Poster Abstract: Interface-based design approach for analysis of mode changes in distributed real-time systems

Nikolay Stoimenov Lothar Thiele
Computer Engineering and Networks Laboratory
ETH Zurich, 8092 Zurich, Switzerland
Email: {nikolays, thiele}@tik.ee.ethz.ch

***Abstract*—Schedulability analysis of adaptive real-time systems has been a well-known problem. It is relevant for many practical systems that have different operating modes. It is not a trivial to solve problem because schedulability needs to be shown not only in the individual operating modes, but also during the transitions between the modes. Many approaches have been proposed to address this problem but most of them are restricted to systems with fixed priority scheduling and periodic task activations. Recently, we have proposed an approach based on Real-time calculus that can perform schedulability analysis of systems with fixed and dynamic priorities such as EDF. It is also applicable to arbitrarily complex task activations. However, it is restricted to single processor systems. In this work, we propose an interface-based design approach based on Real-time interfaces which extends our previous work and makes it applicable to open dynamic environments and distributed systems.**

I. Problem statement

Adaptive real-time systems that change their operating mode during runtime have been in use for a long time in the aviation and the automotive industries. Performance and schedulability analysis techniques have been successfully developed for these areas. However, emerging application domains such as software defined radios in modern mobile phones and scalable video coding exhibit complex activations of tasks that require more sophisticated real-time analysis methods. Such methods are not trivial because adaptive systems require that they meet real-time constraints not only in the individual operating modes, but also during the transitions between modes.

A system may require a mode change because it needs to enter an emergency state, it needs to adapt to changes in the environment, or it needs to reduce its resource usage. A mode change may involve changes in the set of executing tasks, changes in the parameters of tasks e.g. execution times and deadlines, or changes in the activation patterns of tasks e.g. periodic, periodic with jitter, sporadic, etc.

In general, mode changes can lead to violations in the real-time constraints of a system. For example, if a task is triggered by an event stream, immediately switching from one stream to another can cause missed deadlines because bursts of events from the two streams may appear close to the switching time and lead to transient overloads of the system. Such overloads are usually overcome by introducing a delay between the two streams.

The work presented in this paper was partially supported by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322.

II. Related work

Schedulability of mode changes has been addressed previously, see e.g. [1]. Analysis methods for mode changes on single processor systems with rate-monotonic and deadline-monotonic scheduling have been introduced in [2] and [3], respectively. Later in [4], transition delays have been included in the model which help to avoid overload situations. However, all these approaches are limited to a strictly periodic task activation scheme. An approach proposed recently in [5] overcomes this limitation. It models event streams with three parameters: period, jitter, and minimum interarrival distance between events. Even though this model captures more complex activation patterns, it still describes only a limited set of event streams. In [5] the authors identify problems with the mode change analysis of an example distributed system however, a detailed analysis procedure for the general distributed case is missing.

In [6], [7], we have proposed an approach that can deal with fixed priority as well as with dynamic priority scheduling policies such as earliest deadline first (EDF). It uses the concepts of arrival curves α and service curves β from Network and Real-time calculus [8], [9] to model arbitrarily complex task activation patterns and resource availabilities. The method is component-based which means that the schedulability of the system is derived from the individual properties of the components. It can be used to analyze the schedulability of a mode change in a system given the parameters for the different modes and the parameters of the system. In order to deal with system overloads during a mode transition, the method can analyze schedulability of the system with a delay between the streams from the old and the new modes, and it can search for the minimum delay $\delta \geq 0$ that will make a mode transition schedulable, i.e. no deadlines will be violated in the old mode (mode I), the new mode (mode II), and during the transition between them.

For example, to check whether events are processed within their deadlines by a greedy processing component (GPC) in a fixed priority scheduling we need to check whether the inequalities $\text{Del}(\alpha_I^u, \tilde{\beta}^l) \leq d_I$ and $\text{Del}(\alpha_{II}^u + \text{Buf}(\alpha_I^u, \beta_I^l) - \tilde{\beta}^l(\delta), \tilde{\beta}^l) \leq d_{II}$ are satisfied where α_I^u and α_{II}^u are the maximum loads in mode I and mode II, $\tilde{\beta}^l$ is the minimum service available during the mode transition, d_I and d_{II} are the deadlines in mode I and mode II, and δ is the delay parameter defining an interval where new events from mode I are no longer accepted but old events that arrived before the start of this interval are processed, and after the end of this interval events from mode II start being accepted and processed. Note

that it is possible to overlap execution of events from both modes if δ is small. Del describes the maximum delay that an event can experience in a GPC and is defined as follows

$$\sup_{\lambda \geq 0} \left\{ \inf\{\tau \geq 0 : \alpha^u(\lambda) \leq \beta^l(\lambda + \tau)\} \right\} \stackrel{def}{=} \text{Del}(\alpha^u, \beta^l),$$

and Buf is the maximum backlog of a GPC

$$\sup_{\lambda \geq 0} \{\alpha^u(\lambda) - \beta^l(\lambda)\} \stackrel{def}{=} \text{Buf}(\alpha^u, \beta^l).$$

The cumulative load for such a component during a mode transition is upper bounded by

$$\begin{aligned} \tilde{\alpha}^u &= \max\left\{ \alpha^u_{II}(\Delta), \sup_{0 \leq \lambda \leq \Delta} \{\alpha^u_I(\Delta - \lambda) + \alpha^u_{II}(\lambda - \delta)\} \right\} \\ &= \max\left\{ \alpha^u_{II}(\Delta), (\alpha^u_I \overline{\otimes} \alpha^u_{II})(\Delta - \delta) \right\} \end{aligned} \quad (1)$$

which result can be used to compute a safe bound for the minimum service available $\tilde{\beta}'^l$ to lower priority components.

We can derive similar results for a component processing N number of streams with different deadlines using an EDF scheduling policy, see [6], [7]. To the best of our knowledge, this is the first result in mode change schedulability for EDF systems. However, our approach is only limited to single processor systems.

III. Proposed approach

Interface-based design is considered a key approach for tackling the increased complexity of modern embedded systems. The basic idea behind it is that a designer should only know about a component's interface and not about the internal implementation of the component. Knowledge about the interfaces of two components should be sufficient in order to decide whether they can be connected together and work properly. Toward this we have proposed the method of Real-time interfaces [10], [11] which combines the theory of assume/guarantee (A/G) interfaces [12], Real-time calculus and constraint propagation. Compatibility of the interfaces of two components in the context of Real-time interfaces would imply that the real-time constraints of the components would be satisfied if the components are connected. Such constraints could be on buffer underflow and overflow, end-to-end delays, throughput, utilization, and others.

For example, to check whether component F output rate is compatible with component G input rate, see e.g. [13], we need to check whether for all $\Delta \geq 0$ the inequality $\alpha^{uG}_F(\Delta) \leq \alpha^{uA}_G(\Delta)$ is satisfied, where component F guarantees that the actual output rate will never exceed α^{uG}_F and component G makes an assumption toward the environment that the input rate will never exceed α^{uA}_G. For example, this will be sufficient to show that the input buffer in component G will never overflow.

In this work, we propose how to extend the method in [6], [7] with an interface-based design approach by using the concept of Real-time interfaces. This will extend our results from [6], [7] in several ways. We will be able to find whether a mode change requested by the environment is feasible for the system by simply checking the interface of the system, i.e. check whether the transition load computed with (1) satisfies the input assume for the system, $\tilde{\alpha}^{uG} \leq \alpha^{uA}$, where α^{uA} is the assume value for the system input. Similarly, schedulability for mode I and mode II is checked with the inequalities $\alpha^{uG}_I \leq \alpha^{uA}$ and $\alpha^{uG}_{II} \leq \alpha^{uA}$, respectively. This is unlike our approach in [6], [7] where each different new mode required a reevaluation of the whole system with the new mode parameters. This approach can be applied to distributed systems with multiple communication and computation resources.

We will be able to analyze a system and find in advance bounds on the possible parameters for a mode change and include them in the system interface. By using information about what the system assumes at an input and what the environment currently guarantees at this input, we can compute a bound on what the environment may guarantee in the future after a mode change. This involves precomputing the curve $\alpha_{II}(\Delta) = (\alpha^{uA} \overline{\oslash} \alpha^{uG}_I)(\Delta)$. Then, given the bounds α^{uG}_{II} for mode II, we can find the minimum delay parameter δ that guarantees feasibility of the mode change with the inequality $\alpha_{II}(\Delta + \delta) \geq \alpha^{uG}_{II}$. This can be done efficiently by computing the maximal horizontal distance between the two curves with Del. This effectively skips the binary search step in the method proposed in [6], [7]. It makes our approach applicable at runtime which can be an advantage for open environments where the parameters of the different modes are not known during the design phase. Again this approach is applicable to distributed systems.

We will be able to augment the interface of a component with information about the necessary delay parameter between the old and new modes δ^A and propagate this information between the different components in a system. Let us consider two components F and G such that the output of F is connected to the input of G. If component G requires a certain minimum delay δ^A_G between the events from mode I and mode II in order to perform a schedulable mode transition, then component F needs to guarantee this delay between the events. Therefore, the computation of δ^A_F will need to take into account δ^A_G and the minimum delay parameter that makes the mode transition feasible for component F. Doing this can improve the tightness of the mode change analysis of distributed systems.

References

[1] J. Real and A. Crespo, "Mode change protocols for real-time systems: A survey and a new proposal," *Real-Time Systems*, vol. 26, no. 2, 2004.

[2] L. Sha, R. Rajkumar, J. Lehoczky, and K. Ramamritham, "Mode change protocols for priority-driven preemptive scheduling," *Real-Time Systems*, vol. 1, no. 3, 1989.

[3] K. W. Tindell, A. Burns, and A. J. Wellings, "Mode changes in priority pre-emptively scheduled systems," in *RTSS*, 1992.

[4] P. Pedro and A. Burns, "Schedulability analysis for mode changes in flexible real-time systems," in *ECRTS*, 1998.

[5] R. Henia and R. Ernst, "Scenario aware analysis for complex event models and distributed systems," in *RTSS*, 2007.

[6] S. Perathoner, N. Stoimenov, and L. Thiele, "Reliable mode changes in real-time systems with fixed priority or edf scheduling," ETH Zurich, TIK Report 292, 2008.

[7] N. Stoimenov, S. Perathoner, and L. Thiele, "Reliable mode changes in real-time systems with fixed priority or edf scheduling," in *DATE*, 2009, accepted.

[8] J. Y. Le Boudec and P. Thiran, *Network calculus: A Theory of Deterministic Queuing Systems for the Internet*. Springer, 2001.

[9] L. Thiele, S. Chakraborty, and M. Naedele, "Real-time calculus for scheduling hard real-time systems," in *ISCAS*, 2000.

[10] E. Wandeler and L. Thiele, "Interface-based design of real-time systems with hierarchical scheduling," in *RTAS*, 2006.

[11] L. Thiele, E. Wandeler, and N. Stoimenov, "Real-time interfaces for composing real-time systems," in *EMSOFT*, 2006.

[12] L. de Alfaro and T. Henzinger, "Interface theories for component-based design," in *EMSOFT*, 2001.

[13] S. Chakraborty, Y. Liu, N. Stoimenov, L. Thiele, and E. Wandeler, "Interface-based rate analysis of embedded systems," in *RTSS*, 2006.