An FPTAS for Interface Selection in the Periodic Resource Model

Nathan Fisher
Department of Computer Science
Wayne State University
fishern@cs.wayne.edu

Abstract

The periodic resource model of Shin and Lee [18] provides a flexible, simple framework for designing compositional real-time systems. Each component in the periodic resource model has an interface which specifies the period and capacity of the resource used to schedule the component. Unfortunately, the best-known exact algorithms for determining the interface parameters for a component in the periodic resource model potentially require exponential or pseudo-polynomial time. In this paper, we obtain an FPTAS for the problem of selecting an interface period given a capacity-determination algorithm. We also apply our approach to obtain an FPTAS for the problem of selecting both a period and capacity for components consisting of sporadic task systems. Our algorithms obtain interface parameters with interface bandwidth at most \((1 + \epsilon)\) times the optimal for any \(\epsilon > 0\).

1 Introduction

Compositional analysis for real-time systems has recently received considerable research attention due to its well-known benefits of reducing system-design complexity. For component-based systems, reduction in system complexity is typically achieved via component abstraction. Component abstraction hides the complexity and internal details of each component from developers of other components and only exposes information necessary to use the component via an interface. Numerous compositional frameworks have been proposed to support the design and analysis of compositional real-time systems (for a non-exhaustive list see [1, 7, 10, 18]). In each of these compositional frameworks, a component expresses its computational requirements to the system via a real-time interface. One important attribute of a real-time interface is the interface bandwidth. The interface bandwidth simultaneously quantifies the fraction of the total system resource supply that a component \(C\) will require to meet its real-time constraints and the component \(C\)’s “interference” on the resource supply provided to other system components. Thus, an important fundamental problem in design and analysis of compositional real-time systems is the minimization of real-time interface bandwidth (MIB-RT).

One simple, yet flexible, real-time compositional framework is the periodic resource model [18]. A periodic resource, denoted by \(\Gamma = (\Pi, \Theta)\), guarantees that a component \(C\) executed upon resource \(\Gamma\) will receive at least \(\Theta\) units of execution (not necessarily contiguous) between successive time points \(\{t \equiv t_0 + \ell \Pi \mid \ell \in \mathbb{N}\}\), given some initial resource start-time \(t_0\). The parameters \(\Pi\) and \(\Theta\) are respectively referred to as the period and capacity of the periodic resource \(\Gamma\). The ratio \(\frac{\Theta}{\Pi}\) represents the interface bandwidth of resource \(\Gamma\). A system-level scheduling algorithm allocates the processor time among the different periodic resources that share the same processor, such that each resource receives (for every period) aggregate processor time equivalent to its capacity. A subsystem’s tasks are then hierarchically scheduled by a subsystem-level scheduling algorithm upon the processing time supplied to resource \(\Gamma\). If the system-level scheduling algorithm is earliest-deadline-first (EDF), then it is known (e.g., see [18]) that periodic resources \(\{\Gamma_1, \Gamma_2, \ldots, \Gamma_m\}\) can successfully guarantee the
capacity parameters to their respective components, if and only if, the total system bandwidth does not exceed one; i.e., \( \sum_{i=1}^{m} \frac{1}{\Pi_i} \leq 1 \). From the aforementioned condition, it is clear that minimizing the interface bandwidth of each resource increases system schedulability.

In this paper, we specifically consider the problem of determining the optimal choice of period parameter (i.e., \( \Pi \)) for resource \( \Gamma \), given a component \( C \) and a capacity-determination algorithm \( A \). A capacity-determination algorithm returns the minimum-schedulable capacity for a given component \( C \) to meet all deadlines upon a periodic resource with a fixed period \( \Pi \). Such algorithms have already been devised for sporadic tasks; e.g., see [9, 12, 18]. Let \( \Theta_{\min}^A(\Pi, C) \) be the value returned by capacity-determination algorithm \( A \) for a given \( C \) and \( \Pi \). Furthermore, let us consider only integer values of \( \Pi \) in the range \( \{\Pi_{\text{lower}}, \ldots, \Pi_{\text{upper}}\} \) as possible periods. The optimal choice period for the MIB-RT problem is:

\[
\Pi^*(A, C) \overset{\text{def}}{=} \arg\min_{\Pi \in \mathbb{N}^+} \left\{ \Theta_{\min}^A(\Pi, C) \mid \Pi_{\text{lower}} \leq \Pi \leq \Pi_{\text{upper}} \right\}
\]

Let \( \text{OPT} \) denote the optimal capacity-determination algorithm. The problem of exactly determining \( \Pi^*(\text{OPT}, C) \) for the period resource model (and slightly more general models) has been extensively studied by Easwaran [8]. However, each of the proposed methods has runtime dependent on the difference between \( \Pi_{\text{lower}} \) and \( \Pi_{\text{upper}} \). If this difference is large, determining \( \Pi^*(A, C) \) may be prohibitively expensive, especially if the capacity-determination algorithm \( A \) also has significant computational complexity.

§Our Contributions. Our main objective is to find an approximation to \( \Pi^*(A, C) \) with bounded deviation from the optimal solution to MIB-RT. In this paper, we present an approximation scheme with the following guarantee:

Given a component \( C \), capacity-determination algorithm \( A \), range of possible period values \( \{\Pi_{\text{lower}}, \ldots, \Pi_{\text{upper}}\} \), and an accuracy parameter \( \epsilon > 0 \), if our algorithm returns \( \hat{\Pi} \) for the given parameters,

\[ \frac{\Theta_{\min}^A(\Pi^*(A, C), C)}{\Pi^*(A, C)} \leq \frac{\Theta_{\min}^A(\hat{\Pi}, C)}{\hat{\Pi}} \leq (1 + \epsilon) \cdot \frac{\Theta_{\min}^A(\Pi^*(A, C), C)}{\Pi^*(A, C)}. \]

Furthermore, our algorithm runs in time polynomial in the representation of \( C \), \( \Pi_{\text{lower}}, \Pi_{\text{upper}}, \frac{1}{\epsilon} \), and the complexity of \( A \).

In other words, our algorithm is a fully-polynomial-time approximation scheme (FPTAS) [19] for the MIB-RT problem (with respect to the computation complexity of \( A \)). The \((1 + \epsilon)\) factor is called the approximation ratio of the produced solution. We also give an FPTAS for determining both the period and capacity of a component consisting of sporadic tasks.

§Organization. The remainder of the paper is organized as follows. Section 2 briefly describes prior related research on the MIB-RT problem. Section 3 presents our proposed approximation scheme for period selection (with respect to any given capacity-determination algorithm) and proves its associated approximation ratio. Section 4 shows how the approximation scheme given in the Section 3 may be used to obtain an overall approximation scheme (with respect to the optimal capacity-determination) for components consisting of sporadic real-time tasks.

2 Related Work

The MIB-RT problem has previously been studied for the periodic resource model and the explicit-deadline periodic resource (EDP) model [9] where each component is represented by a sporadic task system [15]. Easwaran et al. [9] also obtain exact solutions to MIB-RT in this context (i.e., if the bandwidth provided by the system to component \( C \) is less than the exact solution to MIB-RT, then some real-time constraint will be violated for \( C \)). This solution is based upon exact schedulability techniques for uniprocessor real-time systems [4, 13], which may be computationally expensive. Easwaran [8] also explored period selection in the EDP model. Shin and Lee [18] have obtained \( O(n) \)-time, sufficient solutions to MIB-RT for the periodic resource. The advantage of this approach is that bandwidth allocation may be determined quickly for a component \( C \). However, our previous analysis [11] of Shin and Lee’s linear-time algorithm.
showed that, for a fixed resource period, there exists sporadic task systems that would cause the algorithm to return a bandwidth that is a factor of 1.5 greater than the optimal bandwidth. (It is also shown that the most that the returned bandwidth could exceed optimal is by a factor of 3). As an intermediate solution between computationally-expensive, exact solutions and efficient, inexact solutions, Fisher and Dewan [12] obtain an FPTAS for capacity-determination in the periodic resource model with fixed resource periods. In this paper, we remove the assumption of fixed resource periods.

A number of other results on MIB-RT for different compositional models exist. For components scheduled by fixed-priority on temporal partitions, Lipari and Bini [14], developed an exact, pseudo-polynomial-time algorithm for MIB-RT. While Almeida and Pedreiras [3] developed sufficient, polynomial-time bandwidth allocation techniques for fixed-priority scheduling upon temporal partitions. Recently, researchers have also focused on characterizing components by processor-demand curves which describe the minimum amount of processing that a component requires over any interval. For example, Wandeler and Thiele [20] proposed the concept of interface-based design which uses real-time calculus [6] to compute demand curves and service curves for each component in a compositional real-time system. In another demand-based model, Albers et al. [1] have developed parametric algorithms for MIB-RT (without known approximation ratios) for the hierarchical event stream model.

3 An Algorithm for Selecting the Interface Period

In this section, we describe an algorithm for selecting a “near-optimal” interface period for a component \( C \), with respect to a given capacity-determination algorithm \( A \). However, to guarantee that the approximation ratio of our proposed algorithm is not too large, we need to formally restrict the types of capacity-determination algorithms that we consider. Informally, we will only consider algorithms where the capacity grows with increasing periods. The formal definition is given in the following.

**Definition 1** For any component \( C \), an algorithm \( A \) is a monotonically non-decreasing capacity-determination algorithm over \( \{\Pi_{\text{lower}}, \ldots, \Pi_{\text{upper}}\} \), if for all \( \Pi_1, \Pi_2 \in \{\Pi_{\text{lower}}, \ldots, \Pi_{\text{upper}}\} \) such that \( \Pi_1 \leq \Pi_2 \), it must be that \( \Theta^A_{\text{min}}(\Pi_1, C) \leq \Theta^A_{\text{min}}(\Pi_2, C) \).

Note that this definition does not place a constraint on the types of problems that can be solved by our period-selection algorithm, as all known capacity-determination algorithms [9, 12, 18] for the periodic resource model possess this property. Section 4 will formally show that the algorithm of [12] is monotonically non-decreasing in \( \Pi \). In the remainder of this section, we will briefly describe our algorithm (Section 3.1) and prove its correctness (Section 3.2).

3.1 Algorithm Description

The period-selection algorithm, \( \text{SelectInterface}(C, \mathcal{A}, \Pi_{\text{lower}}, \Pi_{\text{upper}}, \epsilon) \), is simple; the algorithm evaluates \( \Theta^A_{\text{min}}(\Pi, C) \) for select values of \( \Pi \) between \( \Pi_{\text{lower}} \) and \( \Pi_{\text{upper}} \) and returns the \( \Pi \) and \( \Theta^A_{\text{min}}(\Pi, C) \) with the minimum \( \Theta^A_{\text{min}}(\Pi, C) \). The values of \( \Pi \) are selected based on an accuracy parameter \( \epsilon \). Pseudocode for \( \text{SelectInterface}(C, \mathcal{A}, \Pi_{\text{lower}}, \Pi_{\text{upper}}, \epsilon) \) is given below.

Lines 1 through 5 initialize the first choice for \( \Pi \) (equal to \( \Pi_{\text{lower}} \)) and other variables. The while loop of Lines 6 and 21 iterate through successive choices of \( \Pi \) that have capacity ratios (according to \( \mathcal{A} \)) of at most \((1 + \epsilon)\). To find the next choice of \( \Pi \), we use a binary search (Line 7) over the range of remaining values of \( \Pi \) that have not been selected. The binary search returns the largest \( \Pi \) such that the capacity of \( \Pi \) is no more than \((1 + \epsilon)\) times the capacity of the previously-selected value of \( \Pi \). \( \Pi \) is incremented (Line 8) within the while loop to ensure then next capacity-value for \( \Pi \) is more than \((1 + \epsilon)\) times the previous. Finally, the while loop terminates when our returned capacity exceeds the capacity of \( \Pi_{\text{upper}} \). The values \( \Theta \) and \( \Pi \) maintain the interface parameters of the resource with the minimum capacity of all values of \( \Pi \) that have been evaluated. The algorithm returns the minimum-bandwidth interface over all evaluated choices of \( \Pi \).
Lemma 1 Consider any \( \Pi_i, \Pi_j \in \{ \Pi_{\text{lower}}, \ldots, \Pi_{\text{upper}} \} \) where \( \Pi_i \leq \Pi_j \). Given any monotonically non-decreasing capacity-allocation algorithm \( \mathcal{A} \) and component \( C \), the following is true.

\[
\frac{\Theta_{\text{min}}^\mathcal{A}(\Pi_i, C)}{\Pi_j} \leq \min_{\Pi \in \{ \Pi_{\text{lower}}, \ldots, \Pi_{\text{upper}} \}} \left\{ \frac{\Theta_{\text{min}}^\mathcal{A}(\Pi, C)}{\Pi} \right\}. 
\]

(2)

Proof: Observe that, since \( \mathcal{A} \) is monotonically non-decreasing over \( \{ \Pi_i, \ldots, \Pi_j \} \), \( \Theta_{\text{min}}^\mathcal{A}(\Pi_i, C) \leq \Theta_{\text{min}}^\mathcal{A}(\Pi, C) \) for all \( \Pi \in \{ \Pi_i, \ldots, \Pi_j \} \). Also, \( \frac{1}{\Pi_j} \leq \frac{1}{\Pi} \) is trivially true for all \( \Pi \in \{ \Pi_i, \ldots, \Pi_j \} \). Equation 2 follows from these two observations.

We now show that if we select period values from \( \{ \Pi_{\text{lower}}, \ldots, \Pi_{\text{upper}} \} \) such that consecutive choices from this domain are either adjacent (i.e., the values are different by one) or have computed capacities different by at most a multiplicative factor of \((1 + \epsilon)\), then the minimum bandwidth resulting from these choices is at most a factor of \((1 + \epsilon)\) greater than the optimal minimum bandwidth.

Lemma 2 Consider monotonically non-decreasing capacity-allocation algorithm \( \mathcal{A} \), component \( C \), and real number \( \epsilon > 0 \). Let \( \{ \Pi_1, \ldots, \Pi_m \} \) be any (ordered) subset \( \{ \Pi_{\text{lower}}, \ldots, \Pi_{\text{upper}} \} \) such that \( \Pi_1 = \Pi_{\text{lower}}, \Pi_m = \Pi_{\text{upper}} \), and, for all \( i : 1 \leq i < m, \) either

\[
\Pi_{i+1} = \Pi_i + 1, 
\]

(3)

or

\[
\Theta_{\text{min}}^\mathcal{A}(\Pi_i, C) \leq \Theta_{\text{min}}^\mathcal{A}(\Pi_{i+1}, C) \leq (1 + \epsilon) \cdot \Theta_{\text{min}}^\mathcal{A}(\Pi_i, C). 
\]

(4)

is true. Then, the following inequalities holds:

\[
\frac{\Theta_{\text{min}}^\mathcal{A}(\Pi_i, C)}{\Pi_i(\mathcal{A}, C)} \leq \min_{\Pi \in \{ \Pi_1, \ldots, \Pi_m \}} \left\{ \frac{\Theta_{\text{min}}^\mathcal{A}(\Pi, C)}{\Pi} \right\} \leq (1 + \epsilon) \cdot \frac{\Theta_{\text{min}}^\mathcal{A}(\Pi_{i+1}, C)}{\Pi_{i+1}(\mathcal{A}, C)}. 
\]

(5)

Proof: There are two cases to consider:

1. \( \Pi^*(\mathcal{A}, C) \in \{ \Pi_1, \ldots, \Pi_m \} \); or
2. \( \Pi^*(\mathcal{A}, C) \notin \{ \Pi_1, \ldots, \Pi_m \} \).
For Case 1, it obvious that 
\[ \Theta_{\text{min}}(\Pi, C) = \Theta_{\text{min}}(A, C) = \Theta_{\text{min}}(A, (C), C) \] 
\[ \frac{\Theta_{\text{min}}(A, (C), C)}{\Pi^*(A, C)} \]
Equation 5 follows.

For Case 2, there must exist adjacent values \( \Pi_i, \Pi_{i+1} \in \{\Pi_1, \ldots, \Pi_m\} \) such that \( \Pi_i \leq \Pi^*(A, C) \leq \Pi_{i+1} \). Furthermore, \( \Pi_i \neq \Pi_{i+1} \); otherwise, \( \Pi^*(A, C) \) would equal either \( \Pi_i \) or \( \Pi_{i+1} \), and Case 1 would apply. By Lemma 1,
\[ \Theta_{\text{min}}(\Pi, C) = \Theta_{\text{min}}(A, (C), C) \]
\[ \frac{\Theta_{\text{min}}(A, (C), C)}{\Pi^*(A, C)} \]
\[ \leq \min_{\Pi \in \{\Pi_1, \ldots, \Pi_m\}} \left\{ \Theta_{\text{min}}(A, (C), C) \right\} \]
\[ \leq (1+\epsilon) \Theta_{\text{min}}(A, (C), C) \]
\[ \frac{\Theta_{\text{min}}(A, (C), C)}{\Pi^*(A, C)} \]
\[ \Rightarrow \Theta_{\text{min}}(\Pi, C) \leq (1+\epsilon) \Theta_{\text{min}}(A, (C), C) \]
\[ \frac{\Theta_{\text{min}}(A, (C), C)}{\Pi^*(A, C)} \]
\[ \text{(by Equation 4)}. \]

The final inequality implies
\[ \min_{\Pi \in \{\Pi_1, \ldots, \Pi_m\}} \left\{ \Theta_{\text{min}}(A, (C), C) \right\} \leq (1+\epsilon) \Theta_{\text{min}}(A, (C), C) \]
\[ \frac{\Theta_{\text{min}}(A, (C), C)}{\Pi^*(A, C)} \]
by definition of \( \Pi^*(A, C) \). From the preceding two inequalities, Equation 5 of the lemma follows.

Finally, we use Lemma 1 and 2 to show that \( \text{SelectInterface}(C, A, \Pi_{\text{lower}}, \Pi_{\text{upper}}, \epsilon) \) is an approximation scheme with approximation ratio \( (1+\epsilon) \). Furthermore, we quantify the running time of the algorithm in the following theorem.

**Theorem 1** SelectInterface\((C, A, \Pi_{\text{lower}}, \Pi_{\text{upper}}, \epsilon)\) has an approximation ratio of \((1+\epsilon)\) (where \( 0 < \epsilon \leq 1 \)) for period selection with respect to any monotonically, non-decreasing capacity- 

\[ \text{allocation algorithm } A. \] 

Furthermore, the algorithm has time complexity equal to
\[ O\left( \chi^A(C) \cdot \log \left( \frac{\Theta_{\text{upper}}}{\Theta_{\text{lower}}} \right) \cdot \log (\Pi_{\text{upper}}) \right), \] 
\[ (7) \]

where \( \Theta_{\text{lower}} \overset{\text{def}}{=} \Theta_{\text{min}}(\Pi_{\text{lower}}, C), \Theta_{\text{upper}} \overset{\text{def}}{=} \Theta_{\text{min}}(\Pi_{\text{upper}}, C), \) and \( \chi^A(C) \) equals the time complexity of the capacity-determination algorithm \( A \) given a component \( C. \)

**Proof Sketch:** Let \( \{\Pi_1, \ldots, \Pi_m\} \) be the ordered set of values that variables \( \Pi_{\text{last}} \) and \( \Pi \) (set in Line 7) are assigned throughout the execution of \( \text{SelectInterface}(C, A, \Pi_{\text{lower}}, \Pi_{\text{upper}}, \epsilon) \). Obviously, \( \{\Pi_1, \ldots, \Pi_m\} \subseteq \{\Pi_{\text{lower}}, \ldots, \Pi_{\text{upper}}\} \) such that \( \Pi_1 = \Pi_{\text{lower}} \) and \( \Pi_m = \Pi_{\text{upper}} \).

Furthermore, it is easy to verify that subsequent values of \( \Pi_i, \Pi_{i+1} \in \{\Pi_1, \ldots, \Pi_m\} \) satisfy either Equation 3 (see Line 8) or Equation 4 (see Line 7) of Lemma 2. Thus, by Equation 5 of Lemma 2, the \( \hat{\Pi} = (\hat{\Pi}, \hat{\Theta}) \) computed by Lines 4, 5, 13, 17, and 22 is at most \( (1+\epsilon) \cdot \Theta_{\text{min}}(\Pi^*(A, C), C) \). This shows that \( \text{SelectInterface}(C, A, \Pi_{\text{lower}}, \Pi_{\text{upper}}, \epsilon) \) has an approximation ratio of at most \((1+\epsilon)\).

For determining the complexity of \( \text{SelectInterface}(C, A, \Pi_{\text{lower}}, \Pi_{\text{upper}}, \epsilon) \), observe that the running time is dominated by the \textit{while} loop in Lines 6 through 21. The complexity of the \textit{while} loop can (informally) be determined by
\[ (\text{Number of iterations of while loop}) \]
\[ \times (\text{Number of } \Pi \text{ values to be checked in binary search}) \]
\[ \times (\text{Execution time to check value of } \Pi) \].

From (8) the number of iterations of the \textit{while} loop can be determined by observing that \( \Theta_{\text{last}} \) increases by at least a factor of \((1+\epsilon)\) upon every iteration of the \textit{while} loop of Lines 6 to 21. Thus, the number of iterations is equal to smallest integer value of \( \ell \) such that the following equation is true:
\[ (1+\epsilon)\ell \cdot \Theta_{\text{lower}} \geq \Theta_{\text{upper}} \]
Solving for \( \ell \),
\[ \ell = \left\lfloor \frac{\log(1+\epsilon) \cdot \Theta_{\text{upper}}}{\Theta_{\text{lower}}} \right\rfloor \]
\[ = \left\lfloor \frac{\ln \left( \frac{\Theta_{\text{upper}}}{\Theta_{\text{lower}}} \right)}{\ln(1+\epsilon)} \right\rfloor \]
\[ \leq \frac{(1+\epsilon) \cdot \ln \left( \frac{\Theta_{\text{upper}}}{\Theta_{\text{lower}}} \right)}{\epsilon} \]
\[ = O\left( \frac{\ln \left( \frac{\Theta_{\text{upper}}}{\Theta_{\text{lower}}} \right)}{\epsilon} \right). \]

The third equality above follows from the well-known identity \( x \leq \ln(1+x) \) for all \( x > -1 \).

The \textit{binary search} of Line 7 searches over the range \( \{\Pi_{\text{last}}, \ldots, \Pi_{\text{upper}}\} \). This range contains at most \( \Pi_{\text{upper}} \) number of values. Thus, the number of values of \( \Pi \in \{\Pi_{\text{last}}, \ldots, \Pi_{\text{upper}}\} \) that have to calculate \( \Theta_{\text{min}}^A(\Pi, C) \) is
\[ O \left( \lg (\Pi_{\text{upper}}) \right) \]. \hspace{1cm} (10)

Finally, the execution time for each calculation of \( T_{\text{min}}(\Pi, C) \) is equal to the execution cost of algorithm \( A \) for a given component, which is denoted \( \chi^A(C) \). Combining this observation with Equations 8, 9, and 10, implies the running time given in the lemma. \( \blacksquare \)

4 Interface Selection for Sporadic Task Systems

In this section, we explore an application of the algorithm proposed in the previous section to determine the minimum-bandwidth interface for a component consisting of sporadic tasks scheduled by EDF. We will show that we may, in fact, obtain an FPTAS for the MIB-RT problem in the context of sporadic tasks. The remainder of the section is organized as follows. In Section 4.1, we introduce notations and prior analytic results for the task, workload, and periodic resource models. In Section 4.2, we state the capacity-determination algorithm we use in deriving our FPTAS; the capacity-determination algorithm was previously proposed in [12]. In Section 3.1, we give a description of our FPTAS and prove its correctness.

4.1 Models and Notation

In this subsection, we present background and notation for the task model, workload functions, and periodic resource model that we use throughout the remainder of this section.

§Sporadic Task Model. A sporadic task \( \tau_i = (e_i, d_i, p_i) \) is characterized by a worst-case execution requirement \( e_i \), a (relative) deadline \( d_i \), and a minimum inter-arrival separation \( p_i \), which is, for historical reasons, also referred to as the period of the task. Such a sporadic task generates a potentially infinite sequence of jobs, with successive job-arrivals separated by at least \( p_i \) time units. Each job has a worst-case execution requirement equal to \( e_i \) and a deadline that occurs \( d_i \) time units after its arrival time. We will assume that task parameters are positive integers. Furthermore, obviously, \( e_i \leq d_i \) and \( e_i \leq p_i \) for any task \( \tau_i \); otherwise, cannot be scheduled to meet its deadline by any scheduling algorithm. A sporadic task system \( \tau = \{ \tau_1, \ldots, \tau_n \} \) is a collection of \( n \) such sporadic tasks. A useful metric for a sporadic task \( \tau_i \) is the task utilization \( u_i = e_i / p_i \). The system utilization is denoted \( U(\tau) = \sum_{\tau_i \in \tau} u_i \).

The following lemma on system utilization will be useful in defining an FPTAS.

Lemma 3 For any sporadic task system \( \tau \) with positive integer parameters,

\[ U(\tau) \geq \frac{n}{p_{\text{max}}} \] \hspace{1cm} (11)

where \( p_{\text{max}} \) is the maximum of the relative deadlines.

Proof: By definition, \( U(\tau) \) equals \( \sum_{\tau_i \in \tau} \frac{e_i}{p_i} \). Equation 11 follows from observing that \( e_i \geq 1 \) and \( p_i \leq p_{\text{max}} \) for all \( \tau_i \in \tau \).

§Workload Functions. For determining schedulability of a sporadic task system, it is often useful to quantify the maximum amount of execution that must complete over any given interval. For this purpose, researchers [5] have derived the demand-bound function, defined below.

Definition 2 (Demand-Bound Function)

For any \( t > 0 \) and task \( \tau_i \), the demand-bound function (dbf) quantifies the maximum cumulative execution requirement of all jobs of \( \tau_i \) that could have both an arrival time and deadline in any interval of length \( t \). Baruah et al. [5] have shown that, for sporadic tasks, dbf can be calculated as follows.

\[ \text{dbf}(\tau_i, t) = \max \left( 0, \left\lfloor \frac{t - d_i}{p_i} \right\rfloor + 1 \right) \cdot e_i. \] \hspace{1cm} (12)

Figure 1 gives a visual depiction of the demand-bound function for a sporadic task \( \tau_i \). Observe from the above definition and Figure 1 that the dbf is a right continuous function with discontinuities at time points of the form \( t = d_i + a \cdot p_i \) where \( a \in \mathbb{N} \). Let DBF(\( \tau_i, t \)) = \sum_{\tau_i \in \tau} \text{dbf}(\tau_i, t). It has been shown [5] that condition DBF(\( \tau_i, t \)) \leq t, \forall t \geq 0 \) is necessary and sufficient for sporadic task system \( \tau \) to be EDF-schedulable upon a preemptive uniprocessor platform of unit speed. Furthermore, it has also been shown that the aforementioned condition needs to be verified at only time points in the following ordered set (with elements are in non-decreasing order):

\[ \text{TS}(\tau) = \bigcup_{\tau_i \in \tau} \{ t \equiv d_i + a \cdot p_i \mid (a \in \mathbb{N}) \land (t \leq P(\tau)) \}. \] \hspace{1cm} (13)
Let \( \mathbf{DBF}(\tau, t, k) \overset{\text{def}}{=} \sum_{\tau_i \in \tau} \mathbf{dbf}(\tau_i, t, k) \). Albers and Slomka show [2], for any fixed \( k \in \mathbb{N}^+ \), the condition \( \mathbf{DBF}(\tau, t, k) \leq t, \forall t \geq 0 \) is sufficient for sporadic task \( \tau \) to be EDF-schedulable upon a preemptive uniprocessor platform of unit speed. The ordered testing set of this condition is reduced to

\[
\mathbf{TS}(\tau, k) \overset{\text{def}}{=} \bigcup_{\tau_i \in \tau} \{ t = d_i + a \cdot p_i \mid (a \in \mathbb{N}) \land (a < k) \land (t \leq P(\tau)) \}.
\] (15)

\textbf{§Periodic Resource Model.} Throughout this section, we assume that each component \( C \) is a sporadic task system. Let \( C \) be composed of a sporadic task system \( \tau \) that is to be EDF-scheduled upon periodic resource \( \Gamma = (\Pi, \Theta) \). (From now on, we use \( \tau \) in the context of component \( C \).) We will now present some concepts that have been previously-introduced by researchers to determine whether \( \tau \) will meet all deadlines when scheduled upon \( \Gamma \).

\textbf{Definition 3 (Supply-Bound Function)}

For any \( t > 0 \), the supply-bound function (sbf) quantifies the minimum execution supply that a component executed upon periodic resource \( \Gamma \) may receive over any interval of length \( t \). Shin and Lee [18] have quantified the supply bound function for an periodic resource in the following (using the notation of Easwaran et al. [9]):

\[
sbf(\Gamma, t) = \begin{cases} y_{\Gamma} \cdot \Theta + \max (0, t - x_{\Gamma} - y_{\Gamma} \cdot \Pi), & \text{if } t \geq \Pi - \Theta \\ 0, & \text{otherwise.} \end{cases}
\] (16)

where \( y_{\Gamma} = \left\lfloor \frac{\Theta - (\Pi - \Theta)}{\Pi} \right\rfloor \) and \( x_{\Gamma} = (2\Pi - 2\Theta) \).

EDF-schedulability conditions for periodic resource \( \Gamma \) have been developed [9, 16, 17], as given in the following theorem.

\textbf{Theorem 2 (from [9])} A sporadic task system \( \tau \) is EDF-schedulable upon an EDP resource \( \Gamma = (\Pi, \Theta) \), if and only if,

\[
\mathbf{DBF}(\tau, t, \leq \text{sbf}(\Gamma, t), \forall t \leq P(\tau)) \land \left( U(\tau) \leq \frac{\Theta}{\Pi} \right)
\] (17)

where \( P(\tau) \) equals \( \text{lcm}_{\tau_i \in \tau} \{ p_i \} + \max_{\tau_i \in \tau} \{ d_i \} \).

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure1}
\caption{The step function denotes a plot of \( \mathbf{dbf}(\tau_i, t) \) as a function of \( t \). The dashed line represents the function \( \mathbf{dbf}(\tau_i, t, k) \), approximating \( \mathbf{dbf}(\tau_i, t) \). \( \mathbf{dbf}(\tau_i, t, k) \) is equal to \( \mathbf{dbf}(\tau_i, t) \) for all \( t < d_i + (k-1)p_i \) \((k \text{ equals three in the above graph}).
\end{figure}
4.2 Capacity-Determination Algorithm

The algorithm that we consider for determining the minimum-capacity of a periodic resource (for a fixed period) is given by Algorithm 2. Informally, the algorithm works as follows: for each value \( t \) in the testing set \( \bar{TS}(\tau, k) \), \( \text{MinimumCapacity} \) determines the minimum capacity \( \Theta^\text{min}_t \) such that the \( \text{DBF}(\tau, t) \) is “below” the supply-bound function \( \text{sbf}((\Pi, \Theta^\text{min}_t), \tau) \). The algorithm is exact when \( k \) equals \( \infty \), since \( \bar{TS}(\tau, \infty) \) equals \( TS(\tau) \). More details on the algorithm and a proof of correctness is available from [12]. Please note that the algorithm found in [12] is given for the more general explicit-deadline periodic resource (EDP) model [9]; the algorithm stated here has been adapted to for the periodic resource model.

Algorithm 2 \text{MinimumCapacity}(\Pi, \tau, k).

\begin{algorithm}
\begin{enumerate}
\item \( \Theta^\text{min} \leftarrow U(\tau) \cdot \Pi \)
\item \textbf{for all} \( t \in \bar{TS}(\tau, k) \) \textbf{do}
\item \( D_t \leftarrow \text{DBF}(\tau, t, k) \)
\item \( \alpha_t \leftarrow \sum_{\tau \geq D_t + (k-1)p_i} u_i \)
\item \( \Theta^\text{min}_t \leftarrow \infty \)
\item \textbf{for} \( \ell = \max\{1, \left\lfloor \frac{t}{\Pi} \right\rfloor - 1\} \text{ to } \left\lceil \frac{t}{\Pi} \right\rceil \) \textbf{do}
\item \( \Theta^\text{min}_t \leftarrow \max\left\{ \frac{\alpha_t \Pi}{\ell+1}, \frac{D_t+\alpha_t((\ell+2)\Pi-t)}{\ell+2\alpha_t} \right\} \)
\item \( \Theta^\text{min}_t \leftarrow \min\{\Theta^\text{min}_t, \Theta^\text{min}_t\} \)
\item \textbf{end for}
\item \( \Theta^\text{min} \leftarrow \max\{\Theta^\text{min}, \Theta^\text{min}_t\} \)
\item \textbf{end for}
\item \textbf{return} \( \Theta^\text{min} \)
\end{enumerate}
\end{algorithm}

The following result (restated from [12]) shows that \( \text{MinimumCapacity} \) is an FPTAS for capacity-determination given a fixed period.

Theorem 3 (from [12]) Given \( \Pi, \tau, \text{ and } \epsilon > 0 \), the procedure \( \text{MinimumCapacity}(\Pi, \tau, \max\{1, \left\lceil \frac{1}{\tau} \right\rceil\}) \) returns \( \Theta^\text{min} \) such that

\[ \Theta^*(\Pi, \tau) \leq \Theta^\text{min} \leq (1 + \epsilon) \cdot \Theta^*(\Pi, \tau) \]

where \( \Theta^*(\Pi, \tau) \) is the optimal minimum capacity required to EDF-schedule \( \tau \) upon a periodic resource with period of \( \Pi \). Furthermore, \( \text{MinimumCapacity}(\Pi, \tau, \max\{1, \left\lceil \frac{1}{\tau} \right\rceil\}) \) has time complexity \( O\left(\frac{n \log n}{\epsilon}\right) \).

4.3 An FPTAS for Interface Selection

In this subsection, we present the main result of the section: an FPTAS for interface selection for periodic resources scheduling sporadic tasks. Before we state our main result, we require a couple technical lemmas. The first lemma gives an upper and lower bound on the value returned by \( \text{MinimumCapacity} \).

Lemma 4 For a given \( \Pi \), sporadic task system \( \tau \), and \( k \in \mathbb{N}^+ \cup \{\infty\} \), if \( t \geq \text{DBF}(\tau, t, k) \) for all \( t \in \bar{TS}(\tau, k) \) and \( U(\tau) \leq 1 \), then the capacity \( \Theta^\text{min} \) returned from \( \text{MinimumCapacity}(\Pi, \tau, k) \) satisfies

\[ \Pi \cdot \frac{n}{\rho_{\text{max}}} \leq \Theta^\text{min} \leq \Pi \]

Proof: Line 1 of \( \text{MinimumCapacity} \) sets \( \Theta^\text{min} \) to the minimum default value of \( U(\tau) \cdot \Pi \). Thus, \( \Theta^\text{min} \geq U(\tau) \cdot \Pi \). By Lemma 3, \( \Theta^\text{min} \geq \Pi \cdot \frac{n}{\rho_{\text{max}}} \).

To see the upper bound on \( \Theta^\text{min} \), consider Line 7 when \( \ell \) equals \( \left\lceil \frac{t}{\Pi} \right\rceil \). The value \( \frac{D_t}{\ell} \) is at most \( \Pi \) since \( D_t \leq t \). The value \( \frac{D_t}{\ell} \) is also at most \( \frac{D_t}{\ell} \) since \( \ell = \left\lceil \frac{t}{\Pi} \right\rceil \geq \frac{t}{\Pi} \); since \( t \geq D_t, \frac{D_t}{\ell} \leq \Pi \). Finally, the value \( \frac{D_t}{\ell} \) is increasing in \( \alpha_t \) when \( \ell \) equals \( \left\lceil \frac{t}{\Pi} \right\rceil \); thus, \( \frac{D_t+\alpha_t((\ell+2)\Pi-t)}{\ell+2}\alpha_t \leq \frac{D_t}{\ell+2} \) which is at most \( \Pi \) since \( t \geq D_t \).

We have, thus, shown that \( \Theta^\text{min} \leq \Pi \) when \( \ell \) equals \( \left\lceil \frac{t}{\Pi} \right\rceil \). Line 8 sets \( \Theta^\text{min} \) to the minimum value of \( \Theta^\text{min}_t \) for any \( t \in \bar{TS}(\tau, k) \). The inequality \( \Theta^\text{min} \leq \Pi \) follows.

The second technical lemma states that \( \text{MinimumCapacity} \) is monotonically, non-decreasing.

Lemma 5 For a given \( \tau \) and \( \epsilon > 0 \), \( \text{MinimumCapacity}(\Pi) \) is monotonically, non-decreasing over \( \{\Pi_{\text{lower}}, \ldots, \Pi_{\text{upper}}\} \).

Proof Sketch: The lemma follows from observing that the values of \( \Theta^\text{min} \) and \( \Theta^\text{min}_t \) (set in Lines 1 and 7, respectively) are monotonically, non-decreasing in \( \Pi \).
Finally, we give the FPTAS for the MIB-RT with respect to sporadic tasks. The theorem below uses both \textit{SelectInterface} and \textit{MinimumCapacity} to obtain the FPTAS.

**Theorem 4** Given sporadic task system $\tau$ and accuracy parameter $\epsilon : 0 < \epsilon \leq 1$, if $t \geq DBF(\tau, t, k)$ for all $t \in \mathcal{T}(\tau, k)$ and $U(\tau) \leq 1$, then the procedure \textit{SelectInterface}(\tau, A, \Pi_{\text{lower}}, \Pi_{\text{upper}}, \hat{\tau})$ where $A$ equals \textit{MinimumCapacity} $(\cdot, \tau, \max (1, \lfloor \frac{2}{\epsilon} \rfloor))$ returns $\hat{\Gamma} = (\hat{\Pi}, \Theta^A_{\text{min}}(\hat{\Pi}, \tau))$ such that

$$\frac{\Theta^\text{OPT}_{\min}(\Pi^*(OPT, \tau), \tau)}{\Pi^*(OPT, \tau)} \leq \frac{\Theta^A_{\text{min}}(\Pi, \tau)}{\hat{\Pi}} \leq \frac{\Theta^\text{OPT}_{\min}(\Pi^*(OPT, \tau), \tau)}{\Pi^*(OPT, \tau)} + (1 + \epsilon).$$

Equation 19 follows from the fact that $\frac{\Theta^A_{\text{min}}(\Pi, \tau)}{\Pi} \geq \Theta^A_{\text{min}}(\hat{\Pi}, \tau)$.

The time complexity of the approach follows from Equation 7 of Theorem 1. By Theorem 3, $\chi^A(\tau)$ is $O(n \log n / \epsilon)$. The second term of Equation 7 ($\log (\Theta^A_{\text{min}}(\Pi_{\text{lower}})/\epsilon)$ is upper bounded by $\Theta^A_{\text{min}}(\Pi_{\text{upper}})$) according to Lemma 4; this term is $O((n \log \Pi_{\text{upper}}) + \log(p_{\text{max}}))$. Thus, the entire time complexity of the approach of the Theorem is $O(n \log n (\log^2 \Pi_{\text{upper}} + \log \Pi_{\text{upper}} \log p_{\text{max}}) / \epsilon^2)$, which is polynomial in the number of tasks, number of bits to represent both the task parameters and $\Pi_{\text{upper}}$, and $1/\epsilon$.

**5 Conclusions**

In this paper, we have proposed an approximation algorithm for the minimization of interface bandwidth (MIB-RT) problem in a real-time compositional framework, the periodic resource model. We first propose a general algorithm for determining the interface parameter, given a capacity determination algorithm. This approach is general and can apply to any component task model. Next, we explore interface selection for components consisting of entirely sporadic tasks, and propose an algorithm based on a previous capacity-determination algorithm [12]. Our algorithm returns bandwidth that is at most a factor of $(1 + \epsilon)$ greater than the optimal minimum bandwidth, for any $\epsilon > 0$.Furthermore, it is shown that our algorithm is an FPTAS as it has time complexity that is polynomial in the number of tasks in the sporadic task system, the number of bits to represent the task parameters, the number of bits to represent the maximum task period $\Pi_{\text{upper}}$, and the term $1/\epsilon$. Previous work [8] has shown that exact algorithms for MIB-RT problem on periodic resources may require pseudo-polynomial or exponential time.
Thus, our results may provide a significant reduction in the time necessary to determine the minimum-bandwidth interface parameters.

In future work, we hope to explore interface selection in the presence of shared global resources; we would also like to study the effect of overheads in the choice of interface parameters. We believe that our central idea of our FPTAS proposed in this paper is general enough to extend to these more practical and complex settings.

References


