

Multiprocessor Extensions to Real-Time Calculus

Hennadiy Leontyev¹

Samarjit Chakraborty²

James H. Anderson¹

¹ Department of Computer Science, University of North Carolina at Chapel Hill

² Institute for Real-Time Computer Systems (RCS), TU Munich

leontyev@cs.unc.edu, samarjit@tum.de, anderson@cs.unc.edu

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 compositional framework is presented that is obtained by extending real-time calculus to allow multiprocessor processing elements. The basic theory of the resulting extended framework is presented herein and its utility is demonstrated using a case study.

1. Introduction

The increasing complexity and heterogeneity of modern embedded platforms have led to growing interest in compositional modeling and analysis techniques [11]. 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. in [3] and then subsequently extended in a number of papers (e.g., see [4]). Here, the basic idea is

to represent the timing properties of event streams 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 α^u and α^l corresponding to an event stream arriving at a resource, and the service β^u and β^l offered by it, it is possible to compute the timing properties of the processed stream and remaining processing capacity, i.e., functions $\alpha^{u'}$, $\alpha^{l'}$, $\beta^{u'}$, and $\beta^{l'}$, as illustrated in Fig. 1(a), as well as the maximum backlog and delay experienced by the stream. As shown in the same figure, the computed functions $\alpha^{u'}$ and $\alpha^{l'}$ 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. Fig. 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, is available) to Stream B. On the other hand, for the TDMA policy, the total service β is split between the services available to the two streams. Similar so called *scheduling networks* [4] can be constructed for other scheduling policies as well. Various operations on arrival and service curves α and β , as well as procedures for the

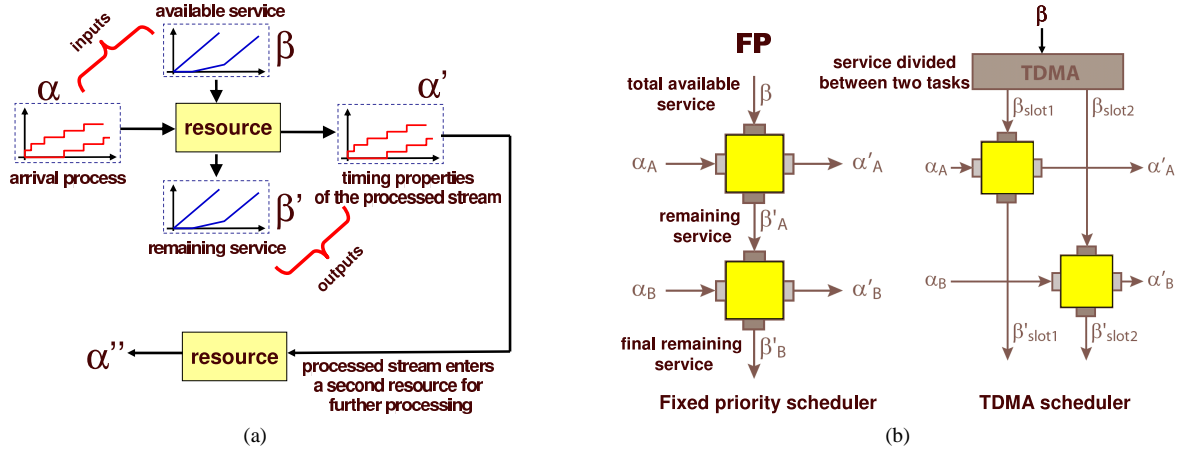


Figure 1. (a) Computing the timing properties of the processed stream using real-time calculus. (b) Scheduling networks for fixed priority and TDMA schedulers.

analysis of scheduling networks on uniprocessors (and partitioned systems) have been implemented in the RTC (real-time calculus) toolbox [13], which is a Matlab-based library that can be used for modeling and analyzing distributed real-time systems using the real-time calculus framework.

Our contribution. Unfortunately, none of the composition techniques described above can be used when the resource in question is a multiprocessor, where the constituent processors are managed according to a *global* multiprocessor scheduling algorithm. There are two reasons why existing composition techniques need to be extended to incorporate such multiprocessors. First, multicore chips are becoming increasingly common. Second, viewing a multiprocessor system as a collection of independent uniprocessors and applying partitioning techniques is unnecessarily restrictive and precludes supporting workloads that fundamentally require global scheduling approaches (such a workload is considered in a case study presented later).

Motivated by these observations, in this paper we present an extension of the real-time calculus framework [3, 4] that incorporates globally-scheduled multiprocessors and is compatible with the RTC toolbox. Given a collection of arrival curves for input streams α^u and α^l , their execution requirements, and the minimum guaranteed available processor time, we compute an upper bound on the maximum event delay for each stream. Using these event delays, we then compute arrival curves for the processed streams, $\alpha^{u'}$ and $\alpha^{l'}$, and the remaining-total-service curve; these curves — as in the original framework — can in turn be used as input for other resources, thereby resulting in a compositional framework (as shown in Fig. 1(a)).

The core of our analysis is a pseudo-polynomial-time procedure that can be used to test whether event delays on such a multiprocessor reside within specified bounds. This test is based upon multiprocessor schedulability tests by

Baruah [1] and Leontyev and Anderson [7]. In some aspects, the presented analysis is also similar to results by Bertogna et al. [2], Shin et al. [12], and Zhang and Burns [14]. 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 [12], where partial availability is considered in the context of hierarchical scheduling.

The rest of the paper is organized as follows. Sec. 2 presents our task model. In Secs. 3 and 4, timing characteristics of processed streams and the remaining supply are computed. In Secs. 5 and 6, the response-time-bound test is presented and its time complexity is discussed. Sec. 8 presents a case study for our analysis and finally Sec. 9 discusses some directions for future work.

2. Task Model

In this paper, we consider a task set $\tau = \{T_1, \dots, T_n\}$. 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 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.

Definition 1. $\gamma_i(k)$ denotes an upper bound on the total execution time of any k consecutive jobs of T_i . (We assume

$\gamma_i(k) = 0$ for all $k \leq 0$ and $\gamma_i(k) \leq \gamma_i(k+1)$.)

Example 1. Suppose that task T_i 's worst-case job execution times follow a pattern 1, 5, 2, 1, 5, 2, \dots . Then, $\gamma_i(1) = 5$, $\gamma_i(2) = 7$, $\gamma_i(3) = 8$, $\gamma_i(4) = 13$, 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 $\Delta > 0$ [4]. (We assume $\alpha_i^u(\Delta) = 0$ for all $\Delta \leq 0$.) $\alpha_i(\Delta)$ denotes the pair $(\alpha_i^u(\Delta), \alpha_i^l(\Delta))$.

Definition 3. Let $\alpha_i^+(\Delta) = \lim_{\epsilon \rightarrow +0} \alpha_i^u(\Delta + \epsilon)$. 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 $\Delta < 0$.)

Example 2. The widely-studied periodic and sporadic task models are subcases of this more general task model. In both models, consecutive job arrivals of T_i are separated by at least p_i time units, where p_i is the *period* of T_i , and each job requires at most e_i^{\max} execution units. Therefore, under both models, $\alpha_i^u(\Delta) = \left\lceil \frac{\Delta}{p_i} \right\rceil$ and $\gamma_i(k) = k \cdot e_i^{\max}$. The next example illustrates the difference between the functions α_i^u and α_i^+ .

Example 3. 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 $\alpha_i^+(2 \cdot p_i) = \lim_{\epsilon \rightarrow +0} \alpha_i^u(2 \cdot p_i + \epsilon) = 3$. In general, under the sporadic task model, $\alpha_i^+(\Delta) = \left\lceil \frac{\Delta}{p_i} \right\rceil + 1$.

Definition 4. 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 $T_{i,j+1}, \dots, T_{i,j+k}$ 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

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

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

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

Also, we assume that there exists $\bar{e}_i > 0$ and v_i , where $\bar{e}_i = \lim_{k \rightarrow +\infty} \frac{\gamma_i(k)}{k}$, such that

$$\gamma_i(k) \leq \bar{e}_i \cdot k + v_i \text{ for all } k \geq 1. \quad (3)$$

(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 \bar{e}_i denotes the average worst-case job execution time of T_i .

Definition 5. Let $u_i = R_i \cdot \bar{e}_i$. 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, $R_i = \lim_{\Delta \rightarrow +\infty} \left(\left\lceil \frac{\Delta}{p_i} \right\rceil + 1 \right) / \Delta = \frac{1}{p_i}$ and $\bar{e}_i = e_i^{\max}$, so $u_i = R_i \cdot \bar{e}_i = \frac{e_i^{\max}}{p_i}$.

Definition 6. Let $\text{supply}_h(t, \Delta)$ 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 $\text{Supply}(t, \Delta) = \sum_{h=1}^m \text{supply}_h(t, \Delta)$ 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.¹ Note that if individual processor supply guarantees are known, a lower bound on the cumulative guaranteed supply can be computed easily.

Definition 7. Let $\mathcal{B}(\Delta) \leq \text{Supply}(t, \Delta)$ 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

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

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, 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 [9].

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

$$U_{sum} \leq \widehat{U} \quad (5)$$

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.

¹In uniprocessor real-time calculus, the available service is described as the number of incoming events processed by a PE during a time interval.

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.

Definition 8. (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 $T_{i,j} \prec T_{k,h}$.

Example 5. 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 $\chi_{i,j} = r_{i,j}$ [6].

The technical contributions of this paper are twofold. First, given per-task bounds on maximum job response times, we characterize the sequence of job completion events for each task T_i in terms of the next-stage arrival functions $\alpha_i^{u'}$ and $\alpha_i^{l'}$, and the remaining processor supply $\mathcal{B}'(\Delta)$; these, in turn, can serve as inputs to subsequent PEs, thereby resulting in a compositional technique.

Second, given a task set $\tau = \{T_1, \dots, T_n\}$ 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

$$\Theta_i \geq \max_{j \geq 1} (\gamma_i(j) - \mathcal{A}_i^{-1}(j-1)). \quad (6)$$

(It can be shown that the maximum job response time of T_i cannot be less than the right-hand-side of (6). Intuitively, $\gamma_i(j)$ is the maximum execution requirement of j consecutive jobs $T_{i,a}, \dots, T_{i,a+j-1}$ and $\mathcal{A}_i^{-1}(j-1)$ is the minimum length of the interval where jobs $T_{i,a+1}, \dots, T_{i,a+j-1}$ are released.) If Θ_i equals the relative deadline of a job, then the 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.

3. Calculating $\alpha_i^{u'}$ and $\alpha_i^{l'}$

Let $\alpha_i^{u'}(\Delta)$ ($\alpha_i^{l'}(\Delta)$) 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 as follows.

Theorem 1. *If the response time of any job of T_i is at most Θ_i , and $e_i^{\min} > 0$ is the best-case job execution time for task T_i , then $\alpha_i^{u'}(\Delta) \leq \min \left(\left\lceil \frac{\Delta}{e_i^{\min}} \right\rceil, \alpha_i^u(\Delta + \Theta_i - e_i^{\min}) \right)$ and $\alpha_i^{l'}(\Delta) \geq \alpha_i^l(\Delta - \Theta_i + e_i^{\min})$.*

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

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

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, $f_{i,j} - r_{i,j}$ is at least e_i^{\min} . From (7), we thus have $r_{i,N_1} > t_1 - \Theta_i$ and $r_{i,N_2} \leq t_2 - e_i^{\min}$. 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 $(t_1 - \Theta_i, t_2 - e_i^{\min}]$. By Def. 2, we have $N_2 - N_1 + 1 \leq \alpha_i^u(t_2 - e_i^{\min} - t_1 + \Theta_i) = \alpha_i^u(\Delta + \Theta_i - e_i^{\min})$. If job $T_{i,j}$ completes at time $f_{i,j}$, then $T_{i,j+1}$ cannot complete earlier than $f_{i,j} + e_i^{\min}$. Thus, job completions are separated by at least e_i^{\min} time units, and hence, at most $\left\lceil \frac{\Delta}{e_i^{\min}} \right\rceil$ jobs can be completed within any interval of length Δ . \square

4. Calculating $\mathcal{B}'(\Delta)$

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

Definition 9. Let $A(T_i, \gamma)$ be the total amount of time for which jobs of task T_i execute within the set of intervals γ .

Lemma 1. *If the response time of T_i 's jobs is at most Θ_i , then $A(T_i, [t, t + \Delta]) \leq \min(\Delta, \gamma_i(\alpha_i^u(\Delta + \Theta_i)))$.*

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 $[t, t + \Delta)$, $A(T_i, [t, t + \Delta])$, is upper-bounded by the maximum execution demand of T_i 's jobs released within the interval $(t - \Theta_i, t + \Delta]$. By Def. 2, there are at most $\alpha_i^u(\Delta + \Theta_i)$ jobs released within $(t - \Theta_i, t + \Delta]$ and, by Def. 1, their total execution demand is at most $\gamma_i(\alpha_i^u(\Delta + \Theta_i))$. We thus have $A(T_i, [t, t + \Delta]) \leq \gamma_i(\alpha_i^u(\Delta + \Theta_i))$. Also, $A(T_i, [t, t + \Delta])$ cannot exceed the length of the interval $[t, t + \Delta)$. \square

Theorem 2. *If the response time of T_i 's jobs is at most Θ_i , then at least*

$$\mathcal{B}'(\Delta) = \sup_{0 \leq y \leq \Delta} (Z(y)) \quad (8)$$

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

Proof. Consider an interval $[t, t + y)$, where $y \leq \Delta$. By Defs. 6 and 9, the supply that is available after scheduling the tasks in τ in this interval is

$$\begin{aligned}
& \text{Supply}(t, y) - \sum_{T_i \in \tau} A(T_i, [t, t + y)) \\
& \quad \{\text{by Def. 7}\} \\
& \geq \max\left(0, \mathcal{B}(y) - \sum_{T_i \in \tau} A(T_i, [t, t + y))\right) \\
& \quad \{\text{by Lemma 1}\} \\
& \geq \max\left(0, \mathcal{B}(y) - \sum_{T_i \in \tau} \min(\Delta, \gamma_i(\alpha_i^u(y + \Theta_i)))\right).
\end{aligned}$$

Additionally, $\text{Supply}(t, \Delta) \geq \sup_{0 \leq y \leq \Delta} (\text{Supply}(t, y))$. 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

$$\begin{aligned}
Z(y) & \geq \max\left(0, \max(0, \widehat{U} \cdot (y - \sigma_{tot}))\right. \\
& \quad \left. - \sum_{T_i \in \tau} \min(y, \gamma_i(\alpha_i^u(y + \Theta_i)))\right) \\
& \geq \max\left(0, \widehat{U} \cdot (y - \sigma_{tot})\right. \\
& \quad \left. - \sum_{T_i \in \tau} \min(y, E_i(\alpha_i^u(y + \Theta_i)))\right) \\
& \quad \{\text{by (2) and (3)}\} \\
& \geq \max\left(0, \widehat{U} \cdot (y - \sigma_{tot})\right. \\
& \quad \left. - \sum_{T_i \in \tau} (\bar{e}_i \cdot (R_i \cdot (y + \Theta_i) + B_i) + v_i)\right) \\
& \quad \{\text{by Def. 5}\} \\
& = \max\left(0, \widehat{U} \cdot (y - \sigma_{tot})\right. \\
& \quad \left. - \sum_{T_i \in \tau} (u_i \cdot y + u_i \cdot \Theta_i + \bar{e}_i \cdot B_i + v_i)\right) \\
& = \max\left(0, \widehat{U} \cdot (y - \sigma_{tot})\right. \\
& \quad \left. - U_{sum} \cdot y + \sum_{T_i \in \tau} (u_i \cdot \Theta_i + \bar{e}_i \cdot B_i + v_i)\right) \\
& = \max\left(0, (\widehat{U} - U_{sum}) \cdot y\right. \\
& \quad \left. - \widehat{U} \cdot \sigma_{tot} - \sum_{T_i \in \tau} (u_i \cdot \Theta_i + \bar{e}_i \cdot B_i + v_i)\right)
\end{aligned}$$

$$\begin{aligned}
& \left. \begin{array}{l} \text{by the definition of } \widehat{U}' \text{ and } \sigma'_{tot} \\ \text{in the statement of the theorem} \end{array} \right\} \\
& = \max\left(0, \widehat{U}' \cdot (y - \sigma'_{tot})\right).
\end{aligned}$$

Finally, by (8), $\mathcal{B}'(\Delta) \geq \sup_{0 \leq y \leq \Delta} (\max(0, \widehat{U}' \cdot (y - \sigma'_{tot}))) = \max(0, \widehat{U}' \cdot (\Delta - \sigma'_{tot}))$. \square

5. Multiprocessor Schedulability Test

Our remaining proof obligation is to find a response-time bound Θ_i for each task T_i . In this section, we present a schedulability test (given in Corollary 1 later in this section) that checks that pre-defined response-time bounds are not exceeded.

As noted earlier, the way jobs are prioritized according to Def. 8 is similar to GEDF. A number of GEDF schedulability tests have been developed assuming that jobs arrive periodically or sporadically (e.g., [1, 2, 8]). In this paper, we extend techniques from [1] and [8] in order to incorporate more general job arrivals and execution models.

Similarly to [5], 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 $T_{a, b} \prec T_{\ell, q}$,

$$f_{a, b} \leq r_{a, b} + \Theta_a. \quad (9)$$

We consider 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 computed for each value of $k \in [1, K_\ell]$ (see Def. 4) and $\delta_\ell(k)$ (defined later in this section), which determine its length, $\delta_\ell(k) + \Theta_\ell$. During this interval, we consider demand due to competing higher-priority jobs that can interfere with $T_{\ell, q}$. We then perform the following three steps:

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

S2: Given a finite upper bound $M^*(\delta_\ell(k), \ell, k, \tau, m)$ 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 a task's response-time bound is not violated by setting $M^*(\delta_\ell(k), \ell, k, \tau, m) + (m - 1) \cdot (E_\ell^*(k) - 1) < \mathcal{B}(\delta_\ell(k) + \Theta_\ell)$.

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

5.1. 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. Since job priorities remain fixed, we also ignore jobs that have lower priority than $T_{\ell,q}$.

We start the derivation by proving the following lemma and claims.

Lemma 2. $r_{\ell,q} - r_{\ell,q-i} \geq \mathcal{A}_{\ell}^{-1}(i)$.

Proof. Let $\Delta' = r_{\ell,q} - r_{\ell,q-i}$. Let

$$\Delta^* = \inf\{\Delta \mid \alpha_{\ell}^+(\Delta) \geq i + 1\}. \quad (10)$$

Because jobs $T_{\ell,q-i}, \dots, T_{\ell,q}$ are released within the interval $[r_{\ell,q-i}, r_{\ell,q}]$, by Def. 3, $\alpha_{\ell}^+(\Delta') \geq i + 1$. Therefore, by (10),

$$r_{\ell,q} - r_{\ell,q-i} = \Delta' \geq \Delta^*. \quad (11)$$

We now consider two cases.

Case 1: $\alpha_{\ell}^u(\Delta^*) > i$. In this case, $\Delta^* \stackrel{\text{by Def. 3}}{\geq} \inf\{\Delta \mid \alpha_{\ell}^u(\Delta) > i\} \stackrel{\text{by Def. 4}}{=} \mathcal{A}_{\ell}^{-1}(i)$. The lemma follows from this and (11).

Case 2: $\alpha_{\ell}^u(\Delta^*) \leq i$. Because $\alpha_{\ell}^u(\Delta)$ is non-decreasing,

$$\alpha_{\ell}^u(\Delta) \leq i \text{ for each } \Delta \leq \Delta^*. \quad (12)$$

Further, by (10),

$$\alpha^+(\Delta) < i + 1, \text{ for each } \Delta < \Delta^* \quad (13)$$

Suppose that for some $\Delta'' > \Delta^*$, $\alpha_{\ell}^u(\Delta'') \leq i$. Because $\alpha_{\ell}^u(\Delta)$ is non-decreasing, $\alpha_{\ell}^u(\Delta_x) \leq i$ for each $\Delta_x \in [\Delta^*, \Delta'']$. The latter implies $\alpha_{\ell}^+(\Delta_x) = \lim_{\epsilon \rightarrow +0} \alpha_{\ell}^u(\Delta_x + \epsilon) \leq i$ for each $\Delta_x \in [\Delta^*, \Delta'']$. From this and (13), we have $\alpha^+(\Delta) < i + 1$ for each $\Delta < \Delta''$. Since $\Delta'' > \Delta^*$, we have a contradiction to (10). Therefore, $\alpha_{\ell}^u(\Delta) > i$ for each $\Delta > \Delta^*$. From this and (12), we have $\Delta^* = \inf\{\Delta \mid \alpha_{\ell}^u(\Delta) > i\} \stackrel{\text{by Def. 4}}{=} \mathcal{A}_{\ell}^{-1}(i)$. The lemma follows from this equality and (11). \square

Claim 1. For $i \geq 1$, $f_{\ell,q-i} \leq r_{\ell,q} + \Theta_{\ell} - \mathcal{A}_{\ell}^{-1}(i)$.

Proof. By (9), $f_{\ell,q-i} \leq r_{\ell,q-i} + \Theta_{\ell} = r_{\ell,q-i} - r_{\ell,q} + r_{\ell,q} + \Theta_{\ell} \stackrel{\text{by Lemma 2}}{\leq} r_{\ell,q} + \Theta_{\ell} - \mathcal{A}_{\ell}^{-1}(i)$. \square

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

Proof. By (1), $-\mathcal{A}_{\ell}^{-1}(K_{\ell}) \leq -\gamma_{\ell}(K_{\ell})$. Setting this and $i = K_{\ell}$ into Claim 1, we get the required result. \square

Claim 3: For $i \geq 1$, $r_{\ell,q-i+1} \leq r_{\ell,q} + \Theta_{\ell} - \gamma_{\ell}(i)$.

Proof.

$$\begin{aligned} r_{\ell,q-i+1} &= r_{\ell,q} - (r_{\ell,q} - r_{\ell,q-i+1}) \\ &\quad \{\text{by Lemma 2}\} \\ &\leq r_{\ell,q} - \mathcal{A}_{\ell}^{-1}(i-1) \\ &\quad \{\text{because, by (6), } \Theta_{\ell} \geq \gamma_{\ell}(i) - \mathcal{A}_{\ell}^{-1}(i-1)\} \\ &\leq r_{\ell,q} + \Theta_{\ell} - \gamma_{\ell}(i) \quad \square \end{aligned}$$

Claim 4: $\Theta_{\ell} \geq \gamma_{\ell}(1)$.

Proof. By (6), $\Theta_{\ell} \geq \max_{j \geq 1}(\gamma_{\ell}(j) - \mathcal{A}_{\ell}^{-1}(j-1)) \geq \gamma_{\ell}(1) - \mathcal{A}_{\ell}^{-1}(0)$. By Def. 4, $\mathcal{A}_{\ell}^{-1}(0) = 0$. \square

Job $T_{\ell,q}$ can violate its response-time bound for the following reasons. If $T_{\ell,q-1}$ completes by time $r_{\ell,q} + \Theta_{\ell} - \gamma_{\ell}(1)$, 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 $r_{\ell,q} + \Theta_{\ell} - \gamma_{\ell}(1)$, which can happen if the minimum job inter-arrival time for T_{ℓ} is less than $\gamma_{\ell}(1)$. 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 4, $\Theta_{\ell} \geq \gamma_{\ell}(1)$). 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 $f_{\ell,q-2} \leq r_{\ell,q} + \Theta_{\ell} - \gamma_{\ell}(2)$, 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 $f_{\ell,q-2} > r_{\ell,q} + \Theta_{\ell} - \gamma_{\ell}(2)$, 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 $T_{\ell,q-1}, \dots, T_{\ell,q-K_{\ell}}$ since, by Claim 2, $f_{\ell,q-K_{\ell}} \leq r_{\ell,q} + \Theta_{\ell} - \gamma_{\ell}(K_{\ell})$. Assuming that, for job $T_{\ell,q-k}$, $f_{\ell,q-k} \leq r_{\ell,q} + \Theta_{\ell} - \gamma_{\ell}(k)$, we define the *problem window* for jobs $T_{\ell,q-k+1}, \dots, T_{\ell,q}$ 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 10. Let $\lambda \in [1, K_{\ell}]$ be the smallest integer such that $f_{\ell,q-\lambda} \leq r_{\ell,q} + \Theta_{\ell} - \gamma_{\ell}(\lambda)$. By Claim 2, such a λ exists.

Claim 5. 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 T_ℓ is ready continuously within $[r_{\ell,q-k+1}, f_{\ell,q})$ for each $k \in [1, \lambda]$. 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 $f_{\ell,q-j} > r_{\ell,q} + \Theta_\ell - \gamma_\ell(j)$ for each $j \in [1, \lambda)$, by the selection of λ . By Claim 3, this implies $f_{\ell,q-j} > r_{\ell,q-j+1}$. 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 11. Let $W(T_{i,y}, t)$ denote the remaining execution time for job $T_{i,j}$ (if any) after time t . Let $W(T_i, t) = \sum_{T_{i,j} \preceq T_{\ell,q}} W(T_{i,y}, t)$. In the appendix, we prove

$$W(T_\ell, r_{\ell,q-\lambda+1}) \leq r_{\ell,q} + \Theta_\ell - r_{\ell,q-\lambda+1}. \quad (14)$$

In Fig. 2, which shows a response-time bound violation for job $T_{\ell,q}$ where $\lambda = 1$, $W(T_\ell, r_{\ell,q-\lambda+1})$ corresponds to the execution demand of job $T_{\ell,q}$ and the unfinished work of job $T_{\ell,q-1}$ at time $r_{\ell,q}$.

Definition 12. Let $\Gamma_\lambda \subseteq [r_{\ell,q-\lambda+1}, r_{\ell,q} + \Theta_\ell)$ be the set of intervals where no available processor is idle as shown in Fig. 2. Let $\overline{\Gamma}_\lambda = [r_{\ell,q-\lambda+1}, r_{\ell,q} + \Theta_\ell) \setminus \Gamma_\lambda$. We let $|\Gamma_\lambda|$ ($|\overline{\Gamma}_\lambda|$) denote the total length of the intervals in Γ_λ , ($\overline{\Gamma}_\lambda$).

Lemma 3. *If the response-time bound for $T_{\ell,q}$ is violated (as we have assumed), then $|\Gamma_\lambda| = r_{\ell,q} + \Theta_\ell - r_{\ell,q-\lambda+1} - W(T_\ell, r_{\ell,q-\lambda+1}) + 1 + \mu$, where $\mu \geq 0$. (Note that, by (14), $|\Gamma_\lambda| > 0$). Additionally, T_ℓ executes within each instant of $\overline{\Gamma}_\lambda$, and $|\overline{\Gamma}_\lambda| = W(T_\ell, r_{\ell,q-\lambda+1}) - 1 - \mu$.*

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

$$|\Gamma_\lambda| < r_{\ell,q} + \Theta_\ell - r_{\ell,q-\lambda+1} - W(T_\ell, r_{\ell,q-\lambda+1}) + 1. \quad (15)$$

Under these conditions, the total length of the intervals in $\overline{\Gamma}_\lambda$, where at least one available processor is idle, is $r_{\ell,q} + \Theta_\ell - r_{\ell,q-\lambda+1} - |\Gamma_\lambda| \stackrel{\text{by (15)}}{>} W(T_\ell, r_{\ell,q-\lambda+1}) - 1$. Thus, this total length is at least $W(T_\ell, r_{\ell,q-\lambda+1})$, as time is integral. By Claim 5, T_ℓ executes at each time $t \in \overline{\Gamma}_\lambda$, and thus completes by time $r_{\ell,q} + \Theta_\ell$, which is a contradiction. \square

Definition 13. Let $\tau_p(t) = \{T_a \mid \text{for some } y, T_{a,y} \text{ is ready at time } t \text{ and } T_{a,y} \preceq T_{\ell,q}\}$. (The subscript p denotes the fact that these jobs have higher or equal priority.)

Definition 14. Let $t_0(k) \leq r_{\ell,q-k+1}$ be the earliest instant such that $\forall t \in [t_0(k), r_{\ell,q-k+1})$, $|\tau_p(t)| \geq m$ or fewer than $|\tau_p(t)|$ tasks from $\tau_p(t)$ execute at time t . If such an instant does not exist, then let $t_0(k) = r_{\ell,q-k+1}$.

Def. 14 generalizes the well-known concept of an *idle instant* in uniprocessor scheduling. We call an interval $[t_1, t_2)$ *busy* if no available processor is idle within it.

Claim 6. *No available processor is idle within $[t_0(\lambda), r_{\ell,q-\lambda+1})$.*

Proof. Suppose that an available processor is idle at time $t \in [t_0(\lambda), r_{\ell,q-\lambda+1})$. Because the scheduler being analyzed is work-conserving, all tasks in $\tau_p(t)$ execute at time t and thus $|\tau_p(t)| \leq m - 1$, which violates Def. 14. \square

Definition 15. Let $\delta_\ell(k) = r_{\ell,q} - t_0(k)$.

Note that, by Def. 14 and Lemma 2, $\delta_\ell(k) = r_{\ell,q} - t_0(k) \geq r_{\ell,q} - r_{\ell,q-k+1} \geq \mathcal{A}_\ell^{-1}(k - 1)$.

Our schedulability test for task T_ℓ is based upon summing the competing demand of tasks in τ within the interval $[t_0(\lambda), r_{\ell,q} + \Theta_\ell)$, which has length $\delta_\ell(\lambda) + \Theta_\ell$, and the unavailable time within this interval.

Definition 16. Let $E_\ell^*(k)$ be a finite function of k such that $W(T_\ell, r_{\ell,q-\lambda+1}) \leq E_\ell^*(\lambda)$. Let $W(t) = \sum_{T_i \in \tau} W(T_i, t)$. Let $M^*(\delta_\ell(k), \ell, k, m, \tau)$ be a finite function of $\delta_\ell(k)$, ℓ , k , m , and τ such that $W(t_0(k)) \leq M^*(\delta_\ell(k), \ell, k, m, \tau)$. (As mentioned earlier at the beginning of Sec. 5, $M^*(\delta_\ell(k), \ell, k, m, \tau)$ and $E_\ell^*(k)$ are calculated in order to test whether the response-time bound of T_ℓ is not violated. Later, in Sec. 5.2, we explain how $M^*(\delta_\ell(k), \ell, k, m, \tau)$ and $E_\ell^*(k)$ are calculated.)

Definition 17. We require that there exists a constant $H_\ell \geq 0$ such that, for all $\delta_\ell(k) \geq \mathcal{A}_\ell^{-1}(k - 1)$,

$$M^*(\delta_\ell(k), \ell, k, m, \tau) \leq U_{sum} \cdot \delta_\ell(k) + H_\ell. \quad (16)$$

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

Definition 18. Let $\delta_\ell^{\max}(k) = \lfloor (H_\ell + (m - 1) \cdot (E_\ell^*(k) - 1) + \widehat{U} \cdot \sigma_{tot} - \Theta_\ell \cdot \widehat{U}) / (\widehat{U} - U_{sum}) \rfloor$.

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

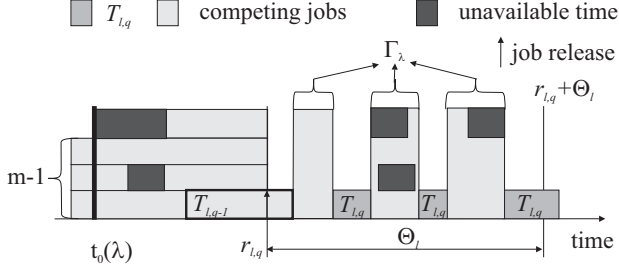


Figure 2. Conditions for response-time bound violation for $\lambda = 1$.

Theorem 3. *If the response-time bound Θ_ℓ is violated for $T_{\ell,q}$ (as we have assumed), then, for some $k \in [1, K_\ell]$ and $\delta_\ell(k) \in [\mathcal{A}_\ell^{-1}(k-1), \delta_\ell^{\max}(k)]$,*

$$M^*(\delta_\ell(k)) + (m-1) \cdot (E_\ell^*(k) - 1) \geq \mathcal{B}(\delta_\ell(k) + \Theta_\ell). \quad (17)$$

Proof. Consider job $T_{\ell,q}$, $k = \lambda$, and time instants $r_{\ell,q-\lambda+1}$ and $t_0(\lambda)$ as defined in Defs. 10 and 14. To establish the necessary condition, we sum the processor allocations within the intervals $[t_0(\lambda), r_{\ell,q-\lambda+1}] \cup \Gamma_\lambda$ and $\overline{\Gamma}_\lambda$. By Def. 12 and Claim 6, the total processor allocation (including unavailable time) within $[t_0(\lambda), r_{\ell,q-\lambda+1}] \cup \Gamma_\lambda$ is $m \cdot (r_{\ell,q-\lambda+1} - t_0(\lambda)) + m \cdot |\Gamma_\lambda|$ (see Fig. 2; note that $r_{\ell,q-\lambda+1} = r_{\ell,q}$ here). Also, Lemma 3 implies that the total processor allocation (including unavailable time) within $\overline{\Gamma}_\lambda$ is at least $W(T_\ell, r_{\ell,q-\lambda+1}) - 1 - \mu$, where $\mu \geq 0$.

The total processor allocation (including unavailable time) within $[t_0(\lambda), r_{\ell,q} + \Theta_\ell]$ is thus at least $m \cdot (r_{\ell,q-\lambda+1} - t_0(\lambda)) + m \cdot |\Gamma_\lambda| + |\overline{\Gamma}_\lambda| \stackrel{\text{by Lemma 3}}{=} m \cdot (r_{\ell,q-\lambda+1} - t_0(\lambda)) + m \cdot (r_{\ell,q} + \Theta_\ell - r_{\ell,q-\lambda+1} - W(T_\ell, r_{\ell,q-\lambda+1}) + 1 + \mu) + W(T_\ell, r_{\ell,q-\lambda+1}) - 1 - \mu = m \cdot (r_{\ell,q} + \Theta_\ell - t_0(\lambda)) - (m-1) \cdot (W(T_\ell, r_{\ell,q-\lambda+1}) - 1) + (m-1) \cdot \mu \stackrel{\text{by Def. 15}}{=} m \cdot (\delta_\ell(\lambda) + \Theta_\ell) - (m-1) \cdot (W(T_\ell, r_{\ell,q-\lambda+1}) - 1) + (m-1) \cdot \mu$.

Let $\text{Res}_h([t_0(\lambda), r_{\ell,q} + \Theta_\ell])$ be the amount of time that is not available on processor h at time instants in the interval $[t_0(\lambda), r_{\ell,q} + \Theta_\ell]$. By Defs. 11 and 16, the allocation of jobs within $[t_0(\lambda), r_{\ell,q} + \Theta_\ell]$ is upper-bounded by $W(t_0(\lambda))$ (recall that we are ignoring lower-priority jobs). Thus,

$$\begin{aligned} W(t_0(\lambda)) + \sum_{h=1}^m \text{Res}_h([t_0(\lambda), r_{\ell,q} + \Theta_\ell]) \\ \geq m \cdot (\delta_\ell(\lambda) + \Theta_\ell) - (m-1) \cdot (W(T_\ell, r_{\ell,q-\lambda+1}) - 1). \end{aligned} \quad (18)$$

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

Def. 6,

$$\begin{aligned} \text{Res}_h([t_0(\lambda), r_{\ell,q} + \Theta_\ell]) \\ = (r_{\ell,q} + \Theta_\ell - t_0(\lambda)) - \text{supply}_h(t_0(\lambda), r_{\ell,q} + \Theta_\ell - t_0(\lambda)) \\ \quad \{\text{by Def. 15}\} \\ = (\delta_\ell(\lambda) + \Theta_\ell) - \text{supply}_h(t_0(\lambda), \delta_\ell(\lambda) + \Theta_\ell). \end{aligned} \quad (19)$$

Summing (19) for all h , we have

$$\begin{aligned} \sum_{h=1}^m \text{Res}_h([t_0(\lambda), r_{\ell,q} + \Theta_\ell]) \\ = \sum_{h=1}^m ((\delta_\ell(\lambda) + \Theta_\ell) - \text{supply}_h(t_0(\lambda), \delta_\ell(\lambda) + \Theta_\ell)) \\ \quad \{\text{by Def. 6}\} \\ = m \cdot (\delta_\ell(\lambda) + \Theta_\ell) - \text{Supply}(t_0(\lambda), \delta_\ell(\lambda) + \Theta_\ell) \\ \quad \{\text{by Def. 7}\} \\ \leq m \cdot (\delta_\ell(\lambda) + \Theta_\ell) - \mathcal{B}(\delta_\ell(\lambda) + \Theta_\ell). \end{aligned} \quad (20)$$

Setting (20) into (18), we have

$$\begin{aligned} W(t_0(\lambda)) + m \cdot (\delta_\ell(\lambda) + \Theta_\ell) - \mathcal{B}(\delta_\ell(\lambda) + \Theta_\ell) \\ \geq m \cdot (\delta_\ell(\lambda) + \Theta_\ell) - (m-1) \cdot (W(T_\ell, r_{\ell,q-\lambda+1}) - 1). \end{aligned}$$

Rearranging the terms in the above inequality, we have

$$\begin{aligned} W(t_0(\lambda)) + (m-1) \cdot (W(T_\ell, r_{\ell,q-\lambda+1}) - 1) \\ \geq \mathcal{B}(\delta_\ell(\lambda) + \Theta_\ell). \end{aligned}$$

Setting $E_\ell^*(\lambda)$ and $M^*(\delta_\ell(\lambda))$ as defined in Def. 16 into the inequality above, we get (17). (Note that, by Def. 10, $\lambda \in [1, K_\ell]$.)

Our remaining proof obligation is to establish the stated range for $\delta_\ell(k)$. By (16) and (17),

$$U_{\text{sum}} \cdot \delta_\ell(k) + H_\ell + (m-1) \cdot (E_\ell^*(k) - 1) \geq \mathcal{B}(\delta_\ell(k) + \Theta_\ell). \quad (21)$$

Applying (4) to (21), we have

$$\begin{aligned} U_{\text{sum}} \cdot \delta_\ell(k) + H_\ell + (m-1) \cdot (E_\ell^*(k) - 1) \\ \geq \max(0, \widehat{U} \cdot (\delta_\ell(k) + \Theta_\ell - \sigma_{\text{tot}})) \\ \geq \widehat{U} \cdot (\delta_\ell(k) + \Theta_\ell - \sigma_{\text{tot}}). \end{aligned}$$

Solving the latter inequality for $\delta_\ell(k)$, we have $\delta_\ell(k) \leq (H_\ell + (m-1) \cdot (E_\ell^*(k) - 1) + \widehat{U} \cdot \sigma_{\text{tot}} - \Theta_\ell \cdot \widehat{U}) / (\widehat{U} - U_{\text{sum}})$. Because $\delta_\ell(k)$ is integral (refer to Def. 15), by Def. 18, $\delta_\ell(k) \leq \delta_\ell^{\max}(k)$. The theorem follows. \square

Corollary 1. (Schedulability Test) *If, for task T_ℓ , (17) does not hold for each $k \in [1, K_\ell]$ and $\delta_\ell(k) \in [\mathcal{A}_\ell^{-1}(k-1), \delta_\ell^{\max}(k)]$, then the response-time bound for T_ℓ is not violated.*

The term $(m - 1) \cdot (E_\ell^*(k) - 1)$ in (17) can be large if u_ℓ and Θ_ℓ are large. For large values of Θ_ℓ and certain schedulers such as GEDF and FIFO, this term can be replaced 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 Def. 7). 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 (see Sec. 7 for details).

5.2. Step S3 (Calculating $M^*(\delta_\ell(k))$ 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 Def. 8. Therefore, Corollary 1 is applicable to all fixed job-priority scheduling policies (these policies include preemptive variants of EDF, FIFO, static-priority policies, and various combinations of these policies; non-preemptive variants can be supported similarly) provided the functions $M^*(\delta_\ell(k))$ (and its linear upper bound in Def. 17) and $E_\ell^*(k)$ are known. $M^*(\delta_\ell(k))$ and $E_\ell^*(k)$ can be derived for a particular algorithm by extending techniques from previously-published papers on the schedulability of sporadic tasks [1, 7] to incorporate more general arrival and execution patterns.

In this section, we derive the functions $E_\ell^*(k)$ and $M^*(\delta_\ell(k))$ for a 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).

Derivation of $M^*(\delta_\ell(k))$. To derive $M^*(\delta_\ell(k))$, we first note that only jobs $T_{a,b} \preceq T_{\ell,q}$ can compete with $T_{\ell,q}$ or its predecessors.

Definition 19. Let $T_{a,b}$ be the earliest pending job of T_a at time $t_0(k)$. We separate the tasks that may compete with $T_{\ell,q}$ into two disjoint sets:

$$\begin{aligned} \mathbf{HC} &= \{T_a :: (T_{a,b} \text{ exists}) \wedge (r_{a,b} < t_0(k)) \wedge (T_{a,b} \preceq T_{\ell,q})\}; \\ \mathbf{NC} &= \{T_a :: (T_{a,b} \text{ does not exist}) \\ &\quad \vee [(r_{a,b} = t_0(k)) \wedge (T_{a,b} \preceq T_{\ell,q})]\}. \end{aligned}$$

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

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

Proof. By Defs. 13 and 19, $\mathbf{HC} \subseteq \tau_p(t_0(k) - 1)$. By Def. 14, 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, $|\mathbf{HC}| \leq m - 1$. \square

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

$$W(t_0(k)) \leq \sum_{T_i \in \mathbf{HC}} W_{\mathbf{HC}}(T_i, \delta_\ell(k)) + \sum_{T_i \in \mathbf{NC}} W_{\mathbf{NC}}(T_i, \delta_\ell(k)). \quad (22)$$

We provide expressions for computing $W_{\mathbf{NC}}(T_i, \delta_\ell(k))$ and $W_{\mathbf{HC}}(T_i, \delta_\ell(k))$ in the following two lemmas. Their proofs can be found in the appendix.

Lemma 4: $W_{\mathbf{NC}}(T_i, \delta_\ell(k)) = \gamma_i(\alpha_i^+(\delta_\ell(k) + D_\ell - D_i))$.

Lemma 5. Let $G_i(S, X) = \min(\gamma_i(S), \max(0, X - \mathcal{A}_\ell^{-1}(S - 1)) + \gamma_i(S - 1))$.

$$W_{\mathbf{HC}}(T_i, \delta_\ell(k)) = G_i(\alpha_i^u(\delta_\ell(k) + D_\ell - D_i + \Theta_i), \delta_\ell(k) + D_\ell - D_i + \Theta_i)$$

Claim 8: $r_{\ell,q} - r_{\ell,q-\lambda+1} \leq \max(0, \gamma_\ell(\lambda - 1) - 1)$.

Proof. If $\lambda = 1$, then $r_{\ell,q} - r_{\ell,q-\lambda+1} = 0$. Alternatively, if $\lambda > 1$, then, by (9), $r_{\ell,q-\lambda+1} + \Theta_\ell \geq f_{\ell,q-\lambda+1} > r_{\ell,q} + \Theta_\ell - \gamma_\ell(\lambda - 1)$, where the last inequality follows from Def. 10. Therefore, $r_{\ell,q} - r_{\ell,q-\lambda+1} \leq \gamma_\ell(\lambda - 1) - 1$, as time is integral. \square

The function $E_\ell^*(k)$ can be derived as a simple corollary of Lemma 5.

Corollary 2. Let $Q(k) = \max(0, \gamma_\ell(k - 1) - 1) + \Theta_\ell$. If

$$E_\ell^*(k) = G_\ell(\alpha_\ell^u(Q(k)), Q(k)),$$

then $E_\ell^*(k)$ complies with Def. 16.

Proof. By Def. 16, the function $E_\ell^*(k)$ upper bounds $W(T_\ell, r_{\ell,q-k+1})$, and hence, it can be computed as $W_{\mathbf{HC}}(T_\ell, \delta_\ell(k))$, for $\delta_\ell(k) = r_{\ell,q} - r_{\ell,q-k+1}$. For $k = \lambda$, we have

$$\begin{aligned} &W(T_\ell, r_{\ell,q-\lambda+1}) \\ &\leq W_{\mathbf{HC}}(T_\ell, r_{\ell,q} - r_{\ell,q-\lambda+1}) \\ &\quad \{\text{by Lemma 5}\} \\ &= G_\ell(\alpha_\ell^u(r_{\ell,q} - r_{\ell,q-\lambda+1} + \Theta_\ell), r_{\ell,q} - r_{\ell,q-\lambda+1} + \Theta_\ell) \\ &\quad \{\text{by Claim 8}\} \\ &\leq G_\ell(\alpha_\ell^u(\max(0, \gamma_\ell(\lambda - 1) - 1) + \Theta_\ell), \\ &\quad \max(0, \gamma_\ell(\lambda - 1) - 1) + \Theta_\ell). \quad \square \end{aligned}$$

To continue our derivation of $M^*(\delta_\ell(k))$, we set

$$\begin{aligned} &M^*(\delta_\ell(k)) \\ &= \max \left(\sum_{T_i \in \mathbf{HC}} W_{\mathbf{HC}}(T_i, \delta_\ell(k)) + \sum_{T_i \in \mathbf{NC}} W_{\mathbf{NC}}(T_i, \delta_\ell(k)) \right), \quad (23) \end{aligned}$$

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

$$\mathbf{NC} \cup \mathbf{HC} \subseteq \tau \quad \mathbf{NC} \cap \mathbf{HC} = \emptyset \quad |\mathbf{HC}| \leq m - 1 \quad \} \quad (24)$$

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

By the selection of λ in Def. 10, (22), and (23), $M^*(\delta_\ell(k))$ upper-bounds $W(t_0(k))$ so it complies with Def. 16. In order to use Corollary 1, we are left to find constant H_ℓ such that (16) holds so that $M^*(\delta_\ell(k))$ given by (23) complies with Def. 17.

Definition 20. Let $L_i(X) = \max(0, u_i \cdot X + \bar{e}_i \cdot B_i) + v_i$ for any X .

Lemma 6. (Proved in the appendix) For all $\delta_\ell(k) \geq \mathcal{A}_\ell^{-1}(k)$, $M^*(\delta_\ell(k)) \leq U_{sum} \cdot \delta_\ell(k) + H_\ell$, where $H_\ell = \sum_{T_i \in \tau} L_i(D_\ell - D_i) + U(m - 1) \cdot \max(\Theta_i)$ and $U(y)$ is the sum of $\min(y, |\tau|)$ largest utilizations.

Given an expression for H_ℓ , we can compute $\delta_\ell^{\max}(k)$ in Def. 18 for any given k . Given expressions for $\delta_\ell^{\max}(k)$, $M^*(\delta_\ell(k))$, and $E_\ell^*(k)$, we can apply Corollary 1 to check that each task $T_\ell \in \tau$ meets its response-time bound. In the next section, we identify conditions under which the test is applicable and discuss its time complexity.

6. Computational Complexity of the Test

According to Corollary 1, (17) needs to be checked for violation for all $k \in [1, K_\ell]$ and $\delta_\ell(k) \in [\mathcal{A}_\ell^{-1}(k - 1), \delta_\ell^{\max}(k)]$.

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

Proof. We start with estimating the complexity of checking (17). The values of $\alpha_i^u(\Delta)$, $\gamma_i(k)$, $\mathcal{A}_i^{-1}(k)$, and $\mathcal{B}(\Delta)$ can be computed in constant time if $\alpha_i^u(\Delta)$ and $\gamma_i(k)$ consist of periodic and aperiodic piecewise-linear parts and $\mathcal{B}(\Delta)$ is also piecewise-linear. These assumptions are used in prior work on the Real-Time Calculus Toolbox [13] and are sufficient for practical purposes.

Under these assumptions, $M^*(\delta_\ell(k))$ for a given value of $\delta_\ell(k)$ can be computed in $O(n)$ time, where n is the number of tasks.

The calculations above need to be repeated for all $k \in [1, K_\ell]$ and all integers in $[\mathcal{A}_\ell^{-1}(k - 1), \delta_\ell^{\max}(k)]$. By Def. 18, $\delta_\ell^{\max}(k)$ is finite if its denominator is nonzero. By

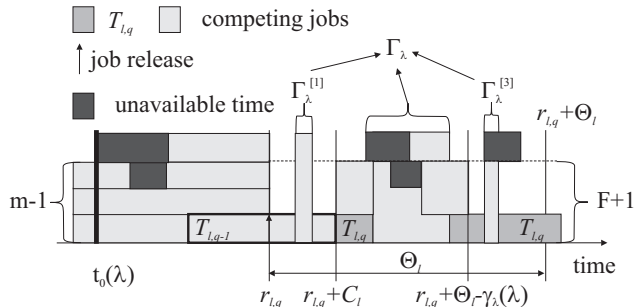


Figure 3. Conditions for response-time bound violation for $\lambda = 1$.

(5), we have $U_{sum} \leq \hat{U}$. Therefore, $\delta_\ell^{\max}(k)$ is finite if (5) is strict. \square

Checking that (17) 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.

7. Schedulability Test for GEDF-like Schedulers

As mentioned earlier, the equation (17) for checking that the response-time bound of T_ℓ is not violated can be unnecessarily pessimistic if u_ℓ and Θ_ℓ are large.

In this section, we improve (17) for a 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).

Definition 21. Let $C_\ell = D_\ell - \min_{T_i \in \tau} (D_i)$.

Claim 9. If $T_{i,y} \preceq T_{\ell,q}$, then $r_{i,y} \leq r_{\ell,q} + D_\ell - D_i$, for $y \geq 0$. No job $T_{i,j} \preceq T_{\ell,q}$, can be released after $r_{\ell,q} + C_\ell$.

Proof. The claim immediately follows from Defs. 8 and 21. \square

To establish the necessary condition, we sum the processor allocations within the intervals $[t_0(\lambda), r_{\ell,q - \lambda + 1}] \cup \Gamma_\lambda$ and $\overline{\Gamma}_\lambda$. In the rest of this section, we assume that $\Theta_\ell > \gamma_\ell(\lambda) + C_\ell$.

Definition 22. Let $\Gamma_\lambda^{[1]} = [r_{\ell,q - \lambda + 1}, r_{\ell,q} + C_\ell] \cap \Gamma_\lambda$ and $\Gamma_\lambda^{[3]} = [r_{\ell,q} + \Theta_\ell - \gamma_\ell(\lambda), r_{\ell,q} + \Theta_\ell] \cap \Gamma_\lambda$, as shown in Fig. 3.

Definition 23. Let A_0 , A_1 , A_2 and A_3 be the total processor allocation (including unavailable time) within $[t_0(\lambda), r_{\ell,q - \lambda + 1}] \cup \Gamma_\lambda$, $[r_{\ell,q - \lambda + 1}, r_{\ell,q} + C_\ell] \cap \overline{\Gamma}_\lambda$, $[r_{\ell,q} + C_\ell, r_{\ell,q} + \Theta_\ell - \gamma_\ell(\lambda)] \cap \overline{\Gamma}_\lambda$, and $[r_{\ell,q} + \Theta_\ell - \gamma_\ell(\lambda), r_{\ell,q} + \Theta_\ell] \cap \overline{\Gamma}_\lambda$, respectively.

Lemma 7: $A_0 = m \cdot (\delta_\ell(\lambda) + \Theta_\ell) - m \cdot (W(T_\ell, r_{\ell, q-\lambda+1} - 1) + m \cdot \mu)$.

Proof. By Def. 12 and Claim 6, the total processor allocation (including unavailable time) within $[t_0(\lambda), r_{\ell, q-\lambda+1}] \cup \Gamma_\lambda$ is

$$\begin{aligned}
A_0 &= m \cdot (r_{\ell, q-\lambda+1} - t_0(\lambda)) + m \cdot |\Gamma_\lambda| \\
&= m \cdot (r_{\ell, q-\lambda+1} - t_0(\lambda)) + m \cdot (r_{\ell, q} + \Theta_\ell - r_{\ell, q-\lambda+1} - W(T_\ell, r_{\ell, q-\lambda+1} + 1 + \mu)) \\
&= m \cdot (r_{\ell, q} - t_0(\lambda) + \Theta_\ell) - m \cdot (W(T_\ell, r_{\ell, q-\lambda+1} - 1) + m \cdot \mu) \\
&\quad \{\text{by Def. 15}\} \\
&= m \cdot (\delta_\ell(\lambda) + \Theta_\ell) - m \cdot (W(T_\ell, r_{\ell, q-\lambda+1} - 1) + m \cdot \mu) \quad \square
\end{aligned}$$

Lemma 8: $A_1 \geq r_{\ell, q} + C_\ell - r_{\ell, q-\lambda+1} - |\Gamma_\lambda^{[1]}|$.

Proof. By Defs. 12 and 22, $[r_{\ell, q-\lambda+1}, r_{\ell, q} + C_\ell] \cap \overline{\Gamma_\lambda} = [r_{\ell, q-\lambda+1}, r_{\ell, q} + C_\ell] \setminus \Gamma_\lambda^{[1]}$. By Claim 5, $T_{\ell, q}$ executes at each instant within $[r_{\ell, q-\lambda+1}, r_{\ell, q} + C_\ell] \setminus \Gamma_\lambda^{[1]}$. \square

Lemma 9: $A_3 \geq \gamma_\ell(\lambda) - |\Gamma_\lambda^{[3]}|$.

Proof. By Defs. 12 and 22, $[r_{\ell, q} + \Theta_\ell - \gamma_\ell(\lambda), r_{\ell, q} + \Theta_\ell] \cap \overline{\Gamma_\lambda} = [r_{\ell, q} + \Theta_\ell - \gamma_\ell(\lambda), r_{\ell, q} + \Theta_\ell] \setminus \Gamma_\lambda^{[3]}$. By Claim 5, $T_{\ell, q}$ executes at each instant within $[r_{\ell, q} + \Theta_\ell - \gamma_\ell(\lambda), r_{\ell, q} + \Theta_\ell] \setminus \Gamma_\lambda^{[3]}$. \square

If $\Gamma_\lambda^{[3]} = \emptyset$, then, because by Def. 10, $f_{\ell, q-\lambda} \leq r_{\ell, q} + \Theta_\ell - \gamma_\ell(\lambda)$, jobs $T_{\ell, q-\lambda+1}, \dots, T_{\ell, q}$ can execute uninterruptedly within $[r_{\ell, q} + \Theta_\ell - \gamma_\ell(\lambda), r_{\ell, q} + \Theta_\ell]$, and hence, $T_{\ell, q}$'s response-time bound will not be violated leading to a contradiction. We henceforth assume $|\Gamma_\lambda^{[3]}| > 0$.

Lemma 10: $A_2 \geq a \cdot (-C_\ell - \gamma_\ell(\lambda) - r_{\ell, q} + r_{\ell, q-\lambda+1} + W(T_\ell, r_{\ell, q-\lambda+1}) - 1 - \mu) + a \cdot (|\Gamma_\lambda^{[1]}| + |\Gamma_\lambda^{[3]}|)$, where $a = \min(F + 1, m)$.

Proof. By Claim 9, no job with priority higher than $T_{\ell, q}$ or its predecessors can be released after $r_{\ell, q} + C_\ell$. If at most F available processors are not idle at some time instant $t' \in [r_{\ell, q} + C_\ell, r_{\ell, q} + \Theta_\ell - \gamma_\ell(\lambda)] \setminus \Gamma_\lambda$, then at each time $t \geq t'$ all tasks with jobs in $\tau_p(t)$ can be accommodated using F fully available processors, and hence $T_{\ell, q}$ will complete by $r_{\ell, q} + \Theta_\ell$. We henceforth assume that at least $a = \min(F + 1, m)$ available processors are not idle at each time within $[r_{\ell, q} + C_\ell, r_{\ell, q} + \Theta_\ell - \gamma_\ell(\lambda)] \setminus \Gamma_\lambda$ (see Fig. 3). Thus, the total processor allocation (including unavailable

time) within $[r_{\ell, q} + C_\ell, r_{\ell, q} + \Theta_\ell - \gamma_\ell(\lambda)] \setminus \Gamma_\lambda$ is at least

$$\begin{aligned}
A_2 &\geq a \cdot |[r_{\ell, q} + C_\ell, r_{\ell, q} + \Theta_\ell - \gamma_\ell(\lambda)] \setminus \Gamma_\lambda| \\
&\quad \{\text{by Defs. 12 and 22}\} \\
&= a \cdot (\Theta_\ell - \gamma_\ell(\lambda) - C_\ell - (|\Gamma_\lambda| - |\Gamma_\lambda^{[1]}| - |\Gamma_\lambda^{[3]}|)) \\
&= a \cdot (\Theta_\ell - \gamma_\ell(\lambda) - C_\ell - |\Gamma_\lambda|) + a \cdot (|\Gamma_\lambda^{[1]}| + |\Gamma_\lambda^{[3]}|) \\
&\quad \{\text{by Lemma 3}\} \\
&= a \cdot (\Theta_\ell - \gamma_\ell(\lambda) - C_\ell - (r_{\ell, q} + \Theta_\ell - r_{\ell, q-\lambda+1} - W(T_\ell, r_{\ell, q-\lambda+1}) + 1 + \mu)) + a \cdot (|\Gamma_\lambda^{[1]}| + |\Gamma_\lambda^{[3]}|) \\
&= a \cdot (-\gamma_\ell(\lambda) - C_\ell - r_{\ell, q} + r_{\ell, q-\lambda+1} + W(T_\ell, r_{\ell, q-\lambda+1}) - 1 - \mu) + a \cdot (|\Gamma_\lambda^{[1]}| + |\Gamma_\lambda^{[3]}|). \quad \square
\end{aligned}$$

Theorem 5. *If the response-time bound Θ_ℓ of $T_{\ell, q}$ is violated, (as we have assumed), then for some $k \in [1, K_\ell]$ and $\delta_\ell(k) \leq \lfloor (H_\ell + (m-a) \cdot (E_\ell^*(k) - 1) + (a-1) \cdot (\gamma_\ell(k) + C_\ell + \max(0, \gamma_\ell(k-1) - 1)) + \widehat{U} \cdot \sigma_{tot} - \Theta_\ell \cdot \widehat{U}) / (\widehat{U} - U_{sum}) \rfloor$,*

$$\begin{aligned}
M^*(\delta_\ell(k)) + (m-a) \cdot (E_\ell^*(k) - 1) \\
+ (a-1) \cdot (\gamma_\ell(k) + C_\ell + \max(0, \gamma_\ell(k-1) - 1)) \\
\geq \mathcal{B}(\delta_\ell(k) + \Theta_\ell). \quad (25)
\end{aligned}$$

Proof. Consider job $T_{\ell, q}$, $k = \lambda$, and time instants $r_{\ell, q-\lambda+1}$ and $t_0(\lambda)$ as defined in Defs. 10 and 14.

By Def. 23, the total processor allocation (including unavailable time) within $[t_0(\lambda), r_{\ell, q} + \Theta_\ell]$ is $A_0 + A_1 + A_2 + A_3$, which, by Lemmas 7–10 is

$$\begin{aligned}
A_0 + A_1 + A_2 + A_3 &\geq m \cdot (\delta_\ell(\lambda) + \Theta_\ell) - m \cdot (W(T_\ell, r_{\ell, q-\lambda+1} - 1) + m \cdot \mu) \\
&\quad + r_{\ell, q} + C_\ell - r_{\ell, q-\lambda+1} - |\Gamma_\lambda^{[1]}| \\
&\quad + a \cdot (-\gamma_\ell(\lambda) - C_\ell - r_{\ell, q} + r_{\ell, q-\lambda+1} + W(T_\ell, r_{\ell, q-\lambda+1}) - 1 - \mu) + a \cdot (|\Gamma_\lambda^{[1]}| + |\Gamma_\lambda^{[3]}|) \\
&\quad + \gamma_\ell(\lambda) - |\Gamma_\lambda^{[3]}| \\
&= m \cdot (\delta_\ell(\lambda) + \Theta_\ell) - (m-a) \cdot (W(T_\ell, r_{\ell, q-\lambda+1} - 1) \\
&\quad + (m-a) \cdot \mu + (a-1) \cdot (|\Gamma_\lambda^{[1]}| + |\Gamma_\lambda^{[3]}|) \\
&\quad + (1-a) \cdot (\gamma_\ell(\lambda) + C_\ell + r_{\ell, q} - r_{\ell, q-\lambda+1})). \quad (26)
\end{aligned}$$

Let $\text{Res}_h([t_0(\lambda), r_{\ell, q} + \Theta_\ell])$ be the amount of time that is not available on processor h at time instants in the interval $[t_0(\lambda), r_{\ell, q} + \Theta_\ell]$. By Defs. 11 and 16, the allocation of jobs within $[t_0(\lambda), r_{\ell, q} + \Theta_\ell]$ is upper-bounded by $W(t_0(\lambda))$ (recall that we are ignoring lower-priority jobs). Thus, by

illustrated in Fig. 4(b) and explained in further detail below. For conciseness, we refer to the systems in these two insets as the (a)- and (b)-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).

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 [4, 10]. We obtained $\gamma_i(k)$ as in Def. 1 by examining a repeating pattern of 19,000 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 $\gamma_i(1) = 164\mu s$, which we set to be the maximum job execution time (the best-case execution time is $e_i^{\min} = 2\mu s$). The function $\alpha_i^u(\Delta)$ in Def. 2 was obtained by examining macroblock arrival times. We computed $\mathcal{A}_i^{-1}(k)$ in Def. 4 as well as linear bounds for $\alpha_i^u(\Delta)$ and $\gamma_i(k)$ as in (2) and (3) using the RTC Toolbox [13].

In the (b)-system, tasks T_1, \dots, T_4 , are scheduled on three fully-available processors. 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. 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 Fig. 4(b), down arrows are used to depict the long-term available utilization on each processor.

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, the arrival curve $\alpha_i^u(\Delta)$ is bursty, i.e., several macroblocks can arrive at the same time instant. Second, while $\bar{e}_i = 17.6\mu s$, the maximum execution time of a single macroblock is $164\mu s$, so assuming that each job executes for its worst-case execution time would result in heavy overprovisioning. The long-term per-task utilization is $u_i = R_i \cdot \bar{e}_i = 0.00396 \cdot 17.6 = 0.7$, where $R_i = 0.00396$ is the long-term arrival rate. Finally, the total utilization is $U = \sum_{i=1}^4 u_i = 2.8$. Therefore, the task set $\{T_1, \dots, T_4\}$ cannot be partitioned onto three processors (four processors are needed, actually), so global scheduling is required.

For the (b)-system, the minimum job inter-arrival time is zero. However, the arriving stream cannot be re-shaped so that the minimum job inter-arrival time is at least $p_i = 25\mu s$ and the long-term arrival rate to be preserved. Because the worst-case job execution time is $e_i^{\max} = \gamma_i(1) = 164\mu s$ and the minimum job inter-arrival time is $p_i = 25\mu s$, we have

$e_i^{\max}/p_i = 6.59 > 1$. Therefore, the (b)-system *cannot be analyzed using prior results for periodic and sporadic task models*, which require $p_i > 0$ and $e_i^{\max}/p_i \leq 1$.

Fig. 4(c) depicts the job completion curve $\alpha_1^{u'}$ (normalized to frames/millisecond assuming 1,584 macroblocks per frame) for task T_1 in the (a)- and (b)-systems, the curve $\alpha_2^{u'}$ for task T_2 in the (b)-system, and the input curve α_1^u . (Note that, in the (b)-system, tasks T_1, \dots, T_4 have the same input curve α_1^u , and the completion curves for T_2, \dots, T_4 are the same.) The curves for T_1 in the (a)- and (b)-systems were obtained using prior results in real-time calculus pertaining to uniprocessor systems as implemented in the RTC Toolbox [13]. 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 incrementally applied Corollary 1 to each of these three tasks to obtain their maximum response-time bounds Θ_i . We then computed completion curves using Theorem 1.

The resulting curves have the same long-term completion rate in both systems. Task T_1 has the shortest possible maximum response time in both the (a)- and (b)-systems. However, the large job response times of tasks T_2, \dots, T_4 in the (b)-system cause a larger degree of burstiness in the output event streams. Such burstiness is mainly due to the fact that multiple jobs of the same task arriving at the same time instant can potentially execute for a significant duration of time, causing jobs of non-executing tasks to wait (or be queued). Overall, the (b)-system has the advantage of needing only *three* processors to accommodate four video streams, at the expense of larger buffers for storing partially decoded macroblocks (for approximately 50 frames). With partitioned scheduling, *four* dedicated processors are required.

9. Concluding Remarks

In this paper, we have presented an extension to the real-time calculus framework. We considered 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 partitioning techniques are unnecessarily restrictive.

We also 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 amount of supply available after scheduling all incoming jobs. These bounds can be used as input for other PEs thereby resulting in a compositional technique.

In our analysis, we assumed that response-time bounds are specified. In the future, we plan to extend the analysis in order to explicitly derive upper bounds on job response

times from the parameters of the incoming streams, task execution times, and service curves under preemptive and non-preemptive scheduling.

Acknowledgment: Work supported by AT&T, IBM, Intel, and Sun Corps., NSF grants CNS 0834270, CNS 0834132, and CNS 0615197, and ARO grant W911NF-06-1-0425. We are grateful to Linh Thi Xuan Phan for her help with experimental data.

References

- [1] S. Baruah. Techniques for multiprocessor global schedulability analysis. In *Proc. of the 28th IEEE Real-Time Systems Symp.*, pages 119–128, 2007.
- [2] M. Bertogna, M. Cirinei, and G. Lipari. Schedulability analysis of global scheduling algorithms on multiprocessor platforms. *IEEE Trans. on Parallel and Distributed Systems*, 20(4):553–566, 2009.
- [3] S. Chakraborty, S. Kunzli, and L. Thiele. A general framework for analysing system properties in platform-based embedded system designs. In *Proc. of Design, Automation and Test in Europe - Volume 1*, page 10190, 2003.
- [4] S. Chakraborty, Y. Liu, N. Stoimenov, L. Thiele, and E. Wandeler. Interface-based rate analysis of embedded systems. In *Proc. of the 27th IEEE Real-Time Systems Symp.*, pages 25–34, 2006.
- [5] U. Devi. *Soft Real-Time Scheduling on Multiprocessors*. PhD thesis, University of North Carolina, Chapel Hill, NC, 2006.
- [6] H. Leontyev and J. Anderson. Tardiness bounds for FIFO scheduling on multiprocessors. In *Proc. of the 19th Euromicro Conf. on Real-Time Systems*, pages 71–80, 2007.
- [7] H. Leontyev and J. Anderson. A hierarchical multiprocessor bandwidth reservation scheme with timing guarantees. In *Proc. of the 20th Euromicro Conf. on Real-Time Systems*, pages 191–200, 2008.
- [8] H. Leontyev and J. Anderson. A unified hard/soft real-time schedulability test for global EDF multiprocessor scheduling. In *Proc. of the 29th IEEE Real-Time Systems Symp.*, pages 375–384, 2008.
- [9] A. K. Mok, X. Feng, and D. Chen. Resource partition for real-time systems. In *Proc. of 7th Real-Time Technology and Applications Symp.*, pages 75–84, 2001.
- [10] L. Phan, S. Chakraborty, and P. Thiagarajan. A multi-mode real-time calculus. In *Proc. of the 29th IEEE Real-Time Systems Symp.*, pages 59–69, December 2008.
- [11] K. Richter, M. Jersak, and R. Ernst. A formal approach to MpSoC performance verification. *IEEE Computer*, 36(4):60–67, 2003.
- [12] I. Shin, A. Easwaran, and I. Lee. Hierarchical scheduling framework for virtual clustering of multiprocessors. In *Proc. of the 20th Euromicro Conf. on Real-Time Systems*, pages 181–190, 2008.
- [13] E. Wandeler and L. Thiele. Real-Time Calculus (RTC) Toolbox, 2006.

- [14] F. Zhang and A. Burns. Schedulability Analysis for Real-Time Systems with EDF Scheduling. Technical Report YCS-2008-426, University of York, Department of Computer Science, 2008.

Appendix

In this appendix, we prove Lemmas 4, 5, and 6.

Claim A1. *If $T_{i,y} \preceq T_{\ell,q}$, then $r_{i,y} \leq r_{\ell,q} + D_\ell - D_i$, for $y \geq 0$.*

Proof. The claim immediately follows from Def. 8. \square

Lemma 4:

$$W_{\mathbf{NC}}(T_i, \delta_\ell(k)) = \gamma_i(\alpha_i^+(\delta_\ell(k) + D_\ell - D_i))$$

Proof. Because $T_i \in \mathbf{NC}$, all of its jobs released prior to $t_0(k)$ are completed by time $t_0(k)$. Thus, the competing demand due to T_i is upper-bounded by the demand due to T_i 's jobs released at or after $t_0(k)$ that have higher priority than $T_{\ell,q}$. For such a job $T_{i,j}$, by Claim A1, $r_{i,j} + D_i \leq r_{\ell,q} + D_\ell$, and hence, $r_{i,j} \leq r_{\ell,q} + D_\ell - D_i$. Therefore, the competing demand due to task T_i , $W(T_i, t_0(k))$, is upper-bounded by the total execution time of T_i 's jobs released within $[t_0(k), r_{\ell,q} + D_\ell - D_i]$. From Defs. 1 and 3, we have

$$\begin{aligned} W(T_i, t_0(k)) &\leq \gamma_i(\alpha_i^+(r_{\ell,q} + D_\ell - D_i - t_0(k))) \\ &\quad \{\text{by Def. 15}\} \\ &= \gamma_i(\alpha_i^+(\delta_\ell(k) + D_\ell - D_i)). \quad \square \end{aligned}$$

Definition A1. Let $T_{i,a}$ be the earliest job of T_i that is pending within $[t_0(k), r_{\ell,q} + \Theta_\ell)$.

Note that, if $T_{i,a}$ does not exist, then $W(T_i, t_0(k)) = 0$. We henceforth assume that $T_{i,a}$ exists.

Claim A2. *If $T_{i,a}$ is defined as in Def. A1, then $f_{i,a} > t_0(k)$ and $r_{i,a} > t_0(k) - \Theta_i$.*

Proof. If $f_{i,a} \leq t_0(k)$, then $T_{i,a}$ is not pending within $[t_0(k), r_{\ell,q} + \Theta_\ell)$, which violates Def. A1. By (9), $f_{i,a} > t_0(k)$ implies $r_{i,a} + \Theta_i > t_0(k)$. \square

Definition A2. Let $\kappa_i = \{T_{i,y} : y \geq a \wedge T_{i,y} \preceq T_{\ell,q} \wedge T_{i,y} \text{ is pending within } [t_0(k), r_{\ell,q} + \Theta_\ell]\}$.

Claim A3. *If $T_{i,y} \in \kappa_i$, then $r_{i,y} \in [r_{i,a}, r_{\ell,q} + D_\ell - D_i]$.*

Proof. By Def. A2, $T_{i,y} \preceq T_{\ell,q}$ holds if $T_{i,y}$ is in κ_i . The claim follows from Claim A1. \square

Claim A4:

$$W(T_i, t_0(k)) = \sum_{T_{i,y} \in \kappa_i} W(T_{i,y}, t_0(k)). \quad (28)$$

Proof. The claim follows immediately from Definitions 11, A1, A2. \square

Claim A5. Let $G_i(S, X) = \min(\gamma_i(S), \max(0, X - \mathcal{A}_i^{-1}(S-1)) + \gamma_i(S-1))$ be as defined in Lemma 5 below. The function $G_i(S, X)$ is a non-decreasing function of the integral argument S .

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

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

Lemma 5. Let $G_i(S, X) = \min(\gamma_i(S), \max(0, X - \mathcal{A}_i^{-1}(S-1)) + \gamma_i(S-1))$.

$$\begin{aligned} & W_{\text{HC}}(T_i, \delta_\ell(k)) \\ &= G_i(\alpha_i^u(\delta_\ell(k) + D_\ell - D_i + \Theta_i), \delta_\ell(k) + D_\ell - D_i + \Theta_i) \end{aligned}$$

Proof. We consider two cases.

Let $T_{i,a}$ be as defined in Def. A1. We first rewrite (28).

$$W(T_i, t_0(k)) = W(T_{i,a}, t_0(k)) + \sum_{T_{i,y} \in \kappa_i \setminus T_{i,a}} W(T_{i,y}, t_0(k)) \quad (29)$$

We now bound the individual terms in (29). By Claim A2, $T_{i,a}$ finishes its execution at time $f_{i,a} > t_0(k)$, and hence,

$$\begin{aligned} & W(T_{i,a}, t_0(k)) \\ &= \min(e_{i,a}, f_{i,a} - t_0(k)) \\ & \quad \{\text{by (9)}\} \\ & \leq \min(e_{i,a}, r_{i,a} + \Theta_i - t_0(k)). \end{aligned} \quad (30)$$

By (29) and (30),

$$\begin{aligned} & W(T_i, t_0(k)) \\ & \leq \min(e_{i,a}, r_{i,a} + \Theta_i - t_0(k)) + \sum_{T_{i,y} \in \kappa_i \setminus a} W(T_{i,y}, t_0(k)) \\ & \leq \min \left(e_{i,a} + \sum_{T_{i,y} \in \kappa_i \setminus T_{i,a}} W(T_{i,y}, t_0(k)), \right. \\ & \quad \left. r_{i,a} + \Theta_i - t_0(k) + \sum_{T_{i,y} \in \kappa_i \setminus T_{i,a}} W(T_{i,y}, t_0(k)) \right). \end{aligned} \quad (31)$$

Let $S_i = |\kappa_i|$. Because, the processor allocation of job $T_{i,y}$ cannot be greater than its execution time, by Def. 1, we have the following.

$$e_{i,a} + \sum_{T_{i,y} \in \kappa_i \setminus T_{i,a}} W(T_{i,y}, t_0(k)) \leq E_i(S_i) \quad (32)$$

$$\sum_{T_{i,y} \in \kappa_i \setminus T_{i,a}} W(T_{i,y}, t_0(k)) \leq E_i(S_i - 1) \quad (33)$$

By (31), (32), and (33), we have

$$W(T_i, t_0(k)) \leq \min(\gamma_i(S_i), r_{i,a} + \Theta_i - t_0(k) + \gamma_i(S_i - 1)). \quad (34)$$

By Claim A3, all jobs $T_{i,y}$ such that $T_{i,y} \in \kappa_i$ are released within $[r_{i,a}, r_{\ell,q} + D_\ell - D_i]$. Let $T_{i,a+S_i-1}$ be the latest job released within this interval. We upper bound $r_{i,a}$ as follows.

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

From the inequality above, we have

$$\begin{aligned} & r_{i,a} + \Theta_i - t_0(k) \\ & \leq \max(0, r_{\ell,q} + D_\ell - D_i - \mathcal{A}_i^{-1}(S_i - 1) + \Theta_i - t_0(k)) \\ & \quad \{\text{by Def. 15}\} \\ & = \max(0, \delta_\ell(k) + D_\ell - D_i + \Theta_i - \mathcal{A}_i^{-1}(S_i - 1)). \end{aligned} \quad (35)$$

By (34) and (35), we have

$$\begin{aligned} & W(T_i, t_0(k)) \\ & \leq \min(\gamma_i(S_i), \max(0, \delta_\ell(k) + D_\ell - D_i + \Theta_i \\ & \quad - \mathcal{A}_i^{-1}(S_i - 1)) + \gamma_i(S_i - 1)) \\ & = G_i(S_i, \delta_\ell(k) + D_\ell - D_i + \Theta_i), \end{aligned} \quad (36)$$

where $G_i(S, X)$ is defined in the statement of the lemma. By Claim A5, the function $G_i(S, X)$ is a non-decreasing function of S . We thus can find an upper bound on $W(T_i, t_0(k))$ by setting an upper bound on S_i into (36).

By Claim A3, $S_i = |\kappa_i|$ is at most the number of jobs released within the interval $[r_{i,a}, r_{\ell,q} + D_\ell - D_i]$, which, by Claim A2, is contained within $(t_0(k) - \Theta_i, r_{\ell,q} + D_\ell - D_i]$. We thus upper-bound S_i using Def. 2.

$$\begin{aligned} S_i &\leq \alpha_i^u(r_{\ell,q} + D_\ell - D_i - t_0(k) + \Theta_i) \\ &\quad \{\text{by Def. 15}\} \\ &= \alpha_i^u(\delta_\ell(k) + D_\ell - D_i + \Theta_i) \end{aligned}$$

Setting this upper bound on S_i into (36), we get the required result. \square

The following claims and lemma are used to prove Lemma 6.

Claim A6: $L_i(X + Y) \leq L_i(X) + u_i \cdot (Y + |a|)$ for all X and $Y \geq a$.

Proof. By Def. 5, $u_i > 0$. We consider two cases.

Case 1: $Y \geq 0$. In this case, by Def. 20,

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

Because $|a| \geq 0$, the required result follows.

Case 2: $Y < 0$. In this case, by Def. 20,

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

From the statement of the claim and the condition of Case 2, we have $a \leq Y < 0$, and hence, $Y + |a| \geq 0$. We thus have $L_i(X + Y) \leq L_i(X) \leq L_i(X) + u_i \cdot (Y + |a|)$. \square

Claim A7: $\gamma_i(\alpha_i^u(X)) \leq \gamma_i(\alpha_i^+(X)) \leq L_i(X)$.

Proof. By Def. 2, $\alpha_i^u(\Delta)$ is a non-decreasing function of Δ . Therefore, $\alpha_i^u(\Delta) \leq \alpha_i^u(\Delta + \epsilon)$ for any $\epsilon > 0$, which implies $\alpha_i^u(\Delta) \leq \lim_{\epsilon \rightarrow +0} \alpha_i^u(\Delta + \epsilon)$. The right-hand side of the latter inequality is $\alpha_i^+(\Delta)$ by Def. 3. Thus, $\alpha_i^u(\Delta) \leq \alpha_i^+(\Delta)$. The first inequality of the claim therefore follows from $\gamma_i(k)$ being a non-decreasing function of k by Def. 1.

We now prove the second inequality. Because $\alpha_i^+(X) \geq 0$ by Def. 3, we have

$$\begin{aligned} &\gamma_i(\alpha_i^+(X)) \\ &= \gamma_i(\max(0, \alpha_i^+(X))) \\ &\quad \{\text{by (3)}\} \\ &\leq \bar{e}_i \cdot (\max(0, \alpha_i^+(X))) + v_i \\ &\quad \{\text{by (2)}\} \\ &\leq \bar{e}_i \cdot (\max(0, R_i \cdot X + B_i)) + v_i \\ &= \max(0, \bar{e}_i \cdot R_i \cdot X + \bar{e}_i \cdot B_i) + v_i \\ &\quad \{\text{by Def. 5}\} \\ &= \max(0, u_i \cdot X + \bar{e}_i \cdot B_i) + v_i \\ &\quad \{\text{by Def. 20}\} \\ &= L_i(X). \end{aligned} \quad \square$$

Lemma A1: $W_{\mathbf{HC}}(T_i, \delta_\ell(k)) \leq L_i(\delta_\ell(k) + D_\ell - D_i) + u_i \cdot \Theta_i$, $W_{\mathbf{NC}}(T_i, \delta_\ell(k)) \leq L_i(\delta_\ell(k) + D_\ell - D_i)$.

Proof. We prove the first inequality. The second inequality is proved similarly.

$$\begin{aligned} &W_{\mathbf{HC}}(T_i, \delta_\ell(k)) \\ &\quad \left\{ \begin{array}{l} \text{by Lemma 5 (note that this inequality holds} \\ \text{for both } i = \ell \text{ and } i \neq \ell \text{ since } k \geq 1) \end{array} \right\} \\ &\leq \gamma_i(\alpha_i^u(\delta_\ell(k) + D_\ell - D_i + \Theta_i)) \\ &\quad \{\text{by Claim A7}\} \\ &\leq L_i(\delta_\ell(k) + D_\ell - D_i + \Theta_i) \\ &\quad \{\text{because } \Theta_i \geq 0, \text{ by Claim A6}\} \\ &\leq L_i(\delta_\ell(k) + D_\ell - D_i + \Theta_i) \\ &\leq L_i(\delta_\ell(k) + D_\ell - D_i) + u_i \cdot \Theta_i. \end{aligned} \quad \square$$

Lemma 6. For all $\delta_\ell(\lambda) \geq \mathcal{A}_\ell^{-1}(\lambda - 1)$, $M^*(\delta_\ell(\lambda)) \leq U_{\text{sum}} \cdot \delta_\ell(\lambda) + H_\ell$, where $H_\ell = \sum_{T_i \in \tau} L_i(D_\ell - D_i) + U(m - 1) \cdot \max(\Theta_i)$ and $U(y)$ is the sum of $\min(y, |\tau|)$ largest utilizations.

Proof. Suppose that the sets \mathbf{HC} and \mathbf{NC} subject to (24) maximize the value of the right-hand side of (23). By (23), we have

$$\begin{aligned} &M^*(\delta_\ell(\lambda)) \\ &= \sum_{T_i \in \mathbf{HC}} W_{\mathbf{HC}}(T_i, \delta_\ell(\lambda)) + \sum_{T_i \in \mathbf{NC}} W_{\mathbf{NC}}(T_i, \delta_\ell(\lambda)) \\ &\quad \{\text{by Lemma A1}\} \\ &\leq \sum_{T_i \in \mathbf{HC}} (L_i(\delta_\ell(\lambda) + D_\ell - D_i) + u_i \cdot \Theta_i) \\ &\quad + \sum_{T_i \in \mathbf{NC}} L_i(\delta_\ell(\lambda) + D_\ell - D_i) \end{aligned}$$

$$\begin{aligned}
& \{\text{since } \mathbf{HC} \cup \mathbf{NC} \subseteq \tau\} \\
& \leq \sum_{T_i \in \tau} L_i(\delta_\ell(\lambda) + D_\ell - D_i) + \sum_{T_i \in \mathbf{HC}} u_i \cdot \Theta_i \\
& \quad \left\{ \begin{array}{l} \text{because } |\mathbf{HC}| \leq m - 1 \text{ by (24), and by the definition} \\ \text{of } U(y) \text{ in the statement of the lemma} \end{array} \right\} \\
& \leq \sum_{T_i \in \tau} [L_i(\delta_\ell(\lambda) + D_\ell - D_i)] + U(m - 1) \cdot \max(\Theta_i) \\
& \quad \left\{ \begin{array}{l} \text{by Claim A6} \\ \text{(note that, by Defs. 14 and 15, } \delta_\ell(\lambda) \geq 0) \end{array} \right\} \\
& \leq \sum_{T_i \in \tau} [L_i(D_\ell - D_i) + u_i \cdot \delta_\ell(\lambda)] \\
& \quad + U(m - 1) \cdot \max(\Theta_i) \\
& \quad \left\{ \begin{array}{l} \text{by Def. 5 and the definition of } H_\ell \\ \text{in the statement of the lemma} \end{array} \right\} \\
& = U_{sum} \cdot \delta_\ell(\lambda) + H_\ell. \quad \square
\end{aligned}$$