17th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications

Real-Time Analysis of Servers for General Job Arrivals

Pratyush Kumar, Jian-Jia Chen♯, Lothar Thiele, Andreas Schranzhofer, and Giorgio C. Buttazzo†
Computer Engineering and Networks Laboratory, ETH Zurich, Switzerland
♯Department of Informatics, Karlsruhe Institute of Technology (KIT), Germany
†Real-Time Systems Laboratory, Scuola Superiore Sant'Anna of Pisa, Italy

***Abstract*—Several servers have been proposed to schedule streams of aperiodic jobs in the presence of other periodic tasks. Standard schedulability analysis has been extended to consider such servers. However, not much attention has been laid on computing the worst-case delay suffered by a given stream of jobs when scheduled via a server. Such analysis is essential for using servers to schedule hard real-time tasks. We illustrate, with examples, that well established resource models, such as supply bound function and models from Real-Time Calculus, do not tightly characterize servers. In this work, we analyze the server algorithm of the Constant Bandwidth Server and compute a provably tight resource model of the server. The approach used enables us to differentiate between the soft and hard variants of the server. A similar approach can be used to characterize other servers; the final results for which are presented.**

I. Introduction

Scheduling of mixed or hybrid task-sets, i.e., task-sets with a combination of periodic and aperiodic tasks, has been a topic of much relevance in the real-time community. Well known scheduling policies such as Rate Monotonic (RM) and Earliest Deadline First (EDF) are usually employed to schedule the periodic tasks. In the presence of periodic tasks, the question of interest is: How to schedule aperiodic tasks, as soon as possible, and without affecting the performance guaranteed to the periodic tasks. To this end, the notion of *servers* was introduced; servers reserve resources which can be used to serve aperiodic tasks. One of the first such reservation schemes was proposed in [1]. Several other servers have been presented since, which are discussed in detail in [2]. Standard schedulability tests can be extended to consider the presence of such servers.

Aperiodic tasks are usually event-driven and may have hard, soft or non-real-time performance requirements, depending on the specific application. In this work, we consider aperiodic tasks with hard real-time performance constraints. When such aperiodic tasks are scheduled via servers, it is essential to validate if the performance constraints of the tasks are met. To this end, a notion of the *aperiodic guarantee* [2] has been introduced. An aperiodic guarantee specifies a condition to verify whether an aperiodic job (an instance of an aperiodic task) of known execution time, arriving at a certain time instant, meets its known deadline, when scheduled via a server. Such aperiodic guarantees are provided for most of the existing servers. The crucial limitation with this approach is that it considers only a single job, instead of a stream of jobs that constitute the aperiodic task. This nullifies the effect of self-interference, i.e., the effect that the execution of one job of a task can delay the execution of subsequent job(s) of the same task.

Another approach to quantify the service provided by servers is to use statistical analysis to compute QoS guarantees expressed in terms of probability for each served job to meet a deadline. For instance, [3] provides such analysis for the Constant Bandwidth Server, proposed in [4]. Such analysis is not suitable for hard real-time tasks, where worst-case deterministic guarantees must be provided for each job.

A third approach is to study the scheduling of specific streams, such as periodic streams, when scheduled via the server. For instance, a Constant Bandwidth Server, characterized by a period P_s and maximum budget Q_s, can serve a periodic task with period $P \geq P_s$, execution time $C \leq Q_s$ and relative deadline equal to the period [2]. Restricting the analysis to only consider streams of periodic jobs belittles the need for servers. This limitation is especially significant because aperiodic tasks by definition can have irregular patterns of job arrivals, which need to be considered in validating if real-time performance requirements are met. For instance, a burst of jobs may appear, potentially delaying the execution of subsequent jobs. Characterizing the possible variability in the patterns of such job arrivals and then analyzing such a characterized task for the performance guarantee provided by the server is necessary.

From the above discussion it follows that analysis of a general stream of jobs when served via a server such as the CBS has not been studied. However, such analysis forms the core of real-time analysis and we can look at existing methods to perform the same. The first step in such methods is to characterize the supply provided by the server. Two well known approaches to characterize the supply provided by a resource are (a) the supply bound function proposed in [5], and (b) resource models used in the framework of Real-Time Calculus and Modular Performance Analysis. As we shall see with the help of examples in the next section, neither of these methods are suitable in tightly characterizing the working of servers such as the CBS. This is the motivation for this work, in which we formally characterize the resource model of the Constant Bandwidth Server (CBS) by analyzing its server algorithm. We then discuss how to use it to tightly compute the worst-case delay suffered by any given stream of aperiodic jobs.

The rest of the paper is organized as follows. We present examples that motivate the presented analysis in Section II. We define the system model and the problem in Section III. We then describe and analyze in detail the Constant Bandwidth Server in Section IV. We present the analysis of the hard variant of CBS in Section V and discuss some of the fine differences between the two variants. We present, without proof, similar results for other servers in Section VI. Finally, we conclude in Section VII.

II. Motivational Examples

In this section, we discuss specific examples where resource models such as the supply bound function [5] are not able to tightly model the Constant Bandwidth Server.

The supply bound function characterizes the minimum service or supply provided by a resource. More specifically, $\mathtt{sbf}(\Delta)$ denotes the minimum amount of service provided by the resource in any interval of length Δ. Such a function can be used to verify if a stream of jobs characterized by an *input arrival trace* can be served by the resource within a given delay bound. We formally define these terms in Section III. In our case, we would like to characterize the supply bound function of a server and use it to compute the worst-case response time of a given stream of aperiodic jobs. For instance, we can determine the supply bound function of a TDMA server for given slot length and cycle period. The supply bound function is such that we can tightly compute the worst-case response time of any stream of aperiodic jobs scheduled via the server.

Now let us consider a different server, namely the Constant Bandwidth Server (CBS) characterized by a period P_s and maximum budget Q_s. We describe the CBS algorithm in Section IV. In contrast to the TDMA server, the CBS is event-driven: the behavior of CBS adapts to the arriving aperiodic jobs. Clearly, if no jobs arrive to be served by the CBS, then the service provided by the server is 0, in any interval of time. In other words, the server, by itself, is not characterized by a supply bound function. Let us redefine supply bound function such that $\mathtt{sbf}(\Delta)$ denotes the minimum amount of service provided by the resource in any time interval of length Δ, within a *backlogged period*. A backlogged period is one where the task queue of pending jobs is non-empty. As we will see, even on using this modified definition of the supply bound function, we cannot derive tight results for the Constant Bandwidth Server.

Consider a specific task-set: a periodic task T_1 with period $P_1 = 20$, execution time $C_1 = 5$, and relative deadline $D_1 = 10$, is scheduled along with a CBS with parameters period $P_s = 2$ and maximum budget $Q_s = 1$. The task-set is schedulable as given by the utilization test, $U_1 + U_s = 1$,

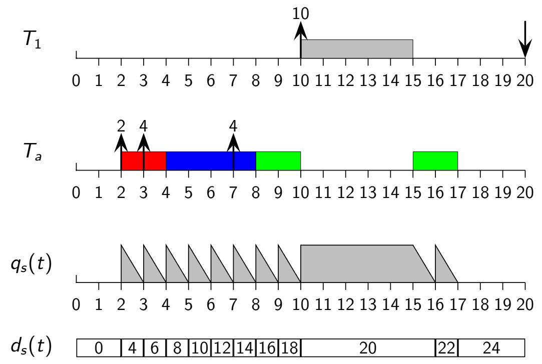

Figure 1. Computation of $\mathtt{sbf}$ for Constant Bandwidth Server. $q_s(t)$ and $d_s(t)$ denote the server budget and deadline, respectively. In the interval $[10, 15]$ the aperiodic task T_a does not receive any service, though there is pending work. Thus, $\mathtt{sbf}(5) = 0$ for the considered CBS. This example can be generalized to show that $\mathtt{sbf}(\Delta) = 0, \forall \Delta \geq 0$.

where $U_1 = C_1/D_1 = 0.5$ and $U_s = P_s/Q_s = 0.5$. Let an aperiodic task T_a be served by the CBS T_s. Consider a specific arrival pattern of the jobs of the aperiodic task as shown in Fig. 1. As can be seen, no periodic job is present in the interval [0, 10], and thus, the resource is fully utilized to serve the pending aperiodic jobs via the server. However, in the process, the deadline of the server is postponed to a large value. Then, a periodic job arrives at time $t = 10$, and gets served in the interval $[10, 15]$. In this interval, the server provides *no* service to the aperiodic tasks, though there are aperiodic jobs waiting to be served. Thus, $\mathtt{sbf}(5) = 0$ for the considered CBS. Indeed, this example can be generalized, by changing the task-set, to show that for arbitrarily long intervals of time, pending aperiodic jobs may receive no service at all via the server. And thus for a CBS $\mathtt{sbf}(\Delta) = 0$ for all $\Delta \geq 0$: a characterization that provides no performance guarantee.

A natural point that may be raised in response to the above discussion is that the considered server is *soft*, in the sense that by serving jobs in a hurry it postpones the server deadline thereby leading to longer stalls. This has been observed earlier in [6]. The Hard-CBS algorithm [4] was proposed to nullify this effect. We formally describe the algorithm of the server in Section V. Let us attempt to compute the supply bound function of a Hard-CBS.

Consider a specific task-set: a periodic task T_1 with period $P_1 = 10$, execution time $C_1 = 8$ and relative deadline $D_1 = P_1$, scheduled along with a Hard-CBS with period $P_s = 5$ and maximum budget $Q_s = 1$. As before, this task-set is schedulable as $U_1 + U_s = 1$, where $U_1 = C_1/D_1 = 0.8$ and $U_s = Q_s/P_s = 0.2$. Now consider a specific trace of the system shown in Fig. 2. In the interval $[1, 9]$, pending aperiodic jobs receive no service. Thus, $\mathtt{sbf}(8) = 0$, for the considered Hard-CBS. Now consider a different trace of aperiodic jobs served via the scheduler, which is periodic with period 5 and execution demand 1. It is required that this

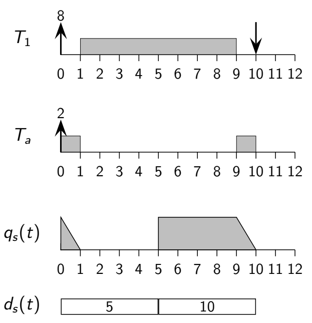

Figure 2. Computation of sbf for Hard Constant Bandwidth Server. In the interval $[1, 9]$ the aperiodic task T_a does not receive any service, though there is pending work. Thus, $\texttt{sbf}(8) = 0$ for the considered Hard-CBS.

set of tasks be executed within a relative deadline of 5. As $\texttt{sbf}(8) = 0$, no job with a relative deadline less than 8, can be guaranteed to meet its deadline. We may conclude that the considered periodic jobs will fail to meet their deadline. As discussed earlier, authors of [2] show that the Hard-CBS can indeed guarantee that these jobs meet their deadlines, as the period and the relative deadline of the jobs equal the server period P_s and the execution demand is equal to the server's maximum budget Q_s. This discrepancy, again, highlights that the considered sbf resource model is not a tight representation of the server.

Instead of using supply bound function, using (strict) service curves as used in Real-Time Calculus (RTC) [7] and Modular Performance Analysis (MPA) [8], also leads to the same conclusion in either of the two cases discussed above. Thus, with models of resource used in standard real-time analysis we have not been able to tightly characterize the resource model of the Constant Bandwidth Server. This is the motivation for the current work, in which we attempt to formally characterize the resource model of the Constant Bandwidth Server (CBS) and to use it to tightly compute the worst-case delay suffered by any given stream of aperiodic jobs.

III. System Models and Problem Definiton

We first formally describe an aperiodic task. An aperiodic task T_a is a stream of jobs, $\mathbf{J} = \{J_i\}$. Each job J_i is characterized by an arrival time a_i and an execution demand c_i. The input arrival trace, denoted as R, of task T_a is defined such that $R(t)$ is the cumulative execution demanded by all jobs that have arrived in the time interval $[0, t)$, i.e.,

$$R(t) := \sum_{\{J_i \in \mathbf{J} \ | \ a_i < t\}} c_i. \tag{1}$$

In this paper, we assume that $R(t)$ is a left-continuous function. By definition, $R(0) = 0$ and $R(t)$ non-decreasing function of time.

The aperiodic task T_a is scheduled on a resource via a server S, parameterized depending on its kind. Other tasks and/or servers may be co-scheduled on the resource. Throughout this paper, we only consider systems which are known to be schedulable. Well known schedulability tests exist for all the servers considered in this work.

We define the output trace, denoted as R', such that $R'(t)$ denotes the cumulative execution provided to the aperiodic task in the time $[0, t]$, i.e., in $[0, t]$ the aperiodic task executes for $R'(t)$ amount of time. In this paper we assume that $R'(t)$ is a right-continuous function. By definition, $R'(0) = 0$ and $R'(t)$ is a non-decreasing function of time.

Let the finish time of job J_i be denoted as f_i. The problem we consider in this paper is to compute an upper-bound on the delay that can be suffered by any of the jobs of the task T_a when served by the server S, i.e., we wish to compute δ such that

$$f_i \le a_i + \delta, \quad \forall J_i \in \mathbf{J}. \tag{2}$$

If the value of δ so computed is no more than the relative deadline of the aperiodic task, then we can claim that the real-time performance requirement of the aperiodic task is met, when served by the server S. The value δ will be referred to as the *delay bound* of the task.

Note that the above constraint must hold for all possible valid behaviors of the other tasks and/or servers of the system. In other words, the server must provide the above guarantee to the aperiodic task independent of the rest of the system, subject to the assumed schedulability condition.

In this work, we assume that the jobs of the aperiodic task are scheduled to be served by the server in First-Come-First-Server (FCFS) policy. This is a reasonable assumption in practice because the jobs belonging to the same task are generally of the same type and priority. For instance, in a video-decoding application, decoding of each individual frame may be considered a separate job. These jobs can be queued according to the FCFS policy while they wait to be served by the server. Technically, this assumption allows us to relate the worst-case response time, δ, to the input and output arrival traces R and R', respectively, as follows

$$\delta \le \texttt{Del}(R, R'), \tag{3}$$

where,

$$\texttt{Del}(R, R') \overset{def}{=} \sup_{t \ge 0} \{\inf\{\Delta \ge 0 : \ R(t - \Delta) \le (R')(t)\}\}. \tag{4}$$

The interpretation of the above formula is that when $R \ge R'$, $\texttt{Del}(R, R')$ gives the maximum horizontal distance between the two curves, R and R'.

IV. Analysis of Constant Bandwidth Server

In this section, we will present the analysis of the Constant Bandwidth Server (CBS), in some detail. This analysis will set the template for the analysis of many other servers which we consider in the subsequent sections.

A Constant Bandwidth Server (CBS) [4] is characterized by a period P_s and a maximum capacity Q_s. It is a dynamic priority server and thus assumes that Earliest Deadline First (EDF) is the underlying scheduling discipline. The quantity Q_s/P_s is usually referred to as the utilization of the server and is denoted as U_s. At each instance of time, the server is characterized by a budget q_s and a deadline d_s.

A. Server Algorithm

The following rules describe the working of a CBS. In all notation below, τ denotes the current time.

1) Initially $q_s(0) := 0$ and $d_s(0) := 0$.
2) Arriving jobs are queued in the FCFS manner in a task queue of pending jobs.
3) When a new job J_i arrives, if the task queue is empty and if $q_s(\tau) \geq (d_s(\tau) - \tau)U_s$, then the server budget is recharged to the maximum value Q_s and the server deadline is changed as $d_s(\tau) := \tau + P_s$.
4) The job (if any) at the head of the task queue is allowed to contend for the resource with a deadline equal to the server deadline $d_s(\tau)$.
5) Whenever a served job executes, the budget $q_s(\tau)$ is decreased by the amount of received execution time.
6) When $q_s(\tau) = 0$, the server budget is recharged to the maximum value Q_s and the server deadline is changed as $d_s(\tau) := d_s(\tau) + P_s$.

B. Definition of Service Curve

In this section, we formally define service curves introduced in Network Calculus [9]. We will use service curves to characterize the servers and to thereby compute the worst-case delay of a given stream of aperiodic jobs.

Definition 1 (Service Curve). *Let the input arrival trace of a stream of jobs be some arbitrary $R(t)$. Let $R'(t)$ be the output arrival trace of the stream of jobs when served by a resource. The resource is said to provide a service curve β if and only if, β is a non-decreasing function of time, $\beta(0) = 0$, and*

$$R' \geq R \otimes \beta, \tag{5}$$

where,

$$(f \otimes g)(t) \overset{def}{=} \inf_{0 \leq u \leq t} \{f(u) + g(t-u)\}. \tag{6}$$

Let us understand the above relations better. For β to be the service curve, we should have

$$\begin{aligned} R'(t) &\geq (R \otimes \beta)(t), && \forall\ t \geq 0, \\ &= \inf_{0 \leq u \leq t} (R(u) + \beta(t-u)), && \forall\ t \geq 0. \end{aligned}$$

In other words, a service curve β is provided if and only if for any given $t \geq 0$, there exists $s \in [0, t]$ such that

$$R'(t) - R(s) \geq \beta(t-s). \tag{7}$$

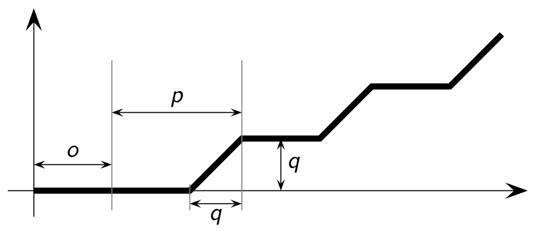

Figure 3. Graphical representation of the function F defined in (9).

The quantity $(R'(t) - R(s))$ is the execution demand of all tasks that both arrive and complete execution in the time interval $[s, t]$. The above condition requires that this quantity be greater than $\beta(t-s)$, for some s for any given t.

Note that the service curve is different in definition to the supply bound function discussed in Section I. While supply bound function provides a guarantee on every interval of time, the service curve only guarantees properties on certain intervals of time, which are free to be chosen. The implications of this difference will be discussed later in this section.

If indeed a resource provides a service curve β, then it can be used to compute the delay bound under the assumption of FCFS queueing thus:

$$\delta \leq \mathtt{Del}(R, R') \leq \mathtt{Del}(R, R \otimes \beta). \tag{8}$$

The above condition follows from the fact that whenever $R_1 \leq R_2 \leq R$, we have $\mathtt{Del}(R, R_2) \leq \mathtt{Del}(R, R_1)$.

C. Service Curve of CBS

If we succeed to show that for some non-negative non-decreasing β function, (5) holds for any input arrival trace R and the corresponding output arrival trace R', when served by a Constant Bandwidth Server (CBS), then we would be able to prove that β is the service curve provided by the CBS.

We first define a function which we shall use to define the service curves of CBS and other servers.

$$\begin{aligned} &F(p, q, o, \Delta) \\ &:= \left\{ (\Delta - o) - \left\lfloor \frac{\Delta - o}{p} \right\rfloor p - (p - q) \right\}^+ + \left\lfloor \frac{\Delta - o}{p} \right\rfloor q, \end{aligned} \tag{9}$$

where

$$x^+ := \max(x, 0). \tag{10}$$

We graphically illustrate this function in Fig. 3. In words, F is a non-decreasing non-negative function with a maximum slope of 1, that periodically with period p (subject to an offset o) increases by a value q.

We now prove that CBS indeed provides a service curve.

Theorem 2. *CBS with a period P_s and maximum budget Q_s provides a service curve β given as*

$$\beta_{\text{CBS}}(\Delta) = F(P_s, Q_s, 0, \Delta). \tag{11}$$

Proof: As discussed earlier, to show that a resource provides a service curve β, we have to compute a number s for any given arbitrary t, such that (7) holds. We now discuss how to compute such an s for any given t for the CBS algorithm for the β_{CBS} described in (11). Two cases arise depending on whether the task queue is empty or not, at the given time t.

Case (a): Task queue is empty at time t.
Since the task queue is empty, we know that all the jobs that have arrived in $[0, t)$ have completed execution by time t, i.e., $R(t) = R'(t)$. Then, by setting $s := t$, we have

$$\begin{aligned} R'(t) - R(s) &= R'(t) - R(t) = 0 \\ &= \beta_{\text{CBS}}(0) = \beta_{\text{CBS}}(t - s). \end{aligned} \tag{12}$$

Thus, we satisfy (7).

Case (b): Task queue is non-empty at time t.
Let s be the latest time before t when step 3) of the server algorithm is executed and the server budget is reset. Such a time always exists, because the arrival time of the first job satisfies the condition and at time t there are pending jobs in the task queue. In the interval $[s, t]$, the only changes to the server parameters occur in Step 6) of the server algorithm, wherein, whenever the deadline is increased by P_s, the server budget is increased by Q_s. Let ρ denote the cumulative added server budget in the interval $(s, t]$. Then, we have

$$d_s(t) = s + \left\lfloor \frac{t-s}{P_s} \right\rfloor P_s + P_s, \text{and} \tag{13}$$

$$\rho = \left(\frac{d_s(t) - d_s(s)}{P_s} \right) Q_s. \tag{14}$$

We know that the system is schedulable, i.e., the task will be executed for time equal to the server budget, if requested, on or before the server deadline. Thus, at the given time t, when the task queue is non-empty

$$q_s(t) \leq \min(Q_s, (d_s(t) - t)). \tag{15}$$

The total server budget consumed in $[s, t]$ which equals the time for which the aperiodic task is executed in the interval, i.e., $R'(t) - R'(s)$ is given as

$$R'(t) - R'(s) = q_s(s) + \rho - q_s(t)$$

By substituting (13), (14) and (15) and by using $d_s(s) = s + P_s$ and $q_s(s) = Q_s$, in the above equation, we have

$$\begin{aligned} &R'(t) - R'(s) \\ &\geq Q_s + \left(\frac{d_s(t) - d_s(s)}{P_s} \right) Q_s - \min(Q_s, (d_s(t) - t)) \\ &= \left\lfloor \frac{t-s}{P_s} \right\rfloor Q_s + (Q_s - q_s(t))^+ \\ &= \left\lfloor \frac{t-s}{P_s} \right\rfloor Q_s + \left((t-s) + (Q_s - P_s) - \left\lfloor \frac{t-s}{P_s} \right\rfloor P_s \right)^+ \\ &= \beta_{\text{CBS}}(t - s). \end{aligned} \tag{16}$$

At time s, a new job arrived while the task queue was empty, and thus, $R'(s) = R(s)$. Substituting this in (16), we have $R'(t) - R(s) \geq \beta_{\text{CBS}}(t - s)$. Hence, we have computed an s such that (7) holds. ■

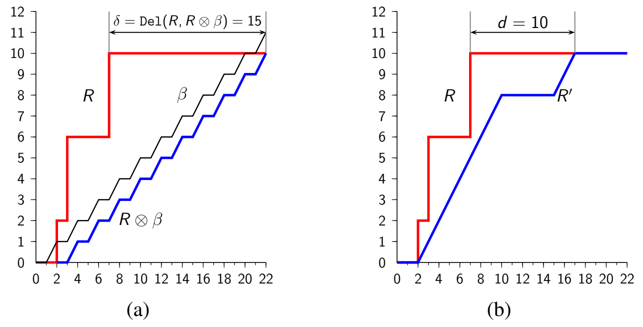

Figure 4. Example validating the service curve provided by CBS. (a) The input arrival curve R is shown along with the bound on the output arrival curve $R \otimes \beta$, where β is as given in (11). Also shown is the delay bound δ. (b) The actual observed output trace R' is shown with the maximum observed delay d.

Let us consider the example in Fig. 1. With the use of the above result, we should be able to compute a lower bound on the output arrival trace and the delay bound. We can verify if these bounds hold for the observed output trace shown in Fig. 1. The CBS we considered in Section I had a period $P_s = 2$ and maximum budget $Q_s = 1$. From the above results, such a CBS provides a service curve of $\beta(\Delta) = F(2, 1, 0, \Delta)$. With this service curve and the input arrival trace considered in the example, we illustrate in Fig. 4 the computed $R \otimes \beta$ and the delay bound δ. Indeed, the observed R', is such that $R' > R \otimes \beta$, and the worst-case delay $d = 10 < 15 = \delta$.

D. Is the Bound Tight?

We have so far shown that the CBS characterized by a period P_s and maximum budget Q_s has a service curve given in (11). This service curve can be used to compute the delay bound, δ, by using (8). However, as the name suggests, δ is only an upper-bound on the delay that can be suffered by a job. In the example shown in Fig. 4, the observed worst-case delay was strictly smaller than the delay bound. This motivates the question, whether this bound is tight, i.e., can some job actually be delayed by the delay bound. In other words, does CBS provide a service curve larger than β_{CBS}. We show with the following result that this is not the case.

Theorem 3. *The service curve of a CBS given in (11) is tight.*

Proof: Recall that the service curve provided by a resource must be valid independent of the other tasks and/or servers in the system, subject to the assumption of schedulability. Thus, to show that the provided service curve is tight, we are free to choose the configuration of the rest of the system as long as the system is schedulable.

Consider a system where a CBS with period P_s and maximum budget Q_s is co-scheduled along with a periodic task T of period P_s and execution time $P_s - Q_s$. The system is schedulable for this configuration. Let an aperiodic job arrive simultaneously with the release of a periodic job. Without loss of generality, we can choose to execute the periodic task whenever the deadline of the periodic job and the server deadline are the same. Then, the output arrival trace of the aperiodic task, until the end of the backlogged period, would exactly follow $R' = R \otimes \beta_{\text{CBS}}$. The longer the backlogged period (which we can increase by increasing the free variable R) the longer the above tightness continues. Hence the computed bound on the worst-case delay is indeed tight. It is noteworthy that this tightness is, in part, due to the tightness of the EDF schedulability constraint. ■

V. Analysis of Hard-CBS

The Hard-CBS, proposed in [4], is a variant of the original CBS algorithm. Like the CBS, a Hard-CBS is characterized by a period P_s, a maximum budget Q_s. At each time instance t, a Hard-CBS is characterized by a server deadline $d_s(t)$ and a server capacity $q_s(t)$. The server works according to the following rules. In all notation below, τ denotes the current time.

1) Initially $q_s(0) := 0$ and $d_s(0) := 0$.
2) Arriving jobs are queued in the FCFS manner in a task queue of pending jobs.
3) When a new job J_i arrives, if the task queue is empty and if $q_s(\tau) \geq (d_s(\tau) - \tau)U_s$, then the server budget is recharged to the maximum value Q_s and the server deadline is changed as $d_s(\tau) := \tau + P_s$.
4) The job (if any) at the head of the task queue is allowed to contend for the resource with a deadline equal to the server deadline $d_s(\tau)$.
5) Whenever a served job executes, the budget $q_s(\tau)$ is decreased by the execution time received by the job.
6) When $q_s(\tau) = 0$, the server is suspended until time $d_s(\tau)$, i.e., jobs served by the server do not contend for resources in $[\tau, d_s(\tau)]$. At time $d_s(\tau)$, the server budget is recharged to the maximum value Q_s and the server deadline is changed as $d_s(d_s(\tau)) := d_s(\tau) + P_s$.

The difference between CBS and Hard-CBS is how either server reacts in step 6) to the budget being exhausted. In CBS the budget is recharged to full immediately, whereas in Hard-CBS the budget is recharged only when the server deadline is reached, and in the meantime the jobs served by the server are not allowed to contend for resources. In spite of this difference, we show below that Hard-CBS provides the same service curve as CBS.

Theorem 4. *Hard-CBS with a period P_s and maximum budget Q_s provides a service guarantee of β_{CBS} as defined in (11).*

Proof: A CBS behaves identical to a Hard-CBS with the same parameters, if the server budget is always exhausted exactly at the server deadline. A CBS can indeed behave like this, as we have argued when showing that the service curve provided by the CBS is tight. When a CBS behaves so, we cannot distinguish between CBS and Hard-CBS by merely looking at the input and output arrival streams. Thus, Hard-CBS with period P_s and maximum budget Q_s provides a service curve β_{CBS} defined in (11). ■

With the presented analysis, we have achieved what we set out to do: to compute the delay bound of an aperiodic task with any given input arrival trace, when scheduled via a (Hard-)CBS with given parameters. We did this by demonstrating that the (Hard-)CBS provides a service curve to the stream of jobs it serves. In addition we showed that the computed bound is tight.

A. Strict Service Curve

The service curves guaranteed by CBS and Hard-CBS with equal parameters are the same. Consequently, the delay bound for any stream of aperiodic jobs is the same, when served by either CBS or Hard-CBS with the same parameters. However, as discussed in Section I, there seems to be some difference in the guarantee provided by either server. We shall try to understand this difference in this section.

Let a stream of aperiodic jobs be served by a CBS with period P_s and maximum budget Q_s. Let at some point of time t, the cumulative remaining execution time of all jobs queued in the task queue be $x(t)$, i.e., $x(t)$ is the backlogged demand, which can be written as $R(t) - R'(t) = x(t)$. We would like to answer the question: what is the earliest time by when this backlogged demand would be served, under the FCFS queueing discipline.

To illustrate the utility of such a computation, we consider an example. Consider a fork system, where jobs can be sent to several parallel servers based on a routing decision. If we could determine the dynamic worst-case finish time of the job when served in each of the servers, then we could route the job to the server that provides the least such finish time. If the queueing discipline is FCFS, this entails computation of the time by when the backlogged demand of each server would have been served. Does the service curve guarantee, characterized by (5), help in answering such a question? The answer is no. Even though the CBS provides a service curve β_{CBS}, it can provide *no* guaranteed time by when the pending jobs with execution demand $x(t)$ would be served. Another example is the use of dynamic voltage scaling (DVS) to adjust the frequency of a system such that it serves the backlogged demand within a specified time window [10]. The service curve again provides us no means to perform such a computation by using only the backlogged demand.

This limitation occurs because the guarantee provided by the service curve says for every time t there is some time $s \leq t$, when a certain service guarantee is provided. This value s can be such that $(t - s)$ is arbitrarily large, and thus to apply this service guarantee we must look back into the history of the aperiodic stream and its execution via the server. No guarantees are provided independent of what has happened in the past. Thus, by merely knowing the quantum of the backlogged demand we cannot compute a guaranteed time by when it would be served. This limitation is overcome if the server provides a strict service curve [9] defined as below.

Definition 5 (Strict Service Curve). *A resource is said to provide a strict service curve $\tilde{\beta}$ if in* any *interval of length Δ, within a backlogged period, the amount of service provided is at least $\tilde{\beta}(\Delta)$.*

Note that the strict service curve is not defined with respect to an input arrival curve. A strict service curve is also a service curve, but the reverse is not always true [9]. To indicate that a service curve is a strict service curve, we use the tilde symbol as $\tilde{\beta}$. We say a resource provides no strict service curve, if $\tilde{\beta}(\Delta) = 0$, for all $\Delta \geq 0$.

We now characterize the considered servers, namely CBS and Hard-CBS, in terms of the strict service curve they provide, if any.

Theorem 6. *CBS provides no strict service curve and Hard-CBS with a period P_s and maximum budget Q_s provides a* strict *service of $\tilde{\beta}_{\text{H-CBS}}$ given as*

$$\tilde{\beta}_{\text{H-CBS}}(\Delta) = F(P_s, Q_s, P_s - Q_s, \Delta). \tag{17}$$

Proof: We can prove that CBS provides no strict service curve by contradiction. Let a CBS with a period P_s and maximum budget Q_s provide a strict service curve $\tilde{\beta}_{\text{CBS}}$ such that $\tilde{\beta}_{\text{CBS}}(\Delta_1) > 0$, for some given Δ_1. We only consider $\Delta_1 > (P_s - Q_s)$, as for any smaller Δ_1 even the tight supply curve $\beta_{\text{CBS}}(\Delta_1) = 0$. Consider a system where the CBS is co-scheduled with a periodic task T with execution time Δ_1 and period $\Delta_1/(1 - U_s)$, where $U_s = Q_s/P_s$. The task-set is schedulable. Let an aperiodic job with execution demand $\Delta_1 U_s/(1 - U_s) + 1$ arrive at time 0 along with the release of a periodic job. It can be shown that the aperiodic job would be executed in the interval $[0, \Delta_1/(1 - U_s) - \Delta_1]$, and then the periodic job would be executed in the interval $[\Delta_1/(1-U_s) - \Delta_1, \Delta_1/(1-U_s)]$. In this interval of length Δ_1, the backlogged aperiodic job receives no service. Hence the contradiction. Thus, for no Δ_1 we can claim that $\tilde{\beta}_{\text{CBS}}(\Delta_1) > 0$.

We now prove the strict service curve provided by Hard-CBS. Let $[t, t+\Delta_1]$ be some given interval of time, when the server is backlogged. For a Hard-CBS, the server deadline $d_s(u) \leq (u + P_s)$, for any $u \geq 0$. Thus, for any time u when the server is backlogged, the server budget must have been reset to the maximum in the interval $[u - P_s, u]$. Let s be the latest time when the server budget was reset to the maximum on or before t. Then, since the budget reduces at a maximum speed of 1, we have $q_s(t) \geq (Q_s - (t - s))^+$. Also, from the server algorithm we can show that

$$d_s(t + \Delta_1) = \left\lfloor \frac{t + \Delta - s}{P_s} \right\rfloor P_s + P_s + s \tag{18}$$

From schedulability of the server, we know that the server budget, if requested, will be served within the server deadline. Using this we have

$$q_s(t + \Delta_1) \leq (d_s(t + \Delta_1) - (t + \Delta_1))^+ . \tag{19}$$

Let ρ denote the cumulative added server budget in the interval $[t, t + \Delta_1]$. Then, ρ is given by

$$\rho = \left(\frac{d_s(t + \Delta_1) - (s + P_s)}{P_s} \right) Q_s. \tag{20}$$

The total execution time provided by the server in the interval $[t, t+\Delta_1]$ is equal to $q_s(t) + \rho - q_s(t+\Delta_1)$. By using the above results, we can show that this quantity is equal to $\tilde{\beta}_{\text{H-CBS}}(\Delta_1)$. Hence, Hard-CBS provides a strict service curve of $\tilde{\beta}_{\text{H-CBS}}$. ■

With the strict service guarantee provided by Hard-CBS, we can conclude that backlogged jobs with execution demand $x(t)$ would be served within time $y(t)$ where $y(t)$ is given as

$$y(t) := \arg\min\{y : \tilde{\beta}_{\text{H-CBS}}(y(t)) \geq x(t)\}. \tag{21}$$

This guarantee holds independent of the input arrival trace of the jobs. No such guarantee is provided by CBS. Thus, while the service curves and consequently the delay bound provided by the two variants of CBS are the same, the Hard-CBS provides a strict service curve which enables dynamic computation of the earliest time when the pending jobs would be completed.

VI. Results for Other Servers

We indicated for the soft and hard variants of Constant Bandwidth Server how to compute the delay bound of streams of aperiodic jobs when scheduled via the servers. A similar approach of analyzing the server algorithm can be used to compute the service curve, and consequently the delay bounds, for other servers. In this section, we present the results for commonly employed other servers, without providing the proofs.

Theorem 7. *A Dynamic Sporadic Server [11, 12] with period P_s and capacity Q_s provides a strict service curve $\tilde{\beta}_{DSS}$ given as*

$$\tilde{\beta}_{DSS}(\Delta) = F(P_s, Q_s, P_s - Q_s, \Delta). \qquad (22)$$

and a service curve β_{DSS} given as

$$\beta_{DSS}(\Delta) = F(P_s, Q_s, 0, \Delta). \qquad (23)$$

Theorem 8. *A Polling Server [13] with period P_s and capacity Q_s provides a strict service curve $\tilde{\beta}_{PS}$ given as*

$$\tilde{\beta}_{PS}(\Delta) = F(P_s, Q_s, P_s - Q_s, \Delta) \qquad (24)$$

Theorem 9. *A Deferrable Server [14] with period P_s and capacity Q_s provides a strict service curve $\tilde{\beta}_{DS}$ given as*

$$\tilde{\beta}_{DS}(\Delta) = F(P_s, Q_s, P_s - Q_s, \Delta). \qquad (25)$$

and a service curve β_{DS} given as

$$\beta_{DS}(\Delta) = F(P_s, Q_s, 0, \Delta). \qquad (26)$$

Theorem 10. *A Sporadic Server [15] with period P_s and capacity Q_s provides a strict service curve $\tilde{\beta}_{SS}$ given as*

$$\tilde{\beta}_{SS}(\Delta) = F(P_s, Q_s, P_s - Q_s, \Delta). \qquad (27)$$

and a service curve β_{SS} given as

$$\beta_{SS}(\Delta) = F(P_s, Q_s, 0, \Delta). \qquad (28)$$

VII. Conclusion

Servers provide means to schedule a stream of aperiodic jobs in the presence of other periodic tasks and/or servers. Temporal isolation amongst tasks is an important consideration in the design of such servers. This temporal isolation enables extension of standard schedulability tests to be applied to servers. However, not much attention has been focussed on computing safe upper-bounds on the worst-case delay suffered by jobs of the aperiodic stream. Such computation is particularly significant when the aperiodic jobs have to finish within hard deadlines.

Towards computing the delay bounds provided by servers, we considered existing resource models in the real-time literature, such as supply bound function and resource models from Real-Time Calculus. However, from our examples, we saw that while in some cases they provide no meaningful delay bounds, in others they provide non-tight delay bounds. We then proposed to use the supply curve as proposed in Network Calculus to model servers. For the specific and representative case of Constant Bandwidth Server (CBS) we showed that by analyzing the server algorithm we can compute a provably safe and tight service curve. As a distinguishing feature the hard variant of CBS also guarantees a strict service curve. A similar approach of analyzing the server algorithm can be employed to characterize other servers. A list of such results were presented for some of the other commonly used servers.

Acknowledgements

This work was supported by EU FP7 projects EURETILE and PRO3D, under grant numbers 247846 and 249776, respectively.

References

[1] Z. Deng and J. W.-S. Liu, "Scheduling real-time applications in an open environment," in *IEEE Real-Time Systems Symposium*, pp. 308–319, IEEE Computer Society, 1997.

[2] G. C. Buttazzo, *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications*. Norwell, MA, USA: Kluwer Academic Publishers, 1997.

[3] L. Abeni and G. C. Buttazzo, "Resource reservation in dynamic real-time systems," *Real-Time Systems*, vol. 27, no. 2, pp. 123–167, 2004.

[4] L. Abeni and G. C. Buttazzo, "Integrating multimedia applications in hard real-time systems," in *IEEE Real-Time Systems Symposium*, pp. 4–13, 1998.

[5] I. Shin and I. Lee, "Compositional real-time scheduling framework," in *RTSS*, pp. 57–67, IEEE Computer Society, 2004.

[6] L. Abeni, L. Palopoli, C. Scordino, and G. Lipari, "Resource reservations for general purpose applications," *IEEE Trans. Industrial Informatics*, vol. 5, no. 1, pp. 12–21, 2009.

[7] L. Thiele, S. Chakraborty, and M. Naedele, "Real-time calculus for scheduling hard real-time systems," in *Circuits and Systems, 2000. Proceedings. ISCAS 2000 Geneva. The 2000 IEEE International Symposium on*, vol. 4, pp. 101 –104 vol.4, 2000.

[8] E. Wandeler, L. Thiele, M. Verhoef, and P. Lieverse, "System architecture evaluation using modular performance analysis: a case study," *STTT*, vol. 8, no. 6, pp. 649–667, 2006.

[9] J.-Y. L. Boudec and P. Thiran, *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet*, vol. 2050 of *Lecture Notes in Computer Science*. Springer, 2001.

[10] A. Maxiaguine, S. Chakraborty, and L. Thiele, "Dvs for buffer-constrained architectures with predictable qos-energy tradeoffs," in *CODES+ISSS* (P. Eles, A. Jantsch, and R. A. Bergamaschi, eds.), pp. 111–116, ACM, 2005.

[11] M. Spuri and G. C. Buttazzo, "Scheduling aperiodic tasks in dynamic priority systems," *Real-Time Systems*, vol. 10, no. 2, pp. 179–210, 1996.

[12] T. M. Ghazalie and T. P. Baker, "Aperiodic servers in a deadline scheduling environment," *Real-Time Systems*, vol. 9, no. 1, pp. 31–67, 1995.

[13] L. Sha, J. P. Lehoczky, and R. Rajkumar, "Solutions for some practical problems in prioritized preemptive scheduling," in *IEEE Real-Time Systems Symposium*, pp. 181–191, IEEE Computer Society, 1986.

[14] J. P. Lehoczky, L. Sha, and J. K. Strosnider, "Enhanced aperiodic responsiveness in hard real-time environments," in *IEEE Real-Time Systems Symposium*, pp. 261–270, IEEE Computer Society, 1987.

[15] B. Sprunt, L. Sha, and J. P. Lehoczky, "Aperiodic task scheduling for hard real-time systems," *Real-Time Systems*, vol. 1, no. 1, pp. 27–60, 1989.