Noname manuscript No.
(will be inserted by the editor)

Multiprocessor Extensions to Real-Time Calculus

**Hennadiy Leontyev · Samarjit Chakraborty ·
James H. Anderson**

the date of receipt and acceptance should be inserted later

Abstract Many embedded platforms consist of a heterogeneous collection of processing elements, memory modules, and communication subsystems. These components often implement different scheduling/arbitration policies, have different interfaces, and are supplied by different vendors. Hence, compositional techniques for modeling and analyzing such platforms are of interest. In prior work, the real-time calculus framework has proven to be very effective in this regard. However, real-time calculus has heretofore been limited to systems with uniprocessor processing elements, which is a serious impediment given the advent of multicore technologies. In this paper, a two-step approach is proposed that allows the power of real-time calculus to be applied in globally-scheduled multiprocessor systems: first, assuming that job response-time bounds are given, determine whether these bounds are met; second, using these bounds, determine the resulting residual processor supply and streams of job completion events using formalisms from real-time calculus. For this methodology to be applied in settings where response-time bounds are not specified, such bounds must be determined. Closed-form expressions for calculating such response-time bounds are presented for a large family of fixed-job-priority schedulers. The utility of the proposed analysis framework is demonstrated using a case study.

Keywords Multiprocessor Scheduling · Streaming Task Model · Response-Time Analysis · Real-Time Calculus

H. Leontyev
Department of Computer Science, University of North Carolina at Chapel Hill
Tel.: +1-919-9621794
E-mail: leontyev@cs.unc.edu

S. Chakraborty
Institute for Real-Time Computer Systems (RCS), Technical University of Munich
E-mail: samarjit@tum.de

J. H. Anderson
Department of Computer Science, University of North Carolina at Chapel Hill
Tel.: +1-919-9621757
Fax.: +1-919-96299
E-mail: anderson@cs.unc.edu

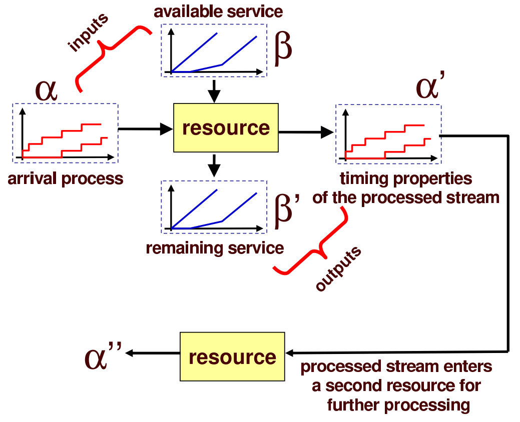
1 Introduction

The increasing complexity and heterogeneity of modern embedded platforms have led to growing interest in compositional modeling and analysis techniques (Richter et al 2003). In devising such techniques, the goal is not only to analyze the individual components of a platform in isolation, but also to compose different analysis results to estimate the timing and performance characteristics of the entire platform. Such analysis should be applicable even if individual processing and communication elements implement different scheduling/arbitration policies, have different interfaces, and are supplied by different vendors. These complicating factors often cause standard event models (e.g., periodic, sporadic, etc.) and schedulability-analysis techniques to lead to overly pessimistic results or to be altogether inapplicable.

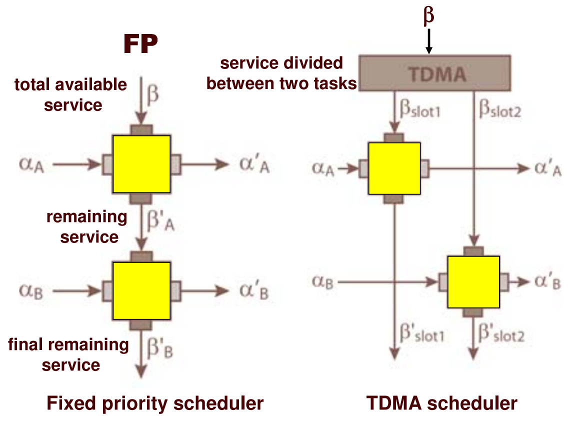
To overcome this difficulty, a compositional framework — often referred to as *real-time calculus* — was proposed by Chakraborty et al (2003) and then subsequently extended in a number of papers (e.g., see (Chakraborty et al 2006)). Real-time calculus is a specialization of *network calculus*, which was proposed by Cruz (1991a,b) and has been widely used to analyze communication networks. Real-time calculus specializes network calculus to the domain of real-time and embedded systems by, for example, adding techniques to model different schedulers and mode/state-based information (e.g., see (Phan et al 2008)). A number of schedulability tests have also been derived based upon network calculus. An overview of these tests can be found in (Wu et al 2005).

In real-time calculus, timing properties of event streams are represented using upper and lower bounds on the number of events that can arrive over any time interval of a specified length. These bounds are given by functions $\alpha^u(\Delta)$ and $\alpha^l(\Delta)$, which specify the maximum and minimum number of events, respectively, that can arrive at a processing/communication resource within any time interval of length Δ (or the maximum/minimum number of possible task activations within any Δ). The service offered by a resource is similarly specified using functions $\beta^u(\Delta)$ and $\beta^l(\Delta)$, which specify the maximum and minimum number of serviced events, respectively, within any interval of length Δ. Given the functions $\alpha^u(\Delta)$ and $\alpha^l(\Delta)$ corresponding to an event stream arriving at a resource, and the service $\beta^u(\Delta)$ and $\beta^l(\Delta)$ offered by it, it is possible to compute the timing properties of the processed stream and remaining processing capacity, i.e., functions $\alpha^{u\prime}(\Delta)$, $\alpha^{l\prime}(\Delta)$, $\beta^{u\prime}(\Delta)$, and $\beta^{l\prime}(\Delta)$, as illustrated in Figure 1(a), as well as the maximum backlog and delay experienced by the stream. As shown in the same figure, the computed functions $\alpha^{u\prime}(\Delta)$ and $\alpha^{l\prime}(\Delta)$ can then serve as inputs to the next resource on which this stream is further processed. By repeating this procedure until all resources in the system have been considered, timing properties of the fully-processed stream can be determined, as well as the end-to-end event delay and total backlog. This forms the basis for composing the analysis for individual resources, to derive timing/performance results for the full system.

Similarly, for any resource with tasks being scheduled according to some scheduling policy, it is also possible to compute service bounds ($\beta^u(\Delta)$ and $\beta^l(\Delta)$) available to its individual tasks. Figure 1(b) shows how this is done for the *fixed-priority* (FP) and *time-division-multiple-access* (TDMA) policies. As shown in this figure, for the FP policy, the *remaining* service after processing Stream A serves as the input (or,



(a)



(b)

Fig. 1: **(a)** Computing the timing properties of the processed stream using real-time calculus. **(b)** Scheduling networks for fixed priority and TDMA schedulers.

is available) to Stream B. On the other hand, for the TDMA policy, the total service $\beta(\Delta)$ is split between the services available to the two streams. Similar so called *scheduling networks* (Chakraborty et al 2006) can be constructed for other scheduling policies as well. Various operations on the arrival and service curves $\alpha(\Delta)$ and $\beta(\Delta)$, as well as procedures for the analysis of scheduling networks on uniprocessors (and partitioned systems) have been implemented in the RTC (real-time calculus) toolbox (Wandeler and Thiele 2006), which is a MATLAB-based library that can be used for modeling and analyzing distributed real-time systems.

Unfortunately, the compositional techniques described above can be used for the analysis of multiprocessor systems only if the multiprocessor under consideration is viewed as a collection of independent uniprocessors and the workload is split using some partitioning technique.

This precludes supporting workloads that fundamentally require *global* scheduling approaches. Under global scheduling, tasks are independently selected by processors from a single ready queue. In particular, when such algorithms are used, processors may be idle even though tasks are available for execution, as tasks must execute sequentially; this situation does not arise on uniprocessors and thus is not addressed in uniprocessor compositional techniques.

There are two reasons why existing compositional techniques need to be extended to incorporate such multiprocessors. First, multicore chips are becoming increasingly common. Second, global schedulers are often less resource-demanding than partitioned ones.

Example 1. Consider a MPEG-2 video decoder application that has been studied previously in (Chakraborty et al 2006; Phan et al 2008). The originally-studied application, shown in Figure 2(a), is partitioned and mapped onto two PEs, PE1 and PE2. PE1 runs the VLD (variable-length decoding) and IQ (inverse quantization) tasks, while PE2 runs the IDCT (inverse discrete cosine transform) and MC (motion compensation) tasks. The (coded) input bit stream enters this system and is stored in the input buffer B. The macroblocks (frame pieces of size 16×16 pixels) in B are first processed by PE1 and the corresponding partially decoded macroblocks are stored in the buffer B' before being processed by PE2. The resulting stream of fully decoded macroblocks is written into a playout buffer B'' prior to transmission by the output video device. In the above system, the coded input event stream arrives at a constant bit-rate.

It has been found from measurements that, in the above application, each task consumes 70% of the time on a dedicated 500 MHz processor. Thus, a single instance of the application requires two processors. Suppose that we want to support the simultaneous decoding of four video streams $S_1, \ldots, S_4$ such that stream S_1 has the lowest possible latency. Such a task system could be used in a virtual reality application, where multiple video streams need to be processed.

If a partitioned approach is used, then eight processors are needed as shown in Figure 2(b). (For conciseness, the intermediate buffers are not shown.) This is because no two tasks can be assigned to a single processor without overloading it. However, if global scheduling is used, then six processors are sufficient. In the system shown in Figure 2(c), the six processors are divided into two groups of three. One processor in each group is dedicated to running tasks processing stream S_1. The remaining time on this processor and the time available on the other two processors in each group is used for processing streams S_2, S_3, and S_4 as shown. The total long-term execution demand of all tasks running on each processor group is $4 \cdot 0.7 = 2.8$, while there are three processors available to each such group. In Figure 2(c), labeled down-arrows indicate the long-term fraction of time available on each processor (i.e., the long-term available per-processor utilization).

The timing properties of stream S_1 can be analyzed using existing uniprocessor real-time calculus methods. However, these methods cannot be applied for the analysis of streams S_2, S_3, and S_4.

In the next section, we briefly overview our proposed extensions to the real-time calculus framework (Chakraborty et al 2003, 2006) that allow systems such as that

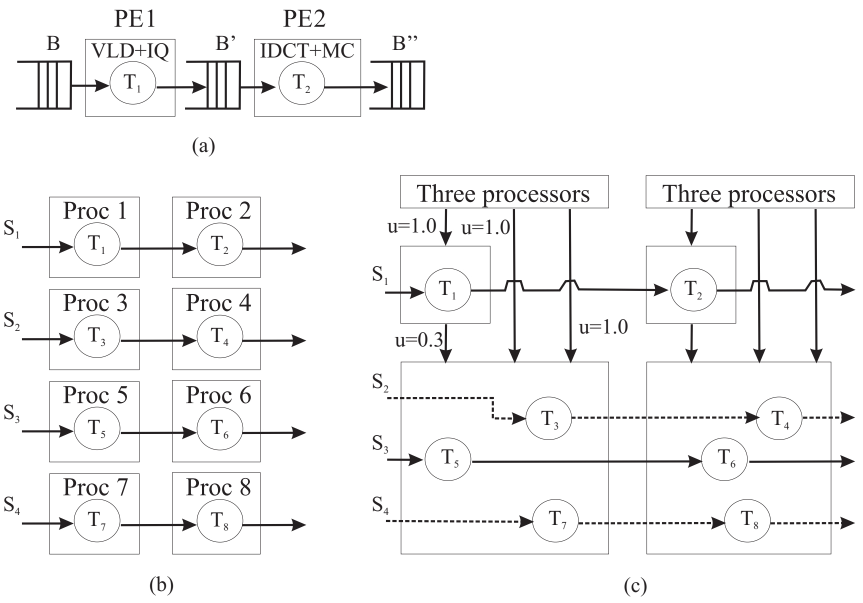


Fig. 2: **(a)** Example MPEG-2 decoder application. Four MPEG decoders running using **(b)** partitioned and **(c)** global scheduling.

presented in Example 1 above to be analyzed. The proposed extension incorporates globally-scheduled multiprocessors and is compatible with the RTC toolbox.

2 Analysis Overview and Prior Work

In this paper, we consider a PE that processes multiple input event streams $\alpha_1, \ldots, \alpha_n$ using the available resource supply $\mathcal{B}(\Delta)$.

The application of our results involves several theorems stated later and is illustrated in Figure 3. The core of our framework is a pseudo-polynomial-time procedure that, given a collection of arrival curves for input streams $\alpha^u(\Delta)$ and $\alpha^l(\Delta)$, their execution requirements, and $\mathcal{B}(\Delta)$, checks that event delays on such a multiprocessor reside within specified bounds. The set of delay bounds $\{\Theta_1, \ldots, \Theta_n\}$, where n is the number of streams, can be:

– specified (e.g., as relative deadlines) for individual tasks;
– calculated using Theorem 7 from the input if task deadlines are not specified or not feasible;
– determined by other means.

We should note that Theorem 7, which can be used to derive event-delay bounds from inputs, is only applicable in settings in which fixed-job-priority schedulers such as earliest-deadline-first (EDF) or first-in-first-out (FIFO) are used. When other schedulers are used, maximum delay bounds should be specified or found using alternative analysis techniques.

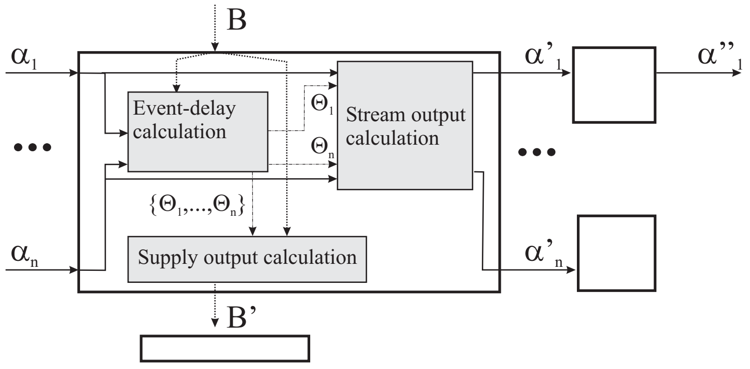


Fig. 3: A multiprocessor PE analyzed using multiprocessor real-time calculus.

In terms of computational complexity, calculating delay bounds using Theorem 7 (where applicable) and comparing those to specified deadlines is less costly than checking whether specified deadline constraints are met using the pseudo-polynomial shedulability test.

As Theorem 7 gives somewhat conservative estimations of delay bounds, they might be further tightened by iteratively decreasing them and applying the schedulability test. However, the details of this tightening procedure are specific to a task set and running a schedulability test multiple times might be time-consuming. In our case study (described later), tightening the bounds obtained using Theorem 7 did not give much improvement.

Once the event delays are identified, we can compute arrival curves for the processed streams $\alpha^{u\prime}(\Delta)$ and $\alpha^{l\prime}(\Delta)$ using Theorem 1. This is done by algebraic manipulations involving the specified input curves. As a result, because the maximum event-delay is bounded, the long-term departure rate of events is the same as the long-term arrival rate, i.e., the long-term growth rate of $\alpha_i^{u\prime}(\Delta)$ is the same as that of $\alpha_i^u(\Delta)$ for each task T_i.

Also, we can compute the remaining-total-service curve $\mathcal{B}'(\Delta)$ using Theorem 2. In this case, we subtract the total execution demand of tasks within an interval of length Δ from the total available supply $\mathcal{B}(\Delta)$. The obtained output curves — as in the uniprocessor case — can in turn be used as input for other resources, thereby resulting in a compositional framework (as shown in Figure 1(a)).

Prior work. Our work is based upon multiprocessor schedulability tests by Baruah (2007) and Leontyev and Anderson (2008). In some aspects, the presented analysis is also similar to results by Bertogna et al (2009), Shin et al (2008), and Zhang and Burns (2008). The main difference between our work and these prior efforts is that we consider more general task arrival and execution models, viz. those supported by the real-time calculus framework. Also, we consider the case when one or more processors can be partially available, which is similar to analysis in (Shin et al 2008), where partial availability is considered in the context of hierarchical scheduling.

Apart from the development of real-time calculus, a number of schedulability tests have been independently proposed for uniprocessor and partitioned settings that consider workload models incorporating long-term arrival patterns and execution requirements. These results pertain to workload models that are similar to that presented in our paper. An example of such analysis along with an overview of related work is presented in (Wu et al 2005). In their paper, Wu et al consider uniprocessor systems and static-priority scheduling. In contrast, we are primarily concerned with multiprocessor scheduling and EDF-like algorithms.

The rest of the paper is organized as follows. Section 3 presents our task model. In Sections 4 and 5, timing characteristics of processed streams and the remaining supply are computed. In Section 6, we present a basic response-time bound test. In Section 7, its time complexity is discussed. In Section 8, we improve the basic test for the case when an EDF-like scheduler is used. In Section 9, closed-form expressions for response-time bounds are derived. Section 10 presents a case study for our analysis. Section 11 summarizes our contributions and discusses some directions for future work.

3 Task Model

In this paper, we consider a task set $\tau = \{T_1, \ldots, T_n\}$. Each task has incoming jobs that are processed by a multiprocessor consisting of $m \geq 2$ unit-speed processors. We assume that $n \geq m$. We also assume that all time quantities except the interval length Δ are integral.

The j^{th} job of T_i, where $j \geq 1$, is denoted $T_{i,j}$. The *arrival* (or *release*) *time* of $T_{i,j}$ is denoted $r_{i,j}$. The *completion time* of $T_{i,j}$ is denoted $f_{i,j}$ and the delay between its start time and completion, $f_{i,j} - r_{i,j}$, is called its *response time*. As in prior work on real-time calculus, we wish to be able to accommodate very general assumptions concerning job executions and arrivals and the available service. Most of the remaining definitions in this section are devoted to formalizing the assumptions we require. Table 1 summarizes the notation introduced in this section.

Definition 1. $\gamma_i^u(k)$ ($\gamma_i^l(k)$) denotes an upper (lower) bound on the total execution time of any k consecutive jobs of T_i. (We assume $\gamma_i^u(k) = \gamma_i^l(k) = 0$ for all $k \leq 0$ and $\gamma_i^u(k) \leq \gamma_i^u(k+1)$ and $\gamma_i^l(k) \leq \gamma_i^l(k+1)$.) These definitions are equivalent to the workload demand curves in (Maxiaguine 2005).

Example 2. Suppose that task T_i's job execution times follow a pattern $1, 5, 2, 1, 5, 2,$ $\ldots$. Then, $\gamma_i^u(1) = 5$, $\gamma_i^u(2) = 7$, $\gamma_i^u(3) = 8$, $\gamma_i^u(4) = 13$, etc. Also, $\gamma_i^l(1) = 1$, $\gamma_i^l(2) = 3$, $\gamma_i^l(3) = 8$, $\gamma_i^l(4) = 9$, etc.

Definition 2. The *arrival function* $\alpha_i^u(\Delta)$ ($\alpha_i^l(\Delta)$) provides an upper (lower) bound on the number of jobs of T_i that can arrive within *any* time interval $(x, x+\Delta]$, where $x \geq 0$ and $\Delta > 0$ (Chakraborty et al 2006). (We assume $\alpha_i^u(\Delta) = 0$ for all $\Delta \leq 0$.) $\alpha_i(\Delta)$ denotes the pair $(\alpha_i^u(\Delta), \alpha_i^l(\Delta))$.

Example 3. The widely-studied periodic and sporadic task models are subcases of this more general task model. In both models, each job of T_i requires at most $e_i^{\max}$

Table 1: Model notation.

Input parameters	
$\alpha_i^u(\Delta)$ ($\alpha_i^l(\Delta)$)	Max. (min.) number of job arrivals of T_i over Δ
$\gamma_i^u(k)$ ($\gamma_i^l(k)$)	Max. (min.) execution demand of any k consecutive jobs of T_i
$\mathcal{B}(\Delta)$	Min. guaranteed cumulative processor supply over Δ
Params. below can be found using the RTC Toolbox	
$\widehat{U}$	Long-term avilable processor utilization
σ_{tot}	Maximum blackout time
F	Number of processors that are always available
$\mathcal{A}_i^{-1}(k)$	Pseudo-inverse of α_i^u
K_i	Min. integer s.t. $\mathcal{A}^{-1}(K_i) \geq \gamma_i^u(K_i)$
$\overline{e_i}$	T_i's average worst-case job execution time
v_i	Burstiness of the execution demand
R_i	Long-term arrival rate of T_i's jobs
B_i	Burstiness of the arrival curve
u_i	T_i's long-term utilization
U_{sum}	Total utilization
Θ_i below can be checked using the test in Section 6	
Θ_i	T_i's response-time bound
Output calculated using the input and $\{\Theta_i\}$	
${\alpha^u}'_i(\Delta)$ (${\alpha^l}'_i(\Delta)$)	Max. (min.) number of job completions of T_i over Δ
$\mathcal{B}'(\Delta)$	Min. guaranteed unused processor supply over Δ

execution units and consecutive job arrivals are separated by at least p_i time, where p_i is the *period* of T_i. Therefore, for both models, $\alpha_i^u(\Delta) = \left\lceil \frac{\Delta}{p_i} \right\rceil$ and $\gamma_i^u(k) = k \cdot e_i^{\max}$.

Definition 3. Let $\mathcal{A}_i^{-1}(k) = \inf\{\Delta \mid \alpha_i^u(\Delta) > k\}$, where $\Delta > 0$. This function characterizes the minimum length of the time interval $(x, x+\Delta]$ during which jobs $T_{i,j+1}, \ldots, T_{i,j+k}$ can be released for some j, assuming $T_{i,j}$ is released at time x. We define $\mathcal{A}_i^{-1}(0) = 0$ and require that there exists $K_i \geq 1$ such that

$$\mathcal{A}_i^{-1}(K_i) \geq \gamma_i^u(K_i). \tag{1}$$

We further require that there exists $R_i > 0$ and $B_i \geq 0$, where $R_i = \lim_{\Delta \to +\infty} \frac{\alpha_i^u(\Delta)}{\Delta}$, such that

$$\alpha_i^u(\Delta) \leq R_i \cdot \Delta + B_i \text{ for all } \Delta \geq 0. \tag{2}$$

Also, we assume that there exists $\overline{e_i} > 0$ and $v_i \geq 0$, where $\overline{e_i} = \lim_{k \to +\infty} \frac{\gamma_i^u(k)}{k}$, such that

$$\gamma_i^u(k) \leq \overline{e_i} \cdot k + v_i \text{ for all } k \geq 1. \tag{3}$$

(1) is needed in order to prevent task T_i from overloading the system. In (2), R_i characterizes the long-term arrival rate of task T_i's jobs and B_i characterizes the degree of burstiness of the arrival sequence. In (3), the parameter $\overline{e_i}$ denotes the average worst-case job execution time of T_i.

Definition 4. Let $u_i = R_i \cdot \overline{e_i}$. This quantity denotes the average long-term utilization of task T_i. We require that $0 < u_i \leq 1$. Let $U_{sum} = \sum_{T_i \in \tau} u_i$.

Example 4. Under the sporadic task model, $R_i = \lim_{\Delta \to +\infty} \left(\left\lfloor \frac{\Delta}{p_i} \right\rfloor + 1 \right) / \Delta = \frac{1}{p_i}$ and $\overline{e_i} = e_i^{\max}$, so $u_i = R_i \cdot \overline{e_i} = \frac{e_i^{\max}}{p_i}$.

Definition 5. Let $\mathsf{supply}_h(t, \Delta)$ be the total amount of processor time available to tasks in τ on processor h in the interval $[t, t + \Delta)$, where $\Delta \geq 0$. Let $\mathsf{Supply}(t, \Delta) = \sum_{h=1}^{m} \mathsf{supply}_h(t, \Delta)$ be the cumulative processor supply in the interval $[t, t + \Delta)$.

Though we desire to make our analysis compatible with the real-time calculus framework, which requires that individual processor supplies be known, there exist many settings in which individual processor supply functions are not known and a lower bound on the cumulative available processor time is provided instead. (In uniprocessor real-time calculus, the available service is described as the number of incoming events processed by a PE during a time interval.) Note that if individual processor supply guarantees are known, a lower bound on the cumulative guaranteed supply can be computed easily.

Definition 6. Let $\mathcal{B}(\Delta) \leq \mathsf{Supply}(t, \Delta)$ be the guaranteed total time that all processors can provide to the tasks in τ during any time interval $[t, t + \Delta)$, where $\Delta \geq 0$. We assume that

$$\mathcal{B}(\Delta) \geq \max(0, \widehat{U} \cdot (\Delta - \sigma_{tot})), \tag{4}$$

where $\widehat{U} \in (0, m]$ and $\sigma_{tot} \geq 0$. We let F be the number of processors that are always available at any time. If all processors have unit speed (as we have assumed), then $F = \max\{y \mid \forall \Delta \geq 0 :: \mathcal{B}(\Delta) \geq y \cdot \Delta\}$.

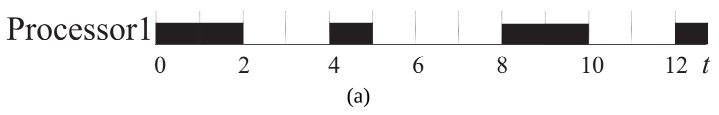
In the above definition, the parameters $\widehat{U}$, which is the total long-term fraction of processor time available to the tasks in τ on the entire platform, and σ_{tot}, which is the maximum duration of time when all processors are unavailable, are similar to those in the bounded delay model (Mok et al 2001).

Example 5. Consider a system with one processor ($m = 1$) that is not fully available. The availability pattern, which repeats every eight time units, is shown in Figure 4(a); intervals of unavailability are shown as shaded regions. For processor 1, the minimum amount of time that is guaranteed to real-time tasks over any interval of length Δ is zero if $\Delta \leq 2$, $\Delta - 2$ if $2 \leq \Delta \leq 4$, and so on. Figure 4(b) shows the minimum amount of time $\mathcal{B}(\Delta)$ that is available on processor 1 for tasks over any interval $[t, t + \Delta]$. It also shows a lower bound $\max(0, \widehat{U}(\Delta - \sigma_{tot}))$, where $\widehat{U} = \frac{5}{8}$ and $\sigma_{tot} = 2$, which bounds $\mathcal{B}(\Delta)$ from below.

We require that (5) below holds for otherwise the system is overloaded and job response times could be unbounded.

$$U_{sum} \leq \widehat{U} \tag{5}$$

We assume that released jobs are placed into a single global ready queue. When choosing a new job to schedule, the scheduler selects (and dequeues) the ready job of highest priority. An unfinished job is *pending* if it is released. A pending job is *ready* if its predecessor (if any) has completed execution. Note that, the jobs of each task execute sequentially. Job priorities are determined as follows.



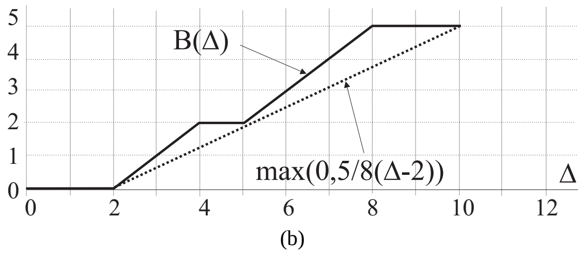


Fig. 4: **(a)** Unavailable time instants and **(b)** service function in Example 5.

Definition 7. (prioritization rules) Associated with each job $T_{i,j}$ is a constant value $\chi_{i,j}$. If $\chi_{i,j} < \chi_{k,h}$ or $\chi_{i,j} = \chi_{k,h} \wedge (i < k \vee (i = k \wedge j < h))$, then the priority of $T_{i,j}$ is higher than that of $T_{k,h}$, denoted $T_{i,j} \prec T_{k,h}$. Additionally, we assume $j < h$ implies $\chi_{i,j} \le \chi_{i,h}$ for each task T_i.

Example 6. Global earliest-deadline-first (GEDF) priorities can be defined by setting $\chi_{i,j} = r_{i,j} + D_i$ for each job $T_{i,j}$, where D_i is T_i's relative deadline. Global first-in-first-out (FIFO) priorities can be defined by setting $\chi_{i,j} = r_{i,j}$, and static priorities can be defined by setting $\chi_{i,j}$ to a constant (Leontyev and Anderson 2009a).

The technical contributions of this paper are threefold. First, given a task set $\tau = \{T_1, \ldots, T_n\}$ and a multiprocessor platform characterized by a cumulative guaranteed processor time $\mathcal{B}(\Delta)$, we develop a sufficient test that verifies whether the maximum job response time of a task $T_i \in \tau$, $\max_j(f_{i,j} - r_{i,j})$, is at most Θ_i, where

$$\Theta_i \ge \max_{j \ge 1}(\gamma_i^u(j) - \mathcal{A}_i^{-1}(j-1)). \tag{6}$$

The right-hand side of (6) is the maximum job response time bound of T_i when it is scheduled on a dedicated processor. Consider a sequence of j consecutive jobs $T_{i,a}, \ldots, T_{i,a+j-1}$ scheduled on a dedicated processor such that $T_{i,a}$ starts its execution at $r_{i,a}$ and $r_{i,k} \le f_{i,k-1}$ for $k \in [a+1, a+j-1]$. The response time of job $T_{i,a+j-1}$ is $f_{i,a+j-1} - r_{i,a+j-1} = (f_{i,a+j-1} - r_{i,a}) - (r_{i,a+j-1} - r_{i,a})$. Because the processor is dedicated and jobs execute back-to-back, $f_{i,a+j-1} - r_{i,a} \le \gamma_i^u(j)$. Below, in Section 6 in Lemma 2, we show that $r_{i,a+j-1} - r_{i,a} \ge \mathcal{A}_i^{-1}(j-1)$. Thus, $f_{i,a+j-1} - r_{i,a+j-1} \le \gamma_i^u(j) - \mathcal{A}_i^{-1}(j-1)$.

If Θ_i equals the relative deadline of a job, then the proposed test will check whether the system is hard-real-time schedulable. Alternatively, if deadlines are allowed to be missed and Θ_i includes the maximum allowed deadline tardiness, then the test will check soft-real-time schedulability. Such a test allows workloads to be considered that fundamentally require global scheduling approaches.

In settings where response-time bounds are not known, they must be determined. Our second contribution is a set of closed-form expressions for calculating response-time bounds directly from task and supply parameters for a large class of scheduling algorithms. These response-time bounds can be directly used for calculating stream and supply outputs. It is also possible to refine the obtained response-time bounds by incrementally decreasing them and running the schedulability test to see if the smaller bounds are also valid.

Finally, given per-task bounds on maximum job response times, we characterize the sequence of job completion events for each task T_i by deriving the next-stage arrival functions $\alpha_i^{u\prime}(\Delta)$ and $\alpha_i^{l\prime}(\Delta)$, and the remaining processor supply $\mathcal{B}'(\Delta)$ (see Figure 3). These functions, in turn, can serve as inputs to subsequent PEs, thereby resulting in a compositional technique.

4 Calculating $\alpha_i^{u\prime}$ and $\alpha_i^{l\prime}$

Let $\alpha_i^{u\prime}(\Delta)$ ($\alpha_i^{l\prime}(\Delta)$) be the maximum (respectively, minimum) number of job completions of task T_i over an interval $(x, x+\Delta]$, where $x \geq 0$. Bounds on these functions can be computed using Theorem 1 below. The following definition is used in the statement of the theorem.

Definition 8. Let ${\gamma^l_i}^{-1}(\Delta) = \inf\{k \mid k \text{ is integral and } \gamma_i^l(k) \geq \Delta\}$, where $\Delta > 0$, be the pseudoinverse function of the lower bound on the execution time function $\gamma_i^l(k)$ (note that we use a non-strict inequality because $\gamma_i^l(k)$ is not defined for non-integral values of k). For $\Delta = 0$, we define ${\gamma^l_i}^{-1}(0) = 0$.

Example 7. The function ${\gamma^l_i}^{-1}(\Delta)$ gives an upper bound on the number of jobs that can complete over any interval $(x, x+\Delta]$, where $\Delta > 0$. For example, if $\Delta = \gamma_i^l(1)$, then at most one job can complete over any interval $(x, x+\gamma_i^l(1)]$. Similarly, if $\Delta = \gamma_i^l(k)$ for some k, then at most k jobs can complete over any interval $(x, x+\gamma_i^l(k)]$.

Theorem 1. *If the response time of any job of T_i is at most Θ_i, then $\alpha_i^{u\prime}(\Delta) \leq \min\left({\gamma^l_i}^{-1}(\Delta),\ \alpha_i^u(\Delta + \Theta_i - \gamma_i^l(1))\right)$ and $\alpha_i^{l\prime}(\Delta) \geq \alpha_i^l(\Delta - \Theta_i + \gamma_i^l(1))$.*

Proof. We first prove the first inequality. Consider an interval $(t_1, t_2]$ such that at least one job of T_i completes within it and let $\Delta = t_2 - t_1$. Let N_1, (N_2) be the index of the first (last) job of T_i completed within $(t_1, t_2]$. Then,

$$f_{i,N_1} > t_1 \quad \text{and} \quad f_{i,N_2} \leq t_2. \tag{7}$$

By the condition of the theorem, job $T_{i,j}$'s response time $f_{i,j} - r_{i,j}$ is at most Θ_i. By the definition of response time and Definition 1, $f_{i,j} - r_{i,j}$ is at least $\gamma_i^l(1)$. From (7), we thus have $r_{i,N_1} > t_1 - \Theta_i$ and $r_{i,N_2} \leq t_2 - \gamma_i^l(1)$. Thus, the number of jobs completed within the interval $(t_1, t_2]$, $N_2 - N_1 + 1$, is at most the number of jobs released within the interval $(t_1 - \Theta_i, t_2 - \gamma_i^l(1)]$. By Definition 2, we have $N_2 - N_1 + 1 \leq \alpha_i^u(t_2 - \gamma_i^l(1) - t_1 + \Theta_i) = \alpha_i^u(\Delta + \Theta_i - \gamma_i^l(1))$. Moreover, from Definition 8, it follows that at most ${\gamma^l_i}^{-1}(\Delta)$ jobs can complete within an interval of length $\Delta > 0$.

We now prove the second inequality. Consider an interval $(t_1, t_2]$ and let $\Delta = t_2 - t_1$. Let N_1, (N_2) be the index of the last (respectively, first) job of T_i completed at or before time t_1 (respectively, after time t_2). Then,

$$f_{i,N_1} \leq t_1 \quad \text{and} \quad f_{i,N_2} > t_2. \tag{8}$$

By the condition of the theorem, job $T_{i,j}$'s response time $f_{i,j} - r_{i,j}$ is at most Θ_i. By the definition of response time and Definition 1, $f_{i,j} - r_{i,j}$ is at least $\gamma_i^l(1)$. From (8), we thus have $r_{i,N_2} > t_2 - \Theta_i$ and $r_{i,N_1} \leq t_1 - \gamma_i^l(1)$. Thus, the number of jobs completed within the interval $(t_1, t_2]$, $N_2 - N_1 - 1$, is at least the number of jobs released within the interval $(t_1 - \gamma_i^l(1), t_2 - \Theta_i]$. By Definition 2, we have $N_2 - N_1 - 1 \geq \alpha_i^l(t_2 - \Theta_i - t_1 + \gamma_i^l(1)) = \alpha_i^l(\Delta - \Theta_i + \gamma_i^l(1))$. □

5 Calculating $\mathcal{B}'(\Delta)$

We now calculate a lower bound $\mathcal{B}'(\Delta)$ on processor time that is available after scheduling tasks $T_1, \ldots, T_n$. We first upper-bound the total allocation of jobs of T_i over any interval of length Δ.

Definition 9. Let $\mathsf{A}(T_{i,y}, I)$ (respectively, $\mathsf{A}(T_i, I)$) be the amount of time for which job $T_{i,y}$ (respectively, task T_i) executes within the set of intervals I.

Lemma 1. *If the response time of any job of T_i is at most Θ_i, then $\mathsf{A}(T_i, [t, t+\Delta)) \leq \min(\Delta, \gamma_i^u(\alpha_i^u(\Delta + \Theta_i)))$.*

Proof. Consider an interval $[t, t+\Delta)$. The condition of the lemma implies that all of T_i's jobs released at or before time $t - \Theta_i$ complete by time t. Thus, the allocation of T_i within $[t, t+\Delta)$, $\mathsf{A}(T_i, [t, t+\Delta))$, is upper-bounded by the maximum execution demand of T_i's jobs released within the interval $(t - \Theta_i, t+\Delta]$. By Definition 2, there are at most $\alpha_i^u(\Delta + \Theta_i)$ such jobs, and by Definition 1, their total execution demand is at most $\gamma_i^u(\alpha_i^u(\Delta+\Theta_i))$. We thus have $\mathsf{A}(T_i, [t, t+\Delta)) \leq \gamma_i^u(\alpha_i^u(\Delta+\Theta_i))$. Also, $\mathsf{A}(T_i, [t, t+\Delta))$ cannot exceed the length of the interval $[t, t+\Delta)$. □

Theorem 2. *If, for each task T_i, the response time of any of its jobs is at most Θ_i, then at least*

$$\mathcal{B}'(\Delta) = \sup_{0 \leq y \leq \Delta} (Z(y)) \tag{9}$$

time units are available over any interval of length $\Delta \geq 0$, where $Z(y) = \max\big(0, \mathcal{B}(y) - \sum_{T_i \in \tau} \min(y, \gamma_i^u(\alpha_i^u(y+\Theta_i)))\big)$. Additionally, (4) *for $\mathcal{B}'(\Delta)$ holds with $\widehat{U}' = \widehat{U} - U_{sum}$ and $\sigma'_{tot} = (\widehat{U} \cdot \sigma_{tot} + \sum_{T_i \in \tau}(u_i \cdot \Theta_i + \overline{e_i} \cdot B_i + v_i))/\widehat{U}'$.*

Proof. Consider an interval $[t, t+y)$, where $y \leq \Delta$. Let $\mathsf{Supply}'(t, y)$ be the amount of supply that is available after scheduling the tasks in τ in this interval. By Definitions 5 and 9, we have

$$\begin{aligned}
\mathsf{Supply}'(t, y) &= \mathsf{Supply}(t, y) - \sum_{T_i \in \tau} \mathsf{A}(T_i, [t, t + y)) \\
&\quad \{\text{by Definition 6}\} \\
&\geq \max\left(0, \mathcal{B}(y) - \sum_{T_i \in \tau} \mathsf{A}(T_i, [t, t + y))\right) \\
&\quad \{\text{by Lemma 1}\} \\
&\geq \max\left(0, \mathcal{B}(y) - \sum_{T_i \in \tau} \min(y, \gamma_i^u(\alpha_i^u(y + \Theta_i)))\right) \\
&\quad \{\text{by the definition of } Z(y) \text{ in the statement of the theorem}\} \\
&= Z(y). \qquad (10)
\end{aligned}$$

Additionally, $\mathsf{Supply}'(t, \Delta) \geq \sup_{0 \leq y \leq \Delta}(\mathsf{Supply}'(t, y))$. From this inequality and (10), we have $\mathsf{Supply}'(t, \Delta) \geq \sup_{0 \leq y \leq \Delta}(Z(y)) = \mathcal{B}'(\Delta)$.

We are left with finding coefficients $\widehat{U}'$ and σ'_{tot} such that (4) holds for $\mathcal{B}'(\Delta)$. Setting (4) (for $\mathcal{B}(\Delta)$) into the definition of $Z(y)$, we have

$$\begin{aligned}
Z(y) &\geq \max\left(0, \max(0, \widehat{U} \cdot (y - \sigma_{tot})) - \sum_{T_i \in \tau} \min(y, \gamma_i^u(\alpha_i^u(y + \Theta_i)))\right) \\
&\geq \max\left(0, \widehat{U} \cdot (y - \sigma_{tot}) - \sum_{T_i \in \tau} \min(y, \gamma_i^u(\alpha_i^u(y + \Theta_i)))\right) \\
&\quad \{\text{by (2) and (3)}\} \\
&\geq \max\left(0, \widehat{U} \cdot (y - \sigma_{tot}) - \sum_{T_i \in \tau} (\overline{e_i} \cdot (R_i \cdot (y + \Theta_i) + B_i) + v_i)\right) \\
&\quad \{\text{by Definition 4}\} \\
&= \max\left(0, \widehat{U} \cdot (y - \sigma_{tot}) - \sum_{T_i \in \tau} (u_i \cdot y + u_i \cdot \Theta_i + \overline{e_i} \cdot B_i + v_i)\right) \\
&= \max\left(0, \widehat{U} \cdot (y - \sigma_{tot}) - U_{sum} \cdot y + \sum_{T_i \in \tau} (u_i \cdot \Theta_i + \overline{e_i} \cdot B_i + v_i)\right) \\
&= \max\left(0, (\widehat{U} - U_{sum}) \cdot y - \widehat{U} \cdot \sigma_{tot} - \sum_{T_i \in \tau} (u_i \cdot \Theta_i + \overline{e_i} \cdot B_i + v_i)\right) \\
&\quad \{\text{by the definition of } \widehat{U}' \text{ and } \sigma'_{tot} \text{ in the statement of the theorem}\} \\
&= \max\left(0, \widehat{U}' \cdot (y - \sigma'_{tot})\right). \qquad (11)
\end{aligned}$$

Finally, by (9) and (11), $\mathcal{B}'(\Delta) \geq \sup_{0 \leq y \leq \Delta}\left(\max\left(0, \widehat{U}' \cdot (y - \sigma'_{tot})\right)\right) = \max\left(0, \widehat{U}' \cdot (\Delta - \sigma'_{tot})\right)$. Thus, (4) holds with $\mathcal{B}'(\Delta)$, $\widehat{U}'$, and σ'_{tot} as defined. $\square$

6 Multiprocessor Schedulability Test

In this section, we present the core analysis of our framework in the form of a schedulability test (given in Corollary 1 later in this section) that checks whether a predefined response-time bound Θ_i is not violated for a task T_i.

As noted earlier, the way jobs are prioritized according to Definition 7 is similar to GEDF or static priority scheduling. A number of GEDF schedulability tests have been developed assuming that jobs arrive periodically or sporadically (e.g., (Baruah 2007; Bertogna et al 2009; Leontyev and Anderson 2008)). In this paper, we extend techniques from (Baruah 2007) and (Leontyev and Anderson 2008) in order to incorporate more general job arrivals and execution models.

Similarly to (Devi 2006), we derive our test by ordering jobs by their priorities and assuming that $T_{\ell,q}$ is the first job for which $f_{\ell,q} > r_{\ell,q} + \Theta_\ell$. We further assume that, for each job $T_{a,b}$ such that $T_{a,b} \prec T_{\ell,q}$,

$$f_{a,b} \leq r_{a,b} + \Theta_a. \tag{12}$$

We first derive a necessary condition for $T_{\ell,q}$ to violate its response-time bound by considering an interval that includes the time when $T_{\ell,q}$ becomes ready and the latest time when $T_{\ell,q}$ is allowed to complete, which is $r_{\ell,q} + \Theta_\ell$. This interval is parametrized by a number $k \in [1, K_\ell]$ (see Definition 3) and δ (defined later in this section), which determine its length, $\delta + \Theta_\ell$. (The range of δ depends on k and ℓ.) In essence, the parameter k defines the number of $T_{\ell,q}$'s predecessors (including $T_{\ell,q}$ itself) that complete "too late" to warrant $T_{\ell,q}$'s timely completion in the presence of other tasks. During this interval, we consider demand due to competing higher-priority jobs that can interfere with $T_{\ell,q}$ or its predecessors. We then perform the following three steps:

S1: Compute the minimum guaranteed supply $\mathcal{B}(\delta + \Theta_\ell)$ over the interval of interest.

S2: Given a finite upper bound $M^*_\ell(\delta, \tau, m)$ on the competing demand and a finite upper bound on the unfinished work due to job $T_{\ell,q}$ and its predecessors, $E^*_\ell(k)$, define a sufficient test for checking whether T_ℓ's response-time bound is not violated by checking that $M^*_\ell(\delta, \tau, m) + (m-1) \cdot (E^*_\ell(k) - 1) < \mathcal{B}(\delta + \Theta_\ell)$ holds for each $k \in [1, K_\ell]$ and δ defined with respect to k and ℓ.

S3: Calculate $M^*_\ell(\delta, \tau, m)$ and $E^*_\ell(k)$ as used in **S2**.

6.1 Steps **S1** and **S2**

To avoid distracting "boundary cases," we henceforth assume that the schedule being analyzed is prepended with a schedule in which response-time bounds are not violated that is long enough to ensure that all predecessor jobs referenced in the proof exist. We begin with the following definition.

Definition 10. Let $\alpha^+_i(\Delta) = \lim_{\epsilon \to +0} \alpha^u_i(\Delta + \epsilon)$. This function provides an upper bound on the number of jobs released within any interval $[x, x + \Delta]$, where $x \geq 0$ and $\Delta \geq 0$. (We assume $\alpha^+_i(\Delta) = 0$ for all $\Delta < 0$.)

The next example illustrates the difference between the functions α_i^u and α_i^+.

Example 8. Consider a task T_i, whose jobs arrive periodically with period p_i. The maximum number of jobs that can arrive within an interval $(x, x + 2 \cdot p_i]$ is thus $\alpha_i^u(2 \cdot p_i) = \left\lceil \frac{2 \cdot p_i}{p_i} \right\rceil = 2$. However, the maximum number of jobs that can arrive within the interval $[x, x + 2 \cdot p_i]$ is $\alpha_i^+(2 \cdot p_i) = \lim_{\epsilon \to +0} \alpha_i^u(2 \cdot p_i + \epsilon) = 3$. In general, under the sporadic task model, $\alpha_i^+(\Delta) = \left\lfloor \frac{\Delta}{p_i} \right\rfloor + 1$.

We start the derivation by proving a lemma and several claims. The following lemma specifies the minimum time between the arrivals of jobs $T_{h,g}$ and $T_{h,g-i}$.

Lemma 2. $r_{h,g} - r_{h,g-i} \geq \mathcal{A}_h^{-1}(i)$.

Proof. Let $\Delta' = r_{h,g} - r_{h,g-i}$. Let

$$\Delta^* = \inf\{\Delta \mid \alpha_h^+(\Delta) \geq i + 1\}. \tag{13}$$

Because jobs $T_{h,g-i}, \ldots, T_{h,g}$ are released within the interval $[r_{h,g-i}, r_{h,g}]$, by Definition 10, $\alpha_h^+(\Delta') \geq i + 1$. Therefore, by (13),

$$r_{h,g} - r_{h,g-i} = \Delta' \geq \Delta^*. \tag{14}$$

We now consider two cases.

Case 1: $\alpha_h^u(\Delta^*) > i$**.** In this case, $\Delta^* \overset{\{\text{by Definition 10}\}}{\geq} \inf\{\Delta \mid \alpha_h^u(\Delta) > i\} \overset{\{\text{by Definition 3}\}}{=} \mathcal{A}_h^{-1}(i)$. The lemma follows from this and (14).

Case 2: $\alpha_h^u(\Delta^*) \leq i$**.** Because $\alpha_h^u(\Delta)$ is non-decreasing, $\alpha_h^u(\Delta^*) \leq i$ implies

$$\alpha_\ell^u(\Delta) \leq i \text{ for each } \Delta \leq \Delta^*. \tag{15}$$

Further, by (13),

$$\alpha_h^+(\Delta) < i + 1, \text{for each } \Delta < \Delta^*. \tag{16}$$

Suppose that for some $\Delta'' > \Delta^*$, $\alpha_h^u(\Delta'') \leq i$. Because $\alpha_h^u(\Delta)$ is non-decreasing, this implies $\alpha_h^u(\Delta_x) \leq i$ for each $\Delta_x \in [\Delta^*, \Delta'')$. The latter implies $\alpha_h^+(\Delta_x) = \lim_{\epsilon \to +0} \alpha_h^u(\Delta_x + \epsilon) \leq i$ for each $\Delta_x \in [\Delta^*, \Delta'')$. From this and (16), we have $\alpha_h^+(\Delta) < i + 1$ for each $\Delta < \Delta''$. Since $\Delta'' > \Delta^*$, we have a contradiction to (13). Therefore, $\alpha_h^u(\Delta) > i$ for each $\Delta > \Delta^*$. From this and (15), we have $\Delta^* = \inf\{\Delta \mid \alpha_h^u(\Delta) > i\} \overset{\{\text{by Definition 3}\}}{=} \mathcal{A}_h^{-1}(i)$. The lemma follows from this equality and (14). $\square$

The next two claims establish a lower bound on the maximum job response time and an upper bound on the finish times of certain jobs that can be used in addition to (12).

Claim 1: $\Theta_i \geq \gamma_i^u(1)$.

Proof. By (6), $\Theta_i \geq \max_{j \geq 1}(\gamma_i^u(j) - \mathcal{A}_i^{-1}(j-1)) \geq \gamma_i^u(1) - \mathcal{A}_i^{-1}(0)$. By Definition 3, $\mathcal{A}_i^{-1}(0) = 0$. $\square$

Claim 2: $f_{\ell,q-K_\ell} \le r_{\ell,q} + \Theta_\ell - \gamma_\ell^u(K_\ell)$.

Proof. By (12), for $i \ge 1$,

$$\begin{aligned} f_{\ell,q-i} &\le r_{\ell,q-i} + \Theta_\ell \\ &= r_{\ell,q-i} - r_{\ell,q} + r_{\ell,q} + \Theta_\ell \\ &\quad \{\text{by Lemma 2}\} \\ &\le r_{\ell,q} + \Theta_\ell - \mathcal{A}_\ell^{-1}(i). \end{aligned} \tag{17}$$

By (1), $-\mathcal{A}_\ell^{-1}(K_\ell) \le -\gamma_\ell^u(K_\ell)$. Setting this and $i = K_\ell$ into (17), we get the required result. □

Job $T_{\ell,q}$ can violate its response-time bound for the following reasons. If $T_{\ell,q-1}$ completes by time $r_{\ell,q}+\Theta_\ell-\gamma_\ell^u(1)$, then $T_{\ell,q}$ may finish its execution after $r_{\ell,q}+\Theta_\ell$ if, after time $\max(f_{\ell,q-1}, r_{\ell,q})$, higher-priority jobs deprive it of processor time or one or more processors are unavailable. Alternatively, $T_{\ell,q-1}$ may complete *after* time $r_{\ell,q} + \Theta_\ell - \gamma_\ell^u(1)$, which can happen if the minimum job inter-arrival time for T_ℓ is less than $\gamma_\ell^u(1)$. In this situation, $T_{\ell,q}$ could violate its response-time bound even if it executes uninterruptedly within $[f_{\ell,q-1}, r_{\ell,q} + \Theta_\ell)$. In this case, T_ℓ's response-time bound is violated because $T_{\ell,q-1}$ completes "late," namely after time $r_{\ell,q}$ (recall that, by Claim 1, $\Theta_\ell \ge \gamma_\ell^u(1)$). However, this implies that T_ℓ is pending continuously throughout the interval $[r_{\ell,q-1}, r_{\ell,q} + \Theta_\ell)$, and hence, we can examine the execution of jobs $T_{\ell,q-1}$ and $T_{\ell,q}$ together. In this case, we need to consider the completion time of job $T_{\ell,q-2}$. If $f_{\ell,q-2} \le r_{\ell,q} + \Theta_\ell - \gamma_\ell^u(2)$, then job $T_{\ell,q}$ may exceed its response-time bound if this job and its predecessor, $T_{\ell,q-1}$, experience interference from higher-priority jobs or some processors are unavailable during the time interval $[\max(f_{\ell,q-2}, r_{\ell,q-1}), r_{\ell,q} + \Theta_\ell)$. On the other hand, if $f_{\ell,q-2} > r_{\ell,q} + \Theta_\ell - \gamma_\ell^u(2)$, then $T_{\ell,q}$ can complete after time $r_{\ell,q}+\Theta_\ell$ even if T_ℓ executes uninterruptedly within $[f_{\ell,q-2}, r_{\ell,q}+\Theta_\ell)$. Continuing by considering predecessor jobs $T_{\ell,q-k}$ in this manner, we will exhaust all possible reasons for the response-time bound violation. Note that it is sufficient to consider only jobs $T_{\ell,q-1}, \ldots, T_{\ell,q-K_\ell}$ since, by Claim 2, $f_{\ell,q-K_\ell} \le r_{\ell,q} + \Theta_\ell - \gamma_\ell^u(K_\ell)$. Assuming that, for job $T_{\ell,q-k}$, $f_{\ell,q-k} \le r_{\ell,q} + \Theta_\ell - \gamma_\ell^u(k)$, we define the *problem window* for jobs $T_{\ell,q-k+1}, \ldots, T_{\ell,q}$ as $[r_{\ell,q-k+1}, r_{\ell,q}+\Theta_\ell)$. (This problem window definition is a significant difference when comparing our analysis to prior analysis pertaining to periodic or sporadic systems.)

Definition 11. Let $\lambda \in [1, K_\ell]$ be the smallest integer such that $f_{\ell,q-\lambda} \le r_{\ell,q}+\Theta_\ell-\gamma_\ell^u(\lambda)$. By Claim 2, such a λ exists.

Claim 3. *T_ℓ is ready (i.e., has a ready job) at each instant of the interval $[r_{\ell,q-k+1}, r_{\ell,q} + \Theta_\ell)$ for each $k \in [1, \lambda]$.*

Proof. To prove the claim, we first show that, for each $k \in [1, \lambda]$, T_ℓ is ready continuously within $[r_{\ell,q-k+1}, f_{\ell,q})$. Because T_ℓ is ready within the interval $[r_{\ell,q}, f_{\ell,q})$, this is true for $k = 1$. If $k > 1$ (in which case $\lambda > 1$), then $f_{\ell,q-j} > r_{\ell,q} + \Theta_\ell - \gamma_\ell^u(j)$

for each $j \in [1, \lambda)$, by the selection of λ. From this, we have

$$\begin{aligned} f_{\ell,q-j} &> r_{\ell,q} + \Theta_\ell - \gamma_\ell^u(j) \\ &\quad \{\text{because, by (6), } \Theta_\ell \geq \gamma_\ell^u(j) - \mathcal{A}_\ell^{-1}(j-1)\} \\ &\geq r_{\ell,q} - \mathcal{A}_\ell^{-1}(j-1) \\ &\quad \{\text{by Lemma 2}\} \\ &\geq r_{\ell,q-j+1}. \end{aligned}$$

Thus, the intervals $[r_{\ell,q-j}, f_{\ell,q-j})$ and $[r_{\ell,q-j+1}, f_{\ell,q-j+1})$, where consecutive jobs of T_ℓ are ready, overlap. Therefore, T_ℓ is ready continuously within $[r_{\ell,q-j}, f_{\ell,q})$ for each $j \in [1, \lambda)$, and hence, T_ℓ is ready continuously within $[r_{\ell,q-k+1}, f_{\ell,q})$ for each $k \in [2, \lambda]$. The claim follows from $[r_{\ell,q-k+1}, r_{\ell,q} + \Theta_\ell) \subset [r_{\ell,q-k+1}, f_{\ell,q})$; to see this, note that $f_{\ell,q} > r_{\ell,q} + \Theta_\ell$ holds, since $T_{\ell,q}$ violates its response-time bound. $\square$

Because $T_{\ell,q}$ violates its response-time bound, after time $r_{\ell,q-k+1}$, there are other higher-priority jobs that deprive T_ℓ of processor time or one or more processors are unavailable.

Definition 12. Let $\tau_p(t) = \{T_h \mid \text{for some } y,\ T_{h,y} \text{ is ready at time } t \text{ and } T_{h,y} \preceq T_{\ell,q}\}$. (The subscript p denotes the fact that these jobs have higher or equal priority.)

To indicate an excessive number of tasks with ready jobs of equal or higher priority at time t we will use the following predicate.

$$\text{IS\_HP}(t) = (|\tau_p(t)| \geq m \text{ or fewer than } |\tau_p(t)| \text{ tasks from } \tau_p(t) \text{ execute at time } t). \tag{18}$$

Definition 13. Let $t_0(k) \leq r_{\ell,q-k+1}$ be the earliest instant such that $\forall t \in [t_0(k), r_{\ell,q-k+1})$, IS_HP$(t)$ holds. If such an instant does not exist, then let $t_0(k) = r_{\ell,q-k+1}$.

The definition below defines the jobs that can compete with $T_{\ell,q}$ or its predecessors.

Definition 14. Let $\mathcal{J}$ be the set of jobs $T_{i,y}$ such that **(i)** $T_{i,y} \preceq T_{\ell,q}$ or **(ii)** $T_{i,y} \succ T_{\ell,q}$, $i \neq \ell$, and $T_{i,y}$ executes at some time $t \in [t_0(k), r_{\ell,q} + \Theta_\ell)$ and IS_HP(t) holds. (More informally, $\mathcal{J}$ includes higher-or-equal-priority jobs and lower-priority jobs that cause non-preemptive blocking.) Note that $\mathcal{J}$ does not contain $T_{\ell,q}$'s successors.

In this paper, we are mainly concerned with fully preemptive scheduling (i.e., $\mathcal{J}$ contains only higher-or-equal-priority jobs). However, the introduced definitions are constructed to support both fully preemptive and non-preemptive execution. Unless stated otherwise, we do not distinguish between the preemptive and non-preemptive cases. However, if non-preemptive execution is assumed, then we assume that all processors are always available. We discuss other differences in the analysis of the non-preemptive case in Section 6.3 and leave the consideration of non-preemptivity in the face of partial availability to future work.

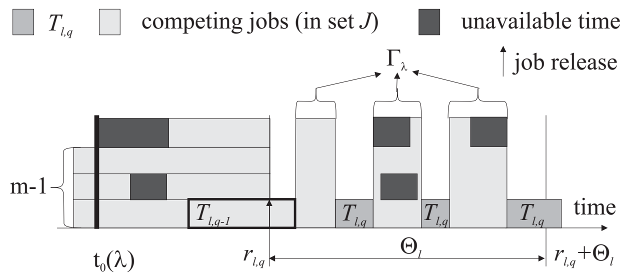


Fig. 5: Conditions for a response-time bound violation for $\lambda = 1$.

Definition 13 generalizes the well-known concept of an *idle instant* in uniprocessor scheduling with respect to jobs in $\mathcal{J}$, as illustrated in Figure 5, which shows the response-time bound violation for job $T_{\ell,q}$ assuming $\lambda = 1$.

Our schedulability test for task T_ℓ is based upon summing the demand of competing jobs as defined above in Definition 14 executing within the interval $[t_0(\lambda), r_{\ell,q} + \Theta_\ell)$, which has length $r_{\ell,q} - t_0(\lambda) + \Theta_\ell$, and the unavailable time within this interval (see Figure 5).

Definition 15. Let $\mathsf{A}_{\mathcal{J}}(T_i, I) = \sum_{T_{i,y} \in \mathcal{J}} \mathsf{A}(T_{i,y}, I)$ be the allocation of task T_i's jobs in $\mathcal{J}$ over a set of intervals I.

Definition 16. Let $\mathsf{Res}_h(I)$ be the amount of time that is not available on processor h at time instants in the set of intervals I.

Definition 17. We call a processor $\mathcal{J}$*-busy* at time t if it executes a job in $\mathcal{J}$ or is unavailable. The total time for which processors are $\mathcal{J}$-busy within a set of intervals I is called the $\mathcal{J}$*-allocation* for I and is defined as $\widehat{\mathsf{A}_{\mathcal{J}}}(I) = \sum_{T_i \in \tau} \mathsf{A}_{\mathcal{J}}(T_i, I) + \sum_{h=1}^{m} \mathsf{Res}_h(I)$.

The following definition is used to calculate $\widehat{\mathsf{A}_{\mathcal{J}}}([t_0(\lambda), r_{\ell,q} + \Theta_\ell))$.

Definition 18. Let $\Gamma_\lambda \subseteq [r_{\ell,q-\lambda+1}, r_{\ell,q} + \Theta_\ell)$ be the set of intervals where all processors are $\mathcal{J}$-busy as shown in Figure 5. Let $\overline{\Gamma_\lambda} = [r_{\ell,q-\lambda+1}, r_{\ell,q} + \Theta_\ell) \setminus \Gamma_\lambda$. We let $|\Gamma_\lambda|$ (respectively, $|\overline{\Gamma_\lambda}|$) denote the total length of the intervals in Γ_λ (respectively, $\overline{\Gamma_\lambda}$).

We next calculate the $\mathcal{J}$-allocations for Γ_λ, $[t_0(\lambda), r_{\ell,q-\lambda+1})$, and $\overline{\Gamma_\lambda}$. Because all processors are $\mathcal{J}$-busy within Γ_λ, $\widehat{\mathsf{A}_{\mathcal{J}}}(\Gamma_\lambda) = m \cdot |\Gamma_\lambda|$. We now consider the interval $[t_0(\lambda), r_{\ell,q-\lambda+1})$.

Claim 4. *All processors are $\mathcal{J}$-busy within $[t_0(\lambda), r_{\ell,q-\lambda+1})$; that is, $\widehat{\mathsf{A}_{\mathcal{J}}}([t_0(\lambda), r_{\ell,q-\lambda+1})) = m \cdot (r_{\ell,q-\lambda+1} - t_0(\lambda))$.*

Proof. Suppose that a processor is not $\mathcal{J}$-busy at time $t' \in [t_0(\lambda), r_{\ell,q-\lambda+1})$. Then it is either available and idle or executes a job that is not in $\mathcal{J}$. If the processor is idle at time t', then, because the scheduler being analyzed is work-conserving, all tasks in $\tau_p(t)$ execute at time t' and thus $|\tau_p(t')| \leq m - 1$. Thus, by (18), IS_HP(t') is *false*, which violates Definition 13. Alternatively, if, at time t', the processor executes a job $T_{x,y} \notin \mathcal{J}$, then, by Definition 14, $T_{x,y} \succ T_{\ell,q}$ and, by (18), IS_HP(t') is *false*, which also violates Definition 13. The given expression for $\widehat{\mathsf{A}_{\mathcal{J}}}([t_0(\lambda), r_{\ell,q-\lambda+1}))$ follows. □

Finally, we consider the interval set $\overline{\Gamma_\lambda}$.

Claim 5. *Task T_ℓ executes at each time $t \in \overline{\Gamma_\lambda}$, and hence, $\mathsf{A}_{\mathcal{J}}(T_\ell, \overline{\Gamma_\lambda}) = |\overline{\Gamma_\lambda}|$.*

Proof. Suppose to the contrary that T_ℓ does not execute at some time $t \in \overline{\Gamma_\lambda}$. By Definition 18, there exists processor P that is not $\mathcal{J}$-busy at time t. By Claim 3, task T_ℓ is ready at each time $t \in [r_{\ell,q-\lambda+1}, r_{\ell,q} + \Theta_\ell)$. If P is idle, then, because the scheduler is work-conserving, all tasks in $\tau_p(t)$, including T_ℓ execute at time t. If P executes job $T_{x,y} \notin \mathcal{J}$, then, by Definition 14 and (18), IS_HP(t) is *false*, which implies that all tasks in $\tau_p(t)$, including T_ℓ execute at time t. In either case, we have a contradiction. □

Lemma 3: $\widehat{\mathsf{A}_{\mathcal{J}}}([t_0(\lambda), r_{\ell,q} + \Theta_\ell)) \geq m \cdot (r_{\ell,q-\lambda+1} - t_0(\lambda)) + m \cdot |\Gamma_\lambda| + |\overline{\Gamma_\lambda}|$.

Proof. We sum up the $\mathcal{J}$-allocations for intervals $[t_0(\lambda), r_{\ell,q-\lambda+1})$, Γ_λ, and $\overline{\Gamma_\lambda}$ (see Figure 5; note that $r_{\ell,q-\lambda+1} = r_{\ell,q}$ here).

$$\begin{aligned}
&\widehat{\mathsf{A}_{\mathcal{J}}}([t_0(\lambda), r_{\ell,q} + \Theta_\ell)) \\
&= \widehat{\mathsf{A}_{\mathcal{J}}}([t_0(\lambda), r_{\ell,q-\lambda+1})) + \widehat{\mathsf{A}_{\mathcal{J}}}(\Gamma_\lambda) + \widehat{\mathsf{A}_{\mathcal{J}}}(\overline{\Gamma_\lambda}) \\
&\quad \{\text{by Claim 4 and Definition 18}\} \\
&= m \cdot (r_{\ell,q-\lambda+1} - t_0(\lambda)) + m \cdot |\Gamma_\lambda| + \widehat{\mathsf{A}_{\mathcal{J}}}(\overline{\Gamma_\lambda}) \\
&\quad \{\text{by Definition 17}\} \\
&\geq m \cdot (r_{\ell,q-\lambda+1} - t_0(\lambda)) + m \cdot |\Gamma_\lambda| + \mathsf{A}_{\mathcal{J}}(T_\ell, \overline{\Gamma_\lambda}) \\
&\quad \{\text{by Claim 5}\} \\
&= m \cdot (r_{\ell,q-\lambda+1} - t_0(\lambda)) + m \cdot |\Gamma_\lambda| + |\overline{\Gamma_\lambda}|
\end{aligned}$$
□

The values of $|\Gamma_\lambda|$ and $|\overline{\Gamma_\lambda}|$ depend on the amount of competing work due to $T_{\ell,q}$'s predecessors (including $T_{\ell,q}$ itself), which is determined as follows.

Definition 19. Let $W(T_{i,y}, t)$ denote the remaining execution time for job $T_{i,y}$ (if any) at time t. Let $\mathsf{W}_{\mathcal{J}}(T_i, t) = \sum_{T_{i,y} \in \mathcal{J}} W(T_{i,y}, t)$. In Figure 5, $\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1})$ corresponds to the execution demand of job $T_{\ell,q}$ and the unfinished work of job $T_{\ell,q-1}$ at time $r_{\ell,q}$.

Claim 6. (Proved in an appendix.) $\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) \leq r_{\ell,q} + \Theta_\ell - r_{\ell,q-\lambda+1}$.

The following lemma establishes constraints on the total length of the intervals Γ_λ and $\overline{\Gamma_\lambda}$.

Lemma 4. *If the response-time bound for $T_{\ell,q}$ is violated* (*as we have assumed*)*, then $|\Gamma_\lambda| = r_{\ell,q} + \Theta_\ell - r_{\ell,q-\lambda+1} - \mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) + 1 + \mu$, where $\mu \geq 0$.* (*Note that, by* Claim 6, *this implies that $|\Gamma_\lambda| > 0$*)*. Additionally, $|\overline{\Gamma_\lambda}| = \mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) - 1 - \mu$.*

Proof. Suppose, contrary to the statement of the lemma, that the response-time bound for $T_{\ell,q}$ is violated and $\mu < 0$, i.e.,

$$|\Gamma_\lambda| < r_{\ell,q} + \Theta_\ell - r_{\ell,q-\lambda+1} - \mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) + 1. \tag{19}$$

Under these conditions, the total length of the intervals in $\overline{\Gamma_\lambda}$, where at least one available processor is not $\mathcal{J}$-busy, is $r_{\ell,q}+\Theta_\ell-r_{\ell,q-\lambda+1}-|\Gamma_\lambda| \overset{\{\text{by (19)}\}}{>} \mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1})-1$. Thus, this total length is at least $\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1})$, as time is integral. By Claim 5, job $T_{\ell,q}$ or one of its predecessors executes at each time $t \in \overline{\Gamma_\lambda}$. Thus, job $T_{\ell,q}$ completes by time $r_{\ell,q} + \Theta_\ell$, which is a contradiction. Hence, $\mu \geq 0$. $|\overline{\Gamma_\lambda}|$ can be found as $|\overline{\Gamma_\lambda}| = r_{\ell,q} + \Theta_\ell - r_{\ell,q-\lambda+1} - |\Gamma_\lambda| = \mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) - 1 - \mu$. $\square$

In the statement of Theorem 3, which defines a schedulability condition, the functions defined below are used.

Definition 20. Let $E_\ell^*(k)$ be a finite function of k such that $\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) \leq E_\ell^*(\lambda)$.

Definition 21. Let $M_\ell^*(\delta, \tau, m)$ be a finite function of δ, τ, and m such that

$$\sum_{T_i \in \tau} \mathsf{A}_{\mathcal{J}}(T_i, [t_0(\lambda), r_{\ell,q} + \Theta_\ell)) \leq M_\ell^*(r_{\ell,q} - t_0(\lambda), \tau, m).$$

(As mentioned earlier at the beginning of Section 6, $M_\ell^*(\delta, \tau, m)$ and $E_\ell^*(k)$ are calculated in order to test whether the response-time bound of T_ℓ is not violated. Later, in Section 6.2, we explain how $M_\ell^*(\delta, \tau, m)$ and $E_\ell^*(k)$ are calculated.)

Definition 22. We require that there exists a constant $H_\ell \geq 0$ such that, for all $\delta \geq 0$,

$$M_\ell^*(\delta, \tau, m) \leq U_{sum} \cdot \delta + H_\ell. \tag{20}$$

This requirement is reasonable because the growth rate of the total demand over the interval of interest, which has length $\delta + \Theta_\ell$, cannot be larger than the total long-term utilization of the tasks in τ for large values of δ. This also allows us to upper-bound our test's computational complexity. Henceforth, we omit the last two arguments of M_ℓ^*.

Definition 23. Let $\delta_\ell^{\max}(k) = \left\lfloor (H_\ell + (m-1)\cdot(E_\ell^*(k)-1) + \widehat{U}\cdot\sigma_{tot} - \Theta_\ell \cdot \widehat{U})/(\widehat{U} - U_{sum}) \right\rfloor$.

We next calculate an upper bound on $\mathsf{Res}_h([t_0(\lambda), r_{\ell,q} + \Theta_\ell))$. For processor h and the interval $[t_0(\lambda), r_{\ell,q} + \Theta_\ell)$, by Definition 5,

$$\mathsf{Res}_h([t_0(\lambda), r_{\ell,q} + \Theta_\ell)) = (r_{\ell,q} - t_0(\lambda) + \Theta_\ell) - \mathsf{supply}_h(t_0(\lambda), r_{\ell,q} - t_0(\lambda) + \Theta_\ell). \tag{21}$$

Summing (21) for all h, we have

$$\begin{aligned}
&\sum_{h=1}^{m} \mathsf{Res}_h([t_0(\lambda), r_{\ell,q}+\mathit{\Theta}_\ell)) \\
&\quad = \sum_{h=1}^{m} \big((r_{\ell,q}-t_0(\lambda)+\mathit{\Theta}_\ell) - \mathsf{supply}_h(t_0(\lambda), r_{\ell,q}-t_0(\lambda)+\mathit{\Theta}_\ell)\big) \\
&\qquad \{\text{by Definition 5}\} \\
&\quad = m\cdot(r_{\ell,q}-t_0(\lambda)+\mathit{\Theta}_\ell) - \mathsf{Supply}(t_0(\lambda), r_{\ell,q}-t_0(\lambda)+\mathit{\Theta}_\ell) \\
&\qquad \{\text{by Definition 6}\} \\
&\quad \leq m\cdot(r_{\ell,q}-t_0(\lambda)+\mathit{\Theta}_\ell) - \mathcal{B}(r_{\ell,q}-t_0(\lambda)+\mathit{\Theta}_\ell). \qquad (22)
\end{aligned}$$

The following theorem will be used to define our schedulability test.

Theorem 3. *If the response-time bound $\mathit{\Theta}_\ell$ is violated for $T_{\ell,q}$* (*as we have assumed*), *then, for $k=\lambda$ and some $\delta \in [\mathcal{A}_\ell^{-1}(\lambda-1), \delta_\ell^{\max}(\lambda)]$* (*such that $\delta = r_{\ell,q}-t_0(\lambda)$*),

$$M_\ell^*(\delta)+(m-1)\cdot(E_\ell^*(k)-1) \geq \mathcal{B}(\delta+\mathit{\Theta}_\ell). \qquad (23)$$

Proof. Consider job $T_{\ell,q}$, $k=\lambda$, and time instants $r_{\ell,q-\lambda+1}$ and $t_0(\lambda)$ as defined in Definitions 11 and 13. To establish (23), we consider the total $\mathcal{J}$-allocation within the interval $[t_0(\lambda), r_{\ell,q}+\mathit{\Theta}_\ell)$. By Definition 17 and Lemma 3,

$$\begin{aligned}
&\sum_{T_i\in\tau} \mathsf{A}_{\mathcal{J}}(T_i, [t_0(\lambda), r_{\ell,q}+\mathit{\Theta}_\ell)) + \sum_{h=1}^{m} \mathsf{Res}([t_0(\lambda), r_{\ell,q}+\mathit{\Theta}_\ell)) \\
&\quad \geq m\cdot(r_{\ell,q-\lambda+1}-t_0(\lambda)) + m\cdot|\mathit{\Gamma}_\lambda| + |\overline{\mathit{\Gamma}_\lambda}| \\
&\qquad \{\text{by Lemma 4}\} \\
&\quad = m\cdot(r_{\ell,q-\lambda+1}-t_0(\lambda))+m\cdot(r_{\ell,q}+\mathit{\Theta}_\ell-r_{\ell,q-\lambda+1}-\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1})+1+\mu) \\
&\qquad + \mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) - 1 - \mu \\
&\quad = m\cdot(r_{\ell,q}-t_0(\lambda)+\mathit{\Theta}_\ell) - (m-1)\cdot(\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1})-1) + (m-1)\cdot\mu \\
&\qquad \{\text{because } \mu \geq 0\} \\
&\quad \geq m\cdot(r_{\ell,q}-t_0(\lambda)+\mathit{\Theta}_\ell)-(m-1)\cdot(\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1})-1). \qquad (24)
\end{aligned}$$

Setting (22) into (24), we have

$$\begin{aligned}
&\sum_{T_i\in\tau} \mathsf{A}_{\mathcal{J}}(T_i, [t_0(\lambda), r_{\ell,q}+\mathit{\Theta}_\ell)) + m\cdot(r_{\ell,q}-t_0(\lambda)+\mathit{\Theta}_\ell) - \mathcal{B}(r_{\ell,q}-t_0(\lambda)+\mathit{\Theta}_\ell) \\
&\quad \geq m\cdot(r_{\ell,q}-t_0(\lambda)+\mathit{\Theta}_\ell) - (m-1)\cdot(\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1})-1).
\end{aligned}$$

Rearranging the terms in the above inequality, we have

$$\begin{aligned}
&\sum_{T_i\in\tau} \mathsf{A}_{\mathcal{J}}(T_i, [t_0(\lambda), r_{\ell,q}+\mathit{\Theta}_\ell))+(m-1)\cdot(\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1})-1) \\
&\quad \geq \mathcal{B}(r_{\ell,q}-t_0(\lambda)+\mathit{\Theta}_\ell).
\end{aligned}$$

Setting $E^*_\ell(\lambda)$ and $M^*_\ell(r_{\ell,q} - t_0(\lambda))$ as defined in Definitions 20 and 21 into the inequality above, we have

$$M^*_\ell(r_{\ell,q} - t_0(\lambda)) + (m-1)\cdot(E^*_\ell(\lambda)-1) \ge \mathcal{B}(r_{\ell,q} - t_0(\lambda) + \Theta_\ell).$$

Setting $r_{\ell,q} - t_0(\lambda) = \delta$ into the inequality above, we get (23).

Our remaining proof obligation is to establish the stated range for δ. Note that, by Definition 13, $\delta = r_{\ell,q} - t_0(\lambda) \ge r_{\ell,q} - r_{\ell,q-\lambda+1} \ge \mathcal{A}^{-1}_\ell(\lambda - 1)$, where the last inequality follows from Lemma 2. By (20) and (23), we have for $k = \lambda$,

$$U_{sum}\cdot\delta + H_\ell + (m-1)\cdot(E^*_\ell(k) - 1) \ge \mathcal{B}(\delta + \Theta_\ell). \tag{25}$$

Applying (4) to (25), we have

$$\begin{aligned} U_{sum}\cdot\delta + H_\ell + (m-1)\cdot(E^*_\ell(k) - 1) \ge\ & \max(0, \widehat{U}\cdot(\delta + \Theta_\ell - \sigma_{tot})) \\ \ge\ & \widehat{U}\cdot(\delta + \Theta_\ell - \sigma_{tot}). \end{aligned}$$

Solving the latter inequality for δ, we have $\delta \le (H_\ell + (m-1)\cdot(E^*_\ell(k)-1) + \widehat{U}\cdot\sigma_{tot} - \Theta_\ell\cdot\widehat{U})/(\widehat{U} - U_{sum})$. Because δ is integral (as $r_{\ell,q}$ and $t_0(k)$ are integral), by Definition 23, $\delta \le \delta^{\max}_\ell(k)$. The theorem follows. □

Corollary 1. (Schedulability Test) *If, for each task $T_\ell \in \tau$,* (23) *does not hold for each $k \in [1, K_\ell]$ and $\delta \in [\mathcal{A}^{-1}_\ell(k-1), \delta^{\max}_\ell(k)]$, then no response-time bound is violated.*

Proof. The corollary follows from Theorem 3 and Definition 11, which implies $\lambda \in [1, K_\ell]$. □

In Section 8, we improve the above schedulability test for fixed-job-priority preemptive schedulers such as GEDF and FIFO by replacing the term $(m-1)\cdot(E^*_\ell(k) - 1)$ in (23) with a smaller term proportional to $\max(m - F - 1, 0)\cdot E^*_\ell(k)$, where F is the number of processors that are always available (see Definition 6). This can be done because, under GEDF and FIFO, the problem job $T_{\ell,q}$ and its predecessors cannot be preempted by other jobs after a certain time point unless the competing demand carried from previous time instants is sufficiently large.

6.2 Step **S3** (Calculating $M^*_\ell(\delta)$ and $E^*_\ell(k)$)

Note that we did not make any assumptions above about how jobs are scheduled except that the jobs of each task execute sequentially and jobs are prioritized as in Definition 7. Therefore, Corollary 1 is applicable to all fixed job-priority scheduling policies (these policies include preemptive variants of EDF, FIFO, static-priority policies, and their various combinations; non-preemptive variants can be supported similarly as discussed later in Section 6.3) provided the functions $M^*_\ell(\delta)$ (and its linear upper bound in Definition 22) and $E^*_\ell(k)$ are known. $M^*_\ell(\delta)$ and $E^*_\ell(k)$ can be derived for a particular algorithm by extending techniques from previously-published papers on the schedulability of sporadic tasks (Baruah 2007; Leontyev and Anderson 2008) to incorporate more general arrival and execution patterns.

In this section, we derive the functions $E^*_\ell(k)$ and $M^*_\ell(\delta)$ for a fully preemptive prioritization scheme in which $\chi_{i,j} = r_{i,j} + D_i$, where D_i is a constant (preemptive global EDF and FIFO are the subcases of this scheme). Note that in this case the set $\mathcal{J}$ only contains jobs with higher or equal priority than that of $T_{\ell,q}$. We first prove some properties about jobs in the set $\mathcal{J}$.

Definition 24. Let $C_{i,k} = D_i - D_k$.

Lemma 5. *If $T_{\ell,q}$ violates its response-time bound and job $T_{a,b}$ is in $\mathcal{J}$, then $T_{a,b} \preceq T_{\ell,q}$ and $r_{a,b} \leq r_{\ell,q} + C_{\ell,a}$.*

Proof. Consider job $T_{a,b} \in \mathcal{J}$.

Case 1: $T_{a,b} \preceq T_{\ell,q}$**.** By Definition 7, $r_{a,b} + D_a \leq r_{\ell,q} + D_\ell$. The required result follows from Definition 24.

Case 2: $T_{a,b} \succ T_{\ell,q}$**.** By Definition 14, $T_{a,b}$ executes at some time $t \in [t_0(\lambda), r_{\ell,q} + \Theta_\ell)$ and IS_HP(t) holds. By (18), since $T_{a,b}$ executes at time t, there exists task $T_x \in \tau_p(t)$ such that job $T_{x,y}$ is ready at t, $T_{x,y} \preceq T_{\ell,q} \prec T_{a,b}$ and $T_{x,y}$ does not execute at t. This contradicts the assumption of full preemptivity. □

Derivation of $M^*_\ell(\delta)$**.** To derive $M^*_\ell(\delta)$, we first note that, by Lemma 5, only jobs $T_{a,b} \preceq T_{\ell,q}$ belong to $\mathcal{J}$ and can compete with $T_{\ell,q}$ or its predecessors.

Definition 25. Let T_{h,b_h} be the earliest pending job of T_h at time $t_0(k)$. We separate the tasks that may compete with $T_{\ell,q}$ into two disjoint sets:

$$\mathbf{HC} = \{T_h :: (T_{h,b_h} \text{ exists}) \wedge (r_{h,b_h} < t_0(k)) \wedge (T_{h,b_h} \in \mathcal{J})\};$$
$$\mathbf{NC} = \{T_h :: (r_{h,b_h} \geq t_0(k)) \wedge (T_{h,b_h} \in \mathcal{J})\}.$$

Here, **HC** denotes "high-priority carry-in" and **NC** denotes "non-carry-in".

Claim 7: $|\mathbf{HC}| \leq m - 1$.

Proof. By Definitions 12 and 25, $\mathbf{HC} \subseteq \tau_p(t_0(k) - 1)$. By Definition 13, all tasks in $\tau_p(t_0(k) - 1)$ execute at $t_0(k) - 1$ and $|\tau_p(t_0(k) - 1)| \leq m - 1$. Thus, $|\mathbf{HC}| \leq m - 1$. □

Since the cumulative length of $[t_0(k), r_{\ell,q} + \Theta_\ell)$, depends on the difference $r_{\ell,q} - t_0(k)$, we use $\mathsf{A}_{\mathbf{NC}}(T_i, r_{\ell,q} - t_0(k))$ and $\mathsf{A}_{\mathbf{HC}}\ (T_i, r_{\ell,q} - t_0(k))$ to denote an upper-bound on $\mathsf{A}_{\mathcal{J}}(T_i, [t_0(k), r_{\ell,q} + \Theta_\ell))$ for the case when T_i is in **NC** and **HC**, respectively. With this notation, we have

$$\sum_{T_i \in \tau} \mathsf{A}_{\mathcal{J}}(T_i, [t_0(\lambda), r_{\ell,q} + \Theta_\ell))$$
$$\leq \sum_{T_i \in \mathbf{HC}} \mathsf{A}_{\mathbf{HC}}(T_i, r_{\ell,q} - t_0(\lambda)) + \sum_{T_i \in \mathbf{NC}} \mathsf{A}_{\mathbf{NC}}(T_i, r_{\ell,q} - t_0(\lambda)). \qquad (26)$$

We provide expressions for computing $\mathsf{A}_{\mathbf{NC}}(T_i, \delta)$ and $\mathsf{A}_{\mathbf{HC}}(T_i, \delta)$ in the following two lemmas. Their proofs can be found in the appendix.

Lemma 6: $\mathsf{A}_{\mathbf{NC}}(T_i, \delta) = \min(\delta + \Theta_\ell, \gamma_i^u(\alpha_i^+(\delta + C_{\ell,i})))$.

Definition 26. Let $G_i(S, X) = \min(\gamma_i^u(S), \max(0, X - \mathcal{A}_\ell^{-1}(S-1)) + \gamma_i^u(S-1))$.

Lemma 7: $\mathsf{A}_{\mathbf{HC}}(T_i, \delta) = \min(\delta + \Theta_\ell, G_i(\alpha_i^u(\delta + C_{\ell,i} + \Theta_i), \delta + C_{\ell,i} + \Theta_i))$.

To continue our derivation of $M_\ell^*(\delta)$, we set

$$M_\ell^*(\delta) = \max\left(\sum_{T_i \in \mathbf{HC}} \mathsf{A}_{\mathbf{HC}}(T_i, \delta) + \sum_{T_i \in \mathbf{NC}} \mathsf{A}_{\mathbf{NC}}(T_i, \delta)\right), \tag{27}$$

where $\max$ is taken over each choice of **HC** and **NC** subject to the following constraints.

$$\mathbf{NC} \cup \mathbf{HC} \subseteq \tau \wedge \mathbf{NC} \cap \mathbf{HC} = \emptyset \wedge |\mathbf{HC}| \leq m - 1 \tag{28}$$

The constraint $|\mathbf{HC}| \leq m - 1$ follows from Claim 7. It is easy to check that $0 \leq \mathsf{A}_{\mathbf{NC}}(T_i, \delta)$ and $0 \leq \mathsf{A}_{\mathbf{HC}}(T_i, \delta)$ for each $\delta \geq 0$. Thus, the sets maximizing the value $M_\ell^*(\delta)$ can be found by adding at most $m - 1$ tasks with the largest positive value of $\mathsf{A}_{\mathbf{HC}}(T_i, \delta) - \mathsf{A}_{\mathbf{NC}}(T_i, \delta)$ to **HC** and adding the remaining tasks to **NC**.

By the selection of λ in Definition 11, (26), and (27), $M_\ell^*(r_{\ell,q} - t_0(\lambda))$ upper-bounds $\sum_{T_i \in \tau} \mathsf{A}_{\mathcal{J}}(T_i, [t_0(\lambda), r_{\ell,q} + \Theta_\ell))$ so it complies with Definition 21. In order to use Corollary 1, we are left with finding a constant H_ℓ such that (20) holds, so that $M_\ell^*(\delta)$ given by (27) complies with Definition 22.

Definition 27. Let $L_i(X) = \max(0, u_i \cdot X + \overline{e_i} \cdot B_i) + v_i$ for any X.

Lemma 8. (Proved in the appendix) *For all $\delta \geq 0$, $M_\ell^*(\delta) \leq U_{sum} \cdot \delta + H_\ell$, where $H_\ell = \sum_{T_i \in \tau} L_i(C_{\ell,i}) + U(m-1) \cdot \max(\Theta_i)$ and $U(y)$ is the sum of $\min(y, |\tau|)$ largest utilizations.*

We finally briefly discuss how $E_\ell^*(k)$ can be calculated.

Definition 28. Let $Q_\ell(k) = \max(0, \gamma_\ell^u(k-1) - 1) + \Theta_\ell$.

We set $E_\ell^*(k)$ as follows.

$$E_\ell^*(k) = G_\ell(\alpha_\ell^u(Q_\ell(k)), Q_\ell(k)) \tag{29}$$

In the lemma below, we show that $E_\ell^*(k)$ given by (29) complies with Definition 20.

Lemma 9. (Proved in the appendix) *If $E_\ell^*(k)$ is given by (29), then $E_\ell^*(\lambda) \geq \mathsf{W}_{\mathcal{J}}(T_i, r_{\ell,q-\lambda+1})$.*

Using an expression for H_ℓ given by Lemma 8, we can compute $\delta_\ell^{\max}(k)$ in Definition 23 for any given k. Given expressions for $\delta_\ell^{\max}(k)$, $M_\ell^*(\delta)$, and $E_\ell^*(k)$, we can apply Corollary 1 to check that each task $T_\ell \in \tau$ meets its response-time bound. In Section 7, we identify conditions under which the test is applicable and discuss its time complexity.

6.3 Analysis of Non-Preemptive Execution

As mentioned earlier, Corollary 1 is applicable if non-preemptive execution is allowed as well, provided the functions $M_\ell^*(\delta)$ (and its linear upper bound in Definition 22) and $E_\ell^*(k)$ are known. Additionally, all processors have to be fully available to tasks in τ because the semantics of non-preemptivity is not well-defined if a processor that executes a task in τ becomes unavailable. The derivation of $M_\ell^*(\delta)$ and $E_\ell^*(k)$ for the non-preemptive case would be similar to the procedures described above with the exception that $\mathcal{J}$ now may contain some jobs $T_{i,y} \succ T_{\ell,q}$.

7 Computational Complexity of the Test

According to Corollary 1, (23) needs to be checked for violation for all $k \in [1, K_\ell]$ and $\delta \in [\mathcal{A}_\ell^{-1}(k-1), \delta_\ell^{\max}(k)]$.

Theorem 4. *The time complexity of the presented test is pseudo-polynomial if there exists a constant c such that $U_{sum} \leq c < \widehat{U}$.*

Proof. We start with estimating the complexity of checking (23). The values of $\alpha_i^u(\Delta)$, $\gamma_i^u(k)$, $\mathcal{A}_i^{-1}(k)$, and $\mathcal{B}(\Delta)$ can be computed in constant time if $\alpha_i^u(\Delta)$, $\gamma_i^u(k)$, and $\mathcal{B}(\Delta)$ consist of an aperiodic and periodic piecewise-linear parts. These assumptions are used in prior work on the Real-Time Calculus Toolbox (Wandeler and Thiele 2006) and are sufficient for practical purposes. Under these assumptions, $M_\ell^*(\delta)$ for a given value of δ can be computed in $O(n)$ time, where n is the number of tasks, in two steps. First, for all tasks T_i, we calculate the values $\mathsf{A}_{\mathbf{HC}}(T_i, \delta)$ and $\mathsf{A}_{\mathbf{NC}}(T_i, \delta)$ in $O(n)$ time. Second, we calculate $\sum_{T_i \in \tau} \mathsf{A}_{\mathbf{NC}}(T_i, \delta)$ in $O(n)$ time. Third, we select at most $m-1$ largest positive values of $\mathsf{A}_{\mathbf{HC}}(T_i, \delta) - \mathsf{A}_{\mathbf{NC}}(T_i, \delta)$ in $O(n)$ time using linear-time selection (Blum et al 1973) and add their sum to $\sum_{T_i \in \tau} \mathsf{A}_{\mathbf{NC}}(T_i, \delta)$. The cost of checking (23) is thus $O(n)$.

For each task T_ℓ, the inequality (23) needs to be checked for all $k \in [1, K_\ell]$ and all integers in $[\mathcal{A}_\ell^{-1}(k-1), \delta_\ell^{\max}(k)]$. By Definition 23, $\delta_\ell^{\max}(k)$ is finite if its denominator is nonzero. By (5), we have $U_{sum} \leq \widehat{U}$. Therefore, $\delta_\ell^{\max}(k)$ is finite if (5) is strict. Overall, (23) has to be checked at most $n \cdot \max_{T_\ell \in \tau}(K_\ell \cdot \max_{k \leq K_\ell}(\delta_\ell^{\max}(k)))$ times, which implies the pseudo-polynomial time complexity. $\square$

Checking that (23) is violated for each integral value in $[\mathcal{A}_\ell^{-1}(k-1), \delta_\ell^{\max}(k)]$ can be computationally expensive. A fixed-point iterative technique can instead be applied so that only a (potentially small) subset of $[\mathcal{A}_\ell^{-1}(k-1), \delta_\ell^{\max}(k)]$ is checked. In essence, we skip intervals where (23) does not hold. A similar technique was used by Zhang and Burns (2008) for checking schedulability under uniprocessor EDF. The important difference is that our procedure does not rely on the assumptions of the sporadic task model and is applicable in multiprocessor systems.

In Definition 30 below, we define a sequence of values δ within the interval $[\mathcal{A}_\ell^{-1}(k-1), \delta_\ell^{\max}(k)]$ that need to be examined in order to check for a violation of (23) within this interval. We assume that $\mathcal{A}_\ell^{-1}(k-1) \leq \delta_\ell^{\max}(k)$, for otherwise (23) does not hold trivially. We will need an additional definition below.

Definition 29. Let $\mathcal{B}^{-1}(y) = \inf\{\varDelta \mid \mathcal{B}(\varDelta) > y\}$ be the pseudo-inverse function of the total processing capacity of the system.

Example 9. In Example 5, $\mathcal{B}^{-1}(2) = \inf\{\varDelta \mid \mathcal{B}(\varDelta) > 2\} = 5$.

Definition 30. Let $\xi(\delta) = \left\lfloor \mathcal{B}^{-1}(M^*_\ell(\delta) + (m-1)\cdot(E^*_\ell(k)-1))\right\rfloor - \varTheta_\ell$. Let $\{x^{[n]}\}$ be the sequence such that $x^{[n+1]} := \xi(x^{[n]})$ and $x^{[1]} = \delta^{\max}_\ell(k)$.

Because, by Definition 20, $E^*_\ell(k)$ upper-bounds a positive variable (which includes the demand of the problem job $T_{\ell,q}$) and, by Definition 21, $M^*_\ell(\delta)$ upper-bounds a non-negative variable, $M^*_\ell(\delta) + (m-1)\cdot(E^*_\ell(k)-1$ is non-negative for each δ. Therefore, $\mathcal{B}^{-1}(M^*_\ell(\delta) + (m-1)\cdot(E^*_\ell(k)-1))$ (and in turn $\xi(\delta)$) is well-defined for each δ. We henceforth assume that

(L) (23) does not hold for $\delta = \delta^{\max}_\ell(k) = x^{[1]}$.

Otherwise, the test in Corollary 1 fails trivially, when the first evaluation interval is considered.

Claim 8. *$\xi(x)$ is a non-decreasing function of x.*

Proof. The claim follows from the fact that $M^*_\ell(\delta)$ and $\mathcal{B}^{-1}(Y)$ are non-decreasing functions of their arguments. □

Lemma 10: $\xi(x^{[1]}) \le x^{[1]}$.

Proof. Consider $Z_1 = \mathcal{B}^{-1}(M^*_\ell(x^{[1]}) + (m-1)\cdot(E^*_\ell(k)-1))$. By Definition 29,

$$\begin{aligned} Z_1 =\ & \inf\{\varDelta \mid \mathcal{B}(\varDelta) > M^*_\ell(x^{[1]}) + (m-1)\cdot(E^*_\ell(k)-1)\} \\ & \quad \{\text{from (L), we have } M^*_\ell(x^{[1]}) + (m-1)\cdot(E^*_\ell(k)-1) < \mathcal{B}(x^{[1]}+\varTheta_\ell)\} \\ & \le x^{[1]} + \varTheta_\ell. \end{aligned}$$

From the inequality above, we have

$$\begin{aligned} x^{[1]} & \ge Z_1 - \varTheta_\ell \\ & \quad \{\text{by the definition of } Z_1\} \\ & = \mathcal{B}^{-1}(M^*_\ell(x^{[1]}) + (m-1)\cdot(E^*_\ell(k)-1)) - \varTheta_\ell \\ & \ge \left\lfloor \mathcal{B}^{-1}(M^*_\ell(x^{[1]}) + (m-1)\cdot(E^*_\ell(k)-1))\right\rfloor - \varTheta_\ell \\ & = \xi(x^{[1]}). \end{aligned}$$

□

Lemma 11. *$x^{[n+1]} \le x^{[n]}$ for each n.*

Proof. **Base case:** $n = 1$**.** Because, by Definition 30, $x^{[2]} = \xi(x^{[1]})$, the required result immediately follows from Lemma 10.

Induction step: $n > 1$**.** By the induction hypothesis, $x^{[n]} \le x^{[n-1]}$. By Claim 8, we have $\xi(x^{[n]}) \le \xi(x^{[n-1]})$. By Definition 30, this implies $x^{[n+1]} \le x^{[n]}$. □

We next prove an auxiliary lemma.

Lemma 12. *If $y > \mathcal{B}^{-1}(y_0)$, then $\mathcal{B}(y) > y_0$.*

Proof. Let $y^* = \inf\{\varDelta \mid \mathcal{B}(\varDelta) > y_0\}$. This implies that

$$\mathcal{B}(y) \leq y_0 \text{ for each } y < y^*. \tag{30}$$

We now consider two cases.

Case 1: $\mathcal{B}(y^*) > y_0$**.** Because $\mathcal{B}(\varDelta)$ is non-decreasing, by the condition of the case, for $y > y^*$, $\mathcal{B}(y) \geq \mathcal{B}(y^*) > y_0$.

Case 2: $\mathcal{B}(y^*) = y_0$**.** Suppose, contrary to the statement of the lemma, that there exists $y' > y^*$ such that $\mathcal{B}(y') \leq y_0$. Then, because $\mathcal{B}(\varDelta)$ is non-decreasing, $\mathcal{B}(\varDelta) \leq y_0$ for each $\varDelta \in [y^*, y']$, and hence, by (30), $\mathcal{B}(\varDelta) \leq y_0$ for each $\varDelta \leq y'$. Therefore, y^* is not an infimum for the set where $\mathcal{B}(\varDelta) > y_0$, which contradicts the definition of y^*. □

Lemma 13. *If $x^{[n+1]} < x^{[n]}$, then (23) does not hold for each non-negative integral $\delta \in (x^{[n+1]}, x^{[n]}]$.*

Proof. Consider a non-negative $\delta \in (x^{[n+1]}, x^{[n]}]$. We first lower-bound $\delta + \varTheta_\ell$ as follows.

$$\begin{aligned}
\delta + \varTheta_\ell &> x^{[n+1]} + \varTheta_\ell \\
&\quad \{\text{because } x^{[n+1]} = \xi(x^n), \text{ by Definition 30}\} \\
&= \left\lfloor \mathcal{B}^{-1}(M_\ell^*(x^{[n]}) + (m-1)\cdot(E_\ell^*(k)-1)) \right\rfloor - \varTheta_\ell + \varTheta_\ell \\
&= \left\lfloor \mathcal{B}^{-1}(M_\ell^*(x^{[n]}) + (m-1)\cdot(E_\ell^*(k)-1)) \right\rfloor
\end{aligned}$$

Because δ and $\varTheta_\ell$ are integral, $\delta + \varTheta_\ell > \mathcal{B}^{-1}(M_\ell^*(x^{[n]}) + (m-1)\cdot(E_\ell^*(k)-1))$. By Lemma 12, the last inequality implies

$$\begin{aligned}
\mathcal{B}(\delta + \varTheta_\ell) &> M_\ell^*(x^{[n]}) + (m-1)\cdot(E_\ell^*(k)-1) \\
&\quad \{\text{by the selection of } \delta \text{ and } M^* \text{ being non-decreasing}\} \\
&\geq M_\ell^*(\delta) + (m-1)\cdot(E_\ell^*(k)-1).
\end{aligned}$$
□

The following theorem gives a method for checking (23) on the interval $[\mathcal{A}_\ell^{-1}(k-1), \delta_\ell^{\max}(k)]$ which skips sub-intervals where (23) does not hold.

Theorem 5. *Let $\{x^{[n]}\}$ be the sequence defined in Definition 30. If $x^{[n+1]} < \mathcal{A}_\ell^{-1}(k-1)$, then (23) does not hold for each integral δ within the interval $[\mathcal{A}_\ell^{-1}(k–1), \delta_\ell^{\max}(k)]$.*

Proof. The theorem follows from dividing the interval $(x^{[n+1]}, \delta_\ell^{\max}(k)]$ into subintervals $(x^{[i+1]}, x^{[i]}]$ and applying Lemma 13 to each of the subintervals. □

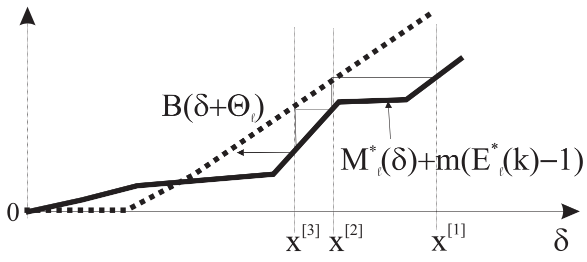


Fig. 6: Iterative process for finding δ_ℓ in Example 10.

We proved the above theorem for the case when time is integral. We defer consideration of continuous time to future work. According to Theorem 5, we can apply Corollary 1 as follows. First, we check whether (23) does not hold for $\delta = \delta_\ell^{\max}(k)$. Second, we construct the sequence $\{x^{[n]}\}$ as defined in Definition 30. If a fixed point $x^{[n]} = x^{[n+1]}$ is not found in the interval $[\mathcal{A}_\ell^{-1}(k-1), \delta_\ell^{\max}(k)]$, then, by Theorem 5, (23) does not hold for each $\delta \in [\mathcal{A}_\ell^{-1}(k-1), \delta_\ell^{\max}(k)]$. If such a fixed point is found, then we conservatively claim that the response-time bound Θ_ℓ is violated.

Example 10. The iteration process described above can be illustrated graphically. Figure 6 shows two functions of δ: $\mathcal{B}(\delta+\Theta_\ell)$ and $M_\ell^*(\delta)+(m-1)\cdot(E_\ell^*(k)-1)$, which are depicted with bold dotted and solid lines, respectively. The iteration process starts with $x^{[1]} = \delta_\ell^{\max}(k)$. At this point, $M_\ell^*(x^{[1]})+(m-1)\cdot(E_\ell^*(k)-1) < \mathcal{B}(x^{[1]}+\Theta_\ell)$. The next step is to set $x^{[2]} = \xi(x^{[1]})$ as shown. Similarly, $x^{[3]}$ is computed. The process continues until a fixed point is found or $x^{[n+1]} < \mathcal{A}_\ell^{-1}(k-1)$ holds. Thus, the iterations skip portions of the interval $[\mathcal{A}_\ell^{-1}(k-1), \delta_\ell^{\max}(k)]$ where (23) is guaranteed to fail.

8 Schedulability Test for GEDF-like Schedulers

In this section, we improve Inequality (23) for a prioritization scheme in which $\chi_{i,j} = r_{i,j} + D_i$, where D_i is a constant. We do this by more carefully estimating $\mathcal{J}$-allocations within the intervals $[t_0(\lambda), r_{\ell,q-\lambda+1}) \cup \Gamma_\lambda$ and $\overline{\Gamma_\lambda}$. We divide these intervals into four non-intersecting sets and estimate the $\mathcal{J}$-allocations individually within these sets in Lemmas 14–17. Using the obtained results, we establish Theorem 6, which gives a necessary condition for a response-time bound violation. This theorem is proved similarly to Theorem 3. Finally, Corollary 3 gives us an improved schedulability test for GEDF-like schedulers.

Definition 31. Let $C_\ell = \max_{T_i \in \tau}(D_\ell - D_i)$.

In Definition 32 and Lemmas 14–17 below, we assume that $\Theta_\ell > \gamma_\ell^u(\lambda) + C_\ell$ holds. In this case, we can improve Inequality (23) by replacing the term $(m-1) \cdot E_\ell^*(k)$ with a smaller term proportional to $\max(m-F-1, 0) \cdot E_\ell^*(k)$, where F is the number of fully available processors. (If $\Theta_\ell \leq \gamma_\ell^u(\lambda) + C_\ell$, then Theorem 3 can be applied to check for a response-time bound violation.)

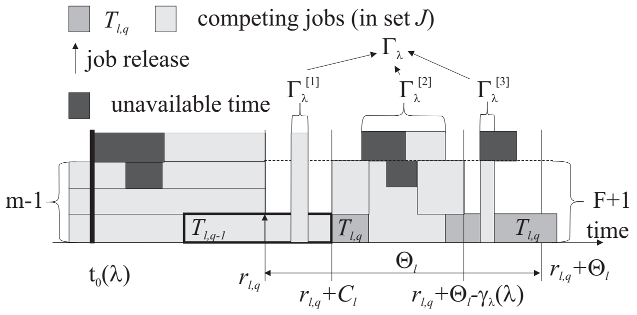


Fig. 7: Conditions for a response-time bound violation for $\lambda = 1$.

Definition 32. Let $\mathit{\Gamma}_\lambda^{[1]} = [r_{\ell,q-\lambda+1}, r_{\ell,q} + C_\ell) \cap \mathit{\Gamma}_\lambda$, $\mathit{\Gamma}_\lambda^{[2]} = [r_{\ell,q} + C_\ell, r_{\ell,q} + \mathit{\Theta}_\ell - \gamma_\ell^u(\lambda)) \cap \mathit{\Gamma}_\lambda$, and $\mathit{\Gamma}_\lambda^{[3]} = [r_{\ell,q} + \mathit{\Theta}_\ell - \gamma_\ell^u(\lambda), r_{\ell,q} + \mathit{\Theta}_\ell) \cap \mathit{\Gamma}_\lambda$, as shown in Figure 7.

Additionally, let $\overline{\mathit{\Gamma}_\lambda}^{[1]} = [r_{\ell,q-\lambda+1}, r_{\ell,q} + C_\ell) \cap \overline{\mathit{\Gamma}_\lambda}$, $\overline{\mathit{\Gamma}_\lambda}^{[2]} = [r_{\ell,q} + C_\ell, r_{\ell,q} + \mathit{\Theta}_\ell - \gamma_\ell^u(\lambda)) \cap \overline{\mathit{\Gamma}_\lambda}$, and $\overline{\mathit{\Gamma}_\lambda}^{[3]} = [r_{\ell,q} + \mathit{\Theta}_\ell - \gamma_\ell^u(\lambda), r_{\ell,q} + \mathit{\Theta}_\ell) \cap \overline{\mathit{\Gamma}_\lambda}$.

Note that, by Definition 32,

$$[t_0(\lambda), r_{\ell,q} + \mathit{\Theta}_\ell) = [t_0(\lambda), r_{\ell,q-\lambda+1}) \cup \mathit{\Gamma}_\lambda \cup \overline{\mathit{\Gamma}_\lambda}^{[1]} \cup \overline{\mathit{\Gamma}_\lambda}^{[2]} \cup \overline{\mathit{\Gamma}_\lambda}^{[3]}. \tag{31}$$

In the rest of this section we let μ be defined as in Lemma 4.

Lemma 14: $\widehat{\mathsf{A}_\mathcal{J}}([t_0(\lambda), r_{\ell,q-\lambda+1}) \cup \mathit{\Gamma}_\lambda) = m \cdot (r_{\ell,q} - t_0(\lambda) + \mathit{\Theta}_\ell) - m \cdot (\mathsf{W}_\mathcal{J}(T_\ell, r_{\ell,q-\lambda+1}) - 1) + m \cdot \mu$.

Proof. By Definition 17, we have

$$\begin{aligned}
&\widehat{\mathsf{A}_\mathcal{J}}([t_0(\lambda), r_{\ell,q-\lambda+1}) \cup \mathit{\Gamma}_\lambda) \\
&= \widehat{\mathsf{A}_\mathcal{J}}([t_0(\lambda), r_{\ell,q-\lambda+1})) + \widehat{\mathsf{A}_\mathcal{J}}(\mathit{\Gamma}_\lambda) \\
&\quad \{\text{by Definition 18 and Claim 4}\} \\
&= m \cdot (r_{\ell,q-\lambda+1} - t_0(\lambda)) + m \cdot |\mathit{\Gamma}_\lambda| \\
&\quad \{\text{by Lemma 4}\} \\
&= m \cdot (r_{\ell,q-\lambda+1} - t_0(\lambda)) + m \cdot (r_{\ell,q} + \mathit{\Theta}_\ell - r_{\ell,q-\lambda+1} - \mathsf{W}_\mathcal{J}(T_\ell, r_{\ell,q-\lambda+1}) + 1 + \mu) \\
&= m \cdot (r_{\ell,q} - t_0(\lambda) + \mathit{\Theta}_\ell) - m \cdot (\mathsf{W}_\mathcal{J}(T_\ell, r_{\ell,q-\lambda+1}) - 1) + m \cdot \mu.
\end{aligned}$$ □

Lemma 15: $\widehat{\mathsf{A}_\mathcal{J}}(\overline{\mathit{\Gamma}_\lambda}^{[1]}) \geq r_{\ell,q} + C_\ell - r_{\ell,q-\lambda+1} - |\mathit{\Gamma}_\lambda^{[1]}|$.

Proof. By Definitions 18 and 32, $\overline{\mathit{\Gamma}_\lambda}^{[1]} = [r_{\ell,q-\lambda+1}, r_{\ell,q} + C_\ell) \cap \overline{\mathit{\Gamma}_\lambda} = [r_{\ell,q-\lambda+1}, r_{\ell,q} + C_\ell) \setminus \mathit{\Gamma}_\lambda = [r_{\ell,q-\lambda+1}, r_{\ell,q} + C_\ell) \setminus \mathit{\Gamma}_\lambda^{[1]}$. By Claim 5, T_ℓ executes at each instant within $\overline{\mathit{\Gamma}_\lambda}$, and hence, at each instant within $[r_{\ell,q-\lambda+1}, r_{\ell,q} + C_\ell) \setminus \mathit{\Gamma}_\lambda^{[1]}$. Thus, $\mathsf{A}_\mathcal{J}(T_\ell, [r_{\ell,q-\lambda+1}, r_{\ell,q} + C_\ell) \setminus \mathit{\Gamma}_\lambda^{[1]}) = r_{\ell,q} + C_\ell - r_{\ell,q-\lambda+1} - |\mathit{\Gamma}_\lambda^{[1]}|$. The required result follows from Definition 17. □

Lemma 16: $\widehat{\mathsf{A}_{\mathcal{J}}}(\overline{\mathit{\Gamma}_\lambda}^{[3]}) \geq \gamma_\ell^u(\lambda) - |\mathit{\Gamma}_\lambda^{[3]}|$.

Proof. By Definitions 18 and 32, $\overline{\mathit{\Gamma}_\lambda}^{[3]} = [r_{\ell,q} + \mathit{\Theta}_\ell - \gamma_\ell^u(\lambda), r_{\ell,q} + \mathit{\Theta}_\ell) \cap \overline{\mathit{\Gamma}_\lambda} = [r_{\ell,q} + \mathit{\Theta}_\ell - \gamma_\ell^u(\lambda), r_{\ell,q} + \mathit{\Theta}_\ell) \setminus \mathit{\Gamma}_\lambda = [r_{\ell,q} + \mathit{\Theta}_\ell - \gamma_\ell^u(\lambda), r_{\ell,q} + \mathit{\Theta}_\ell) \setminus \mathit{\Gamma}_\lambda^{[3]}$. By Claim 5, T_ℓ executes at each instant within $\overline{\mathit{\Gamma}_\lambda}$, and hence, at each instant within $[r_{\ell,q} + \mathit{\Theta}_\ell - \gamma_\ell^u(\lambda), r_{\ell,q} + \mathit{\Theta}_\ell) \setminus \mathit{\Gamma}_\lambda^{[3]}$. Thus, $\mathsf{A}_{\mathcal{J}}(T_\ell, [r_{\ell,q} + \mathit{\Theta}_\ell - \gamma_\ell^u(\lambda), r_{\ell,q} + \mathit{\Theta}_\ell) \setminus \mathit{\Gamma}_\lambda^{[3]}) = \gamma_\ell^u(\lambda) - |\mathit{\Gamma}_\lambda^{[3]}|$. The required result follows from Definition 17. □

If $\mathit{\Gamma}_\lambda^{[3]} = \emptyset$, then, because by Definition 11, $f_{\ell,q-\lambda} \leq r_{\ell,q} + \mathit{\Theta}_\ell - \gamma_\ell^u(\lambda)$, jobs $T_{\ell,q-\lambda+1}, \ldots, T_{\ell,q}$ can execute uninterruptedly within $[r_{\ell,q} + \mathit{\Theta}_\ell - \gamma_\ell^u(\lambda), r_{\ell,q} + \mathit{\Theta}_\ell)$. As their total execution time is at most $\gamma_\ell^u(\lambda)$, $T_{\ell,q}$ will finish by $r_{\ell,q} + \mathit{\Theta}_\ell$ leading to a contradiction. We henceforth assume $|\mathit{\Gamma}_\lambda^{[3]}| > 0$.

From Lemma 5, the corollary below follows.

Corollary 2. *No job in $\mathcal{J}$ is released after $r_{\ell,q} + \max_{T_i \in \tau}(D_\ell - D_i)$.*

Proof. Consider job $T_{i,j} \in \mathcal{J}$. By Lemma 5, $r_{i,j} \leq r_{\ell,q} + C_{\ell,i}$. Thus, $r_{i,j} \leq r_{\ell,q} + D_\ell - D_i \leq r_{\ell,q} + \max_{T_i \in \tau}(D_\ell - D_i)$. □

Definition 33. Let $a = \min(F+1, m)$. (Recall that F is the number of fully available processors as defined in Definition 6.)

Lemma 17: $\widehat{\mathsf{A}_{\mathcal{J}}}(\overline{\mathit{\Gamma}_\lambda}^{[2]}) \geq a \cdot (-C_\ell - \gamma_\ell^u(\lambda) - r_{\ell,q} + r_{\ell,q-\lambda+1} + \mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) - 1 - \mu) + a \cdot (|\mathit{\Gamma}_\lambda^{[1]}| + |\mathit{\Gamma}_\lambda^{[3]}|)$.

Proof. We first note that, by Definitions 18 and 32, we have

$$\overline{\mathit{\Gamma}_\lambda}^{[2]} = [r_{\ell,q} + C_\ell, r_{\ell,q} + \mathit{\Theta}_\ell - \gamma_\ell^u(\lambda)) \setminus \mathit{\Gamma}_\lambda. \tag{32}$$

By Corollary 2 and Definition 31, nor job in $\mathcal{J}$ nor its predecessors can be released after $r_{\ell,q} + C_\ell$. If at most F available processors execute jobs in $\mathcal{J}$ at some time instant $t' \in [r_{\ell,q} + C_\ell, r_{\ell,q} + \mathit{\Theta}_\ell - \gamma_\ell^u(\lambda)) \setminus \mathit{\Gamma}_\lambda$, then at each time $t \geq t'$ all tasks in $\tau_p(t)$ with ready jobs in $\mathcal{J}$ can be accommodated using F fully available processors. By Claim 3, this implies that jobs of T_ℓ execute uninterruptedly within $[t', r_{\ell,q} + \mathit{\Theta}_\ell)$. The completion time of $T_{\ell,q}$ is thus

$$\begin{aligned}
f_{\ell,q} \leq\ & \max(t', f_{\ell,q-\lambda}) + \gamma_\ell^u(\lambda) \\
& \quad \{\text{by Definition 11}\} \\
\leq\ & \max(t', r_{\ell,q} + \mathit{\Theta}_\ell - \gamma_\ell^u(\lambda)) + \gamma_\ell^u(\lambda) \\
& \quad \{\text{by the selection of } t'\} \\
\leq\ & r_{\ell,q} + \mathit{\Theta}_\ell - \gamma_\ell^u(\lambda) + \gamma_\ell^u(\lambda) \\
=\ & r_{\ell,q} + \mathit{\Theta}_\ell,
\end{aligned}$$

leading to a contradiction.

We henceforth assume that at least $a = \min(F+1, m)$ available processors execute jobs in $\mathcal{J}$ at each time within $[r_{\ell,q} + C_\ell, r_{\ell,q} + \Theta_\ell - \gamma_\ell^u(\lambda)) \setminus \Gamma_\lambda$ (see Figure 7). Thus,

$$\begin{aligned}
\widehat{\mathsf{A}_{\mathcal{J}}}(\overline{\Gamma_\lambda}^{[2]}) &\geq a \cdot |\overline{\Gamma_\lambda}^{[2]}| \\
&\quad \{\text{by (32)}\} \\
&= a \cdot |[r_{\ell,q} + C_\ell, r_{\ell,q} + \Theta_\ell - \gamma_\ell^u(\lambda)) \setminus \Gamma_\lambda| \\
&= a \cdot (\Theta_\ell - \gamma_\ell^u(\lambda) - C_\ell - (|\Gamma_\lambda| - |\Gamma_\lambda^{[1]}| - |\Gamma_\lambda^{[3]}|)) \\
&= a \cdot (\Theta_\ell - \gamma_\ell^u(\lambda) - C_\ell - |\Gamma_\lambda|) + a \cdot (|\Gamma_\lambda^{[1]}| + |\Gamma_\lambda^{[3]}|) \\
&\quad \{\text{by Lemma 4}\} \\
&= a \cdot (\Theta_\ell - \gamma_\ell^u(\lambda) - C_\ell - (r_{\ell,q} + \Theta_\ell - r_{\ell,q-\lambda+1} \\
&\quad - \mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) + 1 + \mu)) + a \cdot (|\Gamma_\lambda^{[1]}| + |\Gamma_\lambda^{[3]}|) \\
&= a \cdot (-\gamma_\ell^u(\lambda) - C_\ell - r_{\ell,q} + r_{\ell,q-\lambda+1} \\
&\quad + \mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) - 1 - \mu) + a \cdot (|\Gamma_\lambda^{[1]}| + |\Gamma_\lambda^{[3]}|). \qquad \square
\end{aligned}$$

Claim 9: $r_{\ell,q} - r_{\ell,q-\lambda+1} \leq \max(0, \gamma_\ell^u(\lambda-1) - 1)$. (Note that this result does not depend on the scheduler being assumed.)

Proof. If $\lambda = 1$, then $r_{\ell,q} - r_{\ell,q-\lambda+1} = 0$. Alternatively, if $\lambda > 1$, then, by (12), $r_{\ell,q-\lambda+1} + \Theta_\ell \geq f_{\ell,q-\lambda+1} > r_{\ell,q} + \Theta_\ell - \gamma_\ell^u(\lambda-1)$, where the last inequality follows from Definition 11. Therefore, $r_{\ell,q} - r_{\ell,q-\lambda+1} \leq \gamma_\ell^u(\lambda-1) - 1$, as time is integral. $\square$

The following definition is used to define the schedulability test for GEDF-like schedulers in Theorem 6 and Corollary 3 below.

Definition 34. Let

$$Z_h(k) = \begin{cases} (m-1)\cdot(E_h^*(k)-1) & \text{if } \Theta_h \leq \gamma_h^u(k) + C_\ell, \\ \min\big((m-1)\cdot(E_h^*(k)-1), \\ (m-a)\cdot(E_h^*(k)-1) + (a-1)\cdot(\gamma_h^u(k) + \max(0, \gamma_h^u(k-1)-1) + C_h)\big) & \\ & \text{otherwise,} \end{cases}$$

where a is defined as in Definition 33.

Theorem 6. *If the response-time bound Θ_ℓ of $T_{\ell,q}$ is violated* (*as we have assumed*), *then for some $k = \lambda$ and δ such that $\delta \geq \mathcal{A}_\ell^{-1}(k-1)$ and $\delta \leq \lfloor (H_\ell + Z_\ell(k) + \widehat{U} \cdot \sigma_{tot} - \Theta_\ell \cdot \widehat{U})/(\widehat{U} - U_{sum}) \rfloor$,* (33) *below holds.*

$$M_\ell^*(\delta) + Z_\ell(k) \geq \mathcal{B}(\delta + \Theta_\ell), \tag{33}$$

Proof. Consider job $T_{\ell,q}$, $k = \lambda$, and time instants $r_{\ell,q-\lambda+1}$ and $t_0(\lambda)$ as defined in Definitions 11 and 13. We let $\delta = r_{\ell,q} - t_0(\lambda)$. We consider two cases.

Case 1: $\mathit{\Theta}_\ell \leq \gamma_\ell^u(\lambda) + \mathit{\Theta}_\ell$**.** By Theorem 3, (34) below holds

$$M_\ell^*(\delta) + (m-1) \cdot (E_\ell^*(\lambda) - 1) \geq \mathcal{B}(\delta + \mathit{\Theta}_\ell). \tag{34}$$

Case 2: $\mathit{\Theta}_\ell > \gamma_\ell^u(\lambda) + \mathit{\Theta}_\ell$**.** By Definition 17, we have,

$$\begin{aligned}
&\sum_{T_i \in \tau} \mathsf{A}_{\mathcal{J}}(T_i, [t_0(\lambda), r_{\ell,q} + \mathit{\Theta}_\ell)) + \sum_{h=1}^{m} \mathsf{Res}([t_0(\lambda), r_{\ell,q} + \mathit{\Theta}_\ell)) \\
&\quad = \widehat{\mathsf{A}_{\mathcal{J}}}(T_i, [t_0(\lambda), r_{\ell,q} + \mathit{\Theta}_\ell)) \\
&\qquad \{\text{by (31)}\} \\
&\quad = \widehat{\mathsf{A}_{\mathcal{J}}}([t_0(\lambda), r_{\ell,q-\lambda+1}) \cup \mathit{\Gamma}_\lambda) + \widehat{\mathsf{A}_{\mathcal{J}}}(\overline{\mathit{\Gamma}_\lambda}^{[1]}) + \widehat{\mathsf{A}_{\mathcal{J}}}(\overline{\mathit{\Gamma}_\lambda}^{[2]}) + \widehat{\mathsf{A}_{\mathcal{J}}}(\overline{\mathit{\Gamma}_\lambda}^{[3]}) \\
&\qquad \{\text{by Lemmas 14–17}\} \\
&\quad \geq m \cdot (r_{\ell,q} - t_0(\lambda) + \mathit{\Theta}_\ell) - m \cdot (\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) - 1) + m \cdot \mu \\
&\qquad + r_{\ell,q} + C_\ell - r_{\ell,q-\lambda+1} - |\mathit{\Gamma}_\lambda^{[1]}| \\
&\qquad + a \cdot (-C_\ell - \gamma_\ell^u(\lambda) - r_{\ell,q} + r_{\ell,q-\lambda+1} + \mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) - 1 - \mu) \\
&\qquad + a \cdot (|\mathit{\Gamma}_\lambda^{[1]}| + |\mathit{\Gamma}_\lambda^{[3]}|) + \gamma_\ell^u(\lambda) - |\mathit{\Gamma}_\lambda^{[3]}| \\
&\quad = m \cdot (r_{\ell,q} - t_0(\lambda) + \mathit{\Theta}_\ell) - (m-a) \cdot (\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) - 1) \\
&\qquad + (m-a) \cdot \mu + (a-1) \cdot (|\mathit{\Gamma}_\lambda^{[1]}| + |\mathit{\Gamma}_\lambda^{[3]}|) \\
&\qquad + (1-a) \cdot (\gamma_\ell^u(\lambda) + C_\ell + r_{\ell,q} - r_{\ell,q-\lambda+1}) \\
&\qquad \{\text{because } \mu \geq 0 \text{ and } |\mathit{\Gamma}_\lambda^{[1]}| + |\mathit{\Gamma}_\lambda^{[3]}| \geq 0\} \\
&\quad \geq m \cdot (r_{\ell,q} - t_0(\lambda) + \mathit{\Theta}_\ell) - (m-a) \cdot (\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) - 1) \\
&\qquad + (1-a) \cdot (\gamma_\ell^u(\lambda) + C_\ell + r_{\ell,q} - r_{\ell,q-\lambda+1}).
\end{aligned} \tag{35}$$

Setting (22) into (35), we have

$$\begin{aligned}
&\sum_{T_i \in \tau} \mathsf{A}_{\mathcal{J}}(T_i, [t_0(\lambda), r_{\ell,q} + \mathit{\Theta}_\ell)) \\
&\qquad + m \cdot (r_{\ell,q} - t_0(\lambda) + \mathit{\Theta}_\ell) - \mathcal{B}(r_{\ell,q} - t_0(\lambda) + \mathit{\Theta}_\ell) \\
&\quad \geq m \cdot (r_{\ell,q} - t_0(\lambda) + \mathit{\Theta}_\ell) - (m-a) \cdot (\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) - 1) \\
&\qquad + (1-a) \cdot (\gamma_\ell^u(\lambda) + C_\ell + r_{\ell,q} - r_{\ell,q-\lambda+1}).
\end{aligned}$$

Rearranging the terms in the above inequality, we have

$$\begin{aligned}
&\sum_{T_i \in \tau} \mathsf{A}_{\mathcal{J}}(T_i, [t_0(\lambda), r_{\ell,q} + \mathit{\Theta}_\ell)) + (m-a) \cdot (\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) - 1) \\
&\qquad + (a-1) \cdot (\gamma_\ell^u(\lambda) + C_\ell + r_{\ell,q} - r_{\ell,q-\lambda+1}) \\
&\quad \geq \mathcal{B}(r_{\ell,q} - t_0(\lambda) + \mathit{\Theta}_\ell).
\end{aligned}$$

From Claim 9, we therefore have

$$\sum_{T_i \in \tau} \mathsf{A}_{\mathcal{J}}(T_i, [t_0(\lambda), r_{\ell,q} + \Theta_\ell)) + (m-a) \cdot (\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) - 1)$$
$$+ (a-1) \cdot (\gamma_\ell^u(\lambda) + C_\ell + \max(0, \gamma_\ell^u(\lambda-1) - 1))$$
$$\geq \mathcal{B}(r_{\ell,q} - t_0(\lambda) + \Theta_\ell).$$

Setting $E_\ell^*(\lambda)$ and $M_\ell^*(r_{\ell,q} - t_0(\lambda))$ as defined in Definitions 20 and 21 into the inequality above, we get

$$M_\ell^*(r_{\ell,q} - t_0(\lambda))$$
$$+(m-a)\cdot(E_\ell^*(\lambda)-1)+(a-1)\cdot(\gamma_\ell^u(\lambda)+C_\ell+\max(0,\gamma_\ell^u(\lambda-1)-1))$$
$$\geq \mathcal{B}(r_{\ell,q} - t_0(\lambda) + \Theta_\ell).$$

Setting $\delta = r_{\ell,q} - t_0(\lambda)$ in the inequality above, we have

$$M_\ell^*(\delta)+(m-a)\cdot(E_\ell^*(\lambda)-1)+(a-1)\cdot(\gamma_\ell^u(\lambda)+C_\ell+\max(0,\gamma_\ell^u(\lambda-1)-1))$$
$$\geq \mathcal{B}(\delta + \Theta_\ell). \tag{36}$$

Additionally, (34) holds by Theorem 3. Combining (34) and (36) using Definition 34, we get (33). The stated range for δ can further be found similarly to Theorem 3. □

From Theorem 6, an improved schedulability test follows.

Corollary 3. (Improved Schedulability Test) *Let* $\delta_h^{\max}(k)' = \lfloor (H_h + Z_h(k) + \widehat{U} \cdot \sigma_{tot} - \Theta_h \cdot \widehat{U})/(\widehat{U} - U_{sum}) \rfloor$. *If, for each task* $T_h \in \tau$, $M_h^*(\delta) + Z_h(k) < \mathcal{B}(\delta + \Theta_h)$ *for each* $k \in [1, K_h]$ *and* $\delta \in [\mathcal{A}_h^{-1}(k-1), \delta_h^{\max}(k)']$, *then no response-time bound is violated.*

By Definition 34, for large values of the response-time bound Θ_h such that $\Theta_h > \gamma_h^u(k) + C_h$, $Z_h(k)$ is $\min\big((m-1)\cdot(E_h^*(k)-1), (m-a)\cdot(E_h^*(k)-1) + (a-1)\cdot(\gamma_h^u(k) + \max(0, \gamma_h^u(k-1)-1) + C_h)\big)$. This value is smaller than $(m-1)\cdot(E_h^*(k)-1)$ for large values of Θ_h because $E_h^*(k)$ is proportional to Θ_h by Lemma 9. Thus, the schedulability test given in Corollary 3 is less pessimistic for large response-time bounds than the test in Corollary 1. In the next section, we use the improved schedulability test to derive closed-form expressions for response-time bounds.

9 Closed-Form Expressions for Response-Time Bounds

Though the iterative procedure described in Section 7 can significantly reduce the time needed to check response-time bounds using Corollaries 1 and 3, the verification time can still be large if the task set is large and tasks have complex job arrival and execution-time patterns. In this section, we further reduce the computation time by deriving closed-form expressions for the response-time bounds Θ_i under GEDF-like schedulers. In prior work (Devi 2006; Leontyev and Anderson 2009a), it has been shown that GEDF (and many other schedulers) ensures a maximum response-time bound of $x + p_i + e_i^{max}$, where $x \geq 0$, for each sporadic task $T_i \in \tau$, if tasks have implicit deadlines, all processors are fully available, and $U_{sum} \leq m$. In this paper, we prove a similar result for systems specified as in Section 3. We will be seeking response-time bounds of the form $\Theta_i = x + \gamma_i^u(K_i) + C_i$, where $x > 0$, and K_i and C_i are as defined in Definitions 3 and 31. In the rest of this section, we derive x based upon the task parameters and resource availability. The derivation process is similar to finding an upper bound on δ in Theorem 3. In Lemmas 18 and 19 below, we first establish upper bounds on $E_\ell^*(k)$ and $M_\ell^*(\delta)$ as functions of x for the case when the response-time bound is a function of x. We then set the obtained expressions into the schedulability test and solve the resulting inequality for x.

Definition 35. Let $Y_\ell = L_\ell(\max(0, \gamma_\ell^u(K_\ell - 1) - 1) + \gamma_\ell^u(K_\ell) + C_\ell)$, where L is defined as in Definition 27.

Lemma 18. (Proved in the appendix) *If $\Theta_\ell = x + \gamma_\ell^u(K_\ell) + C_\ell$, then $E_\ell^*(k) \leq Y_\ell + u_\ell \cdot x$ for $k \in [1, K_\ell]$.*

Definition 36. Let $\mathcal{W}$ be the sum of $m - 1$ largest values $u_i \cdot (\gamma_i^u(K_i) + C_i)$.

Lemma 19. (Proved in the appendix) *If $\Theta_i = x + \gamma_i^u(K_i) + C_i$ for each task T_i and $\delta \geq 0$, then $M_\ell^*(\delta) \leq U_{sum} \cdot \delta + U(m-1) \cdot x + \mathcal{W} + \sum_{T_i \in \tau} L_i(C_{\ell,i})$, where $U(m-1)$ is the sum of $m - 1$ largest task utilizations.*

Theorem 7. *If $\widehat{U} - (m - a) \cdot \max(u_i) - U(m-1) > 0$ and $U_{sum} \leq \widehat{U}$, then, under a* GEDF*-like scheduler, the maximum response time of any job of T_i is at most $x + \gamma_i^u(K_i) + C_i$, where*

$$x = \max_{T_h \in \tau} \left(\frac{\mathcal{W} + \widehat{U} \cdot \sigma_{tot} + V_h + \sum_{T_i \in \tau} L_i(C_{h,i})}{\widehat{U} - (m-a) \cdot u_h - U(m-1)} \right) + 1 \tag{37}$$

and $V_h = (m-a) \cdot (Y_h - 1) + (a - 1 - \widehat{U}) \cdot (\gamma_h^u(K_h) + C_h) + (a-1) \cdot \max(0, \gamma_h^u(K_h - 1) - 1)$.

Proof. Suppose to the contrary that task T_ℓ violates its response-time bound $\Theta_\ell = x + \gamma_\ell^u(K_\ell) + C_\ell$. Because $x > 0$, and $\gamma_\ell^u(K_\ell) \geq \gamma_\ell^u(k)$ for each $k \in [1, K_\ell]$, we have

$$\Theta_\ell > \gamma_\ell^u(k) + C_\ell \text{ for each } k \in [1, K_\ell]. \tag{38}$$

By Theorem 6, for some $k \in [1, K_\ell]$ (particularly, for $k = \lambda$ as defined in Definition 11) and $\delta \geq 0$, (33) holds. (Note that $\delta \geq \mathcal{A}_\ell^{-1}(k-1)$ by Theorem 6 and $\mathcal{A}_\ell^{-1}(k-1) \geq 0$ by Definition 3.) Setting $k = \lambda$ and the bound for $\mathcal{B}$ given by (4) into (33), we have

$$M_\ell^*(\delta) + Z_\ell(\lambda) \geq \widehat{U} \cdot (\delta + \Theta_\ell - \sigma_{tot}).$$

Because $\Theta_\ell > \gamma_\ell^u(\lambda) + C_\ell$ by (38), from Definition 34 and the inequality above, we have

$$\begin{aligned} &M_\ell^*(\delta) + (m-a) \cdot (E_\ell^*(\lambda) - 1) + (a-1) \cdot (\gamma_\ell^u(\lambda) + C_\ell + \max(0, \gamma_\ell^u(\lambda - 1) - 1)) \\ &\quad \geq \widehat{U} \cdot (\delta + \Theta_\ell - \sigma_{tot}). \end{aligned}$$

By the selection of Θ_ℓ,

$$M^*_\ell(\delta)+(m-a)\cdot(E^*_\ell(\lambda)-1)+(a-1)\cdot(\gamma^u_\ell(\lambda)+C_\ell+\max(0,\gamma^u_\ell(\lambda-1)-1))$$
$$\geq \widehat{U}\cdot(\delta+x+\gamma^u_\ell(K_\ell)+C_\ell-\sigma_{tot}).$$

Setting the bounds on $E^*_\ell(\lambda)$ and $M^*_\ell(\delta)$ given by Lemmas 18 and 19 into the inequality above, we have

$$U_{sum}\cdot\delta+U(m-1)\cdot x+\mathcal{W}+\sum_{T_i\in\tau}L_i(C_{\ell,i})+(m-a)\cdot(Y_\ell(\lambda)+u_\ell\cdot x-1)$$
$$+(a-1)\cdot(\gamma^u_\ell(\lambda)+C_\ell+\max(0,\gamma^u_\ell(\lambda-1)-1))$$
$$\geq \widehat{U}\cdot(\delta+x+\gamma^u_\ell(K_\ell)+C_\ell-\sigma_{tot}).$$

Because $U_{sum}\leq\widehat{U}$ by the statement of the theorem and $\delta\geq 0$, we have

$$U(m-1)\cdot x+\mathcal{W}+\sum_{T_i\in\tau}L_i(C_{\ell,i})+(m-a)\cdot(Y_\ell(\lambda)+u_\ell\cdot x-1)$$
$$+(a-1)\cdot(\gamma^u_\ell(\lambda)+C_\ell+\max(0,\gamma^u_\ell(\lambda-1)-1))$$
$$\geq \widehat{U}\cdot(x+\gamma^u_\ell(K_\ell)+C_\ell-\sigma_{tot}).$$

After regrouping, we have

$$\mathcal{W}+\sum_{T_i\in\tau}L_i(C_{\ell,i})+(m-a)\cdot(Y_\ell(\lambda)-1)$$
$$+(a-1)\cdot(\gamma^u_\ell(\lambda)+C_\ell+\max(0,\gamma^u_\ell(\lambda-1)-1))-\widehat{U}\cdot(\gamma^u_\ell(K_\ell)+C_\ell-\sigma_{tot})$$
$$\geq x\cdot(\widehat{U}-(m-a)\cdot u_\ell-U(m-1)).$$

Solving the above inequality for x, we have

$$x\leq\frac{\mathcal{W}+\widehat{U}\cdot\sigma_{tot}+V_\ell(\lambda)+\sum_{T_i\in\tau}L_i(C_{\ell,i})}{\widehat{U}-(m-a)\cdot u_\ell-U(m-1)}, \qquad (39)$$

where $V_\ell(\lambda)=(m-a)\cdot(Y_\ell-1)+(a-1)\cdot(\gamma^u_\ell(\lambda)+C_\ell+\max(0,\gamma^u_\ell(\lambda-1)-1)-\widehat{U}\cdot(\gamma^u_\ell(K_\ell)+C_\ell)$. From Definition 33, we have $m-a\geq 0$ and $a\geq 1$. Thus, since the function $\gamma^u_h(k)$ is non-decreasing, $V_h(k)\leq V_h$, where V_h is defined in the statement of the theorem. Maximizing the right-hand side of (39) by task T_ℓ, we have

$$x\leq\max_{T_h\in\tau}\left(\frac{\mathcal{W}+\widehat{U}\cdot\sigma_{tot}+V_h+\sum_{T_i\in\tau}L_i(C_{h,i})}{\widehat{U}-(m-a)\cdot u_h-U(m-1)}\right).$$

This contradicts (37). □

The result of Theorem 7 is closely related to the results of Devi (2006) and Leontyev and Anderson (2009a), in which the maximum deadline tardiness in sporadic task systems under different schedulers is studied. In particular, the requirement $\widehat{U}-(m-a)\cdot\max(u_i)-U(m-1)$ to be positive is a sufficient condition for maximum job response times (deadline tardiness) to be bounded.

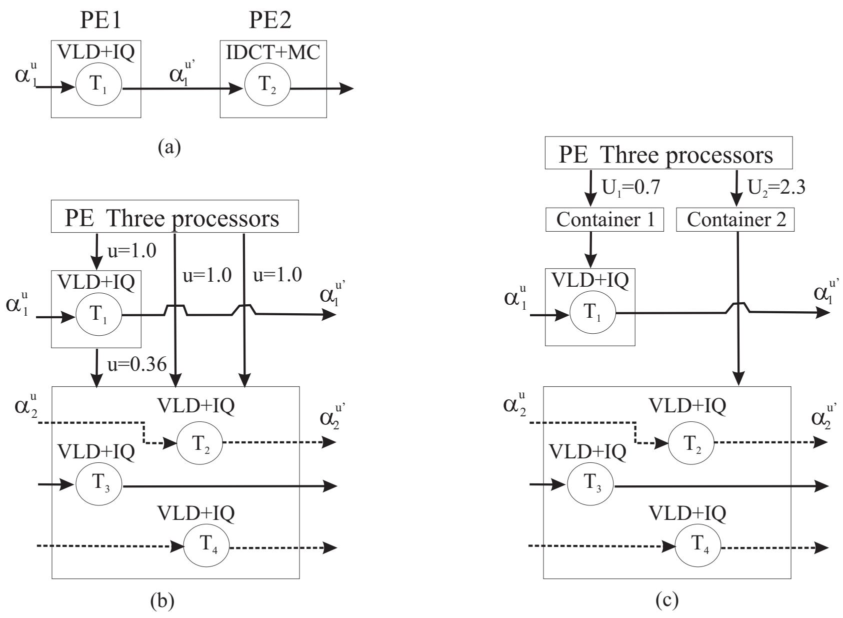


Fig. 8: **(a)** A video-processing application. Experimental setup **(b)** without and **(c)** with containers.

10 Multiprocessor Analysis: A Case Study

Our analysis can be used to derive response-time bounds for workloads that partitioning schemes cannot accommodate and for workloads that cannot be efficiently analyzed under the widely-studied periodic and sporadic models. To illustrate this, we applied our analysis to a part of a MPEG-2 video decoder application presented in Example 1 in Section 1.

Experimental setup. In our experiments, we considered two variants of the previously-studied system shown in Figure 8(a) in which PE1 is a three-processor system running four identical VLD+IQ tasks, T_1, T_2, T_3, and T_4. The two modified systems are illustrated in insets (b) and (c) of Figure 8 and explained in further detail below. For conciseness, we refer to the systems in these three insets as the (a)-, (b)-, and (c)-systems, respectively. To assess the usefulness of our analysis, we computed output curves of the four tasks so that they can be used in further analysis. We assumed zero scheduling and system overheads (the inclusion of such overheads in our analysis is beyond the scope of this paper).

The goal of our experiments was to compare different ways of implementing and analyzing the (b)- and (c)-systems. As we shall see, both systems can be implemented on three processors if global scheduling is used; in this case, they can be analyzed using the techniques of this paper but not using prior global schedulability analysis methods. Moreover, if the system is instead partitioned (allowing uniprocessor real-time calculus to be applied on each processor), then four processors are required.

In the analysis, we used a trace of 6×10^5 macroblock processing events obtained in prior work for the VLD+IQ task during a simulation of the (a)-system using a SimpleScalar architecture (Chakraborty et al 2006; Phan et al 2008).

We first determined execution times of macroblock instructions by examining a repeating pattern of 228,096 consecutive macroblock instruction lengths in the middle of the trace and assuming a 500 MHz processor frequency. We found that all macroblock processing times in the trace are under $164\mu s$ and the best-case macroblock processing time is $2\mu s$. These values are comparable to characteristic preemption and migration costs for multiprocessor systems measured in recent studies for architectures with higher processor frequencies (Brandenburg et al 2008; Brandenburg and Anderson 2009). Therefore it is not practical to invoke a job for each arriving macroblock. We thus assumed that a job is invoked for processing a single frame, which consists of 1,584 macroblocks, and obtained $\gamma_i^u(k)$ and $\gamma_i^l(k)$ as in Definition 1 for frames. We found that all frame processing times in the trace were under $\gamma_i^u(1) = 62ms$, which we set to be the maximum job execution time (the best-case execution time is $\gamma_i^l(1) = 17ms$). The function $\alpha_i^u(\Delta)$ in Definition 2 was obtained by examining macroblock and frame arrival times. We computed $\mathcal{A}_i^{-1}(k)$ in Definition 3 as well as linear bounds for $\alpha_i^u(\Delta)$ and $\gamma_i^u(k)$ as in (2) and (3) using the RTC Toolbox (Wandeler and Thiele 2006).

In the (b)- and (c)-systems, three fully-available processors are used for scheduling tasks $T_1, \ldots, T_4$. However the scheduling algorithms in these two systems are different.

In the (b)-system, task T_1 is statically prioritized over the other tasks. In such a system, task T_1 can process a time-critical video stream and tasks T_2, T_3, and T_4 can process low-priority video streams. The remaining tasks T_2, T_3, and T_4 are scheduled by GEDF using the supply from two fully-available processors and that remaining on a third processor after accommodating task T_1. In Figure 8(b), down arrows are used to depict the long-term available utilization on each processor.

In the (c)-system, task T_1 and tasks $T_2, \ldots, T_4$ are encapsulated into two containers C_1 and C_2, respectively, as shown in Figure 8(c). The available processor time is distributed among these two containers as follows. Two processors are dedicated for scheduling tasks in C_2. The time on the third processor is allocated using periodic server tasks S_1 and S_2 with execution times e_1 and e_2 and periods p_1 and p_2. The jobs of S_1 and S_2 are scheduled using uniprocessor EDF. When task S_1 is scheduled, a job of T_1 is scheduled. Tasks $T_2, \ldots, T_4$ are scheduled by GEDF using the supply from two fully-available processor and the time available on the third processor when S_2 is scheduled. To ensure schedulability of the underlying tasks, the execution times and periods for server tasks should be selected as follows. $e_1/p_1 = \widehat{U_1} \geq u_1$, $2 + e_2/p_2 = \widehat{U_2} \geq u_2 + u_3 + u_4$, and $e_1/p_1 + e_2/p_2 = 1$. In Figure 8(c), down arrows to the container boxes denote the long-term guaranteed utilization in the respective container. The scheme described above is an application of hierarchical scheduling techniques developed by two of the authors earlier (Leontyev and Anderson 2009b). In contrast to the (b)-system, in the (c)-system, task T_1 is temporally isolated from the other tasks.

Results. To show that existing analysis techniques are inapplicable or are too pessimistic in the given setup, some of the properties of the input streams and the VLD+IQ task need to be emphasized.

First, for both (b)- and (c)-systems, the minimum job inter-arrival time is $18ms$. Because the long-term arrival rate is $R_i = 0.03$, the arriving stream cannot be reshaped to achieve a minimum job inter-arrival time greater than $p_i = 1/R_i = 33.3ms$ so that the long-term arrival rate is preserved.

Second, while the long-term worst-case execution time is $\overline{e_i} = 25.5ms$ (see Definition 1 and (3)), the maximum processing time of a single frame is $62ms$, so assuming that each job executes for its worst-case execution time would result in heavy overprovisioning. The long-term per-task utilization is $u_i = R_i \cdot \overline{e_i} = 0.03 \cdot 25.5 = 0.64$. Finally, the total utilization is $U = \sum_{i=1}^{4} u_i = 2.56$. Therefore, the task set $\{T_1, \ldots, T_4\}$ cannot be partitioned onto three processors (four processors are needed, actually), so global scheduling is required.

Because the worst-case job execution time is $e_i^{\max} = \gamma_i^u(1) = 62ms$ and the minimum job inter-arrival time is $p_i = 18ms$, we have $e_i^{\max}/p_i = 3.44 > 1$. Therefore, both (b)- and (c)-systems *cannot be analyzed using prior results for periodic and sporadic task models*, which require $p_i > 0$ and $e_i^{\max}/p_i \leq 1$.

Figure 9 depicts the job completion curve $\alpha_1^{u\prime}$ for task T_1 in the (a)- and (b)-systems, the curve $\alpha_2^{u\prime}$ for task T_2 in the (b)-system, and the input curve α_1^u. (Note that, in the (b)- and (c)-systems, tasks $T_1, \ldots, T_4$ have the same input curve α_1^u, and the completion curves for $T_2, \ldots, T_4$ are the same (within the respective system).) Figure 10 shows the input and completion curves for tasks T_1 and T_2 in the (c)-system.

Because task T_1 is effectively scheduled on a dedicated processor in the (a)- and (b)-systems, the output curves for T_1 in these two systems were obtained using prior results in real-time calculus for uniprocessor systems.

For the (b)-system, we calculated the maximum response time for T_1 and then applied Theorem 2 to find the supply available to tasks T_2, T_3, and T_4. We then calculated their response-time bounds Θ_i' using Theorem 7. After that, we set $\Theta_i = \lfloor \Theta_i' \cdot 0.83 \rfloor = 989ms$, which is the minimum value such that the conclusion of Corollary 3 still holds. The multiplier 0.83 was found by running a binary search procedure. We then computed completion curves using Theorem 1.

For the (c)-system, we constructed two periodic server tasks S_1 and S_2 with execution times $7ms$ and $3ms$, respectively, and period $10ms$. These values for execution times and periods are the smallest multiples of a typical quantum length of $1ms$ that give an approximate task utilization of 0.64. We used prior results to calculate the guaranteed processor time to each of the containers C_1 and C_2 (Leontyev and Anderson 2009b).

Because, in the (c)-system, task T_1 is effectively scheduled on one processor with limited availability, we calculated the output curve for task T_1 using prior results in uniprocessor real-time calculus. Given the supply guaranteed to container C_2, we calculated for tasks T_2, T_3, and T_4 the response-time bound $\Theta_i = 949ms$ and the completion curves similarly to those in the (b)-system.

The resulting curves have the same long-term completion rate in all the three systems. Task T_1 has the shortest possible maximum response time in both the (a)-

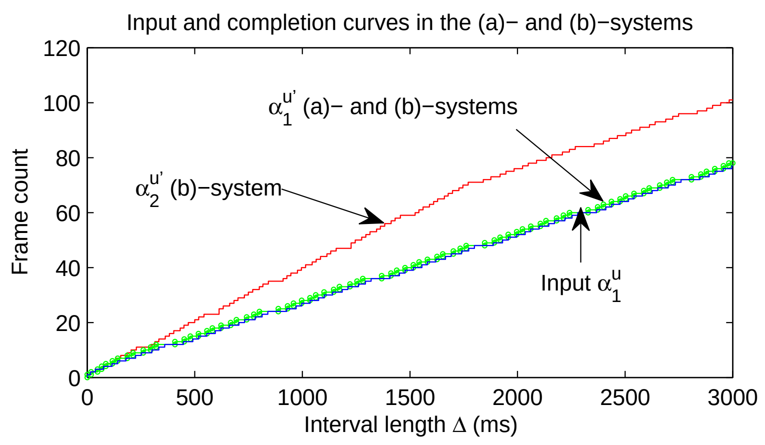


Fig. 9: Job arrival curve α^u and completion curves $\alpha^{u\prime}$ for tasks T_1 and T_2 in the (a)- and (b)-systems.

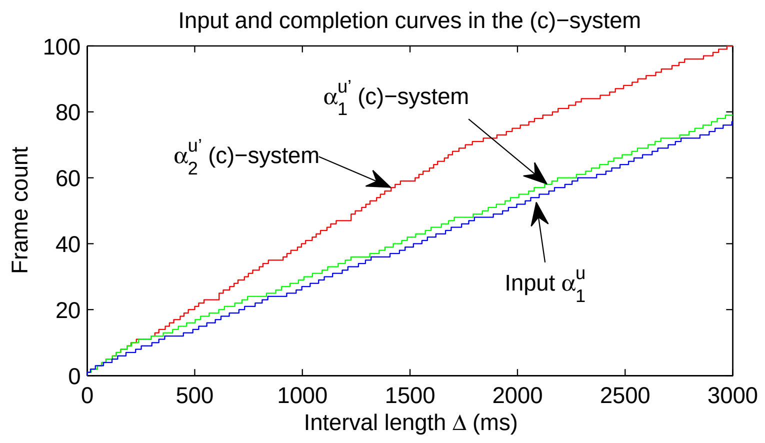


Fig. 10: Job arrival curve α^u and completion curves $\alpha^{u\prime}$ for tasks T_1 and T_2 in the (c)-system.

and (b)-systems. However, the large job response times of tasks $T_2, \ldots, T_4$ in the (b)- and (c)-systems cause a larger degree of burstiness in the output event streams. This burstiness is a result of conservatism in the analysis.

Overall, the (b)- and (c)-systems have the advantage of needing only *three* processors to accommodate four video streams, while with partitioned scheduling, *four* dedicated processors are required. This advantage in the number of processors comes at the expense of larger buffers for storing partially decoded macroblocks for tasks T_2, T_3, and T_4. (The additional buffer size is the maximum difference between the output curves $\alpha_1^{u\prime}$ in the (b)- and (c)- and (a)- systems.) The output buffer for tasks $T_2, \ldots, T_4$ should be at least 25 additional frames, which is 1 second worth of video.

We conclude this section with a few comments about the running time of the analysis procedures. We have implemented these procedures as a set of MATLAB functions extending the RTC Toolbox. Though the procedure presented in Section 6 has pseudo-polynomial time complexity (like many other schedulability tests presented elsewhere), the time needed to verify response times using Corollary 1 can be large, especially for complex arrival and execution-time patterns. In our experimental study of the (b)- and (c)-systems, we found that the required response-time bounds could be calculated in about a couple of minutes, by using Theorem 7 to obtain initial bounds, which were then refined using Corollary 1 (on a 1.7 GHz single-processor desktop system).

11 Conclusion

In this paper, we have studied a multiprocessor PE, where (partially available) processors are managed by a global scheduling algorithm and jobs are triggered by streams of external events. This work is of importance because it allows workloads to be analyzed for which existing schedulability analysis methods are completely inapplicable (e.g., the system cannot be described efficiently using conventional periodic/sporadic task models) and for which partitioning techniques are unnecessarily restrictive.

The research in this paper is part of a broader effort, the goal of which is to produce a practical compositional framework, based on real-time calculus, for analyzing multiprocessor real-time systems. Towards this goal, the contributions of this paper are as follows. We designed a pseudo-polynomial-time procedure that can be used to test whether job response times occur within specified bounds. Given these bounds, we computed upper and lower bounds on the number of job completion events over any interval of length Δ and a lower bound on the supply available after scheduling all incoming jobs. These bounds can be used as inputs for other PEs thereby resulting in a compositional analysis framework.

A number of unresolved issues of practical importance remain. First, *efficient* methods are needed for determining response-time bounds when they are not specified — this is probably the most important unresolved issue left by this paper. As a partial solution, we provided closed-form expressions for computing response-time bounds, but we do not know how pessimistic they are. Second, the schedulability test itself could possibly be improved by incorporating information about lower bounds on job arrivals and execution times and upper bounds on supply. Third, real-time interfaces as in (Chakraborty et al 2006) need to be derived for the multiprocessor case to achieve full compatibility with uniprocessor real-time calculus. Fourth, the inherent pessimism introduced by applying real-time calculus methods on multiprocessors needs to be assessed.

Acknowledgment: Work supported by AT&T, IBM, and Sun Corps.; NSF grants CNS 0834270 and CNS 0834132; ARO grant W911NF-09-1-0535; and AFOSR grant FA9550-09-1-0549. We are grateful to Linh Thi Xuan Phan for her help with experimental data.

References

Baruah S (2007) Techniques for multiprocessor global schedulability analysis. In: Proceedings of the 28th IEEE Real-Time Systems Symposium, pp 119–128

Bertogna M, Cirinei M, Lipari G (2009) Schedulability analysis of global scheduling algorithms on multiprocessor platforms. IEEE Transactions on Parallel and Distributed Systems 20(4):553–566

Blum M, Floyd R, Pratt V, Rivest R, Tarjan R (1973) Time bounds for selection. Journal of Computer and System Sciences 7(4):448–461

Brandenburg B, Anderson J (2009) On the implementation of global real-time schedulers. In: Proceedings of the 30th IEEE Real-Time Systems Symposium, pp 214–227

Brandenburg B, Calandrino J, Anderson J (2008) On the scalability of real-time scheduling algorithms on multicore platforms: A case study. In: Proceedings of the 29th IEEE Real-Time Systems Symposium, pp 157–169

Chakraborty S, Kunzli S, Thiele L (2003) A general framework for analysing system properties in platform-based embedded system designs. In: Proceedings of Design, Automation and Test in Europe - Volume 1, p 10190

Chakraborty S, Liu Y, Stoimenov N, Thiele L, Wandeler E (2006) Interface-based rate analysis of embedded systems. In: Proceedings of the 27th IEEE Real-Time Systems Symposium, pp 25–34

Cruz R (1991a) A calculus for network delay, Part I: Network elements in isolation. IEEE Transactions on Information Theory 37(1)

Cruz R (1991b) A calculus for network delay, Part II: Network analysis. IEEE Transactions on Information Theory 37(1)

Devi U (2006) Soft real-time scheduling on multiprocessors. PhD thesis, University of North Carolina, Chapel Hill, NC

Leontyev H, Anderson J (2008) A unified hard/soft real-time schedulability test for global EDF multiprocessor scheduling. In: Proceedings of the 29th IEEE Real-Time Systems Symposium, pp 375–384

Leontyev H, Anderson J (2009a) Generalized tardiness bounds fo multiprocessor scheduling. Real-Time Systems To appear

Leontyev H, Anderson J (2009b) A hierarchical multiprocessor bandwidth reservation scheme with timing guarantees. Real-Time Systems 43(1):191–200

Maxiaguine A (2005) Modeling multimedia workloads for embedded system design. PhD thesis, ETH Zurich

Mok AK, Feng X, Chen D (2001) Resource partition for real-time systems. In: Proceedings of 7th IEEE Real-Time Technology and Applications Symposium, pp 75–84

Phan L, Chakraborty S, Thiagarajan P (2008) A multi-mode real-time calculus. In: Proceedings of the 29th IEEE Real-Time Systems Symposium, pp 59–69

Richter K, Jersak M, Ernst R (2003) A formal approach to MpSoC performance verification. IEEE Computer 36(4):60–67

Shin I, Easwaran A, Lee I (2008) Hierarchical scheduling framework for virtual clustering of multiprocessors. In: Proceedings of the 20th Euromicro Conference on Real-Time Systems, pp 181–190

Wandeler E, Thiele L (2006) Real-Time Calculus (RTC) Toolbox. URL http://www.mpa.ethz.ch/Rtctoolbox

Wu J, Liu JC, Zhao W (2005) On schedulability bounds of static priority schedulers. In: Proceedings of the 11th IEEE Real Time and Embedded Technology and Applications Symposium, pp 529–540

Zhang F, Burns A (2008) Schedulability Analysis for Real-Time Systems with EDF Scheduling. Tech. Rep. YCS-2008-426, University of York, Department of Computer Science

Appendix

In this appendix, we prove Claim 6, Lemmas 6, 7, 8, 18, and 19. We first prove Claim 6.

Claim 6: $\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) \le r_{\ell,q} + \Theta_\ell - r_{\ell,q-\lambda+1}$.

Proof. Each job $T_{\ell,q-k}$, where $k \geq \lambda$ completes by $f_{\ell,q-\lambda}$. Thus,

$$\begin{aligned}
&\sum_{k\geq\lambda} W(T_{\ell,q-k}, r_{\ell,q-\lambda+1}) \\
&\quad \{\text{by Definition 19}\} \\
&\leq f_{\ell,q-\lambda} - r_{\ell,q-\lambda+1} \\
&\quad \{\text{by Definition 11}\} \\
&\leq r_{\ell,q} + \mathit{\Theta}_\ell - \gamma_\ell^u(\lambda) - r_{\ell,q-\lambda+1}. \qquad (40)
\end{aligned}$$

Also, by Definition 19,

$$\begin{aligned}
&\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) \\
&= \sum_{T_{\ell,j}\in\mathcal{J}} W(T_{\ell,j}, r_{\ell,q-\lambda+1}) \\
&\quad \{\text{because } \mathcal{J} \text{ does not contain } T_{\ell,q}\text{'s successors}\} \\
&= \sum_{k\geq 0} W(T_{\ell,q-k}, r_{\ell,q-\lambda+1}) \\
&= \sum_{k\geq\lambda} W(T_{\ell,q-k}, r_{\ell,q-\lambda+1}) + \sum_{k\in[0,\lambda-1]} W(T_{\ell,q-k}, r_{\ell,q-\lambda+1}) \\
&\quad \{\text{by (40)}\} \\
&\leq r_{\ell,q} + \mathit{\Theta}_\ell - \gamma_\ell^u(\lambda) - r_{\ell,q-\lambda+1} + \sum_{k\in[0,\lambda-1]} W(T_{\ell,q-k}, r_{\ell,q-\lambda+1}) \\
&\quad \{\text{by Definitions 1 and 19}\} \\
&\leq r_{\ell,q} + \mathit{\Theta}_\ell - \gamma_\ell^u(\lambda) - r_{\ell,q-\lambda+1} + \gamma_\ell^u(\lambda) \\
&= r_{\ell,q} + \mathit{\Theta}_\ell - r_{\ell,q-\lambda+1}. \qquad \square
\end{aligned}$$

Because the allocation of a task over a set of intervals cannot exceed the cumulative length of these intervals, the claim below follows.

Claim A1: $\mathsf{A}_{\mathcal{J}}(T_i, [t_0(k), r_{\ell,q} + \mathit{\Theta}_\ell)) \leq r_{\ell,q} - t_0(k) + \mathit{\Theta}_\ell$.

Lemma 6: $\mathsf{A}_{\mathbf{NC}}(T_i, \delta) = \min(\delta + \mathit{\Theta}_\ell, \gamma_i^u(\alpha_i^+(\delta + C_{\ell,i})))$.

Proof. The competing demand due to T_i is upper-bounded by the demand due to T_i's jobs in $\mathcal{J}$ (refer to Definition 14). Because $T_i \in$ **NC**, all such jobs released prior to $t_0(k)$ are completed by time $t_0(k)$. For any $T_{i,j} \in \mathcal{J}$, by Lemma 5, $r_{i,j} \leq r_{\ell,q} + C_{\ell,i}$. Therefore, the allocation $\mathsf{A}_{\mathcal{J}}(T_i, [t_0(k), r_{\ell,q} + \mathit{\Theta}_\ell))$ is upper-bounded by the total execution time of T_i's jobs released within $[t_0(k), r_{\ell,q} + C_{\ell,i}]$. From Definitions 1 and 10, we have

$$\begin{aligned}
\mathsf{A}_{\mathcal{J}}(T_i, [t_0(k), r_{\ell,q} + \mathit{\Theta}_\ell)) &\leq \gamma_i^u(\alpha_i^+(r_{\ell,q} + C_{\ell,i} - t_0(k))) \\
&= \gamma_i^u(\alpha_i^+(r_{\ell,q} - t_0(k) + C_{\ell,i})).
\end{aligned}$$

By Claim A1 and the inequality above, $\mathsf{A}_{\mathbf{NC}}(T_i, \delta)$ upper-bounds $\mathsf{A}_{\mathcal{J}}(T_i, [t_0(k), r_{\ell,q} + \mathit{\Theta}_\ell))$ for $\delta = r_{\ell,q} - t_0(k)$. $\square$

The following claim and a lemma will be used to prove Lemmas 7 and 9.

Claim A2. *The function $G_i(S, X)$ as defined in Definition 26 is a non-decreasing function of the integral argument S.*

Proof. Suppose that $S \geq 1$ is fixed. We compute $G_i(S+1, X)$. By Definition 26,

$$\begin{aligned} G_i(S+1, X) = &\ \min(\gamma_i^u(S+1), \max(0, X - \mathcal{A}_\ell^{-1}(S)) + \gamma_i^u(S)) \\ &\quad \{\text{because } \gamma_i^u(S) \text{ is a non-decreasing function}\} \\ \geq &\ \gamma_i^u(S) \\ \geq &\ \min(\gamma_i^u(S), \max(0, X - \mathcal{A}_\ell^{-1}(S-1)) + \gamma_i^u(S-1)) \\ = &\ G_i(S, X). \end{aligned}$$

□

Lemma A1. *If $t_x \leq r_{\ell,q}$, then*

$$\mathsf{W}_{\mathcal{J}}(T_i, t_x) \leq G_i(\alpha_i^u(r_{\ell,q} - t_x + C_{\ell,i} + \Theta_i), r_{\ell,q} - t_x + C_{\ell,i} + \Theta_i)$$

Proof. Let $T_{i,c} \in \mathcal{J}$ be the earliest job of T_i that is pending at or after time t_x. Note that if $T_{i,c}$ does not exit, then $\mathsf{W}_{\mathcal{J}}(T_i, t_x) = 0$. From the selection of $T_{i,c}$, we have $f_{i,c} > t_x$. If $T_{i,c} \neq T_{\ell,q}$, then $T_{i,c} \prec T_{\ell,q}$, which, by (12), implies

$$f_{i,c} > t_x \wedge r_{i,c} + \Theta_i > t_x. \tag{41}$$

If $T_{i,c} = T_{\ell,q}$, then

$$f_{i,c} > t_x \wedge r_{i,c} + \Theta_i - t_x \geq \gamma_i^u(1). \tag{42}$$

The predicate above holds because $t_x \leq r_{\ell,q}$ by the condition of the lemma, and $\Theta_i \geq \gamma_i^u(1) > 0$ by Claim 1. Note that (42) implies (41). We define the job set $\mathcal{J}_i$ as follows.

$$\text{Let } \mathcal{J}_i = \{T_{i,y} : y \geq c \wedge T_{i,y} \in \mathcal{J}\}. \tag{43}$$

To establish an upper-bound on $\mathsf{W}_{\mathcal{J}}(T_i, t_x)$, we first rewrite $\mathsf{W}_{\mathcal{J}}(T_i, t_x)$ as follows.

$$\begin{aligned} \mathsf{W}_{\mathcal{J}}(T_i, t_x) &= W(T_{i,c}, t_x) + \sum_{T_{i,y} \in \mathcal{J} \setminus T_{i,c}} W(T_{i,y}, t_x) \\ &= W(T_{i,c}, t_x) + \sum_{T_{i,y} \in \mathcal{J}_i \setminus T_{i,c}} W(T_{i,y}, t_x) \end{aligned} \tag{44}$$

We now bound the $W(T_{i,c}, t_x)$ term in (44) by considering two cases.

Case 1: $T_{i,c} = T_{\ell,q}$. By (42), $r_{\ell,q} + \Theta_\ell - t_x \geq \gamma_\ell^u(1) \geq e_{\ell,q}$, in which case $W(T_{\ell,q}, t_x) \leq e_{\ell,q}$. Thus,

$$W(T_{\ell,q}, t_x) \leq \min(e_{\ell,q}, r_{\ell,q} + \Theta_\ell - t_x). \tag{45}$$

Case 2: $T_{i,c} \neq T_{\ell,q}$**.** By (41), $T_{i,c}$ finishes its execution at time $f_{i,c} > t_x$, and hence,

$$\begin{aligned} W(T_{i,c}, t_x) \leq\ & \min(e_{i,c}, f_{i,c} - t_x) \\ & \{\text{if } T_{i,c} \prec T_{\ell,q}, \text{ by (12)}\} \\ \leq\ & \min(e_{i,c}, r_{i,c} + \Theta_i - t_x). \end{aligned} \tag{46}$$

By (44), (45), and (46),

$$\begin{aligned} & \mathsf{W}_{\mathcal{J}}(T_i, t_x) \\ & \leq\ \min(e_{i,c}, r_{i,c} + \Theta_i - t_x) + \sum_{T_{i,y} \in \mathcal{J}_i \setminus T_{i,c}} W(T_{i,y}, t_x) \\ & \leq\ \min\left(e_{i,c} + \sum_{T_{i,y} \in \mathcal{J}_i \setminus T_{i,c}} W(T_{i,y}, t_x), r_{i,c} + \Theta_i - t_x + \sum_{T_{i,y} \in \mathcal{J}_i \setminus T_{i,c}} W(T_{i,y}, t_x) \right). \end{aligned} \tag{47}$$

Let

$$S_i = |\mathcal{J}_i|. \tag{48}$$

Because the execution demand of job $T_{i,y}$ cannot be greater than its execution time, by Definition 1, we have the following.

$$e_{i,c} + \sum_{T_{i,y} \in \mathcal{J}_i \setminus T_{i,c}} W(T_{i,y}, t_x) \leq \gamma_i^u(S_i) \tag{49}$$

$$\sum_{T_{i,y} \in \mathcal{J}_i \setminus T_{i,c}} W(T_{i,y}, t_x) \leq \gamma_i^u(S_i - 1) \tag{50}$$

By (47), (49), and (50), we have

$$\mathsf{W}_{\mathcal{J}}(T_i, t_x) \leq \min(\gamma_i^u(S_i), r_{i,c} + \Theta_i - t_x + \gamma_i^u(S_i - 1)). \tag{51}$$

We next establish an upper bound on $r_{i,c}$ in (51). From (43) above and Lemma 5, we have

(R) If $T_{i,y} \in \mathcal{J}_i$, then $r_{i,y} \in [r_{i,c}, r_{\ell,q} + C_{\ell,i}]$.

By (48), $T_{i,c+S_i-1}$ is the latest job of T_i released within $[r_{i,c}, r_{\ell,q} + C_{\ell,i}]$. We upper bound $r_{i,c}$ as follows.

$$\begin{aligned} r_{i,c} &= r_{i,c+S_i-1} + r_{i,c} - r_{i,c+S_i-1} \\ & \quad \{\text{by the definition of } T_{i,c+S_i-1}\} \\ &\leq r_{\ell,q} + C_{\ell,i} + r_{i,c} - r_{i,c+S_i-1} \\ & \quad \{\text{by Lemma 2}\} \\ &\leq r_{\ell,q} + C_{\ell,i} - \mathcal{A}_i^{-1}(S_i - 1) \end{aligned}$$

From the inequality above, we have

$$\begin{aligned} r_{i,c} + \mathit{\Theta}_i - t_x \leq &\ \max(0, r_{\ell,q} + C_{\ell,i} - \mathcal{A}_i^{-1}(S_i - 1) + \mathit{\Theta}_i - t_x) \\ = &\ \max(0, r_{\ell,q} - t_x + C_{\ell,i} + \mathit{\Theta}_i - \mathcal{A}_i^{-1}(S_i - 1)). \end{aligned} \quad (52)$$

By (51) and (52), we have

$$\begin{aligned} &\mathsf{W}_{\mathcal{J}}(T_i, t_x) \\ &\leq \min(\gamma_i^u(S_i), \max(0, r_{\ell,q} - t_x + C_{\ell,i} + \mathit{\Theta}_i - \mathcal{A}^{-1}(S_i - 1)) + \gamma_i^u(S_i - 1)) \\ &= G_i(S_i, r_{\ell,q} - t_x + C_{\ell,i} + \mathit{\Theta}_i), \end{aligned} \quad (53)$$

where $G_i(S, X)$ is defined as in Definition 26. By Claim A2, the function $G_i(S, X)$ is a non-decreasing function of S. We thus can find an upper bound on $\mathsf{W}_{\mathcal{J}}(T_i, t_x)$ by setting an upper bound on S_i into (53).

By (R), $S_i = |\mathcal{J}_i|$ is at most the number of jobs of T_i released within the interval $[r_{i,c}, r_{\ell,q} + C_{\ell,i}]$, which, by (41), is contained within $(t_x - \mathit{\Theta}_i, r_{\ell,q} + C_{\ell,i}]$. We thus upper bound S_i using Definition 2.

$$S_i \leq \alpha_i^u(r_{\ell,q} - t_x + C_{\ell,i} + \mathit{\Theta}_i)$$

Setting this upper bound on S_i into (53), we get the conclusion of the lemma. □

Using the result of the lemma above, we next prove Lemma 7.

Lemma 7: $\mathsf{A}_{\mathbf{HC}}(T_i, \delta) = \min(\delta + \mathit{\Theta}_\ell, G_i(\alpha_i^u(\delta + C_{\ell,i} + \mathit{\Theta}_i), \delta + C_{\ell,i} + \mathit{\Theta}_i))$.

Proof. Consider $T_i \in$ **HC**. The allocation of T_i's jobs from $\mathcal{J}$ cannot exceed their cumulative demand. From Definitions 15 and 19, we have

$$\begin{aligned} \mathsf{A}_{\mathcal{J}}(T_i, [t_0(k), r_{\ell,q} + \mathit{\Theta}_\ell)) &\leq \mathsf{W}_{\mathcal{J}}(T_i, t_0(k)) \\ &\quad \{\text{by Lemma A1}\} \\ &\leq G_i(\alpha_i^u(t_0(k) - r_{\ell,q} + C_{\ell,i} + \mathit{\Theta}_i), t_0(k) - r_{\ell,q} + C_{\ell,i} + \mathit{\Theta}_i) \\ &\quad \{\text{setting } \delta = t_0(k) - r_{\ell,q}\} \\ &= G_i(\alpha_i^u(\delta + C_{\ell,i} + \mathit{\Theta}_i), \delta + C_{\ell,i} + \mathit{\Theta}_i). \end{aligned}$$

By the inequality above and Claim A1, $\mathsf{A}_{\mathbf{HC}}(T_i, \delta)$ upper-bounds $\mathsf{A}_{\mathcal{J}}(T_i, [t_0(k), r_{\ell,q} + \mathit{\Theta}_\ell))$ for $\delta = r_{\ell,q} - t_0(k)$. □

The following claims and lemma are used to prove Lemma 8.

Claim A3: $L_i(X + Y) \leq L_i(X) + u_i \cdot Y$ *for all* X *and* $Y \geq 0$.

Proof. By Definition 4, $u_i > 0$. By the condition of the claim, $Y \geq 0$. Thus, by Definition 27,

$$\begin{aligned} L_i(X + Y) = &\ \max(0, u_i \cdot (X + Y) + \overline{e_i} \cdot B_i) + v_i \\ \leq &\ \max(0, u_i \cdot X + \overline{e_i} \cdot B_i) + v_i + u_i \cdot Y \\ = &\ L_i(X) + u_i \cdot Y. \end{aligned}$$

□

Claim A4: $\alpha_i^+(X) \leq R_i \cdot X + B_i$ *for* $X \geq 0$.

Proof. By Definition 10,

$$\begin{aligned}\alpha_i^+(X) &= \lim_{\epsilon\to+0} \alpha_i^u(X+\epsilon)\\ &\quad \{\text{by (2)}\}\\ &\leq \lim_{\epsilon\to+0} R_i \cdot (X+\epsilon) + B_i\\ &= R_i \cdot X + B_i.\end{aligned}$$ □

Claim A5: $\gamma_i^u(\alpha_i^u(X)) \leq \gamma_i^u(\alpha_i^+(X)) \leq L_i(X)$ *for all* X.

Proof. By Definition 2, $\alpha_i^u(\varDelta)$ is a non-decreasing function of $\varDelta$. Therefore, $\alpha_i^u(\varDelta) \leq \alpha_i^u(\varDelta + \epsilon)$ for any $\epsilon > 0$, which implies $\alpha_i^u(\varDelta) \leq \lim_{\epsilon\to+0} \alpha_i^u(\varDelta+\epsilon)$. The right-hand side of the latter inequality is $\alpha_i^+(\varDelta)$ by Definition 10. Thus, $\alpha_i^u(\varDelta) \leq \alpha_i^+(\varDelta)$. The first inequality of the claim therefore follows from $\gamma_i^u(k)$ being a non-decreasing function of k by Definition 1. We now prove the second inequality by considering two cases.
Case 1: $X < 0$**.** In this case, by Definition 10, $\alpha_i^+(X) = 0$. By Definition 1, $\gamma_i^u(\alpha_i^+(X)) = 0$. The required result follows from Definition 27 and $v_i \geq 0$ (see (3)).
Case 2: $X \geq 0$**.** By Definition 10, because $\alpha_i^+(X) \geq 0$, we have

$$\begin{aligned}\gamma_i^u(\alpha_i^+(X)) &= \gamma_i^u(\max(0, \alpha_i^+(X)))\\ &\quad \{\text{by (3)}\}\\ &\leq \overline{e_i} \cdot (\max(0, \alpha_i^+(X))) + v_i\\ &\quad \{\text{by Claim A4}\}\\ &\leq \overline{e_i} \cdot (\max(0, R_i \cdot X + B_i)) + v_i\\ &= \max(0, \overline{e_i} \cdot R_i \cdot X + \overline{e_i} \cdot B_i) + v_i\\ &\quad \{\text{by Definition 4}\}\\ &= \max(0, u_i \cdot X + \overline{e_i} \cdot B_i) + v_i\\ &\quad \{\text{by Definition 27}\}\\ &= L_i(X).\end{aligned}$$ □

Lemma A2: $\mathsf{A}_{\textbf{HC}}(T_i, \delta) \leq L_i(\delta + C_{\ell,i}) + u_i \cdot \varTheta_i$ *and* $\mathsf{A}_{\textbf{NC}}(T_i, \delta) \leq L_i(\delta + C_{\ell,i})$.

Proof. We prove the first inequality.

$$\begin{aligned}
&\mathsf{A}_{\mathbf{HC}}(T_i,\delta)\\
&\quad\{\text{by Lemma 7}\}\\
&=\ \min(\delta{+}\Theta_\ell, G_i(\alpha_i^u(\delta{+}C_{\ell,i}{+}\Theta_i), \delta{+}C_{\ell,i}{+}\Theta_i))\\
&\le G_i(\alpha_i^u(\delta{+}C_{\ell,i}{+}\Theta_i), \delta{+}C_{\ell,i}{+}\Theta_i)\\
&\quad\{\text{by Definition 26}\}\\
&\le \gamma_i^u(\alpha_i^u(\delta + C_{\ell,i} + \Theta_i))\\
&\quad\{\text{by Claim A5}\}\\
&\le L_i(\delta + C_{\ell,i} + \Theta_i)\\
&\quad\{\text{because } \Theta_i \ge 0\text{, by Claim A3}\}\\
&\le L_i(\delta + C_{\ell,i}) + u_i \cdot \Theta_i
\end{aligned}$$

The second inequality is proved similarly.

$$\begin{aligned}
&\mathsf{A}_{\mathbf{NC}}(T_i,\delta)\\
&\quad\{\text{by Lemma 6}\}\\
&=\ \min(\delta + \Theta_\ell, \gamma_i^u(\alpha_i^+(\delta + C_{\ell,i})))\\
&\le \gamma_i^u(\alpha_i^+(\delta + C_{\ell,i}))\\
&\quad\{\text{by Claim A5}\}\\
&\le L_i(\delta + C_{\ell,i})
\end{aligned}$$

□

Lemma 8. *For all $\delta \ge 0$, $M_\ell^*(\delta) \le U_{sum} \cdot \delta + H_\ell$, where $H_\ell = \sum_{T_i \in \tau} L_i(C_{\ell,i}) + U(m-1) \cdot \max(\Theta_i)$ and $U(y)$ is the sum of $\min(y, |\tau|)$ largest utilizations.*

Proof. Suppose that the sets **HC** and **NC** subject to (28) maximize the value of the right-hand side of (27). By (27), we have

$$\begin{aligned}
M_\ell^*(\delta) &= \sum_{T_i \in \mathbf{HC}} \mathsf{A}_{\mathbf{HC}}(T_i,\delta) + \sum_{T_i \in \mathbf{NC}} \mathsf{A}_{\mathbf{NC}}(T_i,\delta)\\
&\quad\{\text{by Lemma A2}\}\\
&\le \sum_{T_i \in \mathbf{HC}} (L_i(\delta + C_{\ell,i}) + u_i \cdot \Theta_i) + \sum_{T_i \in \mathbf{NC}} L_i(\delta + C_{\ell,i})\\
&\quad\{\text{since } \mathbf{HC} \cup \mathbf{NC} \subseteq \tau \text{ and } L_i(X) \ge 0 \text{ for all } X\}\\
&\le \sum_{T_i \in \tau} L_i(\delta + C_{\ell,i}) + \sum_{T_i \in \mathbf{HC}} u_i \cdot \Theta_i\\
&\quad\left\{\begin{array}{l}\text{because } |\mathbf{HC}| \le m-1 \text{ by (28), and using the definition}\\ \text{of } U(y) \text{ in the statement of the lemma}\end{array}\right\}\\
&\le \sum_{T_i \in \tau} [L_i(\delta + C_{\ell,i})] + U(m-1) \cdot \max(\Theta_i)\\
&\quad\{\text{by Claim A3 (note that, by the statement of the lemma, } \delta \ge 0)\}
\end{aligned}$$

$$\le \sum_{T_i \in \tau} [L_i(C_{\ell,i}) + u_i \cdot \delta] + U(m-1) \cdot \max(\Theta_i)$$
$$\left\{ \begin{array}{l} \text{by Definition 4 and the definition of } H_\ell \\ \text{in the statement of the lemma} \end{array} \right\}$$
$$= U_{sum} \cdot \delta + H_\ell. \qquad \square$$

We now derive a lower bound for $E_\ell^*(\lambda)$ given by Lemma 9 and prove Lemma 18.

Lemma 9. *If $E_\ell^*(k)$ is given by (29), then $E_\ell^*(\lambda) \ge \mathsf{W}_{\mathcal{J}}(T_i, r_{\ell,q-\lambda+1})$.*

Proof. By Definition 20, the function $E_\ell^*(\lambda)$ upper bounds $\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1})$, which is the amount of work due to unfinished jobs of T_ℓ in $\mathcal{J}$ at time $r_{\ell,q-\lambda+1}$. By Lemma A1,

$$\begin{aligned}
&\mathsf{W}_{\mathcal{J}}(T_\ell, r_{\ell,q-\lambda+1}) \\
&\le G_\ell(\alpha_\ell^u(r_{\ell,q} - r_{\ell,q-\lambda+1} + C_{\ell,\ell} + \Theta_\ell), r_{\ell,q} - r_{\ell,q-\lambda+1} + C_{\ell,\ell} + \Theta_\ell) \\
&\quad \{\text{because } C_{\ell,\ell} = 0 \text{ by Definition 24}\} \\
&= G_\ell(\alpha_\ell^u(r_{\ell,q} - r_{\ell,q-\lambda+1} + \Theta_\ell), r_{\ell,q} - r_{\ell,q-\lambda+1} + \Theta_\ell) \\
&\quad \{\text{by Claim 9}\} \\
&\le G_\ell(\alpha_\ell^u(\max(0, \gamma_\ell^u(\lambda-1)-1) + \Theta_\ell), \max(0, \gamma_\ell^u(\lambda-1)-1) + \Theta_\ell) \\
&\quad \{\text{by (29)}\} \\
&= E_\ell^*(\lambda). \qquad \square
\end{aligned}$$

Lemma 18. *If $\Theta_\ell = x + \gamma_\ell^u(K_\ell) + C_\ell$, where $x \ge 0$, then $E_\ell^*(k) \le Y_\ell + u_\ell \cdot x$ for $k \in [1, K_\ell]$.*

Proof. By (29),

$$\begin{aligned}
E_\ell^*(k) &= G_\ell(\alpha_\ell^u(Q(k)), Q(k)) \\
&\quad \{\text{by Definition 26}\} \\
&\le \gamma_\ell^u(\alpha_\ell^u(Q(k))) \\
&\quad \{\text{by Claim A5}\} \\
&\le L_\ell(Q(k)) \\
&\quad \{\text{by Definition 28}\} \\
&= L_\ell\big(\max(0, \gamma_\ell^u(k-1)-1) + \Theta_\ell\big) \\
&\quad \{\text{by the condition of the Lemma}\} \\
&= L_\ell\big(\max(0, \gamma_\ell^u(k-1)-1) + x + \gamma_\ell^u(K_\ell) + C_\ell\big) \\
&\quad \{\text{by Claim A3}\} \\
&\le L_\ell\big(\max(0, \gamma_\ell^u(k-1)-1) + \gamma_\ell^u(K_\ell) + C_\ell\big) + u_\ell \cdot x \\
&\quad \left\{ \begin{array}{l} \text{because } L_\ell \text{ and } \gamma_\ell^u \text{ are non-decreasing} \\ \text{functions of their arguments} \end{array} \right\} \\
&\le L_\ell\big(\max(0, \gamma_\ell^u(K_\ell-1)-1) + \gamma_\ell^u(K_\ell) + C_\ell\big) + u_\ell \cdot x \\
&\quad \{\text{by Definition 35}\}
\end{aligned}$$

$$= Y_\ell + u_\ell \cdot x. \qquad \square$$

Lemma 19. *If $\Theta_i = x + \gamma_i^u(K_i) + C_i$ for each task T_i and $\delta \geq 0$, then $M_\ell^*(\delta) \leq U_{sum} \cdot \delta + U(m-1) \cdot x + \mathcal{W} + \sum_{T_i \in \tau} L_i(C_{\ell,i})$, where $U(m-1)$ is the sum of $m-1$ largest task utilizations.*

Proof. Suppose that the sets **HC** and **NC** subject to (28) maximize the value of the right-hand side of (27). By (27), we have

$$\begin{aligned}
&M_\ell^*(\delta) \\
&= \sum_{T_i \in \mathbf{HC}} \mathsf{A}_{\mathbf{HC}}(T_i, \delta) + \sum_{T_i \in \mathbf{NC}} \mathsf{A}_{\mathbf{NC}}(T_i, \delta) \\
&\quad \{\text{by Lemma A2}\} \\
&\leq \sum_{T_i \in \mathbf{HC}} (L_i(\delta + C_{\ell,i}) + u_i \cdot \Theta_i) + \sum_{T_i \in \mathbf{NC}} L_i(\delta + C_{\ell,i}) \\
&\quad \{\text{since } \mathbf{HC} \cup \mathbf{NC} \subseteq \tau \text{ and } L_i(X) \geq 0 \text{ for all } X\} \\
&\leq \sum_{T_i \in \tau} L_i(\delta + C_{\ell,i}) + \sum_{T_i \in \mathbf{HC}} u_i \cdot \Theta_i \\
&\quad \{\text{by the selection of } \Theta_i \text{ in the statement of the Lemma}\} \\
&= \sum_{T_i \in \tau} L_i(\delta + C_{\ell,i}) + \sum_{T_i \in \mathbf{HC}} u_i \cdot (x + \gamma_i^u(K_i) + C_i) \\
&\quad \left\{\begin{array}{l}\text{because } |\mathbf{HC}| \leq m-1 \text{ by (28), and using the} \\ \text{definition of } U(y) \text{ in the statement of the lemma}\end{array}\right\} \\
&\leq \sum_{T_i \in \tau} L_i(\delta + C_{\ell,i}) + U(m-1) \cdot x + \sum_{T_i \in \mathbf{HC}} u_i \cdot (\gamma_i^u(K_i) + C_i) \\
&\quad \{\text{because } |\mathbf{HC}| \leq m-1 \text{ by (28), and by Definition 36}\} \\
&\leq \sum_{T_i \in \tau} L_i(\delta + C_{\ell,i}) + U(m-1) \cdot x + \mathcal{W} \\
&\quad \left\{\begin{array}{l}\text{by Claim A3} \\ \text{(note that, by the condition of the lemma, } \delta \geq 0\text{)}\end{array}\right\} \\
&\leq \sum_{T_i \in \tau} [L_i(C_{\ell,i}) + u_i \cdot \delta] + U(m-1) \cdot x + \mathcal{W} \\
&\quad \{\text{by Definition 4}\} \\
&= U_{sum} \cdot \delta + \sum_{T_i \in \tau} L_i(C_{\ell,i}) + U(m-1) \cdot x + \mathcal{W}. \qquad \square
\end{aligned}$$