12-3-2007

Compositional Analysis Framework using EDP Resource Models

Arvind Easwaran
University of Pennsylvania, arvinde@cis.upenn.edu

Madhukar Anand
University of Pennsylvania, anandm@cis.upenn.edu

Insup Lee
University of Pennsylvania, lee@cis.upenn.edu

Compositional Analysis Framework using EDP Resource Models

Abstract
Compositional schedulability analysis of hierarchical scheduling frameworks is a well studied problem, as it has wide-ranging applications in the embedded systems domain. Several techniques, such as periodic resource model based abstraction and composition, have been proposed for this problem. However these frameworks are sub-optimal because they incur bandwidth overhead. In this work, we introduce the Explicit Deadline Periodic (EDP) resource model, and present compositional analysis techniques under EDF and DM. We show that these techniques are bandwidth optimal, in that they do not incur any bandwidth overhead in abstraction or composition. Hence, this framework is more efficient when compared to existing approaches.

Comments
Copyright 2007 IEEE. Presented at *28th International IEEE Real-Time Systems Symposium*, December 2007, 10 pages.

Compositional Analysis Framework using EDP Resource Models *

Arvind Easwaran, Madhukar Anand, and Insup Lee
Department of Computer and Information Science,
University of Pennsylvania,
Philadelphia, PA, 19104, USA
{arvinde,anandm,lee}@cis.upenn.edu

Abstract

Compositional schedulability analysis of hierarchical scheduling frameworks is a well studied problem, as it has wide-ranging applications in the embedded systems domain. Several techniques, such as periodic resource model based abstraction and composition, have been proposed for this problem. However these frameworks are sub-optimal because they incur bandwidth overhead. In this work, we introduce the Explicit Deadline Periodic (EDP) resource model, and present compositional analysis techniques under EDF *and* DM*. We show that these techniques are bandwidth optimal, in that they do not incur any bandwidth overhead in abstraction or composition. Hence, this framework is more efficient when compared to existing approaches.*

1. Introduction

Real-time embedded systems consist of a combination of different processors and programmable components with deadlines. Their increasing complexity demands advanced design and analysis methods. Component-based engineering is widely accepted as an approach to facilitate their design. It is founded on the paradigm that a complex system can be designed by decomposing it into simpler components, and then composing the components using interfaces that abstract complexities. To take advantage of this component-based design for real-time systems, schedulability analysis of such systems should be addressed.

Component-based real-time systems often involve hierarchical scheduling frameworks that support resource sharing among components under different scheduling algorithms. This framework can be represented as a tree of nodes, where each node denotes a component comprising of some real-time workload and a scheduling policy. In this representation, resources are allocated from a parent node to its children. For such frameworks, it is desirable to achieve schedulability analysis *compositionally*, i.e., we should be able to check schedulability of the system by composing interfaces that abstract component-level resource demand. Ideally, these interfaces must use minimum resources to satisfy demand of components. Furthermore, these interfaces should expose only so much information about components as is required for this analysis.

Resource model based component interfaces, and their compositional analysis is a well known technique that addresses the aforementioned problem [14, 19, 20, 21]. A resource model represents the characteristics of a resource supply, and hence can be used for schedulability analysis. A periodic resource model $\Gamma = (\Pi, \Theta)$, represents a resource supply that has Θ units of resource capacity in every Π time units. These models have been extensively studied under fixed-priority [1, 3, 14, 19] and EDF [20] scheduling.

In this work, we introduce the *Explicit Deadline Periodic* (EDP) resource model which generalizes the periodic resource model. We use this model for compositional analysis of hierarchical scheduling frameworks that are comprised of EDF and DM schedulers. An EDP resource model $\Omega = (\Pi, \Theta, \Delta)$ repetitively provides Θ units of resource within Δ time units, where the period of repetition is Π. This choice of model is implementation-oriented, because many popular real-time schedulers support its semantics. This model also closely characterizes the resource demand of many real-time applications like avionics systems [7].

Bandwidth of a resource model ($\frac{\Theta}{\Pi}$ for both periodic and EDP models) is a measure of the resource requirement of the model. Therefore, it is desirable to minimize this quantity when abstracting components into resource models. When compared to the EDP model based analysis presented in this paper, periodic resource model based techniques incur bandwidth overhead which can be explained as follows. In order for a resource model to be able to schedule a demand, length of the largest time interval with no supply (henceforth denoted as *starvation length*) must be

*This research was supported in part by NSF CNS-0509327, NSF CNS-0509143, NSF CNS-0720703, and FA9550-07-1-0216.

smaller than the earliest deadline in demand. Since periodic models have implicit deadlines, satisfaction of this requirement depends entirely on its capacity when resource period is fixed. However, for EDP models, starvation length depends on both its capacity and deadline. Therefore, it is possible to adjust the deadline of EDP models and satisfy this scheduling requirement without changing capacity (and hence bandwidth).

In addition to introducing EDP models, the contributions of this work include, (1) an efficient algorithm to compute a bandwidth optimal EDP model based abstraction (henceforth denoted as EDP interface) for a component whose demand comprises of sporadic tasks, (2) an exact transformation of EDP models to sporadic tasks for compositional analysis (this transforms the composition problem to the abstraction problem which can be solved as in (1) above), and (3) bandwidth optimal, priority preserving abstraction and transformation techniques when component priorities are specified by the designer. These optimality results are defined when the interface periods are fixed a priori. For instance, the designer may specify periods taking into consideration preemption overhead.

Related work. For real-time systems, there has been a growing attention to hierarchical scheduling frameworks. Since a two-level hierarchical framework was introduced by Deng and Liu [5], its schedulability has been analyzed under fixed-priority [10] and EDF [13, 15] scheduling. Since in open systems, there can be more than two levels of hierarchy, it is desirable to develop a more general analysis framework. The bounded-delay resource model [18] has been proposed to achieve a clean separation in a multi-level hierarchical scheduling framework, and analysis techniques [8, 21] have been introduced for this resource model. However, no scheduling algorithm is known for such models, and hence compositional analysis has only been addressed w.r.t feasibility. Techniques have also been proposed to support interacting tasks [17] and mutually exclusive resource sharing between components [4] under periodic resource models.

There have been studies [23, 9, 22, 6] on interface theory for hierarchical scheduling frameworks. They use assume/guarantee interfaces to abstract the resource requirement of components in the form of demand functions [23, 22], bounded-delay resource models [9], or periodic resource models [6]. Unlike resource models, demand functions suffer from the problem of requiring large amounts of space.

2. Preliminaries

In this paper we assume that each real-time task is an independent sporadic task. A task $T = (p, e, d)$ has minimum separation p, execution requirement e and relative deadline d such that $e \leq d \leq p$. For compositional schedulability analysis using resource models, the component must export its worst case resource demand which depends on the task model and scheduler. Any task model for which the component can compute this demand, can be used in our framework. A real-time component consists of a real-time workload and a scheduling policy for the workload. In this work, we assume that this workload comprises of sporadic tasks and other real-time components.

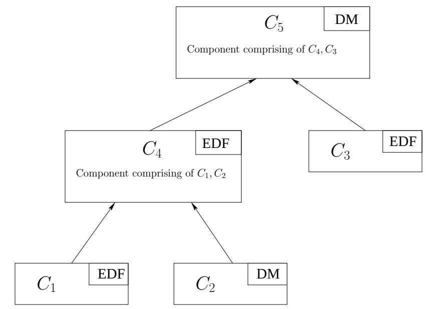


Figure 1. Hierarchical real-time system

Definition 2.1 (Real-time component) *A* real-time component *C is specified as $C = \langle\{C_1, \ldots, C_n\}, S\rangle$, where each C_i is either another* real-time component *or a sporadic task. The workload $C_1, \ldots, C_n$ is scheduled under S, where S is either* DM *or* EDF.

A hierarchical real-time system comprises of one or more components arranged in a scheduling hierarchy. Note that this system is assumed to be free of cycles, and therefore the workload of a component at the bottom of the hierarchy is comprised of only sporadic tasks. Figure 1 shows such a hierarchical system, where the workloads of C_1, C_2 and C_3 are comprised of only sporadic tasks. We assume that this system is scheduled on an uniprocessor platform.

The resource demand of a component with only sporadic tasks, is the collective resource requirement of tasks when they are scheduled under component scheduler. Recall that the demand bound function of a component (dbf) gives the maximum resource demand in a given time interval [2, 11]. Equation (1) gives the dbf for a component C comprising of sporadic tasks $\{T_1, \ldots, T_n\}$ scheduled under EDF [2]. Similarly, Equation (2) gives the dbf for task T_i in C when tasks are scheduled under DM [11]. In this equation, $\mathcal{HP}(T_i)$ denotes the set of tasks in C that have priority higher than T_i.

$$\mathrm{dbf}_C(t) = \sum_{i=1}^{n} \left(\left\lfloor \frac{t + p_i - d_i}{p_i} \right\rfloor e_i \right) \tag{1}$$

$$\mathrm{dbf}_{C,i}(t) = \sum_{T_k \in \mathcal{HP}(T_i)} \left(\left\lceil \frac{t}{p_k} \right\rceil e_k \right) + e_i \tag{2}$$

As described in the introduction, a resource model specifies the timing properties of a resource supply. The supply bound function (sbf) of a resource model gives the minimum amount of resource that the model is guaranteed to provide in a given time interval. For a periodic resource model $\Gamma = (\Pi, \Theta)$, Equation (3) first proposed by Shin and Lee [20], gives its sbf. In this equation, $x = 2(\Pi - \Theta)$ and $y = \left\lfloor \frac{t-(\Pi-\Theta)}{\Pi} \right\rfloor$.

$$\mathrm{sbf}_\Gamma(t) = \begin{cases} y\Theta + \max\{0, t - x - y\Pi\} & t \geq \Pi - \Theta \\ 0 & \text{Otherwise} \end{cases} \quad (3)$$

Compositional analysis of hierarchical systems has been done by abstracting components into resource models such as, periodic resource model [20], and bounded delay resource model [21]. In these analyses, dbf of components and sbf of resource models are used to define schedulability conditions. If the deadlines of a component workload are met when higher priority interference is largest (dbf) and supply from a resource model is least (sbf), then the resource model can always successfully schedule the component. Theorems 2.2 and 2.3 use this property to give schedulability conditions over periodic resource models, when component workload is comprised of only sporadic tasks. In these theorems, LCM_C denotes the least common multiple of minimum separations of tasks in C. They also define the notion of *exactly schedulable*, that identifies the condition under which the resource model uses minimum bandwidth to schedule the component.

Theorem 2.2 [20] *A component* $C = \langle\{T_1 = (p_1, e_1, d_1), \ldots, T_n = (p_n, e_n, d_n)\}, \mathrm{EDF}\rangle$ *is schedulable using a periodic resource model* Γ *iff*

$$\forall t \text{ s.t. } 0 < t \leq LCM_C + \max_{i=1}^{n} d_i, \mathrm{dbf}_C(t) \leq \mathrm{sbf}_\Gamma(t) \quad (4)$$

Furthermore, C *is* exactly schedulable *by* Γ *iff in addition,* $\exists t$ *s.t.* $\min_{i=1}^n d_i \leq t \leq LCM_C + \max_{i=1}^n d_i$ *and* $\mathrm{dbf}_C(t) = \mathrm{sbf}_\Gamma(t)$.

Note that if for any task T_j in the above theorem, $d_j > p_j$, then for schedulability of C over a model $\Gamma = (\Pi, \Theta)$ we also require $\frac{\Theta}{\Pi} \geq \sum_{i=1}^n \frac{e_i}{p_i}$ [2].

Theorem 2.3 [20] *A component* $C = \langle\{T_1 = (p_1, e_1, d_1), \ldots, T_n = (p_n, e_n, d_n)\}, \mathrm{DM}\rangle$ *is schedulable using a periodic resource model* Γ *iff*

$$\forall T_i, \exists t_i \in [0, d_i] \text{ s.t. } \mathrm{dbf}_{C,i}(t_i) \leq \mathrm{sbf}_\Gamma(t_i) \quad (5)$$

Furthermore, C *is* exactly schedulable *by* Γ *iff in addition,* $\forall T_i, \forall t \in [0, d_i]\, \mathrm{dbf}_{C,i}(t) \geq \mathrm{sbf}_\Gamma(t)$.

Shin and Lee [20], use Theorems 2.2 and 2.3 to generate periodic resource model based component abstractions. For analysis of components comprising of other components, they transform a periodic model $\Gamma = (\Pi, \Theta)$ into a sporadic task $T = (\Pi, \Theta, \Pi)$, thereby reducing the composition problem to the abstraction problem. This framework is

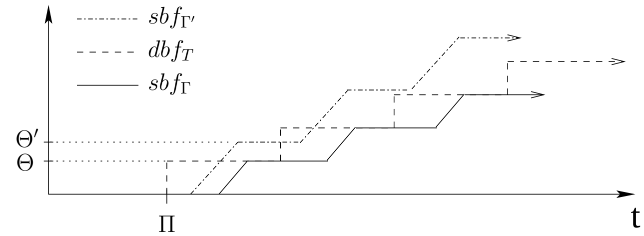


Figure 2. Sub-optimal transformation

sub-optimal for the following reasons: (1) although Theorems 2.2 and 2.3 use sbf_Γ, the authors use a linear lower bound of sbf_Γ to generate abstractions, and hence incur bandwidth overhead. (2) The transformation of periodic resource model based abstraction (Γ) into sporadic task (T) induces demand overhead. This can be explained by observing that any periodic supply that can schedule T must satisfy Γ, but there may exist a supply Γ' which satisfies Γ but cannot schedule T. Figure 2 illustrates this shortcoming for an example, where model $\Gamma' = (\Pi, \Theta')$ with $\mathrm{sbf}_{\Gamma'}(\Pi) < \Theta$ and $\Theta' > \Theta$ cannot schedule T under EDF. In this paper, we overcome these shortcomings, and hence provide an efficient compositional analysis framework.

3. EDP resource model

Recollect that an explicit deadline periodic (EDP) resource model $\Omega = (\Pi, \Theta, \Delta)$ provides Θ units of resource within Δ time units, with this process repeating every Π time units. It is then easy to see that a periodic resource model $\Gamma = (\Pi, \Theta)$ is the EDP model (Π, Θ, Π). In this paper, we focus on EDP models with pre-period deadlines, i.e., $\Delta \leq \Pi$. The supply bound function of Ω is given by the following equation, where $x = (\Pi + \Delta - 2\Theta)$ and $y = \left\lfloor \frac{t-(\Delta-\Theta)}{\Pi} \right\rfloor$. We also define a linear function usbf_Ω as the upper bound for sbf_Ω. We use these functions to generate EDP interfaces. Figure 3 shows the sbf and usbf of Ω. Observe that starvation length $(\Pi + \Delta - 2\Theta)$ of Ω can be smaller than the starvation length $(2\Pi - 2\Theta)$ of the periodic model Γ, even though they have the same bandwidth.

$$\mathrm{sbf}_\Omega(t) = \begin{cases} y\Theta + \max\{0, t - x - y\Pi\} & t \geq \Delta - \Theta \\ 0 & \text{Otherwise} \end{cases} \quad (6)$$

$$\mathrm{usbf}_\Omega(t) = \frac{\Theta}{\Pi}(t - (\Delta - \Theta)) \quad (7)$$

Schedulability conditions for components with workload comprising only of sporadic tasks and over EDP resource models, are identical to Theorems 2.2 and 2.3 when sbf from Equation (6) is used. Therefore, we use these theorems to generate EDP interfaces. Before we present the abstraction procedure, we define the term *bandwidth optimal*

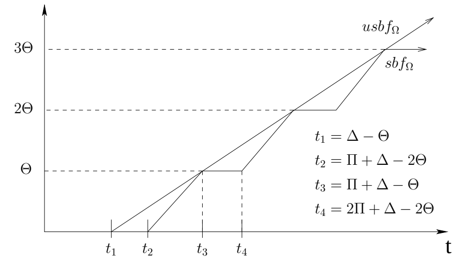


Figure 3. sbf **and** usbf **of model** $\Omega = (\Pi, \Theta, \Delta)$

for an EDP model and also derive conditions under which this optimality is achieved.

Definition 3.1 (Bandwidth optimal) *Given a component C and period Π, model $\Omega = (\Pi, \Theta, \Delta)$ that satisfies schedulability conditions for C is* bandwidth optimal *iff its bandwidth is the minimum over all EDP models that exactly schedule C and have period Π.*

Theorem 3.2 *Let EDP model $\Omega = (\Pi, \Theta, \Delta)$ with $\Delta \leq \Pi$ exactly schedule component $C = \langle \{T_1, \ldots, T_n\}, S \rangle$. Then, Ω is bandwidth optimal if $\Delta = \Theta$.*

Proof We show that if any model $\Omega' = (\Pi, \Theta', \Delta')$ with $\Delta' > \Theta'$ exactly schedules C, then it must be the case that $\Theta' \geq \Theta$. We prove this by contradiction and assume $\Theta' < \Theta$. We first show $\forall t \geq \Pi - \Theta$, $\mathrm{sbf}_{\Omega'}(t) < \mathrm{sbf}_\Omega(t)$. Since $\frac{\Delta' - \Theta'}{\Pi} < 1$, $\left\lfloor \frac{t}{\Pi} \right\rfloor$ can be at most one greater than $\left\lfloor \frac{t - \Delta' + \Theta'}{\Pi} \right\rfloor$. Hence, we consider the following two cases.

Case $\left\lfloor \frac{t}{\Pi} \right\rfloor = \left\lfloor \frac{t - \Delta' + \Theta'}{\Pi} \right\rfloor (= k)$**:**
In this case, $k\Theta > k\Theta'$ and $\max\{0, t - (\Pi - \Theta) - k\Pi\} \geq \max\{0, t - (\Pi + \Delta' - 2\Theta') - k\Pi\}$. The second inequality holds because $\Pi - \Theta < \Pi + \Delta' - 2\Theta'$. Therefore, from Equation (6) we get $\mathrm{sbf}_{\Omega'}(t) < \mathrm{sbf}_\Omega(t)$.

Case $\left\lfloor \frac{t}{\Pi} \right\rfloor = k$, $\left\lfloor \frac{t - \Delta' + \Theta'}{\Pi} \right\rfloor = k - 1$**:**
Here, $\mathrm{sbf}_{\Omega'}(t) = (k-1)\Theta' + \max\{0, t - (\Pi + \Delta' - 2\Theta') - k\Pi + \Pi\}$. Observe that, $\max\{0, t - (\Pi + \Delta' - 2\Theta') - k\Pi + \Pi\} < \Theta'$ for all $t \geq \Pi - \Theta$. This is true because $\Pi - \Theta < \Pi + \Delta' - 2\Theta'$. Therefore, $\mathrm{sbf}_{\Omega'}(t) < (k-1)\Theta' + \Theta' \leq k\Theta \leq \mathrm{sbf}_\Omega(t)$.

We now derive a contradiction by showing that Ω' cannot schedule C. When $S =$ EDF, since Ω exactly schedules C, from Theorem 2.2 we get $\mathrm{sbf}_\Omega(t') = \mathrm{dbf}_C(t')$ for some $t' \geq \min_i d_i \geq \Pi - \Theta$. Now since $\mathrm{sbf}_{\Omega'}(t') < \mathrm{sbf}_\Omega(t')$, we get that Ω' does not schedule C. When $S =$ DM from Theorem 2.3 we get $\forall T_i, \forall t \in [0, d_i]\, \mathrm{dbf}_{C,i}(t) \geq \mathrm{sbf}_\Omega(t)$. Since $\Pi - \Theta < \Pi + \Delta' - 2\Theta'$, this then means that $\forall T_i, \forall t \in [0, d_i]\, \mathrm{dbf}_{C,i}(t) > \mathrm{sbf}_{\Omega'}(t)$. From Theorem 2.3 we then get Ω' does not schedule C. □

4. EDP interface generation

In this section, given interface period Π and a component C whose workload comprises of only sporadic tasks, we generate an EDP interface for C. This interface satisfies the following notion of optimality.

Definition 4.1 (Bandwidth-deadline optimal)
$\Omega = (\Pi, \Theta, \Delta)$ *is a* bandwidth-deadline optimal *EDP interface for a component C, iff*

- *Ω is bandwidth optimal for C, and*
- *For all $\Omega' = (\Pi, \Theta, \Delta')$ such that Ω' is bandwidth optimal for C, $\Delta' \leq \Delta$.*

Among all EDP interfaces that are bandwidth optimal for a component C, the interface which has the largest deadline is desirable. This is so because, a larger deadline implies increased starvation length, which means reduced demand at the next level for compositional analysis. We now describe techniques to generate these interfaces for components scheduled under EDF or DM.

Abstraction under EDF. Let $\Omega_m = (\Pi, \Theta_m, \Delta_m)$ denote the bandwidth-deadline optimal EDP interface for a component C scheduled under EDF, where Π is known a priori. The following abstraction procedure then computes Θ_m and Δ_m (for the computations given below, replace sbf_Γ in Equation (4) with sbf_Ω defined in Equation (6)).

1. Set $\Delta = \Theta$, evaluate Equation (4) for each time interval length, and choose the maximum $\Theta (= \Theta_m)$ over all interval lengths (only those interval lengths at which dbf changes, need to be considered). Note that $(\Pi, \Theta_m, \Theta_m)$ is bandwidth optimal for C.
2. Set $\Theta = \Theta_m$, evaluate Equation (4) for each interval length, and choose the maximum $\Delta (= \Delta_m)$ over all interval lengths. It is then easy to see that Ω_m is bandwidth-deadline optimal for C.

Abstraction under DM. We now compute Θ_m and Δ_m for the case when component C is scheduled under DM. The abstraction procedure consists of the following two steps (for these computations replace sbf_Γ in Equation (5) with sbf_Ω defined in Equation (6)):

1. Set $\Delta = \Theta$ and evaluate Equation (5) for each task T_i in C, and for each interval length (only those interval lengths at which dbf changes, need to be considered). Choose the minimum $\Theta (= \Theta_{m_i})$ over all interval lengths and let $\Theta_m = \max_i \Theta_{m_i}$. Then, $(\Pi, \Theta_m, \Theta_m)$ is bandwidth optimal for C.

2. Set $\Theta = \Theta_m$ and evaluate Equation (5) for each task T_i in C, and for each interval length. Choose the maximum $\Delta(= \Delta_{m_i})$ over all interval lengths and let $\Delta_m = \min_i \Delta_{m_i}$. Then, $\Omega_m = (\Pi, \Theta_m, \Delta_m)$ is bandwidth-deadline optimal for C.

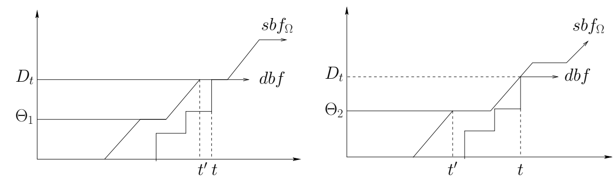
In order to generate the EDP interfaces we have to evaluate Equations (4) and (5) for different values of interval length. Since they involve floor functions, computing Θ that satisfies these equations is non-trivial. Therefore, we now present a procedure that efficiently computes Θ, when interval length t, corresponding dbf value D_t, period Π and deadline Δ are all known.

Algorithm 1 Algorithm for Equations (4) and (5)

Input: t, D_t, Π, Δ
Output: Θ_{opt}
    // Let $t = k\Pi + \alpha$ **and** $t' = \left\lfloor \frac{t-\Delta+\Theta}{\Pi} \right\rfloor \Pi$
1: Set $\Theta = 0$ and compute $s_0 = \mathrm{usbf}_\Omega(t')$. Let $t_0 = \left\lfloor \frac{t-\Delta}{\Pi} \right\rfloor \Pi$.
2: **if** $\alpha < \Delta$ **then**
3:     Set $\Theta = \Delta - \alpha$ and compute $s_1 = \mathrm{usbf}_\Omega(t')$. Let $t_1 = \left\lfloor \frac{t-\alpha}{\Pi} \right\rfloor \Pi$.
4: **else**
5:     Set $\Theta = \Pi + \Delta - \alpha$ and compute $s_1 = \mathrm{usbf}_\Omega(t')$. Let $t_1 = \left\lfloor \frac{t+\Pi-\alpha}{\Pi} \right\rfloor \Pi$.
6: **end if**
7: **if** $s_1 > s_0$ and $s_1 \leq D_t$ **then**
8:     $t_{max} = t_1$
9: **else**
10:     $t_{max} = t_0$
11: **end if**
    // Case D_t **is a multiple of** Θ_{opt}
12: Compute Θ_1 s.t. $\frac{\Theta_1}{\Pi}(t_{max} - (\Delta - \Theta_1)) = D_t$
    // Case D_t **is not a multiple of** Θ_{opt}
13: Compute Θ_2 s.t. $\frac{\Theta_2}{\Pi}(t_{max} + \Pi - (\Delta - \Theta_2)) = D_t + t_{max} + \Pi - t$
14: Return $\Theta_{opt} = \min_i \Theta_i$, where $\Theta_i \in \mathbb{R}^+$ and $i \in \{1, 2\}$.

Algorithm 1 computes Θ_{opt} that satisfies the condition $\mathrm{sbf}_\Omega(t) = D_t$. This procedure exploits the fact that there are only two possible values for the floor function in sbf_Ω. This follows from the observation that $\Theta \leq \Delta \leq \Pi$ and $\alpha < \Pi$, where α is the remainder obtained when t is divided by Π. These values of the floor function are computed in Lines 1,3 and 5 of the algorithm, with Θ set to the smallest capacity of Ω that can generate those values.

In the algorithm, $t' \left(= \left\lfloor \frac{t-\Delta+\Theta}{\Pi} \right\rfloor \Pi\right)$ denotes the interval length at which sbf_Ω is a multiple of Θ. Observe that $\mathrm{sbf}_\Omega(t') = \mathrm{usbf}_\Omega(t')$, and hence the computation of sbf_Ω at t' can be done efficiently. Therefore, in Lines 1,3 and 5 of the algorithm, $\mathrm{sbf}_\Omega(t')$ is computed for the different possible capacities of Ω (corresponding to different values of the floor function). Then, in Lines 7-11, we determine $t_{max}(= t')$ such that the corresponding capacity Θ_{max} is the largest such capacity that is necessary but not sufficient



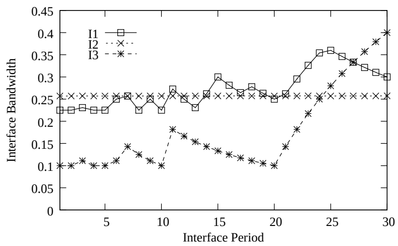
(a) $\Theta_{opt} \mid D_t$ (b) $\Theta_{opt} \nmid D_t$

Figure 4. Figure for Algorithm 1

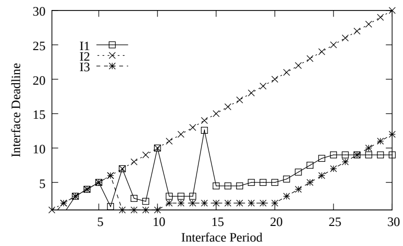
for D_t. In other words, the minimum capacity Θ_{opt} required to satisfy D_t is at least Θ_{max}, and the value of the floor function for Θ_{opt} is the same as its value for Θ_{max}. In Lines 12-13, we compute Θ_{opt} for the two different cases shown in Figure 4 (Θ_{opt} either divides D_t or not). These figures are only for illustration purposes, and give intuition for the two cases under consideration. We do the computations in Lines 12-13 by setting $\mathrm{usbf}_\Omega(t_{max})$ to D_t for the case in Figure 4(a), and by setting $\mathrm{usbf}_\Omega(t_{max} + \Pi)$ to $D_t + t_{max} + \Pi - t$ for the case in Figure 4(b).

Algorithm 1 can be used to compute Θ_m and Δ_m for the abstraction procedures described earlier. Since each invocation of the algorithm runs in constant time, the abstraction procedures have the same execution complexity as the schedulability conditions given in Theorems 2.2 and 2.3. As an aside, since a periodic resource model $\Gamma = (\Pi, \Theta)$ is an EDP model (Π, Θ, Π), these techniques can also be used to generate improved periodic resource model abstractions. This addresses the first concern described in Section 2, for the periodic abstraction framework [20].

Example 1 *Consider components C_1, C_2 and C_3 shown in Figure 1. Let C_1 comprise of the task set* $\{(45, 2, 25), (65, 3, 30), (85, 4, 40)\}$, *$C_2$ comprise of* $\{(35000, 2000, 25000), (55000, 3000, 55000), (75000, 4000, 25000)\}$ *and C_3 comprise of* $\{(45, 1, 45), (75, 2, 20)\}$. *The load for these components* ($\max_t \frac{\mathrm{dbf}(t)}{t}$) *are* $0.225, 0.24$ *and* 0.1, *respectively. We compute EDP interfaces for these components for different values of interface period. These interfaces are plotted in Figure 5 as $I1, I2$ and $I3$ corresponding to components C_1, C_2 and C_3, respectively. We observe that there are period values for which the bandwidth of these interfaces are equal to the corresponding component loads. From these plots we can also see that, as period increases, the bandwidth also gradually increases. Furthermore, from Figure 5(a), it can be seen that the bandwidth has local peaks and troughs. It is then advantageous to choose a period such that the corresponding bandwidth is a trough. Similarly, since the deadline also has local perturbations (Figure 5(b)), it is also desirable to choose a period such that the corresponding deadline is a peak.*



(a) Bandwidth Vs. Period



(b) Deadline Vs. Period

Figure 5. EDP interfaces for components C_1, C_2 and C_3

5. Composition of EDP interfaces

Given an hierarchical scheduling framework as shown in Figure 1, we can generate EDP interfaces for components C_1, C_2 and C_3 using techniques described in previous section. For analysis of the entire system, we must then compose these interfaces iteratively until a single system-level interface is obtained. In this paper, we address this composition problem under EDF and DM schedulers. Our approach to solve this problem is to reduce it to the abstraction problem, by transforming EDP interfaces into sporadic tasks. For the tasks we generate, EDF and DM are optimal dynamic- and static-priority schedulers, respectively [16, 12]. These transformations are also exact, in that they are both necessary and sufficient for schedulability.

5.1. Exact transformation under EDF

We now present a function $\mathcal{T}_{\text{EDF}}$ that takes an EDP interface Ω and returns a sporadic task T. This task is *demand-supply optimal* which can be defined as follows.

Definition 5.1 (Demand-supply optimal under EDF) *A sporadic task T scheduled under* EDF *is* demand-supply optimal *for an EDP interface Ω iff*

- *For any EDP model Ω' that can schedule T under* EDF, $\mathrm{sbf}_{\Omega'}(t) \geq \mathrm{sbf}_{\Omega}(t)$ *for all t, and*
- *For any EDP model Ω' that does not schedule T under* EDF*, there exists a t such that* $\mathrm{sbf}_{\Omega'}(t) < \mathrm{sbf}_{\Omega}(t)$.

Informally, any EDP model that can schedule T must always provide supply at least as much as sbf_{Ω}, and any EDP model that always provides supply greater than sbf_{Ω} must be able to schedule T. Note that the EDF scheduler we mention here is the scheduler at the next level in the hierarchy. For example, to transform the EDP interface of component C_1 in Figure 1, the relevant scheduler is the one for component C_4. Consider a simple transformation that generates the sporadic task $T = (\Pi, \Theta, \Delta)$ for interface $\Omega = (\Pi, \Theta, \Delta)$. An example similar to Figure 2 can be used to show that T is not demand-supply optimal for Ω. Hence, we define $\mathcal{T}_{\text{EDF}}$ as follows.

Definition 5.2 *Given an EDP interface $\Omega = (\Pi, \Theta, \Delta)$, transformation $\mathcal{T}_{\text{EDF}}$ is defined as $\mathcal{T}_{\text{EDF}}(\Omega) = (\Pi, \Theta, \Pi + \Delta - \Theta)$.*

Thus, given multiple EDP interfaces scheduled under EDF, they can be composed by: (1) transforming each interface into a sporadic task using Definition 5.2, and (2) abstracting this set of sporadic tasks into an EDP interface using techniques described in Section 4. We now prove that the transformation $\mathcal{T}_{\text{EDF}}$ is exact.

Theorem 5.3 *Sporadic task generated by $\mathcal{T}_{\text{EDF}}$ given in Definition 5.2, is demand-supply optimal for Ω.*

Proof We first prove the sufficiency condition, i.e., we show for all Ω' that schedules T under EDF, $\mathrm{sbf}_{\Omega'}(t) \geq \mathrm{sbf}_{\Omega}(t)$ for all t. We consider two different types of interval lengths; one where the length is in the range $(n\Pi + \Delta - 2\Theta, n\Pi + \Delta - \Theta)$, and the other where the length is in the range $[n\Pi + \Delta - \Theta, (n+1)\Pi + \Delta - 2\Theta]$, for some $n \geq 1$. Informally, these intervals correspond to the rising and flat portions of sbf_{Ω}, respectively.

Case 1: $t \in (n\Pi + \Delta - 2\Theta, n\Pi + \Delta - \Theta)$

We prove this case by contradiction. Let $\mathrm{sbf}_{\Omega'}(t) < \mathrm{sbf}_{\Omega}(t)$. At $t' = \left\lfloor \frac{t - (\Pi + \Delta - 2\Theta)}{\Pi} \right\rfloor \Pi + (\Pi + \Delta - \Theta)$, we have $\mathrm{sbf}_{\Omega'}(t') \geq \mathrm{dbf}_T(t')$ (Ω' schedules T) and $\mathrm{sbf}_{\Omega}(t') = \mathrm{dbf}_T(t')$. These together imply $\mathrm{sbf}_{\Omega}(t') - \mathrm{sbf}_{\Omega}(t) < \mathrm{sbf}_{\Omega'}(t') - \mathrm{sbf}_{\Omega'}(t)$. This is a contradiction because in the interval $[t, t']$, sbf_{Ω} is always rising with unit slope and no sbf for uniprocessor platforms can rise with a greater slope. Thus, we have shown that for all $t > \Pi + \Delta - 2\Theta$, $\mathrm{sbf}_{\Omega'}(t) \geq \mathrm{sbf}_{\Omega}(t)$. Since $\mathrm{sbf}_{\Omega}(t) = 0$ for all $t \leq \Pi + \Delta - 2\Theta$, we have proved sufficiency.

Case 2: $t \in [n\Pi + \Delta - \Theta, (n+1)\Pi + \Delta - 2\Theta]$

In this case, $\mathrm{dbf}_T(t) = \mathrm{sbf}_{\Omega}(t)$. Since, any Ω' that schedules T must satisfy the condition $\mathrm{sbf}_{\Omega'}(t) \geq \mathrm{dbf}_T(t)$ (Theorem 2.2), the result follows.

The necessary condition follows from the fact that $\forall t > 0, \mathrm{dbf}_T(t) \leq \mathrm{sbf}_{\Omega}(t)$. □

5.2. Exact transformation under DM

In this section, we present a function $\mathcal{T}_{\text{DM}}$ that takes an EDP interface Ω scheduled under DM and returns a demand-supply optimal sporadic task T. The deadline of tasks that we generate must be at most their minimum separation in order for DM to be optimal [12]. Therefore, $\mathcal{T}_{\text{EDF}}$ defined in the previous section is not a good transformation under DM. Hence, we develop another transformation that generates sporadic tasks with deadline at most their minimum separation. This procedure depends on the period of the EDP interface at the next level in the hierarchy. For example in Figure 1, interface transformation for component C_3 depends on the interface period of component C_5. Since interface periods are known a priori, this dependence is not an issue. Let Π' denote this period of the higher level interface. To define $\mathcal{T}_{\text{DM}}$, we first compute a *bandwidth optimal resource supply* $\Omega^* = (\Pi', \Theta^*, \Theta^*)$ for interface $\Omega = (\Pi, \Theta, \Delta)$. Ω^* is such that,

$$\forall t > 0, \text{sbf}_{\Omega^*}(t) \geq \text{sbf}_{\Omega}(t) \qquad \text{(sufficiency)}$$

$$(\exists t > 0, \text{sbf}_{\Omega^*}(t) = \text{sbf}_{\Omega}(t)) \vee \left(\frac{\Theta^*}{\Pi'} = \frac{\Theta}{\Pi}\right) \qquad \text{(necessity)}$$

These conditions imply that any EDP model with period Π', must have capacity at least Θ^* to provide supply as much as Ω (Theorem 3.2). We use Ω^* to generate the demand-supply optimal sporadic task T corresponding to Ω. This notion of optimality for T is similar to the EDF case, except that it depends on Π'.

Definition 5.4 (Demand-supply optimal under DM) *A sporadic task T scheduled under* DM *is demand-supply optimal for an EDP interface Ω iff*

- *For any EDP model Ω' with period Π' that can schedule T under* DM*, it must be the case that $\forall t, \text{sbf}_{\Omega'}(t) \geq \text{sbf}_{\Omega}(t)$, and*
- *For any EDP model Ω' with period Π' that does not schedule T under* DM*, there exists a t such that $\text{sbf}_{\Omega'}(t) < \text{sbf}_{\Omega}(t)$.*

Here, Π' denotes the interface period at the next level from Ω in the hierarchical system.

In the rest of this section, we define capacity Θ^* corresponding to Ω^*, and then present $\mathcal{T}_{\text{DM}}$ using Θ^*.

Theorem 5.5 *Given EDP interface $\Omega = (\Pi, \Theta, \Delta)$ and period Π', the bandwidth optimal resource supply $\Omega^* = (\Pi', \Theta^*, \Theta^*)$ is such that,*

$$\Theta^* = \begin{cases} \Theta + \frac{k\Theta}{\Delta} & \Pi' = \Pi + k, k \geq 0 \\ \frac{\Theta}{k} & \Pi' = \frac{\Pi}{k}, k \geq 2 \\ \Theta_C & \Pi' = \frac{\Pi}{k} + \beta, k \geq 2 \end{cases}$$

where $\beta \in \left(0, \frac{\Pi}{k(k-1)}\right)$ and k is an integer. If there exists $n \in \left(\frac{\Delta-\Theta}{k\beta}, \frac{\Delta-\Theta}{k\beta} + \Pi' + 1\right)$ such that $nk\beta - (\Delta - \Theta) = l\Pi' + \gamma$ and n is the smallest integer satisfying $\gamma \geq \frac{n\Theta+\gamma}{nk-l}$, then $\Theta_C = \frac{n\Theta}{nk-(l+1)}$. Otherwise, $\Theta_C = \frac{\Theta}{k} + \frac{\beta\Theta}{\Pi}$.

Proof We consider the cases $\Pi' \geq \Pi$ and $\Pi' < \Pi$ separately.

Case 1: $\Pi' = \Pi + k, k \geq 0$
Consider the sbf of model $\Omega' = (\Pi', \Theta', \Theta')$ shown in Figure 6(a), where $\Theta' = \Theta + k$. Then, $\text{sbf}_{\Omega'}(\Pi) = \Theta$, $\Pi' - \Theta' = \Pi - \Theta$ and $\frac{\Theta'}{\Pi'} \geq \frac{\Theta}{\Pi}$, and Ω' is a bandwidth optimal resource supply for model (Π, Θ, Θ). Furthermore, Ω' also satisfies the sufficiency condition (of bandwidth optimal resource supply) for $\Omega = (\Pi, \Theta, \Delta)$. Hence, $\Theta^* \leq \Theta + k$.

We now compute a non-negative quantity α such that $\Theta^* = \Theta' - \alpha$. Consider the EDP model $\Omega^* = (\Pi', \Theta' - \alpha, \Theta' - \alpha)$ as shown in the figure. Let $t_1 = \Pi' \left\lfloor \frac{n\Theta}{\Theta^*} \right\rfloor + \Pi' - \Theta^*$ and $t_2 = \Pi' \left\lceil \frac{n\Theta}{\Theta^*} \right\rceil + \Pi' - \Theta^*$ be any two interval lengths for which $n \in \{\mathbb{N} \bigcup \{0\}\}$ is unique (illustrated in Figure 6(a) for $n = 2$). Informally, between t_1 and t_2, sbf_{Ω} has exactly one flat segment. Now since $\Pi' - \Theta^* \geq \Pi - \Theta$, $t_2 - t_1 \geq \Pi - \Theta$. Then, it can never be the case that $\text{sbf}_{\Omega^*}(t_1) = \text{sbf}_{\Omega}(t_1)$ and sbf_{Ω^*} at t_1 touches a rising segment of sbf_{Ω} (if this happens then $\text{sbf}_{\Omega^*}(t_2) < \text{sbf}_{\Omega}(t_2)$). Likewise, it can never be the case that $\text{sbf}_{\Omega^*}(t_2) = \text{sbf}_{\Omega}(t_2)$ and sbf_{Ω^*} at t_2 touches a flat segment of sbf_{Ω} (if this happens then $\text{sbf}_{\Omega^*}(t_1) < \text{sbf}_{\Omega}(t_1)$). This argument can be applied to all such pairs of t_1 and t_2. Hence, if $\text{sbf}_{\Omega^*}(t) = \text{sbf}_{\Omega}(t)$ with sbf_{Ω^*} touching a rising segment of sbf_{Ω}, then $\left\lfloor \frac{n\Theta}{\Theta^*} \right\rfloor \Pi' + \Pi' - \Theta^* = \left\lfloor \frac{m\Theta}{\Theta^*} \right\rfloor \Pi' + \Pi' - \Theta^* (= t)$ for some $m \neq n$, where $m, n \in \{\mathbb{N} \bigcup \{0\}\}$. Likewise, if $\text{sbf}_{\Omega^*}(t) = \text{sbf}_{\Omega}(t)$ with sbf_{Ω^*} touching a flat segment of sbf_{Ω}, then $\left\lfloor \frac{n\Theta}{\Theta^*} \right\rfloor \Pi' + 2\Pi' - \Theta^* = \left\lfloor \frac{m\Theta}{\Theta^*} \right\rfloor \Pi' + 2\Pi' - \Theta^* (= t)$ for some $m \neq n$, where $m, n \in \{\mathbb{N} \bigcup \{0\}\}$. Let l_t denote the index (starting from 0) of t in an increasing sequence of all such relevant interval lengths. Then the following conditions hold for each t.

$$\left(\left\lfloor \frac{t}{\Pi'} \right\rfloor + 1\right) \Theta^* \geq \left(\left\lfloor \frac{t}{\Pi'} \right\rfloor + 2 + l_t\right) \Theta$$

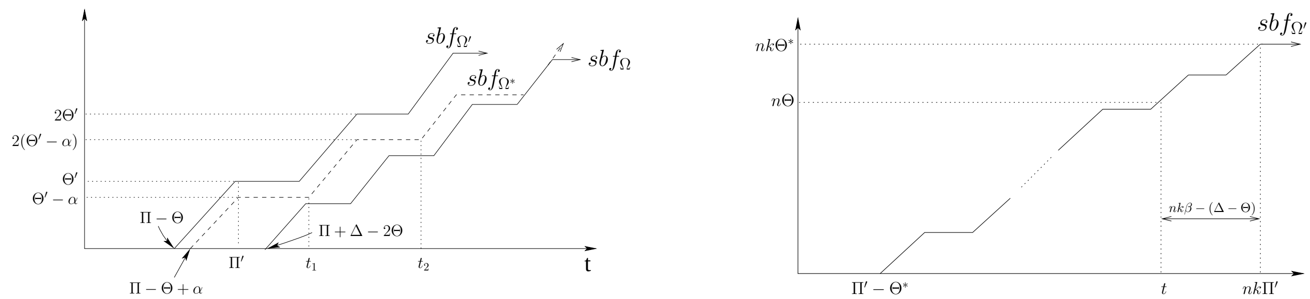
$$l_t(\Pi - \Theta) + (\Delta - \Theta) \geq \left(\left\lfloor \frac{t}{\Pi'} \right\rfloor + 1\right) \alpha$$

Informally, first inequality represents the vertical gap between $\text{sbf}_{\Omega^*}(t + \Pi')$ and the preceding horizontal segment of sbf_{Ω} (case 1 above), and second inequality represents the horizontal gap at $\text{sbf}_{\Omega^*}(t)$ between t and the succeeding rising segment of sbf_{Ω} (case 2 above). Combining these conditions we get,

$$\alpha \leq k \left(1 - \frac{(l_t + 1)\Theta}{l_t \Pi + \Delta}\right)$$

Since this condition must hold for all such t,

$$\alpha \leq \min_{l_t} \left\{ k \left(1 - \frac{(l_t + 1)\Theta}{l_t \Pi + \Delta}\right) \right\}$$



(a) Figure for Case 1

(b) Figure for Case 2

Figure 6. Figures for Theorem 5.5

Observe that the RHS here is minimized when $l_t = 0$ because $\Theta \le \Delta \le \Pi$. Therefore,

$$\Theta^* = \Theta' - k\left(1 - \frac{\Theta}{\Delta}\right) = \Theta + \frac{k\Theta}{\Delta}$$

Case 2: $\Pi' = \frac{\Pi}{k} + \beta, k \ge 2, \beta \in \left[0, \frac{\Pi}{k(k-1)}\right)$
Since Ω^* must be a bandwidth optimal resource supply for Ω, $\frac{\Theta^*}{\Pi'} \ge \frac{\Theta}{\Pi}$ (definition). Therefore, $\Theta^* \ge \frac{\Theta}{k} + \frac{\beta\Theta}{\Pi}$. Also, $\forall n \ge 1$, s.t. $t = n\Pi + \Delta - \Theta$, it must hold that $\mathrm{sbf}_{\Omega^*}(t) \ge n\Theta$. If this condition is satisfied then the sufficiency criteria of bandwidth optimal resource supply is met. Furthermore, if this condition is tight for some n then we also get necessity. We use this condition to compute Θ^*.

If $\Theta^* \ge \frac{\Theta}{k}$, then $\mathrm{sbf}_{\Omega^*}(nk\Pi') = nk\Theta^* \ge n\Theta$. Since $n\Pi + \Delta - \Theta = nk\Pi' - (nk\beta - (\Delta - \Theta)), \forall n$ s.t. $nk\beta - (\Delta - \Theta) \le 0$ we get $\mathrm{sbf}_{\Omega^*}(t) \ge n\Theta$. Therefore, if $n \le \frac{\Delta-\Theta}{k\beta}$ or $\beta = 0$, $\Theta^* \ge \frac{\Theta}{k}$ is sufficient.

We now consider the case when $n > \frac{\Delta-\Theta}{k\beta}$ and $\beta \ne 0$. Here t is smaller than $nk\Pi'$ by $nk\beta - (\Delta - \Theta)$, and we require that $\mathrm{sbf}_{\Omega^*}(t) \ge n\Theta$. In other words, the total resource supply of Ω^* in an interval of length $nk\beta - (\Delta - \Theta)$ immediately preceding $nk\Pi'$ must be at most $nk\Theta^* - n\Theta$. This interval of interest is shown in Figure 6(b). Assuming $nk\beta - (\Delta - \Theta) = l\Pi' + \gamma$ we then require,

$$\forall n, nk\Theta^* - n\Theta \ge l\Theta^* + \min\{\Theta^*, \gamma\}$$
$$\Rightarrow \Theta^* \ge \max_n \left\{\min\left\{\frac{n\Theta}{nk - (l+1)}, \frac{n\Theta + \gamma}{nk - l}\right\}\right\}$$

Consider a n for which $\gamma < \frac{n\Theta+\gamma}{nk-l} (= \Theta^*)$. Since $\Theta^* \le \Theta$ we then get $\frac{\Pi'\Theta}{\Pi} \ge \frac{n\Theta+\gamma}{nk-l}$. Hence, we can ignore the term $\frac{n\Theta+\gamma}{nk-l}$ while computing Θ^*. Then, $\Theta^* = \frac{n\Theta}{nk-(l+1)}$ for the smallest n which satisfies $\gamma \ge \frac{n\Theta+\gamma}{nk-l}$ ($\frac{n\Theta}{nk-(l+1)}$ decreases with increasing n). This n if it exists, lies in $\left(\frac{\Delta-\Theta}{k\beta}, \frac{\Delta-\Theta}{k\beta} + \Pi' + 1\right)$, because γ is the remainder of $nk\beta - (\Delta - \Theta)$ divided by Π'. □

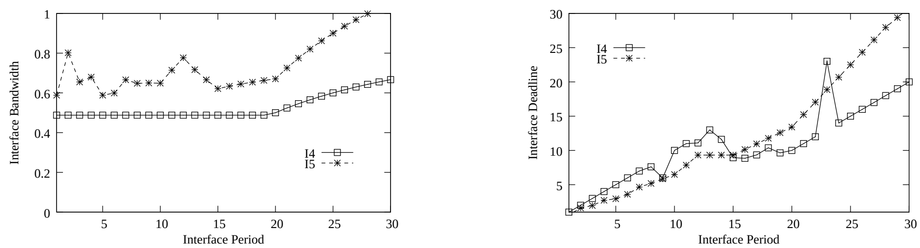
Transformation $\mathcal{T}_{\mathrm{DM}}$ can then be defined as follows.

Definition 5.6 *Given an EDP interface* $\Omega = (\Pi, \Theta, \Delta)$ *and period* Π'*, transformation* $\mathcal{T}_{\mathrm{DM}}$ *is defined as* $\mathcal{T}_{\mathrm{DM}}(\Omega, \Pi') = (\Pi', \Theta^*, \Pi')$.

$\mathcal{T}_{\mathrm{DM}}$ generates sporadic tasks with the same minimum separation and deadline for all the component interfaces at any one level in the hierarchical system. For example in Figure 1, the tasks generated for interfaces of components C_3 and C_4 have the same minimum separation and deadline (capacities may be different). Thus, given multiple EDP interfaces scheduled under DM, they can composed by: (1) transforming each interface into a sporadic task using Theorem 5.5 and Definition 5.6, and (2) abstracting this set of sporadic tasks into an EDP interface using techniques described in Section 4. The following corollary of Theorem 5.5 states that $\mathcal{T}_{\mathrm{DM}}$ is exact.

Corollary 5.7 *Transformation* $\mathcal{T}_{\mathrm{DM}}$ *as given in Definition 5.6 generates a demand-supply optimal sporadic task under* DM *for* Ω.

Example 2 *We now demonstrate our composition techniques on components* C_4 *and* C_5 *in Figure 1. Let the periods of chosen EDP interfaces* $I1$, $I2$ *and* $I3$ *be* 13, 27 *and* 20, *respectively, i.e., let* $I1 = (13, 3, 3), I2 = (27, 6.95, 27)$ *and* $I3 = (20, 2, 2)$ *(from Figure 5). Now,* $I1$ *and* $I2$ *are scheduled under* EDF *(*C_4 *comprises of* C_1 *and* C_2*), and hence the tasks for* C_4 *using Definition 5.2 are* $(13, 3, 13)$ *and* $(27, 6.95, 47.05)$. *Interface* $I4$ *for component* C_4 *is then plotted in Figure 7. We choose a period of* 15 *for* $I4$, *i.e., let* $I4 = (15, 7.1445, 8.9446)$. *For different values of the interface period of component* C_5*, we first transform* $I3$ *and* $I4$ *into sporadic tasks (using Definition 5.6), and then generate interface* $I5$. *Figure 7 also shows the plot for* $I5$ *over different period values. Observe that the total bandwidth of* $I1$ *(period*= 13*) and* $I2$ *(period*= 27*) is* $\frac{3}{13} + \frac{6.95}{27} = 0.4881$, *and the bandwidth of* $I4$ *for period values up to* 19 *is also* 0.4881. *This indicates that our composition and abstraction techniques do not incur any bandwidth overhead for these period values. Similarly, the total*



(a) Bandwidth Vs. Period (b) Deadline Vs. Period

Figure 7. EDP interfaces for components C_4 and C_5

bandwidth of $I3$ *(period* $= 20$*) and* $I4$ *(period* $= 15$*) is* $\frac{2}{20} + \frac{7.1445}{15} = 0.5763$*, and the bandwidth of* $I5$ *for certain periods* $(1, 5$*, etc.) is the same.*

6. Priority preserving EDP interfaces

Analysis techniques described in previous sections, automatically assign deadlines to EDP interfaces. When components are scheduled under DM, this means that their priorities are also automatically fixed. However, it may be desirable that system designers be able to control component priorities. Hence in this section, we describe an analysis technique that respects the designer specified component priorities. We assume that components are scheduled under DM, and therefore present a technique that assigns appropriate deadlines to interfaces and their transformations. We first give the procedure to compute a bandwidth optimal EDP interface $\Omega = (\Pi, \Theta, \Delta)$ for a component C whose workload is comprised of only sporadic tasks. Note that we only generate a bandwidth optimal interface as opposed to the bandwidth-deadline optimal interface for the general case. We then modify $\mathcal{T}_{\text{DM}}$ to generate demand-supply optimal sporadic tasks that respect component priorities.

Algorithm 2 gives the priority preserving abstraction technique. This algorithm assumes that component C_i has higher priority than component C_j for all $i < j$. In Lines 1 and 2 of the algorithm we compute the bandwidth optimal and bandwidth-deadline optimal EDP interfaces, respectively, for each component C_j. These interfaces do not necessarily preserve the component priorities. Therefore, in Lines 4-13 of the algorithm, we modify their deadlines and capacities such that (1) component priorities are preserved, and (2) interface bandwidth is minimized. Capacity Θ_j of interface Ω_j is only increased if its maximum deadline Δ'_j corresponding to Θ_j is smaller than the deadline of Ω_{j-1}. In this case, the deadline of Ω_j is set equal to the deadline of Ω_{j-1} and the corresponding capacity is computed (Lines 10-11). After this process, all interfaces respect the designer specified priorities. In Lines 14-16 of the algorithm, interface deadlines are increased as much as possible without violating priorities or increasing capacities. The following theorem proves that interfaces generated using Algorithm 2 are bandwidth optimal.

Algorithm 2 Priority preserving EDP interface

Input: Components $C_1, \ldots, C_n$ and periods $\Pi_1, \ldots, \Pi_n$
Output: $\Omega_j = (\Pi_j, \Theta_j, \Delta_j)$ for $j = 1, \ldots, n$

1: For each C_j execute Step 1 of abstraction technique for DM and let $(\Pi_j, \Theta_j, \Delta_j)$ be the output.
2: For each C_j execute Step 2 of abstraction technique for DM and let $(\Pi_j, \Theta_j, \Delta'_j)$ be the output.
// Refer to abstraction technique in Section 4 for the previous steps.
3: Let $\Delta = \Theta_1$
4: **for** $j = 2$ to n **do**
5:   **if** $\Theta_j \geq \Delta$ **then**
6:     $\Delta = \Theta_j$
7:   **else if** $\Delta'_j \geq \Delta$ **then**
8:     $\Delta_j = \Delta$
9:   **else**
10:     Compute Θ s.t.(Π_j, Θ, Δ) exactly schedules C_j.
11:     Let $\Theta_j = \Theta$ and $\Delta_j = \Delta$
12:   **end if**
13: **end for**
14: **for** $j = n - 1$ to 1 **do**
15:   $\Delta_j = \max\{\Delta_j, \min\{\Delta_{j+1}, \Delta'_j\}\}$
16: **end for**

Theorem 6.1 *Given that component* C_i *has higher priority than component* C_j *for all* $i < j$*, interface* Ω_j *generated by Algorithm 2 is bandwidth optimal for* C_j*.*

Proof If capacity Θ_j of interface Ω_j is not modified in Line 11 of the algorithm, then from Theorem 3.2 we get Ω_j is bandwidth optimal for C_j. However, if Θ_j is modified then the deadline of Ω_j is set to the deadline of Ω_{j-1}. Since component C_{j-1} is required to have a higher priority than component C_j, this is the minimum deadline for Ω_j (we assume that in case of a tie, priorities are determined by indices). Note that the deadline of an interface is only increased to respect priorities, and hence the deadlines of $\Omega_1, \ldots, \Omega_{j-1}$ are the least possible. From Theorem 3.2 we then get that Ω_j is bandwidth optimal for C_j under the given component priorities. $\Box$

Given an EDP interface $\Omega = (\Pi, \Theta, \Delta)$ and period Π', $\mathcal{T}_{\text{DM}}$ (Definition 5.6) generates a sporadic task whose deadline is Π' for each component. Hence, this transformation does not preserve the designer specified priorities. Assuming components are abstracted using Algorithm 2, we modify $\mathcal{T}_{\text{DM}}$ as follows. In order to respect component priorities, we use the deadline of underlying interfaces to generate sporadic tasks. It is then easy to see that Corollary 5.7 also holds for $\mathcal{T}_{\text{DM}}$ defined below.

Definition 6.2 *Let $\Omega^* = (\Pi', \Theta^*, \Theta^*)$ denote a bandwidth optimal resource supply for interface $\Omega = (\Pi, \Theta, \Delta)$. Then, transformation $\mathcal{T}_{\text{DM}}$ is defined as $\mathcal{T}_{\text{DM}}(\Omega, \Pi') = (\Delta, \text{sbf}_{\Omega^*}(\Delta), \Delta)$.*

7. Conclusion

We proposed a compositional analysis framework based on the explicit deadline periodic resource model. We presented exact schedulability conditions for this model under EDF and DM schedulers, and gave an efficient algorithm to generate bandwidth-deadline optimal interfaces based on this model. We also proposed transformations of these interfaces to sporadic tasks for compositional analysis, and showed that they are demand-supply optimal. In the future, we plan to extend this framework to support incremental analysis of dynamically changing components, as well as for compositional analysis of systems scheduled on multi-core platforms. We also plan to perform a comparative study of EDP models with other resource models.

References

[1] L. Almeida and P. Pedreiras. Scheduling within temporal partitions: response-time analysis and server design. In *Proceedings of ACM International Conference on Embedded Software*, September 2004.

[2] S. Baruah, R. Howell, and L. Rosier. Algorithms and complexity concerning the preemptive scheduling of periodic, real-time tasks on one processor. *Journal of Real-Time Systems*, 2:301–324, 1990.

[3] R. I. Davis and A. Burns. Hierarchical fixed priority preemptive scheduling. In *Proceedings of IEEE Real-Time Systems Symposium*, December 2005.

[4] R. I. Davis and A. Burns. Resource sharing in hierarchical fixed priority pre-emptive systems. In *Proceedings of IEEE Real-Time Systems Symposium*, December 2006.

[5] Z. Deng and J. W.-S. Liu. Scheduling real-time applications in an open environment. In *Proceedings of IEEE Real-Time Systems Symposium*, December 1997.

[6] A. Easwaran, I. Shin, O. Sokolsky, and I. Lee. Incremental schedulability analysis of hierarchical real-time components. In *Proceedings of ACM International Conference on Embedded Software*, October 2006.

[7] Engineering Standards for Avionics and Cabin Systems (AEEC). *ARINC specification 653-2, part I*, 2006.

[8] X. Feng and A. Mok. A model of hierarchical real-time virtual resources. In *Proceedings of IEEE Real-Time Systems Symposium*, December 2002.

[9] T. A. Henzinger and S. Matic. An interface algebra for real-time components. In *Proceedings of IEEE Real-Time Technology and Applications Symposium*, April 2006.

[10] T.-W. Kuo and C. Li. A fixed-priority-driven open environment for real-time applications. In *Proceedings of IEEE Real-Time Systems Symposium*, December 1999.

[11] J. Lehoczky, L. Sha, and Y. Ding. The rate monotonic scheduling algorithm: exact characterization and average case behavior. In *Proceedings of IEEE Real-Time Systems Symposium*, 1989.

[12] J. Leung and J. Whitehead. On the complexity of fixed-priority scheduling of periodic real-time tasks. *Performance Evaluation*, 2:37–250, 1982.

[13] G. Lipari and S. Baruah. Efficient scheduling of real-time multi-task applications in dynamic systems. In *Proceedings of IEEE Real-Time Technology and Applications Symposium*, May 2000.

[14] G. Lipari and E. Bini. Resource partitioning among real-time applications. In *Proceedings of Euromicro Conference on Real-Time Systems*, July 2003.

[15] G. Lipari, J. Carpenter, and S. Baruah. A framework for achieving inter-application isolation in multiprogrammed hard-real-time environments. In *Proceedings of IEEE Real-Time Systems Symposium*, December 2000.

[16] C. Liu and J. Layland. Scheduling algorithms for multi-programming in a hard-real-time environment. *Journal of the ACM*, 20(1):46–61, 1973.

[17] S. Matic and T. A. Henzinger. Trading end-to-end latency for composability. In *Proceedings of IEEE Real-Time Systems Symposium*, December 2005.

[18] A. Mok, X. Feng, and D. Chen. Resource partition for real-time systems. In *Proceedings of IEEE Real-Time Technology and Applications Symposium*, May 2001.

[19] S. Saewong, R. Rajkumar, J. Lehoczky, and M. Klein. Analysis of hierarchical fixed-priority scheduling. In *Proceedings of Euromicro Conference on Real-Time Systems*, June 2002.

[20] I. Shin and I. Lee. Periodic resource model for compositional real-time guarantees. In *Proceedings of IEEE Real-Time Systems Symposium*, December 2003.

[21] I. Shin and I. Lee. Compositional real-time scheduling framework. In *Proceedings of IEEE Real-Time Systems Symposium*, December 2004.

[22] L. Thiele, E. Wandeler, and N. Stoimenov. Real-time interfaces for composing real-time systems. In *Proceedings of ACM International Conference on Embedded Software*, October 2006.

[23] E. Wandeler and L. Thiele. Interface-based design of real-time systems with hierarchical scheduling. In *Proceedings of IEEE Real-Time Technology and Applications Symposium*, April 2006.