

Task Attribute Assignment of Fixed Priority Scheduled Tasks to Reenact Off-line Schedules

Radu Dobrin and Gerhard Fohler
Department of Computer Engineering
Mälardalen University, Sweden
{radu.dobrin@., gerhard.fohler@.}mdh.se

Abstract

A number of industrial applications advocate the use of time-triggered approaches for reasons of predictability, distribution, and particular constraints such as jitter or end-to-end deadlines. The rigid offline scheduling schemes used for time-triggered systems, however, do not provide for flexibility. Fixed priority scheduling can provide more flexibility, but is limited with respect to predictability, as actual times of executions depend on run-time events.

We present a method to combine off-line schedule construction with fixed priority scheduling: by determining task attributes for the off-line scheduled tasks, such that the original schedule is reconstructed if scheduled with FPS at run-time. The method analyzes an off-line schedule together with original task constraints to create sequences and windows of tasks. Priorities and offsets are set to ensure task orders in sequences and relation between windows. As FPS cannot reconstruct all schedules with periodic tasks, our algorithm can split tasks into several instances to achieve consistent task attributes. Lower priority tasks can be added for run-time use.

1 Motivation and Contribution

Fixed priority scheduling (FPS) has been widely studied and used in a number of applications, mostly due by its simple run-time scheduling and resulting small overhead. Modifications to the basic scheme to handle semaphores [9], aperiodic tasks [10], static [11] and dynamic [7] offsets, and precedence constraints [6], have been presented. Consequently, FPS enables good flexibility for tasks with incompletely known attributes. Temporal analysis of FPS algorithms focuses on meeting deadlines, i.e., guarantees that all instances of tasks will finish *before* their deadlines. The actual times of executions of tasks, however, are generally not known and depend largely on run-time events, compromising predictability.

Off-line scheduling for time-triggered systems, on the other hand, provides strong predictability, as all times for task executions are determined and known in advance. In addition, complex constraints can be solved off-line, such as distribution, end-to-end deadlines, precedence, jitter, or instance separation. All this is enabled at the expense of losing run-time flexibility, as all actions have to be planned before.

We present an algorithm to combine off-line schedule construction with fixed priority run-time scheduling. The resulting systems have a time-triggered base that is complemented with even-triggered on-line scheduling. This allows us to combine benefits of off-line scheduling, in particular a distributed system, complex, constrained tasks, and end-to-end deadlines, with online scheduling, which allows flexible task execution. A number of tasks are specified to execute predictable, while allowing flexibility for all others.

Our method works by transforming off-line scheduled tasks with their original constraints into tasks with attributes suited for fixed priority scheduling, i.e., periods, deadlines, and offsets, which will reenact the original offline schedule at runtime. It divides the off-line schedule and its tasks into windows and sequences, sets priorities to ensure execution orders within windows, and determines priorities and offsets to ensure orders and relations between windows. As FPS cannot reconstruct all schedules with periodic tasks, our algorithm can split tasks into several instances to achieve consistent task attributes. Tasks with lower priorities can be added for run-time scheduling.

Priority assignment for FPS tasks has been studied in, e.g., [1], [5], and [8] study the derivation of task attributes to meet a overall constraints, e.g., demanded by control performance. Instead of specific requirements, our algorithm takes an entire off-line schedule and all task requirements to determine task attributes. A method to transform off-line schedules into earliest deadline first tasks has been presented in [4].

[1] N.C. Audsley, "Optimal Priority Assignment and Feasibility of Static Priority Tasks With Arbitrary Start Times", YCS 164, Dept. Computer Science, University of York, December 1991.

- [2] R. Dobrin and Y. Özdemir, "Task Attribute Assignment for Fixed Priority Scheduling", Master's Thesis, Department of Computer Engineering, Malardalen University, Sweden, June 2000.
- [3] G. Fohler, "Flexibility in Statically Scheduled Real-Time Systems". Ph.D. Thesis, Technisch-Naturwissenschaftliche Fakultät, Technische Universität Wien, Wien, Österreich, April 1994.
- [4] G. Fohler, "Joint scheduling of distributed complex periodic and hard aperiodic tasks in statically scheduled systems", Proceedings of the IEEE Real-Time Systems Symposium, 1995. [5] R. Gerber, Seongsoo Hong; M. Saksena, "Guaranteeing real-time requirements with resource-based calibration of periodic processes", IEEE Transactions on Software Engineering, Volume: 21 Issue: 7 July 1995 Page(s): 579–592.
- [6] M. Gonzalez Harbour, M.H Klein, and J.P. Lehoczky, "Fixed Priority Scheduling with Varying Execution Priority", proceedings of the IEEE Real-Time Systems Symposium, December 1991.
- [7] J.C. Palencia, M. Gonzalez Harbour, "Schedulability Analysis for Tasks with Static and Dynamic Offsets", Proceedings of the IEEE Real-Time Systems Symposium, 1998.
- [8] D. Seto, J.P. Lehoczky, Liu Sha, "Task period selection and schedulability in real-time systems" , Proceedings of the IEEE Real-Time Systems Symposium, 1998.
- [9] L. Sha, R. Rajkumar, J. P. Lehoczky, "Priority inheritance protocols: an approach to real-time synchronization", IEEE Transactions on Computers, Volume: 39 Issue: 9, Sept. 1990 Page(s): 1175–1185.
- [10] B. Sprunt, L. Sha, J. P. Lehoczky, " Aperiodic Task Scheduling for Hard Real-Time Tasks", The Journal of Real-Time Systems, 1989.
- [11] K. Tindell, "Adding Time Offsets to Schedulability Analysis", Technical Report YCS 221, Dept. of Computer Science, University of York, England, January 1994.