Advanced Hierarchical Event-Stream Model

Karsten Albers, Frank Bodmann and Frank Slomka
Embedded Systems / Real-Time Systems, Ulm University
{name.surname}@uni-ulm.de

***Abstract*—Analyzing future distributed real-time systems, automotive and avionic systems, is requiring compositional hard real-time analysis techniques. Well known established techniques as SymTA/S and the real-time calculus are candidates solving the mentioned problem. However both techniques use quite simple event models. SymTA/S is based on discrete events the real-time calculus on continuous functions. Such simple models has been choosen because of the computational complexity of the considered mathematical operations required for real-time analysis. Advances in approximation techniques are allowing the consideration of more expressive descriptions of events. In this paper such a new expressive event model and its analysis algorithm are described. It integrates the models of both techniques. It is also possible in this module to integrate an approximative real-time analysis into the event model. This allows to propagate the approximation through the analysis of a distributed system leading to a much more efficient analysis.**

1. Motivation

The module-based design processes make it possible to handle the complexity in software and hardware design. Systems are build using a set of closed modules. These modules can be designed and developed separately. Modules have only designated interfaces and connections to other modules of their set. The purpose of modularisation is to split the challenging job of designing the whole system into multiple smaller jobs, allowing the reuse of modules in different designs or to include IP components of third-party vendors.

Every module-based design concept requires a well defined interface-concept for connecting the modules. Developing real-time systems requires for this interface-concept to cover also the real-time aspects of the modules. A concept for the real-time analysis is required to handle the modules separatly and allows a propagation of the real-time analysis results through the system. It is necessary to propagate the results of the real-time analysis of the different modules in an abstract way. The global analysis is build by connecting the local analyses of the single modules. Therefore it is essiential to have an expressive and efficient interface describing the influence in timing of one module to the next module. One aspect of this interface is the timing description of events which are produced by one module to trigger the next following module. Another aspect is the computation capacity that remains for lower priority modules left over by the higher priority ones.

Consider for example a network packet processor as shown in figure 1. The single packages are processed by chains of tasks τ which can be located on different processing elements P. The processing elements P can be processors, dedicated hardware or the communication network. The events Θ triggering the different tasks are equal to the packages

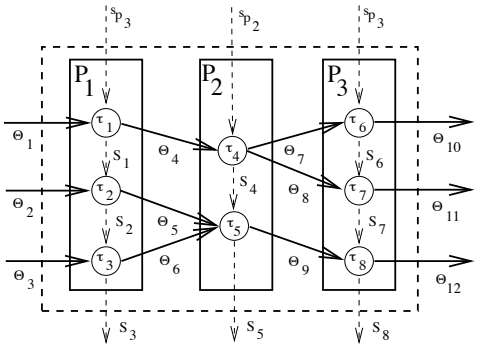

Figure 1. Network processor example

flowing through the network. Each processing unit P uses a fixed-priority scheduling and the task τ on each unit are sorted by their priority level. Each task τ has, as available capacity, the capacity S' left over by the tasks τ with a higher priority located on the same processing unit.

The purpose of this paper is to provide an efficient and flexible approach for the real-time analysis of such a modularized system. Therefore is a powerful and sufficient event model for describing the different time interfaces for the different aspects is necessary.

2. Related work

The most advanced approach for the real-time analysis of such a modulare network is the real-time calculus by Thiele et al. [4], [13]. It is based on the network calculus approach defined by Cruz [5] and Parekh and Gallager [9].

The event pattern is modeled by an sub-additive upper and super-additive lower arrival curve $\alpha_f^u(\Delta t)$ and $\alpha_f^l(\Delta t)$ delivering for every Δt the maximum number of events or the minimum, respectivly. The service curves $\beta_r^u(\Delta t)$ and $\beta_r^l(\Delta t)$ model the upper and lower bound of the computational requirements which can be handled by the resource during Δt. The real-time calculus provides equations to calculate the outgoing arrival and service curves out of the incoming curves of a task. To evaluate the modification equations independently from each other, a good finit description for the curves is needed. The complexity of the equations depends directly on the complexity of this description. In [8] and [4] an approximation with a fixed degree of exactness for the arrival and service curves was proposed in which each curve is described by three straight line segments. One segment describes the initial offset or arrival time, one an initial burst and one the long time rate.

1068-3070/08 $25.00 © 2008 IEEE
DOI 10.1109/ECRTS.2008.19

Δt	interval
T,a, l	period, offset, limitation
k	number of test intervals
Θ	event stream
$\theta = (T,a)$	event element
$\hat{\Theta}$	hierachical event stream
$\hat{\theta} = (T,a,l,G,\hat{\Theta}_{\hat{\theta}})$	hierachical event stream element
s	separation point
$\Upsilon(\Delta t,\hat{\Theta})$, $\Psi(\Delta t,\hat{\Theta})$	event bound function, demand bound function
$\mathscr{I}(\Delta t,\hat{\Theta})$, $\mathscr{B}$	interval bound function, busy period

Table I
LIST OF SYMBOLS

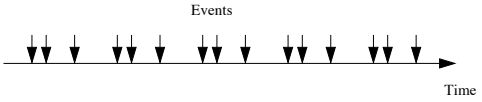

Figure 2. Example Event Stream

As outlined in [3] this approach is too simplified to be suitable for complex systems. No suitable description for the function is known so far. In this paper we will propose a model for the curves having a selectable approximation error. A trade-off between this degree of accuracy and the necessary effort for the analysis becomes possible.

SymTA/S [11],[12] is another approach for the modularized real-time analysis. The idea was to provide a set of interfaces which can connect different event models. Therefore the different modules can use different event models for analysis. Unfortunatly, the event models for which interfaces are provided are quite simple. In [11] an event model covering all these models was described. The problem of these models is that multiple bursts or bursts with different minimum separation times cannot be handled.

However in [10] a real-time analysis problem was formulated, which can't be solved by SymTA/S and the real-time calculus by each technique exclusivly. To solve it, it is necessary to integrate the models of both techniques into one powerful new model.

The event stream model proposed by Gresser [7] with its extension the hierachical event stream model proposed by Albers et al. [1] can model systems with all kinds of bursts efficiently. The problem is that it can only model discrete events and not the continious service function as needed for the real-time calculus. .

2.1. Event stream model

For the event stream model a system is described by a set of communicating tasks τ. Each task is assigned to one resource ρ. $\tau = (\hat{\Theta},c,d)$ is given by the worst-case execution time c_τ, the deadline d_τ and an event pattern $\hat{\Theta}_\tau$ triggering the tasks activations.

The key question is to find a good model for the event pattern $\hat{\Theta}$. For real-time analysis this model has to describe the worst-case densities of all possible event patterns. They lead to the worst-case demand on computation time. Comparing these worst-case demands with the available computation time

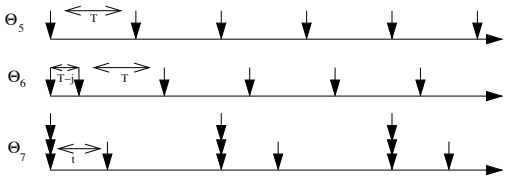

Figure 3. Example event streams ([6])

allows to predict the schedulability of a system. The event stream model gives an efficient general notation for the event bound function.

Definition 1: ([7], [2], [1]) The event bound function $\Upsilon(\Delta t,\Theta)$ *gives an upper bound on the number of events occuring within any interval* Δt.

Lemma 1: ([7]) *The event bound function is a subadditive function, that means for each interval* $\Delta t,\Delta J$*:*

$$\Upsilon(\Delta t+\Delta t',\Theta) \leq \Upsilon(\Delta t,\Theta)+\Upsilon(\Delta t',\Theta)$$

Proof: The events in $\Delta t+\Delta t'$ have to occure either in Δt or in $\Delta t'$. ■

Definition 2: An event stream Θ is a set of event elements $\theta = (T,a)$ given by a period T and an offset a.

$\Theta_1 = \{(6,0),(6,1),(6,3)\}$ (figure 2) describes three events requiring at least an interval $\Delta t = 3$ to occure, two of them have a minimum distance of one time unit. $\hat{\Theta}_1$ is repeated with a period of 6. In cases where the worst-case density of events is unknown for a concrete system an upper bound can be used for the event stream. The model can describe any event sequence. Only those event sequences for which the condition of sub-additivity holds are valid event streams.

Lemma 2: ([7]) *The event bound function for an event sequence* Θ *and an interval I is given by:*

$$\Upsilon(\Delta t,\Theta) = \sum_{\substack{\theta\in\Theta \\ \Delta t\geq a_\theta}} \left\lfloor \frac{\Delta t - a_\theta}{T_\theta} + 1 \right\rfloor$$

Proof: See [2] ■

It is a monotonic non-decreasing function. A larger interval-length cannot lead to a smaller number of events.

In figure 3 some examples for event streams can be found. The first one $\Theta_5 = \{(T,0)\}$ has a strictly periodic stimulus with a period T. The second example $\Theta_6 = \{(\infty,0),\ (T,T-j)\}$ shows a periodic stimulus in which the single events can jitter within a jitter interval of size j. In the third example $\Theta_7 = \{(T,0),\ (T,0)\ ,\ (T,0),\ (T,t)\ \}$ three events occur at the same time and the fourth occurs after a time t. This pattern is repeated with a period of T. Event streams can describe all these examples in an easy and intuitive way. The offset value of the first event element is always zero as this value models the shortest interval in which one single event can occur.

For the real-time analysis for this model let us first repeat the demand bound function definition for the event streams: $\Psi(\Delta t,\Gamma) = \sum_{\forall\tau\in\Gamma}\Upsilon(\Delta t - d_\tau,\Theta_\tau)c_\tau$ $= \sum_{\forall\tau\in\Gamma}\sum_{\substack{\forall\theta\in\Theta_\tau \\ \Delta t\geq a_\theta+d_\tau}} \left\lfloor \frac{\Delta t-a_\theta-d_\tau}{T_\theta}+1 \right\rfloor c_\tau$

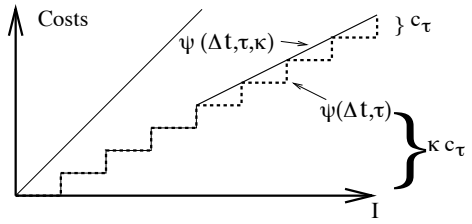

Figure 4. Approximated event stream element

Let θ be an event element belonging to the event stream Θ which belongs to the task τ.

The demand bound function allows a schedulability analysis for single processor systems by testing $\forall \Delta t : \Psi(\Delta t, \Gamma) \leq \mathscr{C}(\Delta t)$. Often an idealized capacity function $\mathscr{C}$ with $\mathscr{C}(\Delta t) = \Delta t$ is assumed. For an efficient analysis an approximation is necessary

2.2. Approximation of event streams

Definition 3: ([2]) Approximated event-bound-function
Let k be a chosen number of steps which should be considered exactly. Let $\Delta t_{\theta,k} = d_\tau + a_\theta + kT$. *We call*

$$\Upsilon'(\Delta t, \theta, \tau, k) = \begin{cases} \Upsilon(\Delta t_{\theta,k}, \theta) + \frac{c_\tau}{T_\theta}(\Delta t - \Delta t) & \Delta t > \Delta t_{\theta,k} \\ \Upsilon(\Delta t, \theta) & \Delta t \leq \Delta t_{\theta,k} \end{cases}$$

the approximated event bound function for task τ.

The function is shown in figure 4. The first k events are evaluated exactly, the remaining events are approximated using the specific utilization $U_\theta = \frac{c_{\tau_\theta}}{T_\theta}$. The interesting point of this function is that the error can be bounded to $\varepsilon_{\theta,k} = \frac{1}{k}$ and therefore does only depend on the chooseable number of steps, and is independent of the concrete values of the parameters of the tasks.

The complete approximated demand-bound-function is $\Psi'(\Delta t,\Gamma,k)=\sum_{\forall \tau \in \Gamma}\sum_{\forall \theta \in \Theta_\tau}\Psi'(\Delta t,\theta,k)$ and has the same error.
The hierachical event stream model [1] extends the event stream model and allows a more efficient description of bursts. In this model an event element describes the arrival not for just one periodic event but of a complete set of periodic events. This set of events can be also modeled by an event sequence having a limitation in the number of events generated by this event sequence. One limit of this model is that it can only describe discrete events. For the approximation it would be appropriate for the model to be capable to describe also the continuous part of the approximated event bound function.

3. Contribution

In this paper we will present an event model covering both, the discrete event model of SymTA/S and the continuous functions of the real-time calculus. It makes the elegant and tighter description of event bursts compared to the SymTA/S approach possible and allows a tighter modeling of the continuous function of the real-time calculus by integrating an approximation with a chooseable degree of exactness into the model. This does not only lead to more flexible and simpler analysis algorithms, it also allows to propagate the approximation together with the event models through the distributed system leading to an efficient, flexible and powerful analysis methodology for distributed real-time systems. The new model can, of course, also model the service functions of the real-time calculus in the same flexible way and allows therefore the integration of the discrete event model of SymTA/S with the continuous service functions.

4. Model

We will define the hierachical event sequence first. The hierarchical event stream is only a specialised hierachical event sequence fulfilling the condition of sub-additivity and can therefore be described by the same model.

Definition 4: A hierachical event sequence $\hat{\Theta} = \{\hat{\theta}\}$ *consists of a set of hierarchical event elements* $\hat{\theta}$ *each describing a pattern of events or of demand which is repeated periodically. The hierarchical event elements are described by:* $\hat{\theta} = (T, a, l, G, \hat{\Theta})$

where T_θ is the period, a_θ is the offset, l_θ is a limitation of the number of events or the amount of demand generated by this element during one period, $G_{\hat{\theta}}$ and $\hat{\Theta}_{\hat{\theta}}$ are the time pattern how the events respectively the demand is generated. The gradient $G_{\hat{\theta}}$ describing a constantly growing set of events, gives the number of events occurring within one time unit. A value $G_{\hat{\theta}} = 1$ means that after one time unit one event has occured, after two time units two events and so on. The gradient allows modeling approximated event streams as well as modeling the capacity of resources. Both cases can be described by a number of events which occurs respectively can be processed within one time unit. $\hat{\Theta}_{\hat{\theta}}$ is again a hierarchical event stream (child event stream) which is recursively embedded in $\hat{\theta}$.

Condition 1: Either $\hat{\Theta}_{\hat{\theta}} = \emptyset$ *or* $G_{\hat{\theta}} = 0$.
Due to this condition it is not necessary to distribute the limitation between the gradient and the sub-element. This simplifies the analysis without restricting the modelling capabilities.

The arrival of the first event occurs after a time units and at $a+T$, $a+2T$, $a+3T$, ..., $a+iT$ $(i \in N)$ the other events occurs.

Definition 5: A hierarchical event stream fulfills for every $\Delta t, \Delta t'$ *the condition* $\Upsilon(\Delta t + \Delta t', \hat{\Theta}) \leq \Upsilon(\Delta t, \hat{\Theta}) + \Upsilon(\Delta t', \hat{\Theta})$
In the following we will give a few examples to show the usage and the possibilities of the new model. A simple periodic event sequence with period 5 can be modeled by : $\hat{\Theta}_1 = \{(5,0,1,0,e)\}$

Lemma 3: Let Θ *be an event stream with* $\Theta = \{\theta_1, ..., \theta_n\}$. Θ *can be modeled by* $\hat{\Theta} = \{\hat{\theta}_1, ..., \hat{\theta}_n\}$ *with* $\hat{\theta}_i = (T_{\theta_i}, a_{\theta_i}, 1, \infty, \emptyset)$

Proof: Each of the hierarchical event elements generates exactly one event at each of its periods following the pattern of the corresponding event element. ■

$\hat{\Theta}_1$ approximated after 10 events would be modeled by: $\hat{\Theta}_1^{10} = \{(\infty,0,10,0,\{(5,2,1,0,e)\}),(\infty,47,\infty,\frac{1}{5},\emptyset)\}$

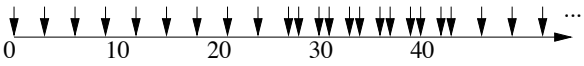

Figure 5. Example for overlapping events of different periods

Note that $47 = 2 + (10-1)\cdot 5$ is the point in time in which the last regular event occurs and therefore the start of the approximation.

One single event is modeled by $\hat{\Theta}_2 = \{(\infty, 0, 1, \infty, \emptyset)\}$. A gradient of ∞ would lead to an infinite number of events but due to the limitation only one event is generated. An event bound function requiring constantly 0.75 time units processor time within each time unit can be described by $\hat{\theta}_2 = (\infty, 0, \infty, 0.75, \emptyset)$.

With the recursively embedded event sequence any possible pattern of events within a burst can be modeled. The pattern consists of a limited set of events repeated by the period of the parent hierarchical event element. For example a burst of five events in which the events have an intra-arrival rate of 2 time units which is repeated after 50 time units can be modeled by $\hat{\Theta}_3 = \{(50, 0, 5, 0, \{(2, 0, 1, \infty, \emptyset)\})\}$.

The child event stream can contain grand-child event streams. For example if $\hat{\Theta}_3$ is used only for 1000 time units and than a break of 1000 time units is required would be modeled by $\hat{\Theta}_4 = \{(2000, 0, 100, 0, \hat{\Theta}_3)\}$.

The length $\Delta t_{\hat{\theta}}$ of the interval for which the limitation of $\hat{\theta}$ is reached can be calculated using a interval bound function $\mathscr{I}(x, \hat{\Theta}) = min(\Delta t | x = \Upsilon(\Delta t, \hat{\Theta}))$ which is the inverse function to the event bound function ($\mathscr{I}(l, \emptyset) = 0$):

$$\Delta t_{\hat{\theta}} = \mathscr{I}(l, \hat{\Theta}_{\hat{\theta}}) + \frac{l_{\hat{\theta}}}{G_{\hat{\theta}}}$$

Note that this calculation requires the condition of the model that either $G_{\hat{\theta}} = 0$ or $\hat{\Theta}_{\hat{\theta}} = \emptyset$ and that the calculation of the interval bound function requires the distribution of $l_{\hat{\theta}}$ on the elements of $\hat{\Theta}_{\hat{\theta}}$.

4.1. Assumptions and Condition

For the analysis it is useful to restrict the model to event sequences having no overlapping periods. Consider for example (figure 5) $\hat{\theta}_5 = \{(28, 0, 15, 0, \{(3, 0, 1, \infty, \emptyset)\})\}$. The limitation interval $\Delta t_{\hat{\theta}_6}$ has the length $\Delta t_{\hat{\theta}_6} = (15-1)\cdot 3 = 42$. The first period $[0, 42]$ and the second period $[28, 70]$ of the event sequence element overlap.

Condition 2: (Separation Condition) $\hat{\theta}$ fulfills the separation condition if the interval in which events are generated by $G_{\hat{\theta}}$ or $\hat{\Theta}_{\hat{\theta}}$ is equal or smaller than its period $T_{\hat{\theta}}$:

$\mathscr{I}(l_{\hat{\theta}}, \hat{\Theta}_{\hat{\theta}}) + \frac{l_{\hat{\theta}}}{G_{\hat{\theta}}} \leq T_{\hat{\theta}}$ or $T_{\hat{\theta}} \leq \Upsilon(T_{\hat{\theta}}, \hat{\Theta}_{\hat{\theta}}) + \frac{T_{\hat{\theta}}}{G_{\hat{\theta}}}$

The condition 2 does not reduce the space of event patterns that can be modeled by a hierarchical event sequence.

Lemma 4: A hierarchical event sequence element $\hat{\theta}$ that does not meet the separation condition can be exchanged with a set of event sequence elements $\hat{\theta}_1, ..., \hat{\theta}_k$ with $k = \left\lceil \frac{\mathscr{I}(l_{\hat{\theta}}, \hat{\theta})}{T_{\hat{\theta}}} \right\rceil$ and $\hat{\theta}_i = (kT_{\hat{\theta}}, (i-1)T_{\hat{\theta}} + a_{\hat{\theta}}, l_{\hat{\theta}}, G_{\hat{\theta}}, \hat{\Theta}_{\hat{\theta}})$.

Proof: The proof is obvious and therefore skipped. ■

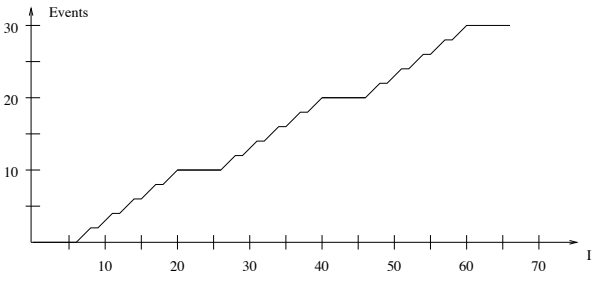

Figure 6. Hierarchical event sequence $\hat{\Theta}_6$

$\hat{\Theta}_5$ can be transferred into $\hat{\Theta}'_5$ meeting the separation condition: $\hat{\Theta}'_5 = \{(56, 0, 15, 0, \{(3, 0, 1, \infty, \emptyset)\}), (56, 28, 15, 0, \{(3, 0, 1, \infty, \emptyset)\})\}$

The separation condition prohibits events of different event sequence elements to overlap. We also do not allow recursion, so no event element can be the child of itself (or a subsequent child element).

4.2. Hierarchical Event Bound Function

The event bound function calculates the maximum number of events generated by $\hat{\Theta}$ within Δt.

Lemma 5: Hierarchical Event Bound Function $\Upsilon(\Delta t, \Theta)$:

Let for any $\Delta t, T$ define $mod(\Delta t, T) = \Delta t - \left\lfloor \frac{\Delta t}{T} \right\rfloor T$ and $\Upsilon(\Delta t, \emptyset) = 0$. Let $\Upsilon(\Delta t, \hat{\Theta}) = \sum_{\substack{\hat{\theta} \in \hat{\Theta} \\ \Delta t \geq a_{\hat{\theta}}}} \Upsilon(\Delta t, \hat{\theta})$ and

$$\Upsilon(\Delta t, \hat{\theta}) = \begin{cases} l_{\hat{\theta}} & T_{\hat{\theta}} = \infty, G_{\hat{\theta}} = \infty \\ \left\lfloor \frac{\Delta t - a_{\hat{\theta}}}{T_{\hat{\theta}}} + 1 \right\rfloor l_{\hat{\theta}} & T_{\hat{\theta}} \neq \infty, G_{\hat{\theta}} = \infty \\ min(l_{\hat{\theta}}, (\Delta t - a_{\hat{\theta}}) G_{\hat{\theta}} & \\ \quad + \Upsilon(\Delta t - a_{\hat{\theta}}, \hat{\Theta}_{\hat{\theta}})) & T_{\hat{\theta}} = \infty, G_{\hat{\theta}} \neq \infty \\ \left\lfloor \frac{\Delta t - a_{\hat{\theta}}}{T_{\hat{\theta}}} \right\rfloor l_{\hat{\theta}} + min(l_{\hat{\theta}}, & \\ \quad mod(\Delta t - a_{\hat{\theta}}, T_{\hat{\theta}}) G_{\hat{\theta}} & \\ \quad + \Upsilon(mod(\Delta t - a_{\hat{\theta}}, T_{\hat{\theta}}), \hat{\Theta}_{\hat{\theta}})) & T_{\hat{\theta}} \neq \infty, G_{\hat{\theta}} \neq \infty \end{cases}$$

Proof: Due to the separation condition it is always possible to include the maximum allowed number of events for completed periods $\left(\left\lfloor \frac{\Delta t - a_{\hat{\theta}}}{T_{\hat{\theta}}} \right\rfloor l_{\hat{\theta}} \right)$. Only the last incomplete fraction of a period has to be considered separately (min(...)). This remaining interval is given by subtracting all complete periods, and the offset a from the interval Δt $\left(mod(\Delta t - a_{\hat{\theta}}, T_{\hat{\theta}}\right)$. For the child event stream, the number of events is calculated by using the same function with now the remaining interval and the new embedded event sequence. In case of the gradient the number of events is simply $G_{\hat{\theta}} \Delta t$. The limitation bounds both values due to the separation condition. ■

Independently of the hierarchical level of an event sequence element it is considered only once during the calculation for one interval. This allows bounding the complexity of the calculation.

Example 1: $\hat{\Theta}_6 = \{(20, 6, 10, 0, \{(3, 0, 2, 1, \emptyset)\}$. $\Upsilon(\Delta t, \hat{\Theta}_7)$ is shown in figure 6. $\Upsilon(33, \hat{\Theta}_6)$ is given by

$$\Upsilon(33, \hat{\Theta}_6) = \left\lfloor \frac{27}{T_{\hat{\theta}}} \right\rfloor l_{\hat{\theta}} + \min(l_{\hat{\theta}}, mod(27, T_{\hat{\theta}}) G_{\hat{\theta}} + \Upsilon(mod(27, T_{\hat{\theta}}), \hat{\Theta}_{\hat{\theta}}))$$

$$= \left\lfloor \frac{27}{20} \right\rfloor \cdot 10 + \min(10, 0 + \Upsilon(7, \hat{\Theta}_{\hat{\theta}}))$$
$$= 10 + \min(10, \Upsilon(7, \hat{\Theta}_{\hat{\theta}}))$$
$$\Upsilon(7, \hat{\Theta}_{\hat{\theta}}) = \Upsilon(7, \hat{\theta}') = \left\lfloor \frac{7}{3} \right\rfloor \cdot 2 + min(2, mod(7,3) \cdot 1 + 0) = 4 + 1 = 5$$
$$\Upsilon(33, \hat{\Theta}_6) = 10 + \min(10, 5) = 15$$

4.3. Reduction and Normalization

In the following we will reduce event streams to a normal form. The hierarchical event stream model allows several different description for the same event pattern. For example an event stream $\hat{\Theta} = \{(100,0,22,0,\hat{\Theta}_a)\}$ with $\hat{\Theta}_a = \{(7,0,3,\infty,\emptyset),(5,3,2,\infty,\emptyset)\}$ can be rewritten as $\hat{\Theta} = \{(100,0,12,0,\hat{\theta}_{a,1}),\ (100,0,8,0,\hat{\theta}_{a,2}),(100,23,2,\infty,\emptyset)\}$ with $\hat{\theta}_{a,1} = (7,0,3,\infty,\emptyset)$ and $\hat{\theta}_{a,2} = (5,3,2,\infty,\emptyset)$.

Lemma 6: An event stream $\hat{\Theta}_a = \{(T_a, a_a, l_a, 0, \hat{\Theta}'_a)\}$ with a child element $\hat{\Theta}'_a = \{(T'_1, a'_1, l'_1, G'_1, \hat{\Theta}_1), ..., (T'_k, a'_k, l'_k, G'_k, \hat{\Theta}_k)\}$ can be transferred into an equivalent event stream $\hat{\Theta}_b$ with $\hat{\Theta}_b = \{\hat{\theta}_{a,1}, \hat{\theta}_{a,2}, ..., \hat{\theta}_{a,n}, \hat{\theta}_{a,x}\}$ having only child event sequences with one element where

$$\hat{\theta}_{b,i} = (T, a, \Upsilon(\Delta t_a, \hat{\theta}'_{a,i}), 0, \hat{\theta}'_{a,i})$$
$$\Delta t_a = \lim_{\substack{\varepsilon \to 0 \\ \varepsilon > 0}} (\mathscr{I}(l_a, \hat{\Theta}'_a) - \varepsilon)$$
$$\hat{\theta}_{a,x} = (\infty, \mathscr{I}(l_a, \hat{\Theta}'_a), l_a - \sum_{\forall \hat{\theta} \in \hat{\Theta}'_a} \Upsilon(\Delta t_a, \hat{\theta}'_{a,i}), \infty, \emptyset)$$

Proof: We have to distribute the limitation l_a on the elements of the child event sequence. First we have to find the interval $\Delta t'$ for which the limitation of the parent element l_a is reached by the child event sequence $\hat{\Theta}'_a$. $\Delta t'$ is given by $\mathscr{I}(l_a, \hat{\Theta}'_a)$. We have to calculate the costs required for each of the child event sequence elements for $\Delta t'$. It is given by $\Upsilon(\Delta t', \hat{\theta}_i)$. The problem is that several elements can have a gradient of ∞ exactly at the end of $\Delta t'$. In this situation the sum of $\Upsilon(\Delta t', \hat{\theta})$ may exceed the allowed limitation l_a of the parent element. The total costs is bounded by the global limitation l_a rather than the limitations l'_i. To take this effect into account we exclude the costs occurring exactly at the end of $\Delta t'$ for each hierarchical event element and we handle these costs seperately modeling them with the hierarchical event element $\hat{\theta}_{a,x}$. To do so we calculate the limitation not by $\Upsilon(\Delta t', \hat{\theta}'_i)$ but by $\Upsilon(\Delta t' - \varepsilon, \hat{\theta}'_i)$ where ε is an infinitly small value excluding only costs occurring at the end of $\Delta t'$ exactly. ■

This allows a better comparison between different hierarchical event streams.

4.4. Capacity Function

The proposed hierarchical event stream model can also model the capacity of processing elements and allows to describe systems with fluctuating capacity over the time. In the standard case a processor can handle one time unit execution time during one time unit real time. For many resources the capacity is not constant. The reasons for a fluctuating capacity can be for example operation-system tasks or variable processor speeds due to energy constraints.

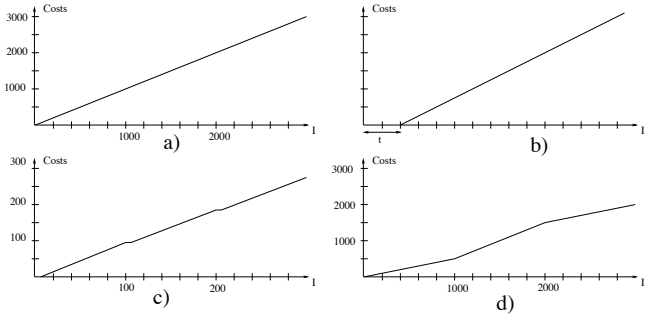

Figure 7. Example service bound functions

Assuming the capacity as constant also does not support a modularization of the analysis. This is especially needed for hierarchical scheduling approaches. Consider for example a fixed priority scheduling. In a modular approach each priority level gets the capacity left over by the previous priority level as available capacity. The remaining capacity can be calculated step-wise for each priority level taking only the remaining capacities of the next higher priority level into account. Such an approach is only possible with a model that can describe the left-over capacities exactly.

Definition 6: The service function $\beta(\Delta t, \rho)$ gives the minimum amount of processing time that is available for processing tasks in any interval of size Δt for a specific resource ρ for each interval Δt. It can also be modeled with the hierarchical event sequence model.

The service function is superadditiv and fulfills the inequation $\beta(\Delta t + \Delta t') \geq \beta(\Delta t) + \beta(\Delta t')$ for all $\Delta t, \Delta t'$. The definition matches the service curves of the real-time calculus. We propose to use the hierarchical event stream model as an explicit description for service curves.

In the following we will show, with a few examples, how to model fluctuating service functions with the hierarchical event streams. The constant capacity, as shown in 7 a) can be modeled by: $\beta_{basic} = \{(\infty, 0, \infty, 1, \emptyset)\}$

Blocking the service for a certain time t (figure 7 b) is done by: $\beta_{block} = \{(\infty, t, \infty, 1, \emptyset)\}$

A constantly growing service curve in which the service is blocked periodically every 100 time units for 5 time units (for example by a task of the operating system): $\beta_{Tblock} = \{(100, 5, 95, 1, \emptyset)\}$ (figure 7 c))

The service for a processor that can handle only 1000 time units with full speed and than 1000 time units with half speed (figure 7 d)): $\beta_{vary} = \{(2000, 1000, 500, \frac{1}{2}, \emptyset), (2000, 0, 1000, 1, \emptyset)\}$

These are only a few examples for the possibilities of the new model.

4.5. Operations

In the following we will introduce some operations on hierarchical event sequences and streams.

Lemma 7: (+ operation) *If $\hat{\Theta}_C = \hat{\Theta}_A + \hat{\Theta}_B$ than for each interval Δt the equation $\Upsilon(\Delta t, \hat{\Theta}_C) = \Upsilon(\Delta t, \hat{\Theta}_A) + \Upsilon(\Delta t, \hat{\Theta}_B)$ is*

true. It can be calculated by the union $\hat{\Theta}_C = \hat{\Theta}_A \cup \hat{\Theta}_B$.
It is also necessary to shift values.

Lemma 8: ($\rightarrow$ shift-operation) *We have*

$$\Upsilon(\Delta t, \hat{\Theta}') = \begin{cases} \Upsilon(\Delta t - t, \hat{\Theta}) & \Delta t \geq t \\ 0 & else \end{cases}$$

if $\hat{\Theta}'$ *contains and only contains for each element* $\hat{\theta} \in \hat{\Theta}$ *an* $\hat{\theta}' \in \hat{\Theta}'$ *with* $\hat{\theta}' = (T_{\hat{\theta}}, a_{\hat{\theta}} + t, l_{\hat{\theta}}, G_{\hat{\theta}}, \hat{\Theta}_{\hat{\theta}})$.

The shift operation ($\leftarrow$) $\Upsilon(\Delta t, \hat{\Theta}') = \Upsilon(\Delta t + t, \hat{\Theta})$ is defined in a similar way with $\hat{\theta}' = (T_{\hat{\theta}}, a_{\hat{\theta}} - t, l_{\hat{\theta}}, G_{\hat{\theta}}, \hat{\Theta}_{\hat{\theta}})$.

This operation ($\leftarrow, \rightarrow$) is associative with the ($+$) operation so we have $(\hat{\Theta}_A + \hat{\Theta}_B) \rightarrow t = (\hat{\Theta}_A \rightarrow t) + (\hat{\Theta}_B \rightarrow t)$ and $(\hat{\Theta}_A + \hat{\Theta}_B) \leftarrow t = (\hat{\Theta}_A \leftarrow t) + (\hat{\Theta}_B \leftarrow t)$. For $(\hat{\Theta} \rightarrow t) \rightarrow v$ we can write also $\hat{\Theta} \rightarrow (t + v)$.

To scale the event stream by a cost value is for example necessary to integration of the worst-case execution times.

Lemma 9: Let $\hat{\Theta}' = c\hat{\Theta}$. *Then for each interval* Δt: $\Upsilon(\Delta t, \hat{\Theta}') = c\Upsilon(\Delta t, \hat{\Theta})$ *if the child set of* $\hat{\Theta}'$ *contains and only contains for each element* $\hat{\theta}$ *of the child set of* $\hat{\Theta}$ *an element* $\hat{\theta}' \in \hat{\Theta}'$ *having* $\hat{\theta}' = (T_{\hat{\theta}}, a_{\hat{\theta}}, cl_{\hat{\theta}}, cG_{\hat{\theta}}, c\hat{\Theta}_{\hat{\theta}})$.

Proof: We do the proof for the add-operation:

$$\begin{aligned} \Upsilon(\Delta t, \hat{\Theta}_C) &= \Upsilon(\Delta t, \hat{\Theta}_A) + \Upsilon(\Delta t, \hat{\Theta}_B) \\ &= \sum_{\hat{\theta} \in \hat{\Theta}_A} \Upsilon(\Delta t, \hat{\theta}) + \sum_{\forall \hat{\theta} \in \hat{\Theta}_B} \Upsilon(\Delta t, \hat{\theta}) \\ &= \sum_{\forall \hat{\theta} \in \hat{\Theta}_A \cup \hat{\Theta}_B} \Upsilon(\Delta t, \hat{\theta}) = \Upsilon(\Delta t, \hat{\Theta}_A \cup \hat{\Theta}_B) \end{aligned}$$

The other proofs can be done in a similar way. ■

4.6. Utilization

Lemma 10: The utilization U_Γ *of a task set in which the event generation patterns are described by hierarchical event streams is given by* $((\forall \tau \in \Gamma) \Lambda (\forall \hat{\theta} \in \hat{\Theta}_\tau) | (l_{\hat{\theta}} \neq \infty \vee T_{\hat{\theta}} = \infty))$*:*

$$U_\Gamma = \sum_{\forall \tau \in \Gamma} \sum_{\substack{\forall \hat{\theta} \in \hat{\Theta}_\tau \\ T_\tau \neq \infty}} \frac{n_{\hat{\theta}}}{T_{\hat{\theta}}} + \sum_{\forall \tau \in \Gamma} \sum_{\substack{\forall \hat{\theta} \in \hat{\Theta}_\tau \\ l_{\hat{\theta}} = \infty \\ T_{\hat{\theta}} = \infty}} \left(U_{\hat{\Theta}_{\hat{\theta}}} + G_{\hat{\theta}} \right)$$

Note that event-elements with an infinite period and a finite limitation do not contribute to the utilization.

5. Schedulability tests

For the schedulability tests of uni-processor system using the hierarchical event stream model analysis, we can integrate the approximation and the available capacity into the analysis.

5.1. Schedulability tests for dynamic priority systems

A system scheduled with EDF is feasible if for all intervals Δt the demand bound function does not exceed the service function $\Psi(\Delta t) \leq \mathscr{C}(\Delta t, \rho)$. Both, the demand bound and the service function can be described by and calculated out of hierarchical event streams. This leads to the test $\sum_{\forall \tau \in \Gamma} \sum_{\forall \hat{\theta} \in \hat{\Theta}_\tau} \Upsilon(\Delta t - d_\tau, \hat{\theta}) c_\tau \leq \mathscr{C}(\Delta t, \rho)$. The analysis can be done using the approximation as proposed in [2]. For the exact analysis an upper bound for Δt, a maximum test interval is required to limit the run-time of the test. For the hierarchical event stream model one maximum test interval available is the busy period.

5.2. Response-time calculation for static priority scheduling

In the following we will show how a worst-case response time analysis for scheduling with static priorities can be performed with the new model. The request bound function Φ calculates the amount of computation time of a higher priority task that can interfere and therefore delays a lower-priority task within an interval Δt. In contrary to the event bound function the request bound function does only contain the events of the start, not the events of the end point of the interval:

$$\Phi(\Delta t, \tau) = \lim_{\substack{\Delta \rightarrow \Delta t \\ 0 \leq \Delta < \Delta t}} (\Upsilon(\Delta, \Theta_\tau) c_\tau)$$

For the hierarchical model it is only necessary to handle the cases $\Delta t = 0$ differently than in the calculation of the event bound function: $\Phi(\Delta t, \Gamma) = \sum_{\forall \tau \in \Gamma} c_\tau \sum_{\forall \hat{\theta} \in \hat{\Theta}_\tau} \Phi(\Delta t, \hat{\theta}, \tau)$ with

$$\Phi(\Delta t, \hat{\theta}, \tau) = \begin{cases} \left\lceil \frac{\Delta t - a_{\hat{\theta}}}{T_{\hat{\theta}}} \right\rceil l_{\hat{\theta}} & T_{\hat{\theta}} = \infty \\ 0 & \Delta t - a_{\hat{\theta}} \leq 0 \\ \left\lfloor \frac{\Delta t - a_{\hat{\theta}}}{T_{\hat{\theta}}} \right\rfloor l_{\hat{\theta}} + \min(l_{\hat{\theta}}, G_{\hat{\theta}}(\Delta t - a_{\hat{\theta}} + \\ \quad \Phi(mod(\Delta t - a_{\hat{\theta}}, T_{\hat{\theta}}), \hat{\Theta}_{\hat{\theta}})) & else \end{cases}$$

With this function it is possible to calculate the worst-case response times for the tasks:

Lemma 11: Let τ *be scheduled with fixed priorities and* $\Gamma_{hp(\tau)}$ *containing all task with a higher priority than* τ. *The response time* $r(\tau_{i,1})$ *for the first event of* τ_i *is given by:* $r(\tau_{i,1}) = \min(\Delta t | \mathscr{C}(\Delta t) \geq c_\tau + \Phi(\Delta t, \Gamma_{hp(\tau)}))$

The value for Δt can be calculated by a fix-point iteration starting with $\Delta t = c_\tau$. To calculate the maximum response time it is necessary to do the calculation for all events within the busy period.

The busy period of a task set is the maximum interval in which the resource is completely busy, so in which does not exists idle time for the resource: $\mathscr{B}(\Gamma) = \min(\Delta t | \mathscr{C}(\Delta t) \geq \Phi(\Delta t, \Gamma))$

Lemma 12: The worst-case response time of τ *can be found in the busy period of any task set containing* τ *and* $\Gamma_{hp(\tau)}$. *It is the maximum response time of all* $r(J, \tau)$ *where:*

$$r(J, \tau) = \min_{\forall 0 \leq \Delta t < \infty} (\Delta t | \mathscr{C}(J + \Delta t) \geq \Upsilon(J) c_\tau + \Phi(J + \Delta t, \Gamma_{hp(\tau)}))$$

$$r(\tau) = max_{\forall 0 \leq J \leq \mathscr{B}(\Gamma)} (r(J, \tau)$$

J is less or equal than the busy period ($J \leq \mathscr{B}(\Gamma)$). This minimum response time has to be lower than the deadline of the task.

6. Approximation

To limit the number of test intervals and therefore the computational complexity we integrate the approximation approach of [2]. We can now integrate the approximation directly into the model. We allow the approximation of an event element to start after the necessary number of test intervals are reached globally for this element, independently in which period of the parent event element this happens. In case that the event element $\hat{\theta}$ is a child element of another (parent) event element $\hat{\theta}'$ we have to distinguish for $\hat{\theta}'$ between those periods in which $\hat{\theta}$ is evaluated exactly and those in which $\hat{\theta}$

is approximated. To do this it is necessary to split $\hat{\theta}'$ at the last exactly considered interval of $\hat{\theta}$.

6.1. Case simple sequence with gradient

Let us consider first a simple hierarchical event element: $\hat{\theta} = \{(T,a,l,G,\emptyset)\}$

$\hat{\theta}^k$ is the approximative counter-part for $\hat{\theta}$ starting with the approximation after k exactly considered test intervals. $\hat{\theta}^k$ is modeled by: $\hat{\theta}^k = \{(\infty,0,l_A,0,\hat{\theta}),(\infty,a_A,l,G,\emptyset),(\infty,a_B,\infty,\frac{l}{T},\emptyset)\}$ with $l_A = kl$, $a_A = a+kT$, $a_B = a_A + \frac{l}{G}$. For the special case with $G = \infty$ we have $a_A = a_B$.

Example 2: Let us, for example consider $\hat{\Theta} = \{(10,0,3,\frac{1}{2},\emptyset)\}$. The approximation $\hat{\Theta}^5$ for $\hat{\Theta}$ after $k = 5$ exactly considered test intervals is given by $\hat{\Theta}^5 = \{(\infty,0,15,0,\{(10,0,3,\frac{1}{2},\emptyset)\}),(\infty,50,3,\frac{1}{2},\emptyset),(\infty,56,\infty,\frac{3}{10},\emptyset)\}$ where $l_A = 5\cdot 3 = 15$, $a_A = 0+5\cdot 10 = 50$, $a_B = 50 + \frac{3}{\frac{1}{2}} = 56$. We can simplify this example to: $\hat{\Theta}^5 = \{(\infty,0,18,0,\{(10,0,3,\frac{1}{2},\emptyset)\}),(\infty,56,\infty,\frac{3}{10},\emptyset)\}$

Consider another example $\hat{\Theta} = \{(10,2,3,\infty,\emptyset)\}$. The approximation $\hat{\Theta}^{10}$ is given by: $\hat{\Theta}^{10} = \{(\infty,0,30,0,\{(10,2,3,\infty,\emptyset)\},(\infty,103,3,\infty,\emptyset),(\infty,103,\infty,\frac{3}{10},\emptyset)\}$ or: $\hat{\Theta}^{10} = \{(\infty,0,33,0,\{(10,2,3,\infty,\emptyset)\},(\infty,103,\infty,\frac{3}{10},\emptyset)\}$

6.2. Approximation of one-level child element

Let us consider a hierarchical event sequence with one child element: $\hat{\theta} = (T,a,l,0,\hat{\theta}')$, $\hat{\theta}' = (T',a',l',G',\emptyset)$

$\hat{\Theta}^k$ is given in this case by:

$\hat{\Theta}^k = \{(\infty,0,l_A,0,\hat{\theta}^\circ),(\infty,a_A,kl - l_A,0,\{(T,a',l',G',\emptyset),(T,a'+\frac{l'}{G'},l-l',\frac{l'}{T'},\emptyset)\}),(\infty,a_B,x,\infty,\emptyset),(\infty,a_B,\infty,\frac{l}{T},\emptyset)\}$

The first element of $\hat{\Theta}^k$ models the part in which the child-element $\hat{\theta}'$ is considered exactly. In case that the first possible approximation interval for $\hat{\theta}'$ occures within the first period of $\hat{\theta}$, we have to start the approximation within this first period of $\hat{\theta}$. Otherwise it would not be possible to find a reasonable bound for the number of considered test intervals for $\hat{\theta}'$. So $\hat{\theta}^\circ$ depends on whether $l \le kl'$ or $l > kl'$. We have

$$\hat{\theta}^\circ = \begin{cases} \hat{\theta} & l \le kl' \\ \{(T,0,l,0,\hat{\theta}'^k)\} & l > kl' \end{cases}$$

$$\hat{\theta}'^k = \{(\infty,0,kl_{\hat{\theta}},0,\hat{\theta}),(\infty,kT_{\hat{\theta}},l_{\hat{\theta}},G_{\hat{\theta}},\emptyset),(\infty,kT_{\hat{\theta}}+\frac{l_{\hat{\theta}}}{G_{\hat{\theta}}},\infty,\frac{l_{\hat{\theta}}}{T_{\hat{\theta}}},\emptyset)\}$$

The calculation of l_A, l_B, a_A, a_B and a_C are done as follows:

$$l_A = \begin{cases} \left\lceil \frac{kl'}{l} \right\rceil l & l \le kl' \\ l & l > kl' \end{cases}$$

$$a_A = \begin{cases} \left\lceil \frac{kl'}{l} \right\rceil T + a & l \le kl' \\ T + a & l > kl' \end{cases}$$

$$a_B = kT + a + a'$$

The approximation of $\hat{\theta}'$ can be done by an element $\hat{\theta}'^k$ with a gradient $G_{\hat{\theta}'^k} = \frac{l'}{T'}$.

When starting finally the approximation of $\hat{\theta}$ a cost-offset x is required to ensure that the approximated function $\Upsilon(\Delta t,\hat{\theta}^k)$ is always equal or higher than the exact function $\Upsilon(\Delta t,\hat{\theta})$.

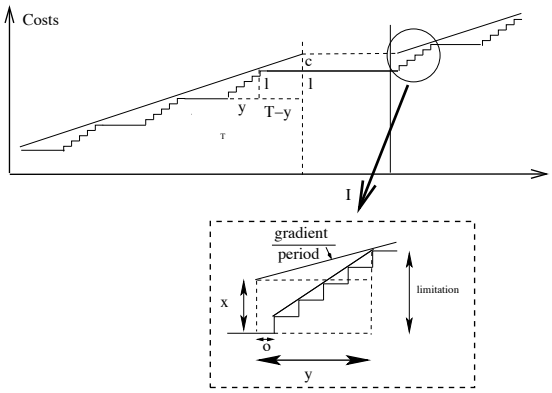

Figure 8. Case $\hat{\theta}'$ approximated, $\hat{\theta}$ not approximated

Figure 8 outlines this situation. This cost-offset is necessary as a new period of the parent element splits the approximation of the child element. The calculation of x can be done as follows:

$$l - x = y\frac{l}{T}$$

$$x = l\left(1 - \frac{y}{T}\right)$$

y gives the interval between the start of the child element $\hat{\theta}'$ and the point in time in which the limitation of $\hat{\theta}$ is reached. The reaching of the limitation is calculated using the approximative description of the child elements of $\hat{\theta}$ with the seperate consideration of every first event of $\hat{\theta}$. For a simple child element $\hat{\theta} = \{(T,a,l,0,\hat{\theta}')\}$ with $\hat{\theta}' = \{(T',a',l',\infty,\emptyset)\}$ this value y is given by

$$(y-a')\cdot(\frac{l'}{T'}) = l - l'$$

$$y = \frac{l-l'}{\frac{l'}{T'}} + a' = T'\frac{l}{l'} - T' + a'$$

Hence for x: $x = l - \frac{T'l^2}{Tl'} + \frac{T'l}{T} - \frac{a'l}{T}$

Example 3: Let us consider the example hierarchical event sequence: $\hat{\Theta}$=$\{(80,2,16,0,\hat{\Theta}')\}$, $\hat{\Theta}'$=$\{(10,2,3,\infty,\emptyset)\}$
For the approximation $\hat{\Theta}^{10}$ we get the values:

$$l_A = \left\lceil \frac{kl'}{l} \right\rceil l = \left\lceil \frac{10\cdot 3}{16} \right\rceil 16 = 32$$

$$a_A = \left\lceil \frac{kl'}{l} \right\rceil T + a = \left\lceil \frac{10\cdot 3}{16} \right\rceil 80 + 2 = 162$$

$$a_B = kT + a = 10\cdot 80 + 2 = 802$$

$$y = T'\frac{l}{l'} - T' + a' = 10\frac{16}{3} - 10 + 2 = 45.3333$$

$$x = l\left(1 - \frac{y}{T}\right) = 16\left(1 - \frac{45.333}{80}\right) = 6.9333$$

$$\hat{\Theta}^{10} = \{(\infty,0,32,0,\{(80,2,16,0,\{(10,2,3,\infty,\emptyset)\})\}),$$
$$(\infty,162,128,0,\{(\infty,2,3,\infty,\emptyset),(\infty,2,\infty,\frac{3}{80},\emptyset),$$
$$(80,2,13,\frac{3}{10},\emptyset)\},(\infty,802,6.9333,\infty,\emptyset),(\infty,802,\infty,\frac{16}{80},\emptyset)\}$$

6.3. Approximation of n-level child element

Let us consider the following hierarchical event element with two levels of child elements $\hat{\theta} = \{(T,a,l,0,\hat{\theta}')\}$, $\hat{\theta}' = \{(T',a',l',0,\hat{\theta}'')\}$, $\hat{\theta}'' = \{(T'',a'',l'',G'',\emptyset)\}$.

We consider the approximation $\hat{\theta}^k$. $\hat{\theta}^k$ is given by

$$\begin{aligned}\hat{\theta}^k = \{&(\infty,0,l_A,0,\hat{\theta}^\circ{}_1),(\infty,a_A,l_B,0,\hat{\theta}^\circ{}_2),\\ &(\infty,a_B,l_C,0,\{(T,a',x',\infty,\emptyset),(T,a',l-x',\frac{l'}{T'},\emptyset)\}),\\ &(\infty,a_C,x,\infty,0),(\infty,a_C,\infty,\frac{l}{T},\emptyset)\}\end{aligned}$$

$\hat{\theta}^\circ{}_1$ depends on whether $l \leq kl''$ or $l > kl''$. We have

$$\hat{\theta}^\circ{}_1 = \begin{cases}\hat{\theta} & l \leq kl'' \\ \{(T,0,l,0,\hat{\theta}'^k)\} & l > kl''\end{cases}$$

$$\hat{\theta}'^k = \{(\infty,0,kl_{\hat{\theta}},0,\hat{\theta}),(\infty,kT_{\hat{\theta}},l_{\hat{\theta}},G_{\hat{\theta}},\emptyset),(\infty,kT_{\hat{\theta}}+\frac{l_{\hat{\theta}}}{G_{\hat{\theta}}},\infty,\frac{l_{\hat{\theta}}}{T_{\hat{\theta}}},\emptyset)\}$$

$\hat{\theta}^\circ{}_2$ depends on whether $l \leq kl'$ or $l > kl'$. We have

$$\hat{\theta}^\circ{}_2 = \begin{cases}\emptyset & l \leq kl' \\ \{(T',a'',l'',G'',0),(T',a''+\frac{l''}{G''},l'-l'',\frac{l''}{T''},\emptyset)\} & l > kl'\end{cases}$$

The calculation of l_A, a_A and l_B:

$$l_A = \begin{cases}\left\lceil \frac{kl''}{l} \right\rceil l & l \leq kl'' \\ l & l > kl''\end{cases}$$

$$l_B = \begin{cases}\left\lceil \frac{kl'}{l} \right\rceil l - l_A & l \leq kl' \\ 0 & l > kl'\end{cases}$$

$$l_C = kl - (l_A + l_B)$$

$$a_A = \begin{cases}\left\lceil \frac{kl''}{l} \right\rceil T + a' + a & l \leq kl'' \\ T + a' + a & l > kl''\end{cases}$$

$$a_B = \begin{cases}\left\lceil \frac{kl'}{l} \right\rceil T & l \leq kl' \\ T & l > kl'\end{cases}$$

$$a_C = kT + a$$

The calculation of x' is the same as the calculation for x in the previous section. We have

$$y' = T''\frac{l'}{l''} - T'' + a''$$

$$x' = l'\left(\frac{T'-y'}{T'}\right)$$

The calculation of x and y is similar but using the approximation of $\hat{\theta}''$. We have

$$(y-a)\cdot(\frac{l'}{T'}) = l - x'$$

$$y = \frac{lT'}{l'} - \frac{x'T'}{l'} + a'$$

$$x = l\left(\frac{T-y}{T}\right)$$

Note that when setting $x'' = l''$ the calculation of x' and y' on the one side and x and y on the other side are the same. Therefore the proposed description for $\hat{\Theta}^k$ can be generalized to handle event sequences with n-level child event sequences. The calculation is visualized in figure 8.

Example 4: Let us consider the example hierarchical event sequence: $\hat{\Theta} = \{(1000,10,100,0,\hat{\Theta}')\}$, $\hat{\Theta}' = \{(80,2,16,0,\hat{\Theta}'')\}$, $\hat{\Theta}'' = \{(10,2,3,\infty,\emptyset)\}$.

For an approximation $\hat{\Theta}^{10}$ in which $k = 10$ test intervals are considered exactly we get the values:

$$y' = \frac{16-3}{(\frac{3}{10})} + 2 = 45.3333$$

$$x' = 16\cdot\left(\frac{80-45.3333}{80}\right) = 6.9333$$

$$y = \frac{100-6.9333}{\left(\frac{16}{80}\right)} + 2 = 467.333$$

$$x = 100\cdot\left(\frac{1000-67.3335}{1000}\right) = 53.2667$$

$$\begin{aligned}\hat{\Theta}^{10} = \{&(\infty,0,100,0,\hat{\Theta}^{10}_{2,1}),(\infty,1012,100,0,\{(\infty,2,3,\infty,\emptyset),\\ &(\infty,2,\infty,\frac{3}{80},\emptyset),(80,2,13,\frac{3}{10},\emptyset)\}),(\infty,2010,800,0,\\ &\{(\infty,2,6.9333,\infty,\emptyset),(\infty,2,\infty,\frac{6.9333}{1000},\emptyset),(1000,2,93.0667,\\ &\frac{16}{80},\emptyset)\}),(\infty,10010,53.2667,\infty,\emptyset),(\infty,10010,\infty,\frac{100}{1000},\emptyset)\}\end{aligned}$$

$$\begin{aligned}\hat{\Theta}^{10}_{2,1} = \{&(\infty,0,32,0,\{(80,2,16,0,\{(10,2,3,\infty,\emptyset)\})\}),\\ &(\infty,162,3,\infty,\emptyset),(\infty,162,\infty,\frac{3}{80},\emptyset),(80,162,13,\frac{3}{10},\emptyset)\}\end{aligned}$$

6.4. Approximation of element with several child elements

A hierarchical event sequence with several child elements can be transferred into a normalized hierarchical event sequence in which each event sequence element has only one child element. Each element matches one of the previous pattern and can therefore be approximated. The overall approximation of the event sequence is than only a merge of the single elements.

6.5. Required number of test intervals

In those cases in which the approximation of the child element starts within the completion of the first period of the parent element we cannot postpone it until the first period of the parent. It would not be possible to bound the number of test intervals for the child hierarchical event element.

Example 5: Consider the following example: $\hat{\theta}_{10} = \{10000,0,4000,0,\{\hat{\theta}_{11}\}\}$, $\hat{\theta}_{11} = \{10,0,5,\infty,\emptyset\}$
Postponing the approximation of the child up to the end of the first period of the parent would cost 3000 additional test intervals. We can still find a simple bound on the required number of test intervals. For those cases in which the approximation does not start within the first period, the number of test intervals for one period of the parent event element has to be less than the approximation bound k. Otherwise the approximation would be allowed somewhere within the first period. Therefore the maximum number of test intervals we have to additionally consider due to the postponing is bounded also by k, so a total bound of $2k$.

6.6. Splitting points

The splitting points are the points in which the parent element is splitted to destinguish between the non-approximated and the approximated part of one of its child elements. In general, the parent element is splitted at the first of its completed period which is greater than the first possible approximation interval of the child element. Each element can require as many splitting points as its total child-set has members. The total child-set contains its children, the children of its children and so on. The parent chain contains the parent element of an element, the parent of the parent element and so on.

For reason of simplification we consider only normalized hierarchical event sequences, in which each $\hat{\theta}$ can only have one direct child element at most.

Let $\hat{\theta}_1$ be the lowest-level child element and $\hat{\theta}_n$ be the highest level parent element. The splitting point for an element $\hat{\theta}_i$ is determined by the upper-most member $\hat{\theta}_j$ of a parent chain for which the first possible approximation interval for k exactly considered test intervals $t_{\hat{\theta}_i,k}$ of $\hat{\theta}_i$ is larger than the end of the first completed period of $\hat{\theta}_j$. This first complete period is given by $a_{\hat{\theta}_j} + T_{\hat{\theta}_j}$, so $t_{\hat{\theta}_i} > a_{\hat{\theta}_j} + T_{\hat{\theta}_j}$. The splitting point is the first start of a new period of $\hat{\theta}_t$ after $t_{\hat{\theta}}$, so $s^k_{i,j} = min(\Delta t | \Delta t = a_{t_i} + kT_{t_i} \wedge \Delta t \geq t_{\hat{\theta}_j,k})$

It is necessary to split each element of the parent-child chain between $\hat{\theta}_t$ and $\hat{\theta}_c$ at this point. All members of the parent chain of $\hat{\theta}_{t_i}$, which are of cause also member of the parent chain of $\hat{\theta}_i$, are splitted at their first period instead, so $\forall j > t \mid s_{i,j} = a_{\hat{\theta}_j} + T_{\hat{\theta}_j}$

In general we get a matrix of possible splitting points:

Lemma 13: (Splitting points) Let $\hat{\theta}_1, .., \hat{\theta}_n$ be a set of hierarchical event elements with $\hat{\theta}_1 = (T_1, a_1, l_1, G_1, \emptyset)$ and $\hat{\theta}_i = \{T_i, a_i, l_i, 0, \hat{\theta}_{i-1})$ for $0 < i \leq n$. Let $s^k_{i,j}$ be the splitting points for element j on the event element $\hat{\Theta}_i$ with the minimum number of k test-intervals considered exactly for $\hat{\theta}_j$. Let $t_{j,k}$ denote the first possible approximated test interval of $\hat{\theta}_j$ after k exact test intervals. $s^k_{i,j}$ can be calculated:

$$
\begin{aligned}
s^{k'}_{i,j} &= min(x | x = a_i + yT_i, y \in N, x \geq t_{j,k}) \\
s^k_{i,j} &= \begin{cases} s^{k'}_{i,j} & s^k_{i,j} < a_{i+1} + T_{i+1} \\ s^k_{i+1,j} & else \end{cases} \\
s^k_{i,0} &= a_i \\
s^k_{n,j} &= s^{k'}_{n,j}
\end{aligned}
$$

Proof: The first completed period of the hierarchical event element $\hat{\theta}_i$ after the first possible approximation start for the hierarchical event element $\hat{\theta}^k_j$ gives the potential splitting point $s^{k'}_{i,j}$. The resulting splitting point $s_{i,j}$ is only in those cases identical to the potential splitting point $s^{k'}_{i,j}$ in which either $\hat{\theta}_i$ is the top-level parent element $(i = n)$ or $s^{k'}{}_{i,j}$ is smaller than the end of the first period of the parent element $\hat{\theta}_{i+1}$. In all other cases, the completion point $s^k_{i,j}$ is identical to the corresponding completion point of the parent element of $\hat{\theta}_i$, $s_{i+1,j}$, which can again be identical to the splitting points of the $(i+1)$-th parent element and so on. ■

We can calculate the approximated hierarchical event streams using these splitting points.

Lemma 14: Let us consider a chain of hierarchical event streams $\hat{\Theta}_1, ..., \hat{\Theta}_n$ with $\hat{\Theta}_{\substack{j \\ j<n}} = \{(T_{\hat{\theta}_j}, a_{\hat{\theta}_j}, l_{\hat{\theta}_j}, 0, \hat{\Theta}_{j+1})\}$ and $\hat{\Theta}_n = \{(T_{\hat{\theta}_j}, a_{\hat{\theta}_j}, l_{\hat{\theta}_j}, G_{\hat{\theta}_n}, \emptyset)\}$. The approximated event elements are given by the following equations ($s_{0,j} = 0$):

$$
\begin{aligned}
\hat{\Theta}^k_j &= \{\hat{\theta}'_{i,j} | i + j \leq n \wedge s_{i,j} \neq s_{i,j-1}\} \cup \hat{\Theta}_{j,j+1} \\
\hat{\Theta}^k_{i,i} &= \{(\infty, s_{i,j-1}, x_{\hat{\theta}_i}, \infty, \emptyset), (\infty, s_{i,j-1}, \infty, \frac{l_{\hat{\theta}_i}}{T_{\hat{\theta}_i}}, \emptyset)\} \\
\hat{\Theta}^k_{i,i+1} &= \{(\infty, s_{i,i}, x_{\hat{\theta}_i}, \infty, \emptyset), (\infty, s_{i,i}, \infty, \frac{l_{\hat{\theta}_j}}{T_{\hat{\theta}_j}}, \emptyset)\} \\
\hat{\theta}_{i,j} &= \begin{cases} (\infty, s_{i,j-1}, \frac{s_{i,j} - s_{i,j-1}}{T_{\hat{\theta}_i}} l_{\hat{\theta}_i}, 0, & s_{i,j} \neq s_{i+1,j} \\ \quad \{(T_{\hat{\theta}_i}, 0, l_{\hat{\theta}_i}, G_{\hat{\theta}_i}, \hat{\Theta}_{i-1,j} & \\ (T_{\hat{\theta}_i}, a_{\hat{\theta}_i}, l_{\hat{\theta}_i}, G_{\hat{\theta}_i}, \hat{\Theta}_{i-1,j} & s_{i,j} = s_{i+1,j} \end{cases} \\
\hat{\Theta}'_{i,j} &= \begin{cases} \{\hat{\theta}'_{i,j}\} & s_{i+1,j} \neq s_{i+1,j+1} \\ \{\hat{\theta}'_{i,j}\} \cup \hat{\Theta}'_{i,j+1} & s_{i+1,j} = s_{i+1,j+1} \end{cases} \\
x_{\hat{\theta}_i} &= l_{\hat{\theta}_i} \left(1 - \frac{y_{i,j}}{T_{\hat{\theta}_i}}\right) \\
y_{\hat{\theta}_i} &= \frac{l_{\hat{\theta}_i} - x_{\hat{\theta}_{i-1}}}{\frac{l_{\hat{\theta}_{i-1}}}{T_{\hat{\theta}_{i-1}}}} + a_{\hat{\theta}_{i-1}} \\
x_{\hat{\theta}_1} &= l_{\hat{\theta}_1}
\end{aligned}
$$

Proof: Only for those splitting points $s^k_{i,j}$ being different from their predecessor splitting point $s^k_{i,j-1}$ a hierarchical event element can be constructed. The other splitting points would lead to elements generating no events. For the construction of the element we have to distinguish, whether the splitting point is identical to the corresponding splitting point of the parent element or whether it is a new value on its own. In the first case $(s^k_{i,j} = s^k_{i+1,j})$, the limitation is simply inherited from the parent element, in the second case $(s^k_{i,j} \neq s^k_{i+1,j})$, the limitation has to be calculated by distributing the previous limitation on the new parts. Note that $\frac{s^k_{i,j} - s^k_{i,j-1}}{T_{\hat{\theta}_i}} \in N$ by definition and therefore the limitation of the new parts are multiple of the limitation of the single elements. ■

The lemma summarizes (and simplifies) the results of the previous sections. Each element of the top-parent event sequence and therefore each chain of elements can be considered seperately.

7. Example

Example 6: Fig. 9 shows the advanced approximation for the event bound function of the event stream $\hat{\Theta}_7 = \{(20, 0, 10, 0, (2, 0, 2, \infty, \emptyset))\}$ and compares it with the description by SymTA/S and by the real-time calculus. For SymTA/S we have used an execution time of 2, a period of 4, a jitter of 10 and a minimum distance between two events of 2 time units. The lines of SymTA/S and the real-time calculus are nearly identical with the exception that SymTA/S models

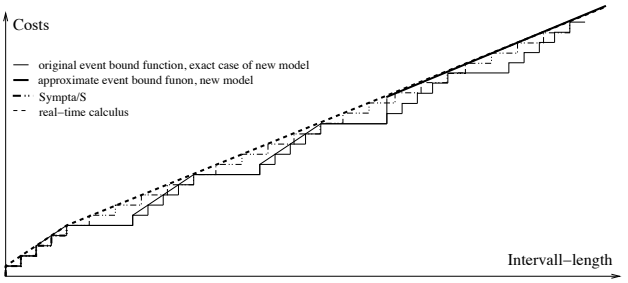

Figure 9. Approximated hierarchical event bound function

discrete events. The line for the new model in is exact form is always equal or below both other lines and in its approximated form it is below and the beginning and than equal to the real-time calculus curve. The degree of approximation is freely selectable. Note, that the event discrete modeling of the SymTA/S approach requires additional effort for the analyis.

The event stream consists of bursts with five events. The advanced approximated event stream with an approximation after three events is has the following separation points: $s_{1,0} = 0$, $s_{1,1} = 20$, $s_{2,0} = 0$, $s_{2,1} = 4$, $s_{2,2} = 60$.

For x and y we have the values:$y = \frac{l-l'}{\frac{l''}{T'}} + a' = \frac{10-2}{\frac{2}{2}} + 0 = 8$, $x = l\left(1 - \frac{y}{T}\right) = 10\left(1 - \frac{8}{20}\right) = 6$

It is given by the following description:$\hat{\Theta}_7^3 = \{(\infty,0,10,0,\{(20,0,6,0,(2,0,2,\infty,\emptyset)),(\infty,10,4,1,\emptyset)\}),(\infty,20,12,0,\{(20,0,2,\infty,\emptyset),(20,0,8,\frac{2}{20},\emptyset)\}),(\infty,60,6,\infty,\emptyset),(\infty,60,\infty,\frac{1}{2},\emptyset)\}$ Such a description limits the maximum number of test intervals for each hierarchical event element separately. In the example five test intervals for the child element and four test intervals for the parent element are required.

8. Conclusion

In this work we presented a new advanced event model especially suitable for the modeling of distributed systems. Such a system consists of several tasks bound on different processing elements and triggering each other. To divide the problem of real-time analysis of the whole system to a problem of real-time analysis of the single tasks, a model efficiently describing the densities of the events triggering the tasks (incoming events) and those events generated by the tasks to trigger other tasks (outgoing events) was required. Additionally, a model for the capacity of the processing elements available for the tasks was necessary. This is especially complicated in the case with a higher priority task already having used up a part of the capacity. In this paper we proposed a unified model for all of this. Additionally this model is capable to introduce approximations into the description of the event densities which guarantees a fast evaluation as well as an upper bound on the approximation error.

The new model integrates the efficient modeling of periodic and aperiodic events, burst of events in various kinds, approximated event streams and the original and the remaining capacites of processors in one single model. It can be seen as an explicit description for the arrival, service and capacity curves of the real-time calculus having the necessary modeling capabilies for them. We have presented the real-time analysis for this model for both, systems with dynamic or static priorities.

In future we will show the concrete integration of this model in the real-time calculus.

Remark 1: This work was funded by the Deutsche Forschungsgemeinschaft (DFG) under grand SL 47/3-1.

References

[1] K. Albers, F. Bodmann, and F. Slomka. Hierachical event streams and event dependency graphs. In *Proceedings of the 18th Euromicro Conference on Real-Time Systems (ECRTS'06)*, pages 97–106, 2006.

[2] K. Albers and F. Slomka. An event stream driven approximation for the analysis of real-time systems. In *IEEE Proceedings of the 16th Euromicro Conference on Real-Time Systems*, pages 187–195, Catania, 2004.

[3] K. Albers and F. Slomka. Efficient feasibility analysis for real-time systems with edf-scheduling. In *Proceedings of the Design Automation and Test Conference in Europa (DATE'05)*, pages 492–497, 2005.

[4] S. Chakraborty, S. Künzli, and L. Thiele. Performance evaluation of network processor architectures: Combining simulation with analytical estimations. *Computer Networks*, 41(5):641–665, 2003.

[5] R.L. Cruz. A calculus for network delay. In *IEEE Transactions on Information Theory*, volume 37, pages 114–141, 1991.

[6] K. Gresser. *Echtzeitnachweis ereignisgesteuerter Realzeitsysteme*. Dissertation, Düsseldorf, 1993.

[7] K. Gresser. An event model for deadline verification of hard real-time systems. In *Proceedings of the 5th Euromicro Workshop on Real-Time Systems*, 1993.

[8] S. Künzli. *Efficient Design Space Exploration for Embedded Systems*. PhD thesis, ETH Zürich No. 16589, 2006.

[9] A.K. Parekh and R.G.Gallager. A generalized processor sharing approach to flow control in integrated service networks. In *IEEE/ACM Transactions on Networking*, volume 1, pages 344–357, 1993.

[10] S. Perathoner, E. Wandler, L. Thiele, A. Hamann, S. Schliecker, R. Henia, R. Racu, R. Ernst, and M. González Harbour. Influence of different system abstractions on the performance analysis of distributed real-time systems. In *EMSOFT 2007*, pages 193–202. IEEE Computer Society Press, 2007.

[11] K. Richter. *Compositional Scheduling Analysis Using Standart Event Models*. Dissertation, TU Braunschweig, 2005.

[12] K. Richter and R. Ernst. Event model interfaces for heterogeneous system analysis. In *Proceedings of the Design Automation and Test Conference in Europe (DATE'02)*, 2002.

[13] L. Thiele, S. Chakraborty, M. Gries, and S. Künzli. Design space exploration for the network processor architectures. In *1st Workshop on Network Processors at the 8th International Symposium for High Performance Computer Architectures*, 2002.