different ranges of $\min(U^{\text{LO}}, U^{\text{HI}})$ when $n = 4$, $\mathsf{CP} = 0.5$ and $\mathsf{CF} = 2$

Simulation results. Fig. 1 in Section I shows the implicit-deadline task sets (marked as red cross)—which were not proven infeasible by any existing studies but which were newly proven infeasible by MC-NFT and MC-NFT-S—in (U^{LO}, U^{HI})-plane when $n = 4, \mathsf{CF} = 2, \mathsf{CP} = 0.5$. In ($U^{\text{LO}}$, U^{HI})-plane, we focus on the region where $0.75 < \max(U^{\text{LO}}, U^{\text{HI}}) \leq 1.0$, yielding 95,000 task sets of interest among 144,000 generated task sets. This is because all task sets in the other regions where $\max(U^{\text{LO}}, U^{\text{HI}}) \leq 0.75$ and $\max(U^{\text{LO}}, U^{\text{HI}}) > 1.00$ are already proved feasible [18] and infeasible [7], [8], [9], respectively. It is observed that MC-NFT and MC-NFT-S can newly find a number of infeasible task sets over a wider range of LO and HI total utilization. It is also observed that MC-NFT and MC-NFT-S identify significantly more infeasible task sets as LO and HI total utilization become close to 1.0; for example, among 1,000 task sets with $0.95 \leq U^{\text{LO}} \leq 1.0$ and $0.95 \leq U^{\text{HI}} \leq 1.0$, MC-NFT finds 325 (32.5%) infeasible task sets. In total, MC-NFT and MC-NFT-S find 1,633 and 671 infeasible task sets, respectively, among 95,000 generated task sets (note that all 95,000 task sets have not been proven infeasible by any existing studies, while some of them may be feasible).

We also generate constrained-deadline task sets to have the same task parameter values as implicit-deadline ones except setting D_i uniformly chosen in $[C^{\text{HI}}_i, T_i]$. Note that, among 144,000 constrained-deadline task sets for given n, CP, and CF, we exclude all task sets that can be proven infeasible by trivial necessary feasibility tests shown in Lemmas 2 and 3 and consider the remaining task sets as task sets of interest. In order to show how many more infeasible task sets can be found by our proposed tests, we use *detection ratio* as performance metric, defined as the percentage of task sets that are deemed infeasible by each individual necessary feasibility test to the total number of task sets of interest.

Fig. 3 plots the detection ratio by MC-NFT and MC-NFT-S while varying $\min(U^{\text{LO}}, U^{\text{HI}})$ from $[0.40, 0.45)$ to $[0.95, 1.0]$. We have the following observations. First, MC-NFT and MC-NFT-S exhibit high capability in finding infeasible task sets in that MC-NFT and MC-NFT-S find 20,679 (56.0%) and 3,041 (8.2%) total infeasible task sets, respectively, among 36,935 task sets of interests. Such high capability can be interpreted as the benefit of dealing with unique issues pertaining to MC task systems. In particular, higher capability for constrained-deadline task sets (than that for implicit-deadline task sets) mainly comes from accurate calculation on the execution contribution of HI to the sub-intervals and precise constraints thereof, in that we generate a constrained-deadline task set

TABLE I. Detection ratio and average running time of MC-NFT for constrained-deadline task sets with different numbers of tasks (n) when $\mathsf{CP} = 0.5$ and $\mathsf{CF} = 2$

n	4	6	8	10
Detection ratio (%)	56.0	77.5	90.3	95.8
Avg. running time (ms)	0.11	0.18	0.20	0.24

by reducing the relative deadline of tasks in the corresponding implicit-deadline task set. Second, both MC-NFT and MC-NFT-S find more infeasible task sets as $\min(U^{\mathtt{LO}}, U^{\mathtt{HI}})$ increases; for example, using MC-NFT, 50.6% and 96.2% of the task sets are proven infeasible with $\min(U^{\mathtt{LO}}, U^{\mathtt{HI}})$ in $[0.4, 0.45)$ and $[0.95, 1.0]$, respectively. This is due to the difficulty in meeting all job deadlines of a task set with high $\min(U^{\mathtt{LO}}, U^{\mathtt{HI}})$. Third, MC-NFT is shown to outperform MC-NFT-S for all values of $\min(U^{\mathtt{LO}}, U^{\mathtt{HI}})$. This is because MC-NFT (i) derives a tighter bound on the demand of HI jobs by considering the relationship between the mode change instant and each HI job's execution window, and (ii) tests many choices of J_k^* (while MC-NFT-S tests one choice of J_k^*). Nevertheless, MC-NFT-S has the same time complexity as in the SC task system case, while finding some infeasible task sets.

Note that although, among generated task sets, there might also exist feasible task sets that can be revealed using existing schedulability analyses under some scheduling algorithms, such as PLRS [3], EDF-VD [4], GREEDY [5], and ECDF [6], we did not identify them in our evaluation. This is because the number of task sets proven infeasible by this paper is independent of those sufficient feasibility tests; instead, if we exclude those feasible task sets from task sets of interest, detection ratio in Fig. 3 will increase, implying that Fig. 3 as of now exhibits the minimum capability of the proposed necessary feasibility tests in finding infeasible task sets.

In this section, we only show the results of $n = 4$, $\mathsf{CP} = 0.5$ and $\mathsf{CF} = 2$, due to space limit, but similar trends have observed for other input combinations. We now briefly show the scalability of of MC-NFT with respect to the number of tasks. Table I shows the detection ratio by MC-NFT and its average running time to identify an infeasible constrained-deadline task set with different numbers of tasks when $\mathsf{CP} = 0.5$ and $\mathsf{CF} = 2$. MC-NFT exhibits higher capability in finding infeasible task sets as the number of tasks increases with a comparable running time. For example, as n increases from 4 to 10, the detection ratio is increased from 56.0% to 95.8%, while the average running time is increased from 0.11 ms to 0.24 ms[3]. This implies that MC-NFT is able to scale to larger task sets.

VIII. Conclusion and Discussion

In this paper, we investigated characteristics of MC necessary feasibility conditions and identified challenges for deriving those conditions. By resolving the challenges, we completed to develop a necessary feasibility test and its simplified version, establishing foundations for necessary feasibility tests for MC task systems, which is the first study that yields non-trivial results for MC necessary feasibility. We demonstrated that the proposed necessary feasibility tests find a number of infeasible task sets which have been proven neither feasible nor infeasible by any existing studies.

[3]The average running time was measured on a machine that has Intel i7-8700K CPU.

While we successfully developed the necessary feasibility tests, we have an interesting remaining issue as follows. As to SC task systems, it has been proven that the collective necessary feasibility test in Section III-A exhibits the highest capability in finding infeasible task sets, by the fact that the scenario of S1, S2 and S3' maximizes the demand. Then, we need to figure out the following questions for MC task systems: (i) does the scenario of S1–S4 maximizes capability in finding infeasible task sets? and (ii) if not, what is the best or at least a better scenario? Although we did not present in this paper, we already observed a counter example of showing that S2 does not maximize the demand, which proves the answer of (i) is no. Therefore, we would like to address (ii) in the future.

Acknowledgement

This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (2019R1A2B5B02001794, 2017H1D8A2031628, 2017M3A9G8084463), as well as the DGIST Start-up Fund Program of the Ministry of Science, ICT and Future Planning (2019010064).

References

[1] C. Liu and J. Layland, "Scheduling algorithms for multi-programming in a hard-real-time environment," *Journal of the ACM*, vol. 20, no. 1, pp. 46–61, 1973.

[2] S. Vestal, "Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance," in *RTSS*, 2007.

[3] N. Guan, P. Ekberg, M. Stigge, and W. Yi, "Effective and efficient scheduling of certifiable mixed-criticality sporadic task systems," in *RTSS*, 2011.

[4] S. Baruah, V. Bonifaci, G. D'Angelo, A. MarchettiSpaccamela, S. van der Ster, and L. Stougie, "Mixed-criticality scheduling of sporadic task systems," in *In Proceedings of the 19th Annual European Symposium on Algorithms*, 2011.

[5] P. Ekberg and W. Yi, "Bounding and shaping the demand of mixed-criticality sporadic tasks," in *ECRTS*, 2012.

[6] A. Easwaran, "Demand-based scheduling of mixed-criticality sporadic tasks on one processor," in *RTSS*, 2013.

[7] S. Baruah, V. Bonifaci, G. D'Angelo, H. Li, A. Marchetti-Spaccamela, S. V. D. Ster, and L. Stougie, "Preemptive uniprocessor scheduling of mixed-criticality sporadic task systems," *Journal of the ACM*, vol. 62, no. 2, pp. 14:1–14:33, April 2015.

[8] S. Baruah, A. Easwaran, and Z. Guo, "MC-Fluid: simplified and optimally quantified," in *RTSS*, 2015.

[9] S. Ramanathan, X. Gu, and A. Easwaran, "The feasibility analysis of mixed-criticality systems," in *Real-Time Scheduling Open Problems Seminar*, 2016.

[10] K. Agrawal and S. Baruah, "Intractability issues in mixed-criticality scheduling," in *ECRTS*, 2018.

[11] A. Burns and R. I. Davis, "Mixed criticality systems - a review," the twelfth edition, 2019.

[12] S. Baruah, A. Mok, and L. Rosier, "Preemptively scheduling hard-real-time sporadic tasks on one processor," in *RTSS*, 1990.

[13] S. Baruah, "Techniques for multiprocessor global schedulability analysis," in *RTSS*, 2007.

[14] R. Pathan, "Schedulability analysis of mixed-criticality systems on multiprocessors," in *ECRTS*, 2012.

[15] H. Baek, N. Jung, H. S. Chwa, I. Shin, and J. Lee, "Non-preemptive scheduling for mixed-criticality real-time multiprocessor systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 8, pp. 1766–1779, August 2018.

[16] H. S. Chwa, K. G. Shin, H. Baek, and J. Lee, "Physical-state-aware dynamic slack management for mixed-criticality systems," in *RTAS*, 2018.

[17] E. Bini and G. Buttazzo, "Measuring the performance of schedulability tests," *Real-Time Systems*, vol. 30, no. 1-2, pp. 129–154, May 2005.

[18] S. Baruah, V. Bonifaci, G. D'Angelo, H. L. A. Marchetti-Spaccamela, S. van der Ster, and L. Stougie, "The preemptive uniprocessor scheduling of mixed-criticality implicit-deadline sporadic task systems," in *ECRTS*, 2012.