



2012

PADS: An Approach to Modeling Resource Demand and Supply for the Formal Analysis of Hierarchical Scheduling

Anna Philippou

University of Cyprus, Nicosia, Cyprus, annap@cs.ucy.ac.cy

Insup Lee

University of Pennsylvania, lee@cis.upenn.edu

Oleg Sokolsky

University of Pennsylvania, sokolsky@cis.upenn.edu

Follow this and additional works at: http://repository.upenn.edu/cis_papers

Recommended Citation

Anna Philippou, Insup Lee, and Oleg Sokolsky, "PADS: An Approach to Modeling Resource Demand and Supply for the Formal Analysis of Hierarchical Scheduling", *Theoretical Computer Science* 413(1), 2-20. January 2012. <http://dx.doi.org/10.1016/j.tcs.2011.08.025>

Manuscript Number: TCS-D-10-00731R2

This paper is posted at ScholarlyCommons. http://repository.upenn.edu/cis_papers/486

For more information, please contact libraryrepository@pobox.upenn.edu.

PADS: An Approach to Modeling Resource Demand and Supply for the Formal Analysis of Hierarchical Scheduling

Abstract

As real-time embedded systems become more complex, resource partitioning is increasingly used to guarantee real-time performance. Recently, several compositional frameworks of resource partitioning have been proposed using real-time scheduling theory with various notions of real-time tasks running under restricted resource supply environments. However, these real-time scheduling-based approaches are limited in their expressiveness in that, although capable of describing resource-demand tasks, they are unable to model resource supply. This paper describes a process algebraic framework PADS for reasoning about resource demand and resource supply inspired by the timed process algebra ACSR. In ACSR, real-time tasks are specified by enunciating their consumption needs for resources. To also accommodate resource-supply processes in PADS, given a resource cpu we write \overline{cpu} to denote the availability of cpu for a requesting task process. Using PADS, we define a supply-demand relation where a pair (T, S) belongs to the relation if the demand process T can be scheduled under supply S . We develop a theory of compositional schedulability analysis as well as a technique for synthesizing an optimal supply process for a set of tasks. Furthermore, we define ordering relations between supplies which describe when a supply offers more resource capacity than another. With this notion it is possible to formally represent hierarchical scheduling approaches that assign more “generous” resource allocations to tasks in exchange for a simple representation. We illustrate our techniques via a number of examples.

Comments

Manuscript Number: TCS-D-10-00731R2

PADS: An approach to modeling resource demand and supply for the formal analysis of hierarchical scheduling

Anna Philippou^a, Insup Lee^b, Oleg Sokolsky^b

^a*Department of Computer Science, University of Cyprus, Nicosia, Cyprus*

^b*Department of Computer and Info. Science, University of Pennsylvania, Philadelphia, PA, U.S.A.*

Abstract

As real-time embedded systems become more complex, resource partitioning is increasingly used to guarantee real-time performance. Recently, several compositional frameworks of resource partitioning have been proposed using real-time scheduling theory with various notions of real-time tasks running under restricted resource supply environments. However, these real-time scheduling-based approaches are limited in their expressiveness in that, although capable of describing resource-demand tasks, they are unable to model resource supply. This paper describes a process algebraic framework PADS for reasoning about resource demand and resource supply inspired by the timed process algebra ACSR. In ACSR, real-time tasks are specified by enunciating their consumption needs for resources. To also accommodate resource-supply processes in PADS, given a resource cpu , we write \overline{cpu} to denote the availability of cpu for a requesting task process. Using PADS, we define a supply-demand relation where a pair (T, S) belongs to the relation if the demand process T can be scheduled under supply S . We develop a theory of compositional schedulability analysis as well as a technique for synthesizing an optimal supply process for a set of tasks. Furthermore, we define ordering relations between supplies which describe when a supply offers more resource capacity than another. With this notion it is possible to formally represent hierarchical scheduling approaches that assign more “generous” resource allocations to tasks in exchange for a simple representation. We illustrate our techniques via a number of examples.

1. Introduction

Component-based design has been widely accepted as a compositional approach to facilitate the design of complex systems. It provides means for decomposing a complex system into simpler components and for composing the components using interfaces that abstract component complexities. Such approaches are increasingly used in practice for real-time systems. For example, ARINC-653 standards by the Engineering Standards for Avionics and Cabin Systems committee specify partition-based design

Email addresses: annap@cs.ucy.ac.cy (Anna Philippou), lee@cis.upenn.edu (Insup Lee), sokolsky@cis.upenn.edu (Oleg Sokolsky)

of avionics applications. Also, hypervisors for real-time virtual machines provide temporal partitions to guarantee real-time performance [15, 11].

To take advantage of the component-based design of real-time systems, schedulability analysis should support compositional analysis using component interfaces. These interfaces should abstract the timing requirements of a component with a minimum resource supply that is needed to meet the resource demand of the component. Component-based real-time systems often involve hierarchical scheduling frameworks that support resource sharing among components as well as associated scheduling algorithms [5, 22]. To facilitate the analysis of such systems, resource component interfaces and their compositional analysis have been proposed [16, 23, 24, 8, 25, 12]

Process algebras are abstract and compositional methodologies for system specification and analysis. They allow to systematically build complex systems from smaller ones via the use of a small number of operators, as well as to reason compositionally about system correctness. As such, they provide a promising framework in which to study compositional scheduling. This paper presents a formal treatment of the problem of compositional hierarchical scheduling by introducing a process algebraic framework, PADS, for modeling resource demand and supply inspired by the timed process algebra ACSR [13, 14]. The notions of resource demand and resource supply are fundamental in defining the meaning of compositional real-time scheduling analysis. Our proposed framework formally defines both of these notions. As in ACSR, a task in our process algebra is specified by describing its consumption needs for resources. To also accommodate resource-supply processes, we extend the notion of a *resource* and given a resource *cpu* we use \overline{cpu} to denote the availability of the resource for consumption by a requesting task. Our formalism then addresses the following issues:

1. *Schedulability*: We define a *supply simulation relation* \models that captures when a task T is schedulable by a supply S , $S \models T$.
2. *Compositionality*: We explore conditions under which we may safely compose schedulable systems. Specifically, we are interested to define functions on supplies, \circ , and appropriate conditions, f , such that if T_1 is schedulable by S_1 and T_2 by S_2 then the parallel composition of T_1 and T_2 is schedulable by $S_1 \circ S_2$, assuming that condition f holds:

$$\frac{S_1 \models T_1, S_2 \models T_2}{S_1 \circ S_2 \models T_1 \parallel T_2}, \quad f(S_1, S_2)$$

3. *Supply Synthesis*: We propose a method by which we can generate a supply process to schedule a set of tasks, assuming that such a scheduler exists. Our method is based on the notion of a *demand* of a task which is a supply that can schedule the task and, at the same time, it is optimal in the sense that (1) it does not reserve more resources than those required and (2) it captures all possible ways in which a task can be scheduled. We then prove that two or more tasks are schedulable if and only if they can be scheduled by the composition of their demands.
4. *Task and Supply Orderings*: We propose an ordering between tasks which defines when a task is more “demanding” than another, meaning that it requires more resources in order to execute correctly. We also propose two orderings between supplies which define when a supply is more “generous” than another

meaning that it offers a greater resource allocation. The main result accompanying these notions is that any supply that schedules a more demanding task may also schedule a less demanding task and that any task schedulable by a less generous supply is also schedulable by a more generous supply. This result comes to complement our supply synthesis approach since it allows us to check whether a supply S schedules a task set as follows: We begin by constructing the optimal supply/demand, D , for the task set and then check whether S is more generous than D . In the affirmative case we may conclude that the task set is also schedulable by S .

Related work. As mentioned above, this work brings together two long-standing lines of research. On the one hand, there has been much work on compositional hierarchical scheduling based on real-time scheduling theory [16, 23, 24, 8, 6, 7]. Typically, such approaches to schedulability analysis rely on over-approximations of task demand using, for example, demand bound functions and under-approximations of resource supply using supply bound functions. Efficient algorithms are developed to ensure that demand never exceeds supply. On the other hand, several formal approaches to scheduling based on process algebras [3, 14, 13, 20, 18, 19], task automata [10, 9], preemptive Petri nets [4], etc., have been developed. To the best of our knowledge, none of these approaches consider the modeling of resource supply explicitly. Instead, sharing of a continuously available processing resource between a set of tasks has been considered.

Our approach to supply synthesis is conceptually similar to the work of Altisen *et al.* on applying controller synthesis to scheduling problems [1, 2]. The difference is that we are not aiming to generate schedulers, but rather an *interface* for a task set, an abstraction that can be used in a component-based approach to real-time system design.

The present paper extends our previous work of [21] as follows. It introduces priorities to the framework, thus allowing us to represent schedulability with respect to particular schedulers and it contains all the proofs missing from [21] adopted for the extended framework. Furthermore, it introduces ordering relations between tasks and supplies and associated results that enable us to formally represent techniques for over-approximating optimal resources as can be found in e.g. [23].

The rest of the paper is structured as follows. Section 2 presents our process algebra and its semantics. Section 3 contains our results on compositional schedulability analysis and interface construction, followed by examples illustrating the application of the theory in Section 3.3. Section 4 presents hierarchies between tasks and supplies and develops their properties and, finally, Section 5 concludes the paper.

2. The Language

In our calculus, PADS (Process Algebra for Demand and Supply), we consider a system to be a set of processes operating on a set of serially reusable resources denoted by R . These processes are (1) the *tasks* of the system, which require the use of resources in order to complete their jobs, and (2) the *supplies*, that specify when each resource is available to the tasks. Based on this, each resource $r \in R$ can be requested by a task, r ,

granted by a supply, \bar{r} , or consumed, \overleftarrow{r} , when a supply and a request for the resource are simultaneously available.

An action in PADS is a set relating to resource requests, grants and consumptions, where each resource may be represented at most once. Resource requests and consumptions are associated with a priority, where priorities are drawn from the nonnegative integers. These priorities are used to arbitrate between actions, the intention being that an action with a higher priority always wins. Supplies of resources are not associated with priorities since a resource can either be supplied or not supplied to a component and cannot be simultaneously offered to two or more tasks in a system. For example, the action $\{(r_1, 1), (r_2, 3)\}$ represents a request for the resources r_1 and r_2 at priorities 1 and 3, respectively, whereas the action $\{\bar{r}_1, (\overleftarrow{r}_2, 2), (r_3, 1)\}$ involves the granting of resource r_1 , consumption of resource r_2 at priority level 2 and request for resource r_3 at priority level 1.

Our framework is intended to capture real-time, resource-aware systems. Such systems have a limited number of shared resources each of which is capable of participating in at most one action at a time. To capture this view and enable reasoning about scheduling such systems, our process algebra contains the notion of time. In particular, we take a discrete time approach: we assume that all actions require one unit of time to complete measured on a global clock, with action \emptyset representing idling for one time unit since no resource is being employed.

We write Act , ranged over by α and β , for the set of all actions and distinguish Act_R , the set of actions involving only resource requests, ranged over by ρ , and Act_G , the set of actions involving only resource grants, ranged over by γ . Given $\alpha \in Act$ we write α^b to remove all priorities from resource-priority pairs in α , e.g. $\{(r_1, 2), \bar{r}_2, (\overleftarrow{r}_3, 1)\}^b = \{r_1, \bar{r}_2, \overleftarrow{r}_3\}$ and $res(\alpha)$ for the set of resources occurring in α , e.g. $res(\{(r_1, 2), \bar{r}_2, (\overleftarrow{r}_3, 1)\}) = \{r_1, r_2, r_3\}$. Finally, given an action α and a resource r , we write $\pi_\alpha(r)$ for the priority at which resource r is employed within action α , where we consider all supplied resources to be employed at priority level 0, e.g. for $\alpha = \{(r_1, 2), \bar{r}_2, (\overleftarrow{r}_3, 4)\}$, we have $\pi_\alpha(r_1) = 2$, $\pi_\alpha(r_2) = 0$ and $\pi_\alpha(r_3) = 4$. (Note that π_α is well defined since we have assumed that each resource may be represented in an action at most once.)

2.1. Syntax

The following grammars define the set of tasks T , the set of supplies S and the set of timed systems P , where we recall that $\rho \in Act_R$ and $\gamma \in Act_G$. Furthermore, C ranges over a set of *task constants*, each with an associated definition of the form $C \stackrel{\text{def}}{=} T$, where T may contain occurrences of C as well as other task constants and D ranges over a similar set of *supply constants*.

$$\begin{aligned} T & ::= \text{FIN} \mid \rho : T \mid T + T \mid C \\ S & ::= \text{FIN} \mid \gamma : S \mid S + S \mid D \\ P & ::= \delta \mid T \mid S \mid P \parallel P \end{aligned}$$

We consider FIN to be the well-terminated process. Then a task process can be FIN, a task constant C , an action-prefixed process $\rho : T$ which executes ρ during

the first time unit and then behaves as T , or a nondeterministic choice $T_1 + T_2$ which offers the choice between executing T_1 or T_2 . Similarly, a supply process can be FIN, a supply constant S , an action-prefixed process $\gamma : S$, or a nondeterministic choice $S_1 + S_2$. We make the assumption that all constants are guarded by an action, that is, it is not possible to define a process such as $C \stackrel{\text{def}}{=} D + \dots$.

Finally, a system can be a deadlocked system, δ , or an arbitrary composition of tasks and supplies. In a parallel composition $P_1 \parallel P_2$, P_1 and P_2 run concurrently and synchronize while executing their actions. Furthermore, whenever one process requests a resource granted by the other, we obtain a consumption of the resource in question. Note that the difference between FIN and δ is that while FIN allows time to pass, δ does not. As a shorthand notation we will write $\sum_{i \in I} P_i$ for $P_{i_1} + \dots + P_{i_n}$, where $I = \{i_1, \dots, i_n\}$. Note that, given our assumption that all process constants occur guarded by an action, any task or supply different to FIN is in fact a guarded choice of the form $\sum_{i \in I} \alpha_i : P_i$.

2.2. Semantics

The semantics of PADS is given in two steps. First, we develop a transition relation in which nondeterminism is resolved in all possible ways, the unprioritized transition relation \rightarrow . Then, we refine \rightarrow into \longrightarrow , the prioritized transition relation, on the basis of a preemption relation which implements a type of “angelic” behavior in the way in which tasks resolve their nondeterminism, choosing the best possible outcome given the available supply and taking priorities into account.

We proceed to consider the unprioritized transition relation \rightarrow defined in Table 1. FIN being a well-terminated (and not a deadlocked) process, it allows time to pass (axiom (IDLE)). Action-prefixed processes first execute their initial action and then proceed according to the continuation ((ActT) and (ActS)). Nondeterministic choice behaves as either of its constituent summands ((SumT) and (SumS)). A constant behaves as the process in its defining equation ((ConstT) and (ConstS)). Finally, rule (Par) specifies the way in which a parallel system evolves. To begin with, we recall that all actions take one time unit thus every step of a parallel composition should capture the actions of each of its components during the first time unit. To achieve this, the components of a parallel composition evolve synchronously and the composition advances only if both of the constituent processes are willing to take a step. Furthermore, the rule enunciates the outcome of the synchronization between two parallel processes, the most important aspect being that a request within one component is satisfied by an available grant in the other. The condition of rule (Par) imposes a restriction on when two actions may take place simultaneously within a system. Specifically, we say that actions α_1 and α_2 are *compatible* with each other if, whenever r occurs in both actions then one occurrence must be a request and the other a supply of the resource. So, for example, it is not possible to simultaneously offer a resource in one component and consume or offer it in another, nor to request it by two different tasks. We capture this requirement as follows:

$$\text{compatible}(\alpha_1, \alpha_2) = \bigwedge_{r \in \text{res}(\alpha_1) \cap \text{res}(\alpha_2)} (r \in \alpha_1^b \wedge \bar{r} \in \alpha_2^b) \vee (r \in \alpha_2^b \wedge \bar{r} \in \alpha_1^b)$$

Table 1: Transition rules for tasks, supplies and systems

(Idle)	$\text{FIN} \xrightarrow{\emptyset} \text{FIN}$		
(ActT)	$\rho : T \xrightarrow{\rho} T$	(ActS)	$\gamma : S \xrightarrow{\gamma} S$
(SumT)	$\frac{T_i \xrightarrow{\alpha} T, i \in \{1, 2\}}{T_1 + T_2 \xrightarrow{\alpha} T}$	(SumS)	$\frac{S_i \xrightarrow{\alpha} S, i \in \{1, 2\}}{S_1 + S_2 \xrightarrow{\alpha} S}$
(ConstT)	$\frac{T \xrightarrow{\alpha} T'}{C \xrightarrow{\alpha} T'} \quad C \stackrel{\text{def}}{=} T$	(ConstS)	$\frac{S \xrightarrow{\alpha} S'}{D \xrightarrow{\alpha} S'} \quad D \stackrel{\text{def}}{=} S$
(Par)	$\frac{P_1 \xrightarrow{\alpha_1} P'_1 \quad P_2 \xrightarrow{\alpha_2} P'_2}{P_1 \parallel P_2 \xrightarrow{\alpha_1 \oplus \alpha_2} P'_1 \parallel P'_2}$	compatible(α_1, α_2)	

We may now combine compatible actions by transforming a simultaneous request and supply of the same resource into a consumption:

$$\begin{aligned} \alpha_1 \oplus \alpha_2 = & \{(r, p) \in \alpha_1 \cup \alpha_2 \mid \bar{r} \notin \alpha_1 \cup \alpha_2\} \cup \{\bar{r} \in \alpha_1 \cup \alpha_2 \mid (r, p) \notin \alpha_1 \cup \alpha_2\} \\ & \cup \{(\overset{\leftrightarrow}{r}, p) \mid (r, p) \in \alpha_i, \bar{r} \in \alpha_{3-i}, i \in \{1, 2\} \text{ or } (\overset{\leftrightarrow}{r}, p) \in \alpha_1 \cup \alpha_2\} \end{aligned}$$

We may show the parallel composition operator to be associative with respect to \rightarrow in the sense that, $P_1 \parallel (P_2 \parallel P_3) \xrightarrow{\alpha} P'_1 \parallel (P'_2 \parallel P'_3)$ if and only if $(P_1 \parallel P_2) \parallel P_3 \xrightarrow{\alpha} (P'_1 \parallel P'_2) \parallel P'_3$. This can be shown by establishing that (1) compatible(α_1, α_2) and compatible($\alpha_1 \oplus \alpha_2, \alpha_3$) if and only if compatible(α_2, α_3) and compatible($\alpha_1, \alpha_2 \oplus \alpha_3$), and (2) the associativity of \oplus . Both of these properties are easy to prove by referring to the definitions.

Example 2.1. Consider the supply $S \stackrel{\text{def}}{=} \{\bar{r}_1, \bar{r}_2\} : S$ which offers resources r_1 and r_2 simultaneously and the following task processes:

$$\begin{aligned} T_1 & \stackrel{\text{def}}{=} \{(r_1, 2)\} : \text{FIN} + \emptyset : \{(r_1, 2)\} : \text{FIN} \\ T_2 & \stackrel{\text{def}}{=} \{(r_1, 1), (r_2, 1)\} : \text{FIN} + \{(r_2, 1), (r_3, 1)\} : T_2 \\ T_3 & \stackrel{\text{def}}{=} \{(r_1, 1), (r_2, 1)\} : \text{FIN} + \{(r_2, 1)\} : \{(r_1, 1)\} : \text{FIN} \end{aligned}$$

Task T_1 places a demand for resource r_1 at priority level 2 during either the first or the second time unit. Task T_2 requires the use of two resources simultaneously during the first time unit, either r_1 and r_2 or r_2 and r_3 . Finally, task T_3 requires the use of resources r_1 and r_2 either simultaneously or in sequence. The transition systems of $T_1 \parallel S$, $T_2 \parallel S$ and $(T_1 \parallel S) \parallel T_3$ are depicted in Figure 1.

Note that $(T_2 \parallel S) \parallel T_3$ has no transitions altogether since both T_2 and T_3 require r_2 during the first time unit. \square

Before we proceed to define the prioritized transition relation of PADS let us draw some motivation from the example above. We may note that these unprioritized transition systems include some unexpected and even undesirable behaviors. For example,

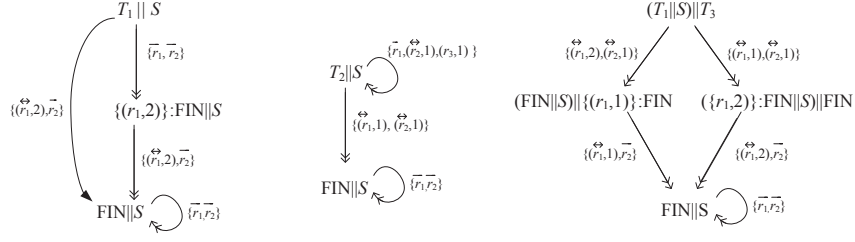


Figure 1: The unprioritized transition systems of $T_1 \parallel S$, $T_2 \parallel S$ and $(T_1 \parallel S) \parallel T_3$

consider task T_1 . Our intention in writing this process is to express that T_1 requests resource r_1 during the first or the second time unit. More precisely, if r_1 is available during the first time unit, then T_1 should employ it and, if not, then it should idle and reiterate its request during the second time unit. However, when placing T_1 in parallel with a process like S that offers r_1 (and r_2) immediately, we observe that $T_1 \parallel S \xrightarrow{\{\bar{r}_1, \bar{r}_2\}} \{(r_1, 2)\} : \text{FIN} \parallel S$, i.e. the semantics allow for T_1 to choose its second alternative of idling instead of employing the available resource, contrary to our intention. Furthermore, consider $T_2 \parallel S$. Again, here we observe that, contrary to what one might expect, the process may choose to execute its action $\{(r_2, 1), (r_3, 1)\}$, thus iterating its resource request for r_3 , instead of consuming the available r_1 and r_2 . Finally, in the transition system of $(T_1 \parallel S) \parallel T_3$, we observe that the initial state enables two transitions whose actions contain the same resources but with one having higher priorities than the other: a treatment of priority is needed to ensure that higher-priority actions take precedence over lower-priority ones.

In order to capture the intended behavior of systems, as discussed above, we define a preemption relation on actions that prunes away undesirable behaviors. This preemption relation focuses on nondeterminism within tasks and it ensures that it is resolved based on the priorities of the resource requests and the following two assumptions:

1. Given a supply, a task should respond “angelically” and, given a nondeterministic set of enabled transitions, it should choose only between the ones that are satisfied by the available supply, assuming that such options exist. For example, $T_2 \parallel S$ above should retain only transition $\{(\overset{\leftrightarrow}{r}_1, 1), (\overset{\leftrightarrow}{r}_2, 1)\}$ in its initial state.
2. In addition, we assume that a task behaves greedily and, at each step, it employs as many of the supplied resources as possible. For example, the composition $T_1 \parallel S$ above should only retain transition $\{(\overset{\leftrightarrow}{r}_1, 2), \bar{r}_2\}$ in its initial state.

Given the above, we define the preemption relation as follows:

Definition 2.2. We define the preemption relation $\prec \in \text{Act} \times \text{Act}$ so that $\alpha \prec \beta$ if one of the following holds:

1. $\{r | \bar{r} \in \alpha^b \text{ or } \overset{\leftrightarrow}{r} \in \alpha^b\} = \{r | \bar{r} \in \beta^b \text{ or } \overset{\leftrightarrow}{r} \in \beta^b\}$, $\alpha^b \cap R \neq \emptyset$ and $\beta^b \cap R = \emptyset$, that is, α and β use the same consumed and offered resources and α contains some additional resource requests whereas β does not.

2. $\text{res}(\alpha) = \text{res}(\beta)$, $\alpha^b \cap R = \beta^b \cap R = \emptyset$ and $\{r \mid \overset{\leftrightarrow}{r} \in \alpha^b\} \subset \{r \mid \overset{\leftrightarrow}{r} \in \beta^b\}$, that is, α and β involve the same resources, neither of them makes any resources requests, but β consumes more resources than α .
3. $\alpha^b = \beta^b$, for all $r \in \text{res}(\alpha)$ $\pi_\alpha(r) \leq \pi_\beta(r)$, and there exists $r \in \text{res}(\alpha)$, $\pi_\alpha(r) < \pi_\beta(r)$, that is, α and β contain the same resources with β giving greater or equal priority to all resource usages, and there exists at least one resource which is associated with a strictly greater priority in β than in α .

Intuitively, an action precludes another if it makes better usage of the same offered resources: According to clause (1), an action that involves no resource requests for an available resource supply preempts an action that makes further requests given the same supply which implies that tasks should behave in an ‘‘angelic’’ manner according to the first assumption above. According to clause (2), given a resource supply as much resource should be consumed as possible, thus tasks behave greedily according to the second assumption above. And, finally, the third clause implements our treatment of priority: if two resources contain exactly the same resources and in the same mode (request, grant or consume) then β preempts α if each resource is employed by β at a priority higher than or equal to α , with at least one resource being implemented at a higher priority.

Note that preemption takes place between two actions only if they contain the same consumed and offered resources. For example, $\{(\overset{\leftrightarrow}{r_1}, 2), (\overline{r_2}, 1)\} < \{(\overset{\leftrightarrow}{r_1}, 1), (\overset{\leftrightarrow}{r_2}, 1)\}$ but $\{(\overset{\leftrightarrow}{r_1}, 1), (\overline{r_2}, 1)\} \not< \{(\overset{\leftrightarrow}{r_1}, 2)\}$ and $\{(\overset{\leftrightarrow}{r_1}, 1), (\overline{r_2}, 1), (r_3, 1)\} \not< \{(\overset{\leftrightarrow}{r_1}, 2)\}$. In other words, our semantics makes an asymmetric treatment between resource requests and resource supplies and, consequently, between task and supply processes. Intuitively, this asymmetry captures the understanding that while supplies control their nondeterminism and may choose to offer any one of their available actions, tasks respond to the supply available and resolve their nondeterminism based on the environment.

We may now define the prioritized transition relation $\overset{\alpha}{\rightarrow}$ by the following rule:

$$\boxed{\frac{P \overset{\alpha}{\rightarrow} Q}{P \overset{\alpha}{\rightarrow} Q}, \text{ there is no } P \overset{\beta}{\rightarrow}, \alpha < \beta}$$

Figure 2 presents the refined versions of the transition systems in Figure 1 after preemption is implemented.

We conclude this section by introducing some notations. We write $P \longrightarrow$ if there exists α such that $P \overset{\alpha}{\rightarrow}$. If $P \not\overset{\alpha}{\rightarrow}$ for all actions α , we write $P = \delta$, where δ is the deadlocked process. We write $P \Longrightarrow P'$ if there exist $\alpha_1, \dots, \alpha_n$ and P_1, \dots, P_n , $n \geq 1$, such that $P \overset{\alpha_1}{\rightarrow} P_1 \overset{\alpha_2}{\rightarrow} \dots P_{n-1} \overset{\alpha_n}{\rightarrow} P_n = P'$. The set of traces of P , $\text{traces}(P)$, is defined to be the set of all infinite sequences $\alpha_1^b \alpha_2^b \dots$ such that $P \overset{\alpha_1}{\rightarrow} P_1 \overset{\alpha_2}{\rightarrow} \dots$. Furthermore, we write κ for elements of 2^R and $\overline{\kappa}$ to transform all resource requests in κ into resource grants, so, $\overline{\{r_1, r_2\}} = \{\overline{r_1}, \overline{r_2}\}$. Extending this notation to traces of the form $w = \kappa_1 \kappa_2 \dots$, we write \overline{w} for $\overline{\kappa_1} \overline{\kappa_2} \dots$. Finally, given $\alpha \in \text{Act}_G$, we write α^\natural to transform all resource grants into resource requests, so, $\{\overline{r_1}, \overline{r_2}\}^\natural = \{r_1, r_2\}$.

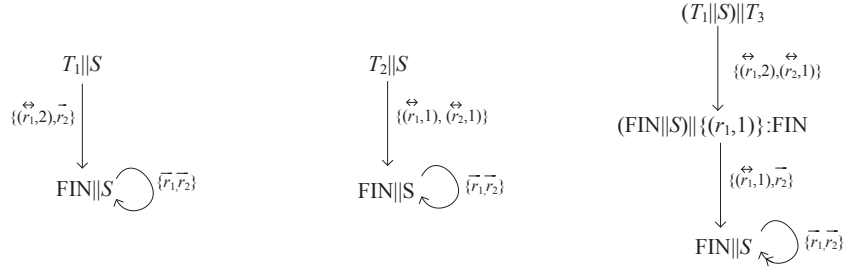


Figure 2: The prioritized transition systems of $T_1||S$, $T_2||S$ and $(T_1||S)||T_3$ from Example 2.1

3. Schedulability

In this section we present a theory of schedulability for our calculus. We begin by defining when a set of tasks is considered to be schedulable by a supply. Then we present an alternative characterization based on a type of simulation relation and we prove the two definitions to be equivalent. In what follows we write T^* for the set containing all processes of the form $T_1||\dots||T_n$, $n \geq 1$, and S^* for the set containing all processes of the form $S_1||\dots||S_n$, $n \geq 1$. For simplicity, we refer to elements of T^* and S^* simply as tasks and supplies, respectively.

Definition 3.1. A task $T \in T^*$ is schedulable by supply $S \in S^*$ if whenever $T||S \Longrightarrow P$ then (i) $P \longrightarrow$ and (ii) for all $P \xrightarrow{\alpha}$ we have $\alpha^b \cap R = \emptyset$.

According to this definition, a task T is schedulable by supply S if at no point during their interaction does the system deadlock (clause (i)) and, moreover, no request for a resource remains unsatisfied (clause (ii)).

Example 3.2. Let

$$\begin{array}{ll}
S_1 \stackrel{\text{def}}{=} \{\bar{r}_2\} : \text{FIN} & T_1 \stackrel{\text{def}}{=} \{(r_1, 1)\} : \text{FIN} \\
S_2 \stackrel{\text{def}}{=} \{\bar{r}_1, \bar{r}_2\} : \text{FIN} & T_2 \stackrel{\text{def}}{=} \{(r_1, 1)\} : \text{FIN} + \{(r_2, 1)\} : \text{FIN} \\
S_3 \stackrel{\text{def}}{=} \{\bar{r}_1\} : \text{FIN} + \{\bar{r}_2\} : \text{FIN} & T_3 \stackrel{\text{def}}{=} \{(r_2, 1)\} : \text{FIN}
\end{array}$$

Consider T_1 . We observe that T_1 is not schedulable by S_1 since $T_1||S_1 \xrightarrow{\{(r_1,1), \bar{r}_2\}}$. Clearly, this is so because S_1 does not offer r_1 as required by T_1 , while S_2 , by offering simultaneously r_1 and r_2 , schedules T_1 as shown in the transition $T_1||S_2 \xrightarrow{\{(\bar{r}_1,1), \bar{r}_2\}}$ $\text{FIN}||\text{FIN}$. However, this is not the case for supply S_3 : although it offers both r_1 and r_2 during the first time unit, it does so in two distinct actions. If the nondeterminism is resolved according to the first summand, we obtain $T_1||S_3 \xrightarrow{\{(r_1,1), \bar{r}_2\}}$ $\text{FIN}||\text{FIN}$ (note that $\{(r_1, 1), \bar{r}_2\}$ and $\{(\bar{r}_1, 1)\}$ are incomparable by \prec thus both actions are enabled in $T_1||S_3$).

Moving on to task T_2 we observe that this is schedulable by all three supplies. In particular, $T_2 \parallel S_1 \xrightarrow{\{(r_2, 1)\}} \text{FIN} \parallel \text{FIN}$ and this is the only transition of $T_2 \parallel S_1$ since $\{\bar{r}_2, (r_1, 1)\} \prec \{(r_2, 1)\}$. Finally, T_3 is schedulable by S_1 and S_2 but not S_3 . \square

Following this example we can make a number of observations regarding the defined notion of schedulability. Regarding supplies, we note that adding resources to the actions of a supply (as S_2 introduces r_2 in the action of supply S_1) appears to increase the supply's ability to schedule tasks since this implies that more resources are offered (S_2 schedules task T_1 whereas S_1 cannot). However, introducing nondeterministic alternatives in a supply reduces this ability; for example S_1 schedules T_3 but S_3 does not. The opposite holds for tasks: extending the actions of a task with resources decreases its ability to be scheduled by a supply since this implies that more resources are required, while extending a task with nondeterministic alternatives increases its ability to be schedulable since additional alternatives instill greater flexibility for the task to execute (T_2 is schedulable by S_3 unlike T_1 and T_3). These observations will be further studied and formalized in Section 4.

We now continue to provide an alternative characterization of schedulability via a type of simulation relations. This definition highlights the conditions under which a task is schedulable by a supply as well as the asymmetry between tasks and supplies discussed above. Before moving on to this definition we introduce some useful notations and results:

Definition 3.3. *Let $\alpha, \beta \in \text{Act}$.*

- *We write $\text{sat}(\beta, \alpha)$ if $\text{res}(\beta) \subseteq \text{res}(\alpha)$. In the case of $\beta \in \text{Act}_R$ and $\alpha \in \text{Act}_G$, we say that request action β is satisfied by grant action α .*
- *For a system P , we write $\beta \trianglelefteq_P \alpha$ if $\text{sat}(\beta, \alpha)$ and there exists no $\gamma \in \text{Act}$ such that $P \xrightarrow{\gamma} P'$, $\text{sat}(\gamma, \alpha)$ and either $\beta^b \subset \gamma^b$ or $\beta^b = \gamma^b$ and $\beta \prec \gamma$. If $\beta \trianglelefteq_P \alpha$ we say that β is a maximal response of P with respect to α .*

Note that, given a resource grant by some supply S , only maximal responses of a task T are relevant responses to the supply. This is because, in the parallel composition of $T \parallel S$, any non-maximal responses will be pruned away by the preemption relation and thus they can be ignored. For example, if $T \stackrel{\text{def}}{=} \{(r_1, 2), (r_2, 1)\} : T_1 + \{(r_1, 3)\} : T_2$ and $S \stackrel{\text{def}}{=} \{\bar{r}_1, \bar{r}_2\} : S'$, we have $\{(r_1, 2), (r_2, 1)\} \trianglelefteq_T \{\bar{r}_1, \bar{r}_2\}$. We may in fact prove that:

Lemma 3.4. *For any $T \in \mathbb{T}^*$, $S \in \mathbb{S}^*$,*

1. *$T \parallel S \xrightarrow{\alpha} T' \parallel S'$ with $\alpha^b \cap R = \emptyset$ if and only if $T \xrightarrow{\alpha_1} T'$, $S \xrightarrow{\alpha_2} S'$, $\alpha = \alpha_1 \oplus \alpha_2$ and $\alpha_1 \trianglelefteq_T \alpha_2$.*
2. *Suppose $T \parallel S \xrightarrow{\alpha} T' \parallel S'$, where $T \xrightarrow{\alpha_1} T'$, $S \xrightarrow{\alpha_2} S'$ and $\alpha = \alpha_1 \oplus \alpha_2$, and, furthermore, there exists β , $\text{sat}(\beta, \alpha_2)$ with $T \xrightarrow{\beta} T''$. Then $\alpha_1 \trianglelefteq_T \alpha_2$.*

PROOF: For the first item of the lemma, consider $T \in \mathbb{T}^*$, $S \in \mathbb{S}^*$, such that $T \parallel S \xrightarrow{\alpha} T' \parallel S'$ with $\alpha^b \cap R = \emptyset$. Then it must be that for some $\alpha_1 \in \text{Act}_R$ and $\alpha_2 \in \text{Act}_G$,

$T \xrightarrow{\alpha_1} T', S \xrightarrow{\alpha_2} S'$ with $\alpha = \alpha_1 \oplus \alpha_2$. Since $\alpha^b \cap R = \emptyset$ it must be that $\text{sat}(\alpha_1, \alpha_2)$. Suppose that there exists γ with $\text{sat}(\gamma, \alpha_2)$ such that either $\alpha_2^b \subset \gamma^b$ or $\beta^b = \gamma^b$ and $\beta \prec \gamma$. In both cases we may see that $\alpha_1 \oplus \alpha_2 \prec \gamma \oplus \alpha_2$ which contradicts the existence of transition $T \parallel S \xrightarrow{\alpha}$. This implies that $\alpha_1 \trianglelefteq_T \alpha_2$ as required. The other direction of the property can be established using similar arguments.

For the second item of the lemma, suppose $T \parallel S \xrightarrow{\alpha} T' \parallel S'$, where $T \xrightarrow{\alpha_1} T', S \xrightarrow{\alpha_2} S'$, and $\alpha = \alpha_1 \oplus \alpha_2$, and, furthermore, there exists β , $\text{sat}(\beta, \alpha_2)$ with $T \xrightarrow{\beta} T''$. Suppose that $\alpha^b \cap R \neq \emptyset$. Then $\alpha \prec \beta \oplus \alpha_2$ which contradicts the existence of the transition $S \parallel T \xrightarrow{\alpha} S' \parallel T'$. Thus, $\alpha^b \cap R = \emptyset$, and by the first part of the lemma $\alpha_1 \trianglelefteq_T \alpha_2$. \square

Definition 3.5. A relation $\mathcal{S} \subseteq T^* \times S^*$ is a supply simulation relation if for all $(T, S) \in \mathcal{S}$, $S \longrightarrow$, and if $S \xrightarrow{\alpha} S'$ then

1. there exists $T \xrightarrow{\beta} T'$ with $\text{sat}(\beta, \alpha)$ and $(T', S') \in \mathcal{S}$, and
2. whenever $T \xrightarrow{\beta} T'$ with $\beta \trianglelefteq_T \alpha$, then $(T', S') \in \mathcal{S}$.

If there exists a supply simulation relation between T and S , then we write $S \models T$.

That is, a task and a supply are related by a supply simulation relation if (i) the supply is able to offer resources to the task ($S \longrightarrow$), (ii) if a supply offers a set of resources then the task will be able to respond by an action that is satisfied by the available supply and to remain schedulable by the resulting state of the supply (clause 1), and (iii) given a set of resources offered by the supply, any maximal transition by which the task can accept the offered supply will result in a state that remains schedulable by the remaining supply (clause 2). Recall that, according to Lemma 3.4(2), only maximal transitions of T with respect to some supply are relevant responses, all other transitions are pruned away by the preemption relation and can thus be ignored.

We may now prove that the two alternative schedulability notions coincide.

Lemma 3.6. A task $T \in T^*$ is schedulable by supply $S \in S^*$ if and only if $S \models T$.

PROOF: To begin with, suppose there exists a supply simulation relation \mathcal{R} between T and S . We will show that if $T \parallel S \xrightarrow{\alpha} T' \parallel S'$ then (i) $\alpha^b \cap R = \emptyset$, (ii) $(T', S') \in \mathcal{R}$ and (iii) $T' \parallel S' \longrightarrow$. Then, by induction on the length of the transition of $T \parallel S \Longrightarrow P$, we may deduce that T is schedulable by S , according to Definition 3.1.

So suppose that $T \parallel S \xrightarrow{\alpha} T' \parallel S'$ where $T \xrightarrow{\alpha_1} T'$ and $S \xrightarrow{\alpha_2} S'$, $\alpha = \alpha_1 \oplus \alpha_2$.

We know that for some β , $\text{sat}(\beta, \alpha_2)$, $T \xrightarrow{\beta} T''$ (Definition 3.5(1)). By Lemma 3.4(2) this implies that $\alpha_1 \trianglelefteq_T \alpha_2$, thus by Definition 3.5 $(T', S') \in \mathcal{R}$. Furthermore, by Lemma 3.4(1) we have that $\alpha^b \cap R = \emptyset$. Finally, since $(T', S') \in \mathcal{R}$, by Definition 3.5 we have that $S' \longrightarrow$ and for each $S' \xrightarrow{\beta_1}$ there exists $T' \xrightarrow{\beta_2}$ with $\text{sat}(\beta_2, \beta_1)$. This implies that $S' \parallel T' \longrightarrow$ which completes the first part of the proof.

Conversely, suppose that task T is schedulable by supply S . We will show that

$$\mathcal{R} = \{(T, S) \mid T \text{ is schedulable by } S\}$$

is a supply simulation relation. Suppose $(T, S) \in \mathcal{R}$. Since $T \parallel S \longrightarrow$, $T \longrightarrow$ and $S \longrightarrow$. Furthermore, if $S \xrightarrow{\alpha} S'$ then, since T is schedulable by S , there exists

$T \xrightarrow{\beta} T'$, $\text{sat}(\beta, \alpha)$. If not, that is for all $T \xrightarrow{\beta} T''$, $\text{res}(\beta) - \text{res}(\alpha) \neq \emptyset$, then $T \parallel S \xrightarrow{\gamma} \gamma^b \cap R \neq \emptyset$ which contradicts our assumption of T being schedulable by S . Next, suppose that $T \xrightarrow{\beta} T'$ and $\beta \triangleleft_T \alpha$. Since $S \xrightarrow{\alpha} S'$ and $T \xrightarrow{\beta} T'$ where $\beta \triangleleft_T \alpha$, by Lemma 3.4(1) $T \parallel S \xrightarrow{\alpha \oplus \beta} T' \parallel S'$, where T' is schedulable by S' , which implies that $(T', S') \in \mathcal{R}$, as required. \square

We define when a task is schedulable and this is done in the following obvious way.

Definition 3.7. A task $T \in \mathsf{T}^*$ is schedulable if there exists a supply S with $S \models T$.

We observe that the crux of the schedulability of a task by a supply lies in the ability of the task to operate acceptably for all possible actions of the supply and in doing so in all its enabled nondeterministic executions that can take place as a response to the supply available. The notion of a cylinder defined below is intended to capture all the possible ways in which a task can respond given an execution of a supply.

Definition 3.8. Given a task $T \in \mathsf{T}^*$ and an infinite trace $w = \kappa_1 \kappa_2 \dots$, with $\kappa_i \subset \mathbb{R}$ for all i , we define the w -cylinder of T to be the set $A = \cup_{i \geq 1} A_i$, where

$$\begin{aligned} A_1 &= \{(T, \alpha_1, P_1) \mid T \xrightarrow{\alpha_1} P_1, \alpha_1 \triangleleft_T \overline{\kappa_1}\} \\ A_i &= \{(P_i, \beta_i, P'_i) \mid P_i \xrightarrow{\beta_i} P'_i, \beta_i \triangleleft_{P_i} \overline{\kappa_i}, \exists (Q, \gamma, P_i) \in A_{i-1}\}, \quad i > 1 \end{aligned}$$

Furthermore, we say that an w -cylinder $A = \cup_{i \geq 1} A_i$ is live if (i) for all $(Q, \alpha, Q') \in A$ then $Q \longrightarrow$, (ii) $A_i \neq \emptyset$ for all i and (3) $\bigcup_{(P, \beta, Q) \in A_i} \beta^b = \kappa_i$.

Thus, a w -cylinder, where $w = \kappa_1 \kappa_2 \dots$, of a task T contains all the possible/maximal responses of T given the supply $\overline{\kappa_1}$ (set A_1), all possible responses of the resulting states given supply $\overline{\kappa_2}$ (set A_2), and so on. For example, consider task T where

$$\begin{aligned} T &\stackrel{\text{def}}{=} \{(r_1, 2)\} : T' \\ T' &\stackrel{\text{def}}{=} \{(r_1, 1)\} : \text{FIN} + \{(r_2, 1)\} : \text{FIN} + \{(r_3, 1)\} : \text{FIN} \end{aligned}$$

For $w = \{r_1, r_2\} \{r_2, r_3\} \emptyset^*$, the w -cylinder of T is $A = \cup_{i \geq 1} A_i$, where

$$\begin{aligned} A_1 &= \{(T, \{(r_1, 2)\}, T')\} \\ A_2 &= \{(T', \{(r_2, 1)\}, \text{FIN}), (T', \{(r_3, 1)\}, \text{FIN})\} \\ A_i &= \{(\text{FIN}, \emptyset, \text{FIN})\}, \quad i \geq 3 \end{aligned}$$

We observe that these are indeed the transitions that will be relevant when scheduling T by a supply of the form $S \stackrel{\text{def}}{=} \{\overline{r_1}, \overline{r_2}\} \{\overline{r_2}, \overline{r_3}\} : S'$. The following result relates live cylinders with task schedulability.

Lemma 3.9. A task $T \in \mathsf{T}^*$ is schedulable if and only if it possesses a live cylinder.

PROOF: Suppose T has a live w -cylinder where $w = \kappa_1 \kappa_2 \dots$. Consider supply S_0 defined by the following set of equations $S_i \stackrel{\text{def}}{=} \overline{\kappa_{i+1}} : S_{i+1}$. Then, we may confirm that $S_0 \models T$. In particular we show that if $A = \cup_{i \geq 1} A_i$ is the w -cylinder of T , then

$$\mathcal{R} = \{(T, S_i) \mid (T, \beta, Q) \in A_i, i \geq 1\}$$

is a supply relation. So, consider $(T, S_i) \in \mathcal{R}$. To begin with, trivially $S_i \longrightarrow$. Further, if $S_i \xrightarrow{\alpha} S_{i+1}$, then since $A_i \neq \emptyset$, there exists $T \xrightarrow{\beta} Q$, $\beta \triangleleft_T \alpha$, and $(Q, S_{i+1}) \in \mathcal{R}$. In fact, this holds for all $T \xrightarrow{\beta} Q$, where $\beta \triangleleft_{T_i} \alpha$ and the result follows.

On the other hand, if T is schedulable, then there exists a supply S that schedules it. Let $w = \overline{\kappa_1 \kappa_2} \dots \in \text{traces}(S)$. We may construct a cylinder $A = \cup_{i \geq 1} A_i$ of T as

$$\begin{aligned} A_1 &= \{(T, \alpha_1, P) \mid T \xrightarrow{\alpha_1} P, \alpha_1 \triangleleft_T \overline{\kappa_1}\} \\ A_i &= \{(P, \beta_i, P') \mid P \xrightarrow{\beta_i} P', \beta_i \triangleleft_P \overline{\kappa_i}, (Q, \gamma, P) \in A_{i-1}\}, i > 1 \end{aligned}$$

Since T is schedulable by S it is straightforward to see that A contains no triple of the form (Q, α, δ) and also that $A_i \neq \emptyset$ for all i . Finally, if we take $\beta_i = \bigcup_{(P, \beta, Q) \in A_i} \beta^b$, we may conclude that $A = \cup_{i \geq 1} A_i$ is a w' -cylinder of T , where $w' = \beta_1 \beta_2 \dots$. \square

3.1. Matching Supplies to Tasks

In this section we focus our attention to the problem of collecting the resource requirements of a task into a matching supply. Specifically, given a task, we would like to generate a supply process which schedules the task and at the same time is optimal in that (1) it does not reserve more resources than those required by the task and (2) it provides resource assignments to capture all possible ways in which the task can be scheduled. Both of these properties are important during the compositional scheduling of real-time tasks. The first property is clearly desirable since conservation of resources becomes critical when real-time components are composed. For the second property, we observe that capturing all possible ways of scheduling a task gives greater flexibility when one tries to compositionally schedule a set of tasks where the challenge is to share the resources between the tasks in ways that are acceptable to each one of them.

We begin by defining a function for combining supplies. This is helpful for a subsequent definition that considers matching supplies to tasks.

Definition 3.10. Given supplies $S_1, S_2 \in \mathcal{S}$ we define $S_1 \otimes S_2$ as

$$S_1 \otimes S_2 = \begin{cases} S_1 & \text{if } S_2 = \text{FIN} \\ S_2 & \text{if } S_1 = \text{FIN} \\ \sum_{i \in I} \sum_{j \in J} \alpha_i \cup \beta_j : (\bigotimes_{k \in I, \alpha_k \triangleleft_{S_1} \alpha_i \cup \beta_j} P_k \otimes \bigotimes_{l \in J, \beta_l \triangleleft_{S_2} \alpha_i \cup \beta_j} Q_l) & \\ & \text{if } S_1 \stackrel{\text{def}}{=} \sum_{i \in I} \alpha_i : P_i \text{ and } S_2 \stackrel{\text{def}}{=} \sum_{j \in J} \beta_j : Q_j \end{cases}$$

Essentially, the joined supply $S_1 \otimes S_2$ joins together the various summands of the individual supplies as follows: in its topmost summand it unites all available grants of S_1 with all available grants of S_2 , while the continuation process consists of the join of those continuations of S_1 and S_2 which appear after ‘‘maximal’’ subsets of the initial action in question. For example we have:

$$\begin{aligned}
\emptyset : \{\bar{r}\} : \emptyset : \text{FIN} \otimes \emptyset : \emptyset : \{\bar{r}\} : \text{FIN} &= \emptyset : \{\bar{r}\} : \{\bar{r}\} : \text{FIN} \\
\emptyset : \{\bar{r}\} : \emptyset : \text{FIN} \otimes (\emptyset : \emptyset : \{\bar{r}\} : \text{FIN} + \{\bar{r}\} : \emptyset : \emptyset : \text{FIN}) & \\
&= \emptyset : \{\bar{r}\} : \{\bar{r}\} : \text{FIN} + \{\bar{r}\} : \{\bar{r}\} : \emptyset : \text{FIN}
\end{aligned}$$

Using this definition we now move to define the *demand* of a task. The demand of a task is intended to capture the optimal supply that can schedule a task in the sense we have already discussed. The main point to note in this definition is that we combine all same-prefixed nondeterministic choices of a task by a singly-prefixed supply.

Definition 3.11. *Given a task $T \in \mathbb{T}^*$, we define its demand as the following element of \mathbb{S} :*

$$\text{demand}(T) \stackrel{\text{def}}{=} \sum_{\alpha: T \xrightarrow{\alpha}} \overline{\alpha^b}: [\bigotimes_{T': T \xrightarrow{\alpha} T'} \text{demand}(T')]$$

Example 3.12. Consider tasks

$$\begin{aligned}
T_1 &= \{(r, 2)\} : \emptyset : \emptyset : T_1 + \emptyset : \{(r, 1)\} : \emptyset : T_1 + \emptyset : \emptyset : \{(r, 3)\} : T_1 \\
T_2 &= \{(r, 1)\} : \emptyset : \emptyset : T_2 + \emptyset : (\{(r, 2)\} : \emptyset : T_2 + \emptyset : \{(r, 2)\} : T_2) \\
T_3 &= \{(r, 1)\} : \{(r, 1)\} : \text{FIN} + \{(r, 2)\} : \emptyset : T_3
\end{aligned}$$

Their demands are given by X_1, X_2, X_3 below, respectively.

$$\begin{aligned}
X_1 &= \{\bar{r}\} : \emptyset : \emptyset : X_1 + \emptyset : \{\bar{r}\} : \{\bar{r}\} : X_1 \\
X_2 &= \{\bar{r}\} : \emptyset : \emptyset : X_2 + \emptyset : (\{\bar{r}\} : \emptyset : X_2 + \emptyset : \{\bar{r}\} : X_2) \\
X_3 &= \{\bar{r}\} : \emptyset : X_3
\end{aligned}$$

□

The next lemma considers the optimality of $\text{demand}(T)$ following the requirements posed at the beginning of this section. We write $w \leq w'$ for the infinite traces $w = \alpha_1 \alpha_2 \dots$ and $w' = \beta_1 \beta_2 \dots$, if $\alpha_j \subseteq \beta_j$ for all $j \geq 1$.

Lemma 3.13. *If $w \in \text{traces}(\text{demand}(T))$ then T possesses a live w -cylinder and if $w \in \text{traces}(T)$ then there exists $w' \in \text{traces}(\text{demand}(T))$ such that $\bar{w} \leq w'$.*

PROOF: Suppose $\text{demand}(T) \xrightarrow{\alpha_1} S_1 \xrightarrow{\alpha_2} S_2 \xrightarrow{\alpha_3} \dots$. We will show that for the w -cylinder $A = \cup_{i \geq 1} A_i$ of T , where $w = \alpha_1^{\sharp} \alpha_2^{\sharp} \dots$, we have $S_i = \bigotimes_{(P, \beta, Q) \in A_i} \text{demand}(Q)$ and A is live. Consider an arbitrary S_i and suppose $S_i = \bigotimes_{(P, \beta, Q) \in A_i} \text{demand}(Q)$ where $A_i \neq \emptyset$ and A_i does not contain elements of the form (P, β, δ) . Then, since $S_i \xrightarrow{\alpha_{i+1}}$, by the definition of \otimes , it must be that

$$\alpha_{i+1} = \bigcup \{ \alpha \mid (P, \beta, Q) \in A_i, \text{demand}(Q) \xrightarrow{\alpha} \}.$$

In addition, $S_i \xrightarrow{\alpha_{i+1}} \bigotimes_{T' \in B} \text{demand}(T')$, $B = \{T' \mid (P, \beta, Q) \in A_i, Q \xrightarrow{\beta} T', \beta \sqsubseteq_Q \alpha_{i+1}\}$. But, $B = A_{i+1}$ and by the construction of α_{i+1} , $A_{i+1} \neq \emptyset$ and A is live, which completes the first part of the proof.

To establish the second part of the proof it is sufficient to note that if $T \xrightarrow{\alpha} T'$ then $\text{demand}(T) \xrightarrow{\bar{\alpha}} \text{demand}(T') \otimes S$ for some S and, further, if $S_1 \xrightarrow{\alpha} S'_1$ then $S_1 \otimes S_2 \xrightarrow{\alpha'} S'_1 \otimes S'_2$, where $\alpha \subseteq \alpha'$ for some S'_2 . Then, by the definition of demand, it is easy to see that if $T \xrightarrow{\alpha_1} T_1 \xrightarrow{\alpha_2} T_2 \xrightarrow{\alpha_3} \dots$, then $\text{demand}(T) \xrightarrow{\beta_1} \text{demand}(T_1) \otimes S_1 \xrightarrow{\beta_2} \text{demand}(T_2) \otimes S_2 \xrightarrow{\beta_3} \dots$, where $\bar{\alpha}_1 \bar{\alpha}_2 \dots \leq \beta_1 \beta_2 \dots$. \square

Thus, we may conclude that a task T is schedulable by $\text{demand}(T)$. Furthermore, $\text{demand}(T)$ is an optimal supply for T since each of its executions schedules exactly a cylinder of T , i.e. it offers exactly the resources necessary for scheduling the cylinder, and each possible schedule of T is captured by an execution of $\text{demand}(T)$.

3.2. Compositional Theory

We proceed to consider the schedulability problem of a set of task components. The first issue we tackle is the compositionality problem: If a component T_1 is schedulable by S_1 and an independent component T_2 by S_2 can we combine S_1 and S_2 into a collective supply that schedules $T_1 \parallel T_2$? We begin by noting a subtlety pertaining to this problem which we need to consider before answering it. Consider the two components below each consisting of one task:

$$\begin{aligned} T_1 &= \{(r, 1)\}:\emptyset:\text{FIN} + \emptyset:\{(r, 1)\}:\text{FIN} \\ T_2 &= \{(r, 1)\}:\emptyset:\text{FIN} + \emptyset:\{(r, 1)\}:\{(r, 1)\}:\text{FIN} \end{aligned}$$

These components are schedulable by supplies $S_1 = \emptyset:\{\bar{r}\}:\text{FIN}$ and $S_2 = \{\bar{r}\}:\emptyset:\text{FIN}$, respectively. That is, it is sufficient for component T_1 to obtain resource r during the second time unit and for component T_2 during the first time unit. However, a supply $S = \{\bar{r}\}:\{\bar{r}\}:\text{FIN}$, offering r during both time units, fails to schedule $T_1 \parallel T_2$. This is due to the fact that the supply for resource r during the first time unit is intended for component T_2 but may be consumed by component T_1 leading to a deadlock of the system during the third time unit. Moreover, if T_1 employed its resources at priority level 2, this would in fact be destined to happen.

To resolve this issue, we associate components with their matching supplies by annotating each resource reference by a number which distinguishes the component in which the resource is employed/supplied. Precisely, we assume that each component is associated with a component identifier and if resource r is requested by a component with identifier i we write $r[i]$ for the request and, similarly, if a supply of r is intended for the component with identifier i we write $\bar{r}[i]$ for the supply. So, we say that component $\{(r[1], 1)\}:\text{FIN}$ is schedulable by supply $\{\bar{r}[1]\}:\text{FIN}$ and component $\{(r[2], 3)\}:\text{FIN}$ by supply $\{\bar{r}[2]\}:\text{FIN}$. However, note that resources $r[1]$ and $r[2]$ do refer to the same resource and for all other purposes should be treated as the same. So, for example, $\{r[1]\} \cap \{r[2]\} \neq \emptyset$. To model this precisely we write:

- $P[i]$ for the process P with all its resources r renamed as $r[i]$ (and, thus, $\text{demand}(P)[i]$ for the process $\text{demand}(P)$ with all its resources r renamed as $r[i]$).
- $\alpha \cap_{\mathbb{R}} \beta$ for $\{r \in \mathbb{R} \mid r[i] \in \text{res}(\alpha) \text{ and } r[j] \in \text{res}(\beta)\}$.

Furthermore, we use the notation $\alpha[i] = \{r[i] \mid r \in \alpha\}$ and, if $w = \alpha_1\alpha_2\dots$, $w[i] = \alpha_1[i]\alpha_2[i]\dots$. We have the following result:

Lemma 3.14. *If T_1 is schedulable by S_1 , T_2 is schedulable by S_2 and $S_1\|S_2$ does not deadlock, then $T_1[1]\|T_2[2]$ is schedulable by $S_1[1]\|S_2[2]$.*

PROOF: We will show that \mathcal{R} , below, is a supply simulation relation.

$$\mathcal{R} = \{(T_1[1]\|T_2[2], S_1[1]\|S_2[2]) \mid S_1 \models T_1, S_2 \models T_2, S_1[1]\|S_2[2] \text{ does not deadlock}\}$$

Let $(T_1[1]\|T_2[2], S_1[1]\|S_2[2]) \in \mathcal{R}$. By the definition of \mathcal{R} , $S_1[1]\|S_2[2] \longrightarrow$. So consider $S_1[1]\|S_2[2] \xrightarrow{\alpha} S'_1[1]\|S'_2[2]$. It must be that $\alpha = \alpha_1[1] \oplus \alpha_2[2]$, where $S_1 \xrightarrow{\alpha_1} S'_1$, $S_2 \xrightarrow{\alpha_2} S'_2$ and $\alpha_1 \cap \alpha_2 = \emptyset$. Since $S_1 \models T_1$, $S_2 \models T_2$, we have $T_1 \xrightarrow{\beta_1} T'_1$, $S'_1 \models T'_1$, and similarly $T_2 \xrightarrow{\beta_2} T'_2$, $S'_2 \models T'_2$. In fact, for all $T_1 \xrightarrow{\beta_1} T'_1$, $\beta_1 \trianglelefteq_{T_1} \alpha_1$, it holds that $S'_1 \models T'_1$, and for all $T_2 \xrightarrow{\beta_2} T'_2$, $\beta_2 \trianglelefteq_{T_2} \alpha_2$, it holds that $S'_2 \models T'_2$. This implies that for all $T_1[1]\|T_2[2] \xrightarrow{\beta} T'_1[1]\|T'_2[2]$, $\beta \trianglelefteq_{T_1[1]\|T_2[2]} \alpha$, $(T'_1[1]\|T'_2[2], S'_1[1]\|S'_2[2]) \in \mathcal{R}$ and there exists at least one such β -transition. This completes the proof. \square

However, note that even if $S_1\|S_2$ deadlocks, it is still possible that the supplies S_1 and S_2 can be combined to produce a supply for $T_1\|T_2$. In particular, we may suspect that every infinite trace of $S_1\|S_2$ is capable of scheduling $T_1\|T_2$, and in fact we can show that the part of the transition system that pertains to non-deadlocking behavior achieves exactly that. The following operator on supplies extracts this type of behavior.

Definition 3.15. *Given supplies S_1 and $S_2 \in \mathcal{S}$ we define their product $S_1 \times S_2$ by*

$$S_1 \times S_2 = \begin{cases} S_1 & \text{if } S_2 = \text{FIN} \\ S_2 & \text{if } S_1 = \text{FIN} \\ (\alpha \cup \beta):(S'_1 \times S'_2) & \text{if } S_1 = \alpha:S'_1, S_2 = \beta:S'_2, \alpha \cap_{\mathbb{R}} \beta = \emptyset, S'_1 \times S'_2 \neq \delta \\ \delta & \text{if } S_1 = \alpha:S'_1, S_2 = \beta:S'_2, \alpha \cap_{\mathbb{R}} \beta \neq \emptyset \text{ or } S'_1 \times S'_2 = \delta \\ \Sigma_{i \in I, j \in J} (S_1^i \times S_2^j) & \text{if } S_1 = \Sigma_{i \in I} S_1^i, S_2 = \Sigma_{j \in J} S_2^j \end{cases}$$

Note that the set of recursive equations used in the definition of $S_1 \times S_2$ may allow more than one solution. Consider, for example, $S_1 = \{\overline{r_1}\} : S_1$ and $S_2 = \{\overline{r_2}\} : S_2$. It is easy to see that $S_1 \times S_2 = \delta$ is a trivial solution. However, we are interested in the maximal solution to this set of equations, which in this case is $S_1 \times S_2 = \{\overline{r_1}, \overline{r_2}\} : S_1 \times S_2$. Intuitively, solutions can be ordered by the set of terms that are set to δ : the fewer terms are deadlocked, the “larger” the solution. We use the following lemma to make this notion precise and show that, for finite-state processes, the maximal solution exists and can be computed iteratively:

Lemma 3.16. *Given supplies S_1 and S_2 the set of equations which arise through $S_1 \times S_2$ has a greatest fixed point.*

PROOF: Consider the term $S_1 \times S_2$. Let $S_{S_1, S_2} = \{S \mid S_1 \Longrightarrow S \text{ or } S_2 \Longrightarrow S\}$ and $S_{S_1, S_2}^\times = S_{S_1, S_2} \cup [S_{S_1, S_2} \times S_{S_1, S_2}]$. That is, S_{S_1, S_2}^\times is a set containing all derivatives of S_1 or S_2 and all pairs of these derivatives. For finite-state processes, S_{S_1, S_2}^\times is finite.

Consider the set of relations on S_{S_1, S_2}^\times , $\mathcal{W} = \{W \mid W \subseteq S_{S_1, S_2}^\times \times S_{S_1, S_2}^\times\}$, ordered by set inclusion. Because \mathcal{W} is a powerset of a finite set, \mathcal{W} is a complete lattice of finite height.

Consider relation W where, if $(w_1, w_2) \in W$, then $w_1 \in [S_{S_1, S_2} \times S_{S_1, S_2}]$ and $w_2 \in S_{S_1, S_2}^\times$ appears on the right-hand side of the equation for w_1 , disregarding the recursive part of the third clause and the fourth clause, so that

- if $S_i = \text{FIN}$ then $(S_1 \times S_2, S_{3-i}) \in W$,
- if $S_1 = \alpha:S'_1, S_2 = \beta:S'_2$ and $(S_1 \times S_2, S'_1 \times S'_2) \in W$, then $\alpha \cap_{\mathbb{R}} \beta \neq \emptyset$,
- if $S_1 = \Sigma_{i \in I} S_1^i, S_2 = \Sigma_{j \in J} S_2^j$ and $(S_1 \times S_2, S'_1 \times S'_2) \in W$ then $S'_1 = S_1^i$ for some $i \in I$ and $S'_2 = S_2^j$ for some $j \in J$.

Thus, such a relation W relates a product $S_1 \times S_2$ with some of its possible derivatives according to the selected part of the definition. Further, suppose that whenever $w \in [S_{S_1, S_2} \times S_{S_1, S_2}]$ and there exists w_1 such that $(w_1, w) \in W$, then there also exists w_2 such that $(w, w_2) \in W$. Then, such W is a fixed point of the set of equations defining $S_1 \times S_2$. This is because, according to the complete definition, $S_1 \times S_2$ has some derivative w , if and only if w has a derivative (i.e. $w \neq \delta$).

Define a function $\mathcal{F} : \mathcal{W} \mapsto \mathcal{W}$ as $\mathcal{F}(W) = W - \{(w_1, w_2) \mid w_2 \in [S_{S_1, S_2} \times S_{S_1, S_2}] \wedge \forall w, (w_2, w) \notin W\}$. Since \mathcal{F} can only remove elements from W , $\mathcal{F}(W) \leq W$. Furthermore, if $W_1 \leq W_2$, then $\mathcal{F}(W_1) \leq \mathcal{F}(W_2)$; that is, \mathcal{F} is monotonic. Let us construct the set W_0 using the definition of $S_1 \times S_2$, again by omitting $S'_1 \times S'_2 \neq \delta$ from clause 3 and $S'_1 \times S'_2 = \delta$ from clause 4. Clearly, any fixed point of $S_1 \times S_2$, $W_{S_1 \times S_2}$ satisfies $W_{S_1 \times S_2} \leq W_0$ since fewer terms are set to δ in W_0 . Since \mathcal{W} is a complete lattice, by the Tarski-Knaster theorem, the greatest fixed point exists and is unique. Since the lattice is of finite height, the fixed point can be computed starting from W_0 and iteratively applying \mathcal{F} until the fixed point is reached. \square

It is easy to see that, if $S_1 \parallel S_2$ does not deadlock then $S_1 \times S_2 \neq \delta$. However, the opposite is not true. By the construction of \times , $S_1 \times S_2$ selects the part of the transition system of $S_1 \parallel S_2$ that does not lead to deadlocked states. For example, consider $S_1 \stackrel{\text{def}}{=} \{\bar{r}\}:\{\bar{r}\}:\text{FIN} + \emptyset:\{\bar{r}\}:\{\bar{r}\}:\text{FIN}$ and $S_2 \stackrel{\text{def}}{=} \emptyset:\{\bar{r}\}:\text{FIN} + \{\bar{r}\}:\emptyset:\text{FIN}$. Then, although $S_1 \parallel S_2 \xrightarrow{\{\bar{r}\}} \{\bar{r}\}:\text{FIN} \parallel \{\bar{r}\}:\text{FIN} = \delta$, $S_1 \times S_2 = \{\bar{r}\}:(\{\bar{r}\}:\{\bar{r}\}:\text{FIN} \times \emptyset:\text{FIN})$, and $(\{\bar{r}\}:\{\bar{r}\}:\text{FIN} \times \emptyset:\text{FIN}) = \{\bar{r}\}:\{\bar{r}\}:\text{FIN}$.

Lemma 3.17. *If T_1 is schedulable by S_1 , T_2 is schedulable by S_2 and $S_1 \times S_2 \neq \delta$, then $T_1[1] \parallel T_2[2]$ is schedulable by $S_1[1] \times S_2[2]$.*

PROOF: The proof is similar to that of Lemma 3.14. \square

At this point we turn our attention to the problem of constructing an interface for a set of mutually schedulable tasks. To do this, we employ the notion of demands and we prove the following:

Lemma 3.18. *If $w \in \text{traces}(T_1[1] \parallel T_2[2])$ then there exists a trace $w' \in \text{traces}(\text{demand}(T_1[1]) \times \text{demand}(T_2[2]))$ such that $\bar{w} \leq w'$.*

PROOF: Suppose that $w \in \text{traces}(T_1[1]||T_2[2])$. It is easy to see that $w[1]$ and $w[2]$ give rise to traces of $T_1[1]$ and $T_2[2]$. Then, by Lemma 3.13, there exist w_1 and w_2 such that $\overline{w[1]} \leq w_1$ and $\overline{w[2]} \leq w_2$ such that $w_1 \in \text{traces}(\text{demand}(T_1[1]))$ and $w_2 \in \text{traces}(\text{demand}(T_2[2]))$. Suppose that $w_1 = \alpha_1\alpha_2\dots$ and $w_2 = \beta_1\beta_2\dots$ and write $w' = \gamma_1\gamma_2\dots$, where $\gamma_i = \alpha_i \cup \beta_i$. Then, from the definition of \times we may conclude that w' a trace of $\text{demand}(T_1[1]) \times \text{demand}(T_2[2])$, and, in addition, $\overline{w} \leq w'$, as required. \square

This result implies that all alternatives of scheduling $T_1[1]||T_2[2]$ will be explored by $\text{demand}(T_1[1]) \times \text{demand}(T_2[2])$. It can be extended to the composition of an arbitrary number of tasks. We are now ready to present our main theorem:

Theorem 3.19. $T_1[1]||T_2[2]$ is schedulable if and only if $\text{demand}(T_1[1]) \times \text{demand}(T_2[2]) \neq \delta$. Moreover, if it is schedulable, then it is schedulable by $\text{demand}(T_1[1]) \times \text{demand}(T_2[2])$.

PROOF: Suppose $T_1[1]||T_2[2]$ is schedulable. Then, by Lemma 3.9, it has a live w -cylinder. Let w_1 be the trace of an execution of $T_1[1]||T_2[2]$ occurring within the cylinder. Then, by Lemma 3.18, there is a trace w_2 , $\overline{w_1} \leq w_2$ such that w_2 is a trace of $\text{demand}(T_1[1]) \times \text{demand}(T_2[2])$. This implies that $\text{demand}(T_1[1]) \times \text{demand}(T_2[2]) \neq \delta$. On the other hand, if $\text{demand}(T_1[1]) \times \text{demand}(T_2[2]) \neq \delta$, then, since, additionally, $\text{demand}(T_1[1])$ schedules $T_1[1]$ and $\text{demand}(T_2[2])$ schedules $T_2[2]$, then, by Lemma 3.17, $T_1[1]||T_2[2]$ is schedulable by $\text{demand}(T_1[1]) \times \text{demand}(T_2[2])$. \square

Based on this result we may determine the schedulability and a related scheduler for a set of tasks T_1, \dots, T_n , as follows: For each task, extract its demand and compute the products $D_1 = \text{demand}(T_1) \times \text{demand}(T_2)$, $D_2 = D_1 \times \text{demand}(T_3), \dots$. If this process does not reduce to some $D_i = \delta$ then the tasks are schedulable by D_{n-1} . Furthermore, according to Theorem 3.19, if they are indeed schedulable then $D_{n-1} \neq \delta$. Thus, this method is guaranteed to produce a schedule if one exists.

3.3. Examples

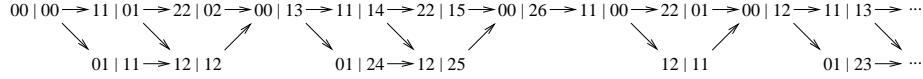
3.3.1. Scheduling periodic tasks

We first consider a simple periodic task with period p and execution time w which requires usage of some resource r , $Task_{w,p} = T_{0,0,w,p}$. This is defined by the following equations where e is the accumulated execution time of resource r during the current period and t the total elapsed time during the current period, and π is the priority of the resource access.

$$T_{e,t,w,p} = \begin{cases} \emptyset : T_{e,t+1,w,p} & \text{if } e = w \text{ and } t < p \\ T_{0,0,w,p} & \text{if } e = w \text{ and } t = p \\ \emptyset : T_{e,t+1,w,p} + \{(r, \pi)\} : T_{e+1,t+1,w,p} & \text{if } e < w \text{ and } w - e < p - t \\ \{(r, \pi)\} : T_{e+1,t+1,w,p} & \text{if } e < w \text{ and } w - e = p - t \end{cases}$$

Note that in our definition, the task cannot idle if idling will make it miss the deadline. If the supply can avoid giving the resource to the task in this case, the system will have an unmet resource request transition that signals non-schedulability (by Definition 3.1). Let us consider an instance of a classical scheduling problem for a set of periodic tasks running on a single processor resource: $Task_{2,3}||Task_{2,\tau}||S$, where

$S = \{\bar{r}\} : S$. In the figure below, we show the initial part of the state space of the example where we assume that the priorities of all resource requests in both tasks are the same and equal to 1. Each state is represented as a tuple $ij|km$, where i and j are the first two parameters of the first task and k and m are the first two parameters of the second task. The other two parameters do not change and are omitted to avoid cluttering the figure. We also omit labels on the transitions: all transitions are labeled by $\{(\bar{r}, 1)\}$.



The tasks are schedulable according to the Definition 3.1 and the transition system of the composite process, shown above, can be seen as the specification of feasible schedulers for the task set. Nondeterminism in the transition system represents different decisions that a scheduler can make. For example, the trace along the top of the figure corresponds to the rate-monotonic scheduling policy, which gives higher priority to $Task_{2,3}$ as it has the smallest period. Indeed, to consider schedulability under a specific scheduling policy, we would simply need to specify the appropriate priorities and check for the schedulability of the system within the new transition system.

We now consider the demand of a periodic task defined above. It is easy to see that the task process is *deterministic*, that is, whenever $T_{e,t,w,p} \xrightarrow{\alpha} T_1$ and $T_{e,t,w,p} \xrightarrow{\alpha} T_2$ then $T_1 = T_2$. For a deterministic task, the demand is obtained by a straightforward replacement of requested resources by matching offered resources. Thus, $\text{demand}(Task_{w,p}) = X_{0,0,w,p}$ is defined below:

$$X_{e,t,w,p} = \begin{cases} \emptyset : X_{e,t+1,w,p} & \text{if } e = w, t < p \\ X_{0,0,w,p} & \text{if } e = w, t = p \\ \emptyset : X_{e,t+1,w,p} + \{\bar{r}\} : X_{e+1,t+1,w,p} & \text{if } e < w, w - e < p - t \\ \{\bar{r}\} : X_{e+1,t+1,w,p} & \text{if } e < w, w - e = p - t \end{cases}$$

It is easy to check that $\text{demand}(Task_{2,3}) \parallel \text{demand}(Task_{2,7})$ does not deadlock and thus can schedule the two tasks according to Lemma 3.14.

Let us now consider a task with variable execution time which takes between b and w time units to complete: $Task_{b,w,p}^v = Task_{b,p} + Task_{b+1,p} + \dots + Task_{w,p}$. One can see that $\text{demand}(Task_{b,w,p}^v) = \text{demand}(Task_{w,p})$. This observation matches the well-known fact from the real-time systems theory that for independent periodic tasks it is sufficient to consider the worst-case execution time of each task [17].

3.3.2. Scheduling with partial supplies

To illustrate compositional analysis with partial supplies, we begin with a simple example of time-partitioned supplies that are widely used in practice. Consider a periodic time partition with period P , duration $D \leq P$, and relative start time t_0 , which essentially offers a resource r for the interval $[t_0, t_0 + D)$ during each period: $Part_{t_0,D,P} = P_{0,t_0,D,P}$ is defined as follows where, again, addition is modulo P :

$$P_{t,t_0,D,P} = \begin{cases} \{\bar{r}\} : P_{t+1,t_0,D,P} & \text{if } t_0 \leq t < t_0 + D \\ \emptyset : P_{t+1,t_0,D,P} & \text{otherwise} \end{cases}$$

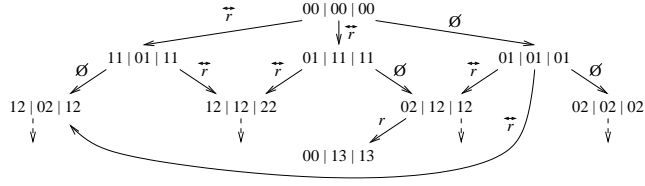


Figure 3: Scheduling with a periodic resource

It is clear that partitions with the same period and non-overlapping service intervals $[t, t + D)$ do not conflict. We can now analyze schedulability of tasks allocated to a partition separately from any other task in the system. It is, for example, trivial to see that partition $Part_{t_0, D, P}$ can schedule a task $Task_{D, P}$ for any t_0 .

We can similarly define more complex partial supplies. Consider, for example, compositional scheduling based on periodic resource models [23, 24]. A periodic resource model is a supply that guarantees w units of resource execution within a period P , however, the availability of the resource within the period is unknown *a priori*. We can straightforwardly model a periodic resource model as $PRM_{w, P} = \text{demand}(Task_{w, P})$. We can then analyze whether a set of tasks is schedulable with respect to this supply. This analysis will not be limited to independent periodic or sporadic tasks, unlike existing approaches in the literature.

As an example, consider the system $T_1 = Task_{1,3} || Task_{1,5} || PRM_{3,5}$ where all priorities of resource requests are fixed to 1. Figure 3 shows the initial state space using the same notation as above, except now the state tuple also includes the parameters e and t of the supply. Note that, in this transition system we have actions pertaining to resource consumption, abbreviated by \overleftarrow{r} , actions pertaining to resource requests, abbreviated by r , and idling actions. Recall that idling and consumed resource actions are incomparable in the preemption relation, while idling preempts unsatisfied resource requests. We see that a poor scheduling decision can make $Task_{1,3}$ miss its deadline. The scenario is seen on the right side of the figure: in the first two time units, one unit of resource goes to $T_{1,5}$ and the other unit of resource is denied to both tasks (this can happen in any order). If on the third step the supply denies access to the resource again, the first task cannot idle, thus we reach a transition labeled by $\{r\}$, which implies that the task misses its deadline, leading to a violation of Definition 3.1.

If instead we wish to consider schedulability of the tasks under an EDF (earliest-deadline-first) policy, we would have to repeat our analysis for periodic tasks with priorities defined as below.

$$T_{e,t,w,p} = \begin{cases} \emptyset : T_{e,t+1,w,p} & \text{if } e = w, t < p \\ T_{0,0,w,p} & \text{if } e = w, t = p \\ \emptyset : T_{e,t+1,w,p} + \{(r, D_{max} - (p - t))\} : T_{e+1,t+1,w,p} & \text{if } e < w, w - e < p - t \\ \{(r, D_{max} - (p - t))\} : T_{e+1,t+1,w,p} & \text{if } e < w, w - e = p - t \end{cases}$$

where D_{max} is a number exceeding the largest period in the task set. In this new setting, the composition $T_2 = Task_{1,3} || Task_{1,5} || PRM_{3,5}$, where $D_{max} = 6$, is schedulable

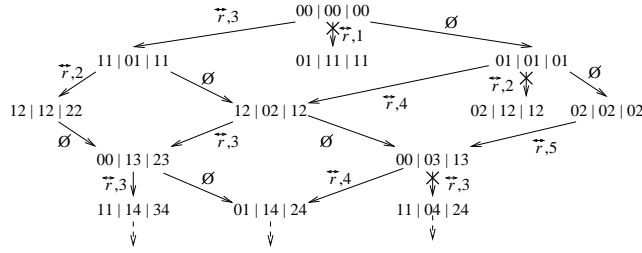


Figure 4: EDF scheduling with a periodic resource

as shown in Figure 4. In the figure, preempted transitions are crossed out. Note that the problematic action r of the previous example is no longer present because, in the initial state, action $(\overleftrightarrow{r}, 1)$ is preempted by action $(\overleftrightarrow{r}, 3)$, and from state $01|01|01$ action $(\overleftrightarrow{r}, 2)$ is preempted by action $(\overleftrightarrow{r}, 4)$, and thus the trees pointed to by these preempted actions in the transition system are pruned away, including the request action r .

4. Hierarchies on tasks and supplies

In the previous section we defined an approach for scheduling a set of tasks via analysis of their demand processes which are supply processes capturing the precise resource allocation required by tasks to complete their execution. In this section we proceed to provide machinery that will allow us to reason about hierarchical approaches to scheduling that rely on approximating the necessary supply, making it more generous than necessary, in exchange to a simple representation. Specifically, we define an ordering relation between tasks and two ordering relations between supplies which describe when a task/supply requires/offers greater resource allocation than another.

4.1. Task demands

We proceed to consider the notion of *task demand* and we define a relation on tasks which characterizes when a task is more “demanding” than another in the sense that it places more requirements on the available supply.

Definition 4.1. A relation $\mathcal{D} \in \mathbb{T} \times \mathbb{T}$ is a demand relation if for all $(T_1, T_2) \in \mathcal{D}$, if $T_1 \xrightarrow{\alpha}$ then

1. there exist $T_2 \xrightarrow{\beta} T_2'$ with $\text{sat}(\beta, \alpha)$, and $T_1 \xrightarrow{\alpha} T_1'$, such that $(T_1', T_2') \in \mathcal{D}$,
2. for all $T_2 \xrightarrow{\beta} T_2'$, if $\beta \preceq_{T_2} \alpha \cup \beta$, $\alpha \preceq_{T_1} \alpha \cup \beta$ and for no γ , $T_1 \xrightarrow{\gamma}$ and $\beta \preceq_{T_2} \gamma$ and $\gamma \preceq_{T_1} \alpha \cup \beta$, then there exists $T_1 \xrightarrow{\alpha} T_1'$ such that $(T_1', T_2') \in \mathcal{D}$.

We write $\preceq_{\mathcal{D}}$ for the largest demand relation and we say that a task T_1 is more demanding than a task T_2 , $T_2 \preceq_{\mathcal{D}} T_1$, if there exists a demand relation \mathcal{D} with $(T_1, T_2) \in \mathcal{D}$.

According to this definition, if T_1 is more demanding than T_2 then for every action α enabled by T_1 , (1) there is a move of T_2 which can be matched by some α -move of T_1 , (2) if β and α are maximal moves of T_2 and T_1 , respectively, with respect to $\alpha \cup \beta$,

and, additionally, there is no γ -move of T_1 such that β is a maximal move of T_2 with respect to γ and γ is a maximal move of T_1 with respect to $\alpha \cup \beta$, then all β -derivatives of T_2 are related to some α -derivative of T_1 .

To better understand the definition, let us first consider the point relating to the *existence* of an α move of T_1 (instead of universality) as required by the first clause: let

$$\begin{aligned} T_1 &\stackrel{\text{def}}{=} \{(r_1, 1)\} : \{(r_2, 2)\} : \text{FIN} + \{(r_1, 1)\} : \{(r_3, 1)\} : \text{FIN} \\ T_2 &\stackrel{\text{def}}{=} \{(r_1, 1)\} : \{(r_2, 1)\} : \text{FIN} \end{aligned}$$

Although T_2 cannot match the second summand of T_1 , it is intuitive that T_1 should be considered as more demanding than T_2 . This is because for T_1 to be scheduled successfully it is imperative that after being offered r_1 it will be offered simultaneously both r_2 and r_3 . Thus, it is sufficient for T_2 to match one of the $\{(r_1, 1)\}$ actions of T_1 .

The second clause of the definition is concerned with combinations of actions $\alpha \cup \beta$ where $T_1 \xrightarrow{\alpha}$ and $T_2 \xrightarrow{\beta}$, and it aims to ensure that, if a supply offers the resources in $\alpha \cup \beta$ then, if it is able to schedule the α -continuation of T_1 it should also be able to schedule the β continuation of T_2 , that is, T_1 should continue to be more demanding than T_2 . Clause (2) of the definition enunciates this requirement assuming that α and β are maximal actions with respect to $\alpha \cup \beta$, since this is necessary for them to constitute relevant responses to a supply of $\alpha \cup \beta$ and furthermore, that no action of T_1 , γ , lies between β and $\alpha \cup \beta$, since, if such as γ exists and $T_1 \xrightarrow{\gamma} T'_1$, it is sufficient that T'_1 is more demanding than T'_2 . For example, for

$$\begin{aligned} T_1 &\stackrel{\text{def}}{=} \{(r_1, 1), (r_3, 1)\} : \{(r_2, 1), (r_3, 1)\} : \text{FIN} \\ T_2 &\stackrel{\text{def}}{=} \{(r_2, 1)\} : \{(r_2, 0)\} : \text{FIN} + \{(r_1, 1), (r_3, 1)\} : \{(r_3, 0)\} : \text{FIN} \end{aligned}$$

we may check that, according to the definition, T_1 is more demanding than T_2 . Note that supply $S \stackrel{\text{def}}{=} \{\bar{r}_1, \bar{r}_2, \bar{r}_3\} : \{\bar{r}_2, \bar{r}_3\} : \text{FIN}$, schedules both tasks. Moreover, for

$$\begin{aligned} T_1 &\stackrel{\text{def}}{=} \{(r_1, 1), (r_2, 1), (r_3, 1)\} : \{(r_2, 1), (r_4, 1)\} : \text{FIN} \\ &\quad + \{(r_1, 1), (r_3, 1), (r_4, 1)\} : \{(r_2, 1), (r_3, 1)\} : \text{FIN} \\ T_2 &\stackrel{\text{def}}{=} \{(r_1, 1)\} : \{(r_2, 0)\} : \text{FIN} + \{(r_1, 1), (r_2, 1)\} : \{(r_2, 1), (r_4, 1)\} : \text{FIN} \end{aligned}$$

if we we apply the definition of a demand relation and take $\alpha = \{(r_1, 1), (r_3, 1), (r_4, 1)\}$ and $\beta = \{(r_1, 1), (r_2, 1)\}$ it is not necessary for the α -derivative of T_1 to be more generous than the β -derivative of T_2 , which is not. This is because $T_1 \xrightarrow{\gamma}$, where $\gamma = \{(r_1, 1), (r_2, 1), (r_3, 1)\}$ and $\beta \preceq_{T_2} \gamma$ and $\gamma \preceq_{T_1} \alpha \cup \beta$. Nonetheless, T_1 is *more* demanding than T_2 according to our definition and for a supply that offers the resources $\alpha \cup \beta$, if it may schedule T_1 then it must schedule both its α and γ derivatives and, consequently, it must also schedule T_2 .

Some further examples follow:

Example 4.2. Consider the following tasks.

$$\begin{aligned}
T_1 &\stackrel{\text{def}}{=} \{(r, 1)\} : \emptyset : T_1 \\
T_2 &\stackrel{\text{def}}{=} \{(r, 1)\} : \emptyset : T_2 + \emptyset : \{(r, 1)\} : T_2 \\
T_3 &\stackrel{\text{def}}{=} \{(r, 1)\} : \emptyset : \emptyset : T_3 + \emptyset : \{(r, 1)\} : \emptyset : T_3 + \emptyset : \emptyset : \{(r, 1)\} : T_3 \\
T_4 &\stackrel{\text{def}}{=} \{(r, 1)\} : \emptyset : \emptyset : T_4 + \emptyset : (\{(r, 1)\} : \emptyset : T_4 + \emptyset : \{(r, 1)\} : T_4) \\
T_5 &\stackrel{\text{def}}{=} \emptyset : \emptyset : \{(r, 1)\} : T_5 \\
T_6 &\stackrel{\text{def}}{=} \emptyset : \emptyset : \{(r, 1)\} : T_6 + \{(r, 1)\} : \{(r, 1)\} : \{(r, 1)\} : T_6
\end{aligned}$$

T_1 and T_2 request resource r once in every two time units with the distinction that T_1 requires the resource during the first time unit whereas T_2 is satisfied with an allocation during either time units. We may verify that T_1 is more demanding than T_2 . Note that action $T_2 \xrightarrow{\emptyset}$ need not be matched by T_1 since, according to the definition, it is not a maximal move of T_2 with respect to $\emptyset \cup \{(r, 1)\}$.

Moving on to tasks T_3 and T_4 we observe that they both require resource r once in every three time units but they pose slightly different nondeterministic requirements: T_3 is defined as the nondeterministic choice between the options of using r during one of the first three time units, whereas T_4 initially offers the choice between acquiring the resource and idling for two time units or idling and then acquiring the resource during one of the next two time units. We may check that T_1 is more demanding than both tasks T_3 and T_4 which demand r once every three time units. In addition, T_2 is more demanding than T_4 but not of T_3 since T_3 may choose to respond to an initial \emptyset action with the third summand which is *not* less demanding than T_2 given that it requests resource r during the third time unit. A comparison between T_3 and T_4 shows that T_3 is more demanding than T_4 . Finally, note that task T_5 is *not* more demanding than task T_6 . Intuitively, we can see that task T_5 can be scheduled by supply $S \stackrel{\text{def}}{=} \{\bar{r}\} : \emptyset : \{\bar{r}\} : S$ but task T_6 cannot. Furthermore, according to the definition, action $T_6 \xrightarrow{\{(r, 1)\}}$ needs to be examined as it is a maximal action of T_6 with respect to $\emptyset \cup \{(r, 1)\}$ and clearly one that illustrates the absence of a demand relation between the two tasks. This example brings out the subtle treatment required for the actions of the less demanding task. \square

We now proceed to justify our notion of *more demanding*. To begin with we may easily prove that \preceq_D is reflexive and transitive. Furthermore, we may verify that more demanding tasks place more requirements on their supplies by proving that if task T is more demanding than task T' then a supply that can schedule T can also schedule T' .

Lemma 4.3. *Suppose that task T_1 is schedulable by supply S and that T_1 is more demanding than T_2 . Then, task T_2 is also schedulable by supply S .*

PROOF: The proof consists of showing that the relation

$$\begin{aligned}
\mathcal{S} = \{ & (T_2, S) | \exists \text{ demand relation } \mathcal{D}, \text{ supply simulation relation } \mathcal{R} \text{ and} \\
& T_1 \in \mathbb{T}, (T_1, T_2) \in \mathcal{D}, (T_1, S) \in \mathcal{R} \}
\end{aligned}$$

is a supply simulation. Suppose $(T_2, S) \in \mathcal{S}$ and T_1 is a task such that $(T_1, T_2) \in \mathcal{D}$, where \mathcal{D} is a demand relation, and $(T_1, S) \in \mathcal{R}$, where \mathcal{R} is a supply simulation relation. Suppose $S \xrightarrow{\alpha} S'$. We confirm that the two clauses of Definition 3.5 are satisfied as follows:

- Since $(T_1, S) \in \mathcal{S}$, there exists T'_1 with $T_1 \xrightarrow{\beta} T'_1$, $\beta \trianglelefteq_{T_1} \alpha$ and $(T'_1, S') \in \mathcal{R}$. Then, by clause (1) of Definition 4.1, there exists T'_2 , such that $T_2 \xrightarrow{\beta'} T'_2$ with $\text{sat}(\beta', \beta)$, and for some T''_1 , $T_1 \xrightarrow{\beta} T''_1$, $(T''_1, T'_2) \in \mathcal{D}$. By Definition 3.5 it is also the case that $(T''_1, S') \in \mathcal{R}$, while, clearly, $\text{sat}(\beta', \alpha)$. This implies that $(T'_2, S') \in \mathcal{S}$ as required.
- Next suppose $T_2 \xrightarrow{\beta_2} T'_2$. Two cases exist:
 - If there exists $T_1 \xrightarrow{\gamma} T'_1$, $\gamma \trianglelefteq_{T_1} \alpha$ and $\beta_2 \trianglelefteq_{T_2} \gamma$. Then $(T'_1, S') \in \mathcal{R}$ and $(T'_1, T'_2) \in \mathcal{D}$. Thus, $(T'_2, S') \in \mathcal{S}$ as required.
 - Now suppose there exists no $T_1 \xrightarrow{\gamma} T'_1$, $\gamma \trianglelefteq_{T_1} \alpha$ and $\beta_2 \trianglelefteq_{T_2} \gamma$. Nonetheless, since S schedules T_1 , there exists $T_1 \xrightarrow{\beta_1} T'_1$, $\beta_1 \trianglelefteq_{T_1} \alpha$ and $(T'_1, S') \in \mathcal{R}$. Now consider $\beta_1 \cup \beta_2$. It must be the case that both $\beta_2 \trianglelefteq_{T_2} \beta_1 \cup \beta_2$ and $\beta_1 \trianglelefteq_{T_1} \beta_1 \cup \beta_2$, otherwise we would have contradictions to our assumptions that $\beta_2 \trianglelefteq_{T_2} \alpha$ and $\beta_1 \trianglelefteq_{T_1} \alpha$. Now, suppose there exists γ such that $T_1 \xrightarrow{\gamma}$ with $\beta_2 \trianglelefteq_{T_2} \gamma$ and $\gamma \trianglelefteq_{T_1} \beta_1 \cup \beta_2$. Since $\gamma \trianglelefteq_{T_1} \beta_1 \cup \beta_2$, $\text{sat}(\beta_1, \alpha)$ and $\text{sat}(\beta_2, \alpha)$, we have that $\text{sat}(\gamma, \alpha)$, which implies that either $\gamma \trianglelefteq_{T_1} \alpha$ or, if not, there exists γ' with $\gamma \prec \gamma'$, $\text{sat}(\gamma', \alpha)$ and $\gamma' \trianglelefteq_{T_1} \alpha$ while $\beta \trianglelefteq_{T_2} \gamma'$. This contradicts the assumption of the case and it implies that there exists no γ as the one just described and, consequently, by Definition 4.1(2), there exists T''_1 such that $T_1 \xrightarrow{\alpha} T''_1$ and $(T''_1, T'_2) \in \mathcal{D}$. By Definition 3.5 it is also the case that $(T''_1, S') \in \mathcal{R}$. Thus $(T'_2, S') \in \mathcal{S}$ which completes the proof. □

4.2. Supply generosity

Similarly to demands, we now proceed to define a hierarchy on supplies. This hierarchy is built on the basis of simulation relations that capture when a supply is more “generous” than another, where the intended meaning of “generosity” is that the more generous a supply the more tasks it can schedule. Below we define two such notions.

4.2.1. Strong generosity

Definition 4.4. A relation $\mathcal{R} \in \mathcal{S} \times \mathcal{S}$ is a strong generosity relation if for all $(S_1, S_2) \in \mathcal{R}$,

1. if $S_2 \longrightarrow$ then $S_1 \longrightarrow$.
2. if $S_2 \longrightarrow$ and $S_1 \xrightarrow{\alpha} S'_1$ then we have that $S_2 \xrightarrow{\alpha} S'_2$ and $(S'_1, S'_2) \in \mathcal{R}$.

We write \preceq_S for the largest strong generosity relation and we say that supply S_1 is strongly more generous than supply S_2 , $S_2 \preceq_S S_1$, if there exists \mathcal{R} with $(S_1, S_2) \in \mathcal{R}$.

According to the definition, S_1 is strongly more generous than S_2 if: (1) whenever S_2 is not deadlocked then S_1 is also not deadlocked, and (2) whenever S_2 is not deadlocked then any action enabled by S_1 is also enabled by S_2 . Intuitively, this definition aims to establish that any task scheduled by the less generous supply, S_2 , can also be scheduled by the more general supply, S_1 . To implement this, S_1 is required to offer a subset of the behaviors of S_2 , in this way it is guaranteed that each of S_1 's executions is also possible in S_2 and, thus, any task schedulable by S_2 will be schedulable by S_1 . Thus, in Example 3.2, S_1 is a strongly more generous supply than S_3 .

Note that the notion of strong generosity captures an earlier observation that the introduction of nondeterministic alternatives in supplies diminishes their potential of scheduling tasks. This is because, as viewed by a task, a supply with more choices constitutes an environment with more uncertainty, and the more ways in which a supply may offer resources implies a need for greater flexibility on behalf of a task. As an example consider

$$T \stackrel{\text{def}}{=} \emptyset : \{(r, 1)\} : \text{FIN} + \{(r, 1)\} : \{(r, 2)\} : \{(r, 1)\} : \text{FIN}$$

and

$$S_1 \stackrel{\text{def}}{=} \emptyset : \{\bar{r}\} : \text{FIN} + \{\bar{r}\} : \{\bar{r}\} : \text{FIN}, \quad S_2 \stackrel{\text{def}}{=} \emptyset : \{\bar{r}\} : \text{FIN}$$

Although S_2 can schedule T , this is not the case with S_1 . The same is true in the case that we allow a supply to offer a wider range of resources. For example, $S_1' \stackrel{\text{def}}{=} \{\bar{r}\} : \{\bar{r}\} : \text{FIN}$ also fails to schedule task T .

It is easy to show that \preceq_S is reflexive and transitive. Furthermore, the following result establishes that generosity preserves schedulability.

Lemma 4.5. *If task T is schedulable by supply S_2 and S_1 is strongly more generous than supply S_2 then T is also schedulable by supply S_1 .*

PROOF: The proof consists of showing that the relation

$$\mathcal{S} = \{(T, S_1) \mid \exists S_2 \in \mathcal{S}, \text{ supply simulation relation } \mathcal{R} \text{ and strong generosity relation } \mathcal{G}, (S_1, S_2) \in \mathcal{G} \text{ and } (T, S_2) \in \mathcal{R}\}$$

is a supply simulation relation. Suppose $(T, S_1) \in \mathcal{S}$ and S_2 is a supply such that $(T, S_2) \in \mathcal{R}$, where \mathcal{R} is a supply simulation relation and $(S_1, S_2) \in \mathcal{G}$ where \mathcal{G} is a strong generosity relation. Suppose $S_1 \xrightarrow{\alpha} S_1'$. By Definition 4.4(2), $S_2 \xrightarrow{\alpha} S_2'$ with $(S_1', S_2') \in \mathcal{G}$. Thus:

1. There exists $T \xrightarrow{\beta} T'$, $\beta \triangleleft_T \alpha$ with $(T', S_2') \in \mathcal{R}$. By definition, $(T', S_1') \in \mathcal{S}$ as required.
2. Suppose $T \xrightarrow{\beta} T'$, $\beta \triangleleft_T \alpha$. Again we have $(T', S_2') \in \mathcal{R}$ and $(T', S_1') \in \mathcal{S}$ which completes the proof. \square

In fact, we can also show that:

Lemma 4.6. *S_1 is strongly more generous than S_2 if and only if each task schedulable by supply S_2 is also schedulable by supply S_1 .*

PROOF: The '⇒' direction follows by the previous lemma. To demonstrate the '⇐' direction we will show that the following relation is a strong generosity relation.

$$\mathcal{R} = \{(S_1, S_2) \mid \forall T \cdot T \text{ schedulable by } S_2 \implies T \text{ schedulable by } S_1\}$$

Suppose $(S_1, S_2) \in \mathcal{R}$. We have the following:

1. Suppose $S_2 \xrightarrow{\alpha} S'_2$ and consider the set of tasks $\{\alpha : T \mid T \text{ schedulable by } S'_2\}$. Then, this set, being schedulable by S_2 , is also schedulable by S_1 , which implies that $S_1 \xrightarrow{\alpha}$ by Definition 3.5, as required.
2. Suppose $S_1 \xrightarrow{\alpha} S'_1$ and in order to reach a contradiction suppose further that $S_2 \not\xrightarrow{\alpha}$. Consider task $T \stackrel{\text{def}}{=} \sum_{S_2 \xrightarrow{\alpha_i}} \alpha_i : \text{FIN} + \alpha : T'$ where T' is not schedulable by S'_1 nor by any of S_2 's derivatives. Then T is schedulable by S_2 but not S_1 , resulting in a contradiction. This implies that $S_2 \xrightarrow{\alpha} S'_2$ and $(S'_1, S'_2) \in \mathcal{R}$ as required. \square

As an example for strong generosity consider supplies S_1 and S_2 below

$$\begin{aligned} S_1 &\stackrel{\text{def}}{=} \{\bar{r}\} : \{\bar{r}\} : \emptyset : S_1 \\ S_2 &\stackrel{\text{def}}{=} \{\bar{r}\} : (\{\bar{r}\} : \emptyset : S_2 + \emptyset : \{\bar{r}\} : S_2) + \emptyset : \{\bar{r}\} : \{\bar{r}\} : S_2 \end{aligned}$$

where S_1 offers supply r during the first two out of every three units of execution and S_2 offers r for two out of every three time units where the precise timing of the offerings is nondeterministic. We may easily verify that S_1 is more generous than S_2 and, as such, it may schedule at least as many tasks as S_2 . Thus, the deterministic nature of S_1 makes it more generous than S_2 .

Generalizing this example, we may also see that a periodic time partition with period P , duration $D \leq P$, and relative start time t_0 , $Part_{t_0, D, P}$, defined in Section 3.3.2, is strongly more generous than the periodic resource model $PRM_{D, P}$ that guarantees D time units of resource usage within every period P . The former presents one of the possible behaviors of the latter, this making it more generous, and able to schedule at least as many tasks.

4.2.2. Weak generosity

It turns out that the definition of strong generosity prevents us from comparing other supply models which one might be interested in comparing. For instance, supply S_1 above which offers a resource during the first two out of every three time units, would be intuitively considered as being more generous than supply $S_3 \stackrel{\text{def}}{=} \{\bar{r}\} : \emptyset : \emptyset : S_3$. However, S_1 is not strongly more generous than S_3 , according to our definition and, for instance, although S_1 offers more resources than S_3 it fails to schedule task T below which is in fact schedulable by the more stingy S_3 :

$$T \stackrel{\text{def}}{=} \{(r, 0)\} : [\emptyset : \emptyset : \text{FIN} + \{(r, 0)\} : \{(r, 0)\} : \text{FIN}]$$

Nonetheless, we would like to relax the notion of supply generosity to encompass a wider set of supplies at the expense of Lemma 4.5. Specifically, below we define a weaker notion of generosity which is subsequently considered within a restricted class of tasks. This definition is as follows.

Definition 4.7. A relation $\mathcal{R} \in \mathcal{S} \times \mathcal{S}$ is a weak generosity relation if for all $(S_1, S_2) \in \mathcal{R}$,

1. if $S_2 \longrightarrow$ then $S_1 \longrightarrow$.
2. if $S_2 \longrightarrow$ and $S_1 \xrightarrow{\alpha} S'_1$ then we have that $S_2 \xrightarrow{\beta} S'_2$, $\beta \subseteq \alpha$ and (S'_1, S'_2) .

We write \preceq_W for the largest weak generosity relation and we say that supply S_1 is weakly more generous than supply S_2 , $S_2 \preceq_W S_1$, if there exists a weak generosity relation \mathcal{R} with $(S_1, S_2) \in \mathcal{R}$.

This definition follows along the lines of that of strong generosity with the exception that we allow the less generous supply S_2 to match the supply of S_1 with a subset of its resources $\beta \subseteq \alpha$. Although we have shown that in this case S_1 is not guaranteed to schedule all tasks schedulable by S_2 , this new notion allows to explore the intuition that offering more resources makes for more generous supplies. The following hold:

- Supplies S_1 and S_3 considered above are such that S_1 is weakly more generous than S_3 .
- The partial supply $Part_{t_0, D, P}$ is weakly more generous than the partial supply $Part_{t_0, D', P}$, where $D' \leq D$.
- The periodic resource model $PRM_{w, P}$, defined in Section 3.3.2, is weakly more generous than the periodic resource model $PRM_{w', P}$, $w' \leq w$.
- The periodic resource model $PRM_{2,4}$ is *not* weakly more generous than the periodic resource model $PRM_{1,2}$. We may confirm this by considering the execution $PRM_{2,4} \xrightarrow{\{\bar{r}\}} \xrightarrow{\{\bar{r}\}} \xrightarrow{\emptyset} \xrightarrow{\emptyset} PRM_{2,4}$ and observing that it cannot be matched by $PRM_{1,2}$ as required by the definition of weak generosity. Note that task $Task_{1,2}$ is schedulable by supply $PRM_{1,2}$ but it is *not* schedulable by $PRM_{2,4}$.

Regarding the ability of weakly more generous supplies to schedule tasks we have the following result. Consider the class of periodic tasks \mathcal{C} with period p and execution time w , $Task_{w,p}$, defined in Section 3.3.1. We may prove that:

Lemma 4.8. *If task $T \in \mathcal{C}$ is schedulable by supply S_2 and S_1 is weakly more generous than supply S_2 , then T is also schedulable by supply S_1 .*

PROOF: The proof consists of showing that the following relation is a supply simulation relation.

$$\mathcal{R} = \{(T_{e,t,w,p}, S_1) \mid \exists S_2 \in \mathcal{S}, \text{ supply simulation relation } \mathcal{S} \text{ and weak generosity relation } \mathcal{W} \cdot (S_1, S_2) \in \mathcal{W} \text{ and } (T_{e',t,w,p}, S_2) \in \mathcal{S}, \text{ for some } e' \leq e\}$$

So, consider $(T_{e,t,w,p}, S_1) \in \mathcal{R}$ and suppose there exist a supply S_2 , a supply simulation relation \mathcal{S} and a weak generosity relation \mathcal{W} , such that $(S_1, S_2) \in \mathcal{W}$ and $(T_{e',t,w,p}, S_2) \in \mathcal{S}$ for some $e' \leq e$. Suppose $S_1 \xrightarrow{\alpha} S'_1$. We will show that $T_{e,t,w,p} \xrightarrow{\beta} T$ where $\beta \preceq_{T_{e,t,w,p}} \alpha$ and $(T, S'_1) \in \mathcal{R}$. First note that since $S_1 \xrightarrow{\alpha} S'_1$ and $(S_1, S_2) \in \mathcal{W}$, $S_2 \xrightarrow{\gamma} S'_2$, $\gamma \subseteq \alpha$ and $(S'_1, S'_2) \in \mathcal{W}$. The following cases exist:

- If $e = w$, $e = e'$ and $t < p$, then $T_{e',t,w,p} \xrightarrow{\emptyset} T_{e',t+1,w,p}$, $T_{e,t,w,p} \xrightarrow{\emptyset} T_{e,t+1,w,p}$, $(T_{e',t,w,p}, S'_2) \in \mathcal{S}$, and, thus, $(T_{e,t+1,w,p}, S'_1) \in \mathcal{R}$ as required.
- If $e = w$, $e < e'$ and $t < p$, then $T_{e',t,w,p} \xrightarrow{\beta'} T_{e'',t+1,w,p}$, where $e'' \in \{e', e' + 1\}$, depending on whether $\bar{r} \in \gamma$. In any case, $e'' \leq e$, $T_{e,t,w,p} \xrightarrow{\emptyset} T_{e,t+1,w,p}$, $(T_{e',t,w,p}, S'_2) \in \mathcal{S}$, and, thus, $(T_{e,t+1,w,p}, S'_1) \in \mathcal{R}$ as required.
- If $e = w$ and $t = p$, then since $T_{e',t,w,p}$ is schedulable by S_2 it must be that $e' = w$ and the proof follows as in the next case.
- If $e < w$, $w - e < p - t$, $w - e' < p - t$ the following cases exist. If $\bar{r} \in \gamma$, then $\bar{r} \in \alpha$ and $T_{e',t,w,p} \xrightarrow{(r,\pi)} T_{e'+1,t+1,w,p}$, $T_{e,t,w,p} \xrightarrow{(r,\pi)} T_{e+1,t+1,w,p}$, where $(T_{e'+1,t+1,w,p}, S'_2) \in \mathcal{S}$, and, thus, $(T_{e+1,t+1,w,p}, S'_1) \in \mathcal{R}$ as required. If $\bar{r} \notin \gamma$ and $\bar{r} \in \alpha$ then $T_{e',t,w,p} \xrightarrow{\emptyset} T_{e',t+1,w,p}$, $T_{e,t,w,p} \xrightarrow{(r,\pi)} T_{e+1,t+1,w,p}$, where $(T_{e',t+1,w,p}, S'_2) \in \mathcal{S}$, and, thus, $(T_{e+1,t+1,w,p}, S'_1) \in \mathcal{R}$ as required. Finally, if $\bar{r} \notin \gamma$ and $\bar{r} \notin \alpha$ then $T_{e',t,w,p} \xrightarrow{\emptyset} T_{e',t+1,w,p}$, $T_{e,t,w,p} \xrightarrow{\emptyset} T_{e,t+1,w,p}$, where $(T_{e',t+1,w,p}, S'_2) \in \mathcal{S}$, and, thus, $(T_{e,t+1,w,p}, S'_1) \in \mathcal{R}$ as required.
- If $e < w$, $w - e < p - t$ and $w - e' = p - t$, then the proof follows similarly to the first case of the previous clause.
- Finally, if $e < w$ and $w - e = p - t$, then, since $T_{e',t,w,p}$ is schedulable by S_2 , $e = e'$, $\bar{r} \in \gamma$ and thus, $\bar{r} \in \alpha$ and $T_{e',t,w,p} \xrightarrow{(r,\pi)} T_{e'+1,t+1,w,p}$, $T_{e,t,w,p} \xrightarrow{(r,\pi)} T_{e+1,t+1,w,p}$, where $(T_{e'+1,t+1,w,p}, S'_2) \in \mathcal{S}$, and, thus, $(T_{e+1,t+1,w,p}, S'_1) \in \mathcal{R}$ which completes the proof. \square

Example 4.9. Consider a system composed of two applications competing for the usage of a single resource, the first consisting of the task set $Task_{1,3} \parallel Task_{1,5}$ running under an EDF scheduler and the second consisting of the task set $Task_{1,6} \parallel Task_{1,5}$ running under a rate-monotonic (RM) scheduler (i.e. the smaller the period the higher the priority). We may verify that the assignment of supply $PRM_{3,5}$ to the first application and $PRM_{2,5}$ to the second application leads to the schedulability of the system. This can be achieved by constructing the demand-processes of the two applications and verifying that

1. $PRM_{3,5}$ is weakly more generous than demand($Task_{1,3} \parallel Task_{1,5}$) and
2. $PRM_{2,5}$ is weakly more generous than demand($Task_{1,6} \parallel Task_{1,5}$). \square

As the above example illustrates, our study of generosity relations complement our compositionality results for schedulability analysis of real-time systems. Specifically, our framework represents a formal approach for hierarchical scheduling which allows us (1) to check compositionally whether a hierarchical system is schedulable and extract appropriate (optimal) supplies for its components via the demand function, and (2) to construct practical schedulers for the components in question by isolating simple supplies that are at least as generous as the component demands. Our framework may also be used to formally represent the hierarchical scheduling approaches based on resource models [23] that rely on approximating the necessary supply, making it more generous than necessary, in exchange for a simple representation.

5. Conclusions

In this paper, we have presented PADS, a process algebra for resource demand and supply. The algebra can be used to describe a task process and its demand on resources necessary for the execution of a real-time task as well as a supply process that describes the behavior of a resource allocator. We have defined precisely the notion of schedulability using demand and supply, that is, when a process can be scheduled under a supply process, and provided a compositional theory of demand-supply schedulability. We believe that PADS is the first process algebra that can describe the behavior of demand and supply processes and compositional schedulability between them.

There are several directions in which the current work can be extended. We are currently developing a tool which implements our techniques for schedulability analysis and compositional scheduling of real-time systems and we are developing the theory of the process algebra via the study of the precongruence properties and the axiomatizations of the preorders proposed in this paper. We plan to extend our work in order to handle dependencies between tasks. Furthermore, we would like to define the notion of a *residual* supply which captures the supply available after a system has its resource demands satisfied and which will enable to perform incremental scheduling of systems. It would also be interesting to explore how to extend the notion of schedulability to the notion of *resource satisfiability* between demand and supply of arbitrary resources that are not shared mutually exclusively. Another extension is to explore demand and supply processes in the presence of probabilistic behavior.

References

- [1] K. Altisen, G. Göbller, A. Pnueli, J. Sifakis, and Y. Yovine. A framework for scheduler synthesis. In *Proceedings of RTSS'99*, pages 154–163. IEEE Computer Society, 1999.
- [2] K. Altisen, G. Göbller, and J. Sifakis. Scheduler modeling based on the controller synthesis paradigm. *Real-Time Systems*, 23(1-2):55–84, 2002.
- [3] H. Ben-Abdallah, J.-Y. Choi, D. Clarke, Y. S. Kim, I. Lee, and H.-L. Xie. A process algebraic approach to the schedulability analysis of real-time systems. *Real-Time Systems*, 15:189–219, 1998.
- [4] G. Bucci, A. Fedeli, L. Sassoli, and E. Vicario. Modeling flexible real time systems with preemptive time Petri nets. In *Proceedings of ECRTS'03*, pages 279–286. IEEE Computer Society, 2003.
- [5] Z. Deng and J. W.-S. Liu. Scheduling real-time applications in an open environment. In *Proceedings of RTSS'97*, pages 308–319. IEEE Computer Society, 1997.
- [6] A. Easwaran, M. Anand, and I. Lee. Compositional analysis framework using EDP resource models. In *Proceedings of RTSS'07*, pages 129–138. IEEE Computer Society, 2007.

- [7] A. Easwaran, I. Lee, and O. Sokolsky. Interface algebra for analysis of hierarchical real-time systems. In *Proceedings of FIT'08*, 2008.
- [8] X. Feng and A. Mok. A model of hierarchical real-time virtual resources. In *Proceedings of RTSS'02*, pages 26–35. IEEE Computer Society, 2002.
- [9] E. Fersman, P. Krcál, P. Pettersson, and W. Yi. Task automata: schedulability, decidability and undecidability. *Information and Computation*, 205(8):1149–1172, 2007.
- [10] E. Fersman, P. Pettersson, and W. Yi. Timed automata with asynchronous processes: schedulability and decidability. In *Proceedings of TACAS'02*, LNCS 2280, pages 67–82, 2002.
- [11] R.-T. S. GmbH. Real-Time Hybervisor, 2010. www.real-time-systems.com.
- [12] T. A. Henzinger and S. Matic. An interface algebra for real-time components. In *Proceedings of RTAS'06*, pages 253–263. IEEE Computer Society, 2006.
- [13] I. Lee, P. Brémont-Grégoire, and R. Gerber. A process algebraic approach to the specification and analysis of resource-bound real-time systems. *Proceedings of the IEEE*, pages 158–171, 1994.
- [14] I. Lee, A. Philippou, and O. Sokolsky. Resources in process algebra. *Journal of Logic and Algebraic Programming*, 72:98–122, 2007.
- [15] Linuxworks. LynxSecure Embedded Hypervisor and Separation Kernel, 2010. www.linuxworks.com/virtualization/hypervisor.php.
- [16] G. Lipari and E. Bini. Resource partitioning among real-time applications. In *Proceedings of ECRTS'03*, pages 151–160. IEEE Computer Society, 2003.
- [17] J. Liu. Real-Time Systems. *Prentice Hall*, 2000.
- [18] M. Mousavi, M. Reniers, T. Basten, and M. Chaudron. PARS: a process algebra with resources and schedulers. In *Proceedings of FORMATS'03*, LNCS 2791, pages 134–150, 2003.
- [19] M. Mousavi, M. Reniers, T. Basten, and M. Chaudron. PARS: a process algebra with resources and schedulers. In *Process Algebra for Parallel and Distributed Processing*, pages 331–358. Chapman and Hall/CRC, 2008.
- [20] M. Nunez and I. Rodriguez. PAMR: A process algebra for the management of resources in concurrent systems. In *Proceedings of FORTE'01*, pages 169–184, 2001.
- [21] A. Philippou, I. Lee, O. Sokolsky, and J.-Y. Choi. A process algebraic framework for modeling resource demand and supply. In *Proceedings of FORMATS'10*, LNCS 6246, pages 183–197, 2010.

- [22] S. Saewong, R. Rajkumar, J. Lehoczky, and M. Klein. Analysis of hierarchical fixed-priority scheduling. In *Proceedings of ECRTS'02*, pages 173–181. IEEE Computer Society, 2002.
- [23] I. Shin and I. Lee. Periodic resource model for compositional real-time guarantees. In *Proceedings of RTSS'03*, pages 2–13. IEEE Computer Society, 2003.
- [24] I. Shin and I. Lee. Compositional real-time scheduling framework. In *Proceedings of RTSS'04*, pages 57–67. IEEE Computer Society, 2004.
- [25] L. Thiele, E. Wandeler, and N. Stoimenov. Real-time interfaces for composing real-time systems. In *Proceedings of EMSOFT '06*. ACM, 2006.