Modifications on Event Streams for the Real-Time Analysis of Distributed Fixed-Priority Systems

Steffen Kollmann, Karsten Albers, Frank Bodmann, Frank Slomka
Department of Computer Science
University of Oldenburg
{steffen.kollmann,karsten.albers,frank.bodmann,frank.slomka}@informatik.uni-oldenburg.de

Abstract

In this paper we present a real-time analysis for complex distributed systems. The event stream model describes the occurrences of events within arbitrary time intervals. We propose a method to explore the modification of these occurrences as the events are processed within a complex task system. By observing the effects of several tasks competing for the same resource, additional insight can be won on the density of the events generated by the individual tasks.

1. Introduction

In this paper we will show how to adopt the real time analysis for distributed systems to the event stream model. In figure 1 a small example is presented where several task are connected to each other. The edges in the picture represent the dependencies within the system. Each edge is weighted by an event stream, which describes the occurrence of events. By means of the event stream model together with [2] we can make a real-time analysis for each task. The main contribution of this paper is the calculation of internal and outgoing event streams regarding their dependency on the scheduling and on the incoming streams.

2. Model

The model is one of the central points in real time analysis. The analysis can only be as accurate as the model allows. Most of the work in this area is based on the periodic task model with jitter like it is used in [5]. However the simplicity of the model leads to a loss of accuracy. For this reason we use the event stream model, because it is more accurate and still allows an efficient analysis.

Event streams were first defined in [3]. The purpose was to give a generalised description for every kind of stimuli. The idea is to notate for each number of events the minimum interval which can include this number of events. The result is a sequence of intervals which shows a non-decreasing behaviour. The reason for this behaviour is, that the minimum interval for n events cannot be smaller than the minimum interval for n-1 events since the first interval also includes n-1 events which therefore would be a contradiction. Each of the single intervals is called event stream element.

Definition 1: *An event stream is a set of event stream elements ψ:*

$$ES = \{\psi_1, \psi_2, ..., \psi_n\}$$

Each event stream element $\psi = \binom{p}{a}$ consists of an offset-interval a and a period p. With a period of infinite it is possible to model aperiodic behaviour. The whole event stream model is described in [3] and [1].

3. Competing Tasks and Event Streams

In this paper solutions will be presented for the scenarios shown in figure 1. All tasks in the system are scheduled

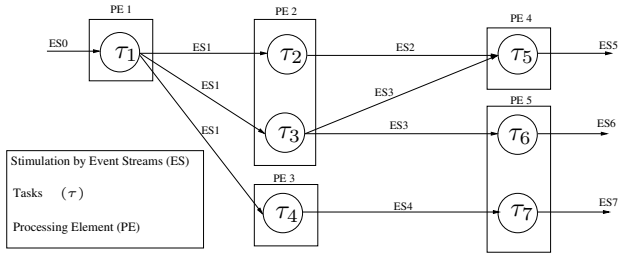

Figure 1. An Example of a Distributed System

by any fixed priority schedule and have no additional dependency, each task generates only an event at the end of its execution and the utilisation of each processing element must be less than or equal to one. Event streams allowed in

the system are characterized as follows.

$$ES_k=\left\{\binom{\infty}{a_{k,1}},\ldots,\binom{\infty}{a_{k,m-1}},\binom{p_k}{a_{k,m}},\ldots,\binom{p_k}{a_{k,n}}\right\}:1\leq m\leq n\wedge a_i\leq a_j\Leftrightarrow i\leq j$$

The idea of the approach which is presented here is that the first execution of a task which is analysed calculates as long as possible and all instances following it run as short as possible. This leads to the fact that the generated events occur in their highest density. Consequently we get the worst case of utilisation, which can be generated. To reach that goal, we need a modification of the response time analysis from [4]. This modified analysis is presented in the next formulas. By means of these formulas we get the response time ($RT(\tau)$) of the first execution.

$$RT(\tau)=min\left\{RT_n^m:RT_n^m=RT_n^{m-1}\right\}$$

$$RT_n^m=c_n+\sum_i^{i\in HP}E(RT_n^{m-1})\cdot c_i:RT_n^0=c_n$$

The event streams within the system or the outgoing event streams consist of an aperiodic part and a periodic part of events. So formulas are required which calculate when the aperiodic part ends and the periodic part starts. These formulas are presented in the next lines.

$$ES^\infty=\left\{\binom{P}{a}|\binom{P}{a}\in ES\wedge p=\infty\right\}$$

$$ES^P=ES\setminus ES^\infty$$

$$N=|ES|\ ;\ N^\infty=|ES^\infty|\ ;\ N^P=|ES^P|$$

$$a(i)=\begin{cases}a_i & i\leq N\\ \left\lfloor\frac{i-N^\infty}{N^P}\right\rfloor\cdot p_{((i-N)mod\ N^P+N^\infty)} & i>N\\ \qquad +a_{((i-N)mod\ N^P+N^\infty)}\end{cases}$$

$$j=min\{\forall i:EDA(i,\tau,ES)\leq a(i)\}$$

The formula ES calculates the event stream for each task. The formula divides itself into three parts. The first part calculates the events, which are affected by the first execution. The other events which are not affected by the first event, are calculated by the other parts. One part calculates the aperiodic events and the other part calculates the events which are repeated.

$$ES=\{\bigcup_{i=1}^{j}\left(\binom{\infty}{EDA(i,\tau,ES)}-RT(\tau)\right),\ \bigcup_{\substack{i=j+1\\p=\infty}}^{N}\left(\binom{\infty}{EDA(i,\tau,ES)}-RT(\tau)\right),\ \bigcup_{\substack{i=j+1\\p\neq\infty}}^{N+j}\left(\binom{p_i}{EDA(i,\tau,ES)}-RT(\tau)\right)\}$$

To get an exact calculation of the event streams and not only a lower bound we have to consider two cases. The first case is shown is figure 1. τ_1 stimulates the two tasks on the PE2. In this case we have a more relaxed event stream than in the case when tasks are stimulated by different event streams as the tasks on PE3. This is caused by the fact that the events, which stimulate the tasks, occur simultaneously. Consequently the distance between the generated events are stretched by tasks with a higher priority. Hence we need different formulas for the cases. For the first case the formulas are presented next. These functions regard the occurrence of the higher priority tasks.

$$EDA(1,\tau,ES)=RT(\tau)$$

$$\forall i\geq 2:EDA(i,\tau,ES)=g(EDA(i-1,\tau,ES),a(i),\sum_j^{j\in HP}b_j,b,RT(\tau))$$

$$g(t,a,b_g,b,RT)=\begin{cases}a+b_g+b & t\leq a,\\ t+b & t>a\wedge a<RT,\\ t+b_g+b & t>a\wedge a\geq RT,\end{cases}$$

The formulas for the other case are presented in the next few lines. This case is stricter than all other cases. So we can use this case as a lower bound for all cases which can occur.

$$EDA(1,\tau,ES)=RT(\tau)$$

$$\forall i\geq 2:EDA(i,\tau,ES)=h(EDA(i-1,\tau,ES),a(i),b)$$

$$h(t,a,b)=\begin{cases}a+b & t\leq a,\\ t+b & t>a,\end{cases}$$

4. Conclusion

The purpose of this paper was to calculate more precise inner event streams for the real-time analysis of fixed priority systems. This was possible by using the more general event stream model. We have given the calculation of the inner and outgoing event streams for the possible scenarios. Although the used model is more powerful this does not lead to an increase of analysis complexity. Together with the results in [2] a complete real time analysis is possible.

References

[1] K. Albers and F. Slomka. An event stream driven approximation for the analysis of real- time systems. In *Proceedings of the 16th Euromicro Conference on Real-Time Systems (ECRTS 04)*, pages 187–195. IEEE, July 2004.

[2] S. K. Baruah. Dynamic- and static-priority scheduling of recurring real-time tasks. *Real-Time Systems*, 24(1):93–128, 2003.

[3] 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.

[4] J. Lehoczky, L. Sha, and Y. Ding. The rate monotonic scheduling algorithm: Exact characterization and average case behavior. In *Proceedings of the Real-Time Systems Symposium*, pages 166–171, 1989.

[5] K. Tindell and J. Clark. Holistic schedulability analysis for distributed hard real-time systems. *Microprocessing and Microprogramming*, 40:117–134, April 1994.