Erratum to

"Supporting Pipelines in Soft Real-Time Multiprocessor Systems"

Cong Liu and James H. Anderson

Department of Computer Science, University of North Carolina at Chapel Hill

Abstract

In [1], we derived a deadline tardiness bound for soft real-time periodic pipeline task systems on a multiprocessor. In that paper, we assumed that each job executes for exactly its worst-case execution time (WCET). We also claim that this assumption can be eased to treat each job's WCET as an upper bound, at the expense of more cumbersome notation because reducing a job's execution cost cannot increase any job's tardiness. Although this claim holds for ordinary periodic or sporadic task systems, it has come to our attention that it does not hold for pipeline task systems. We show a corresponding counterexample in this note.

1 Counterexample

Consider a task system that contains multiple tasks including a two-stage pipeline task T_i with two subtasks T_i^1 and T_i^2. Figure 1(a) shows a possible partial schedule of this task system on two processors when $T_{i,1}^1$, which is the first released job of T_i^1, executes for its WCET and completes at time t. Note that in this example $T_{i,1}^1$ is also tardy at t. Thus, at time t, both jobs $T_{i,2}^1$ and $T_{i,1}^2$ become eligible for execution. We also observe in Figure 1(a) that another job J that has lower priority than $T_{i,1}^1$, $T_{i,2}^1$, and $T_{i,1}^2$ completes before t.

As we can see in Figure 1(b), if $T_{i,1}^1$ executes for less than its WCET and completes earlier, e.g., at some time $t' < t$, then job J gets preempted at time t' and completes after t. In other words, J's response time grows in this case. This execution anomaly happens due to the fact that for pipeline tasks, one job completion may trigger two job executions: (*i*) the next released job of the same subtask, and (*ii*) the job with the corresponding pipeline-based precedence constraint. In this counterexample, $T_{i,1}^1$'s completion triggers the execution of both $T_{i,2}^1$ and $T_{i,1}^2$.

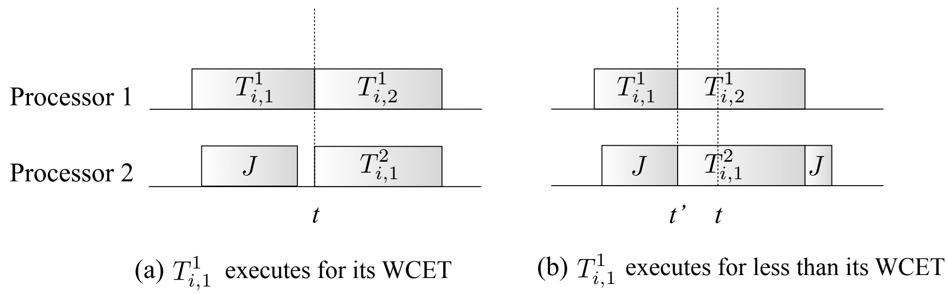

Figure 1. A counterexample.

Due to the above observation, the proof presented in [1] remains correct only if each job is assumed to execute for its WCET. If a job completes x time units earlier than its worst-case at runtime, this assumption can still be enforced by requiring the job to spin for x additional time units (i.e., force the job to keep occupying the processor for x more time units). Alternatively, analysis from [2] can be applied that does not suffer from the anomaly considered herein.

References

[1] C. Liu and J. Anderson. Supporting pipelines in soft real-time multiprocessor systems. In *Proc. of the 21st Euromicro Conf. on Real-Time Sys.*, pp. 269-278, 2009.

[2] C. Liu and J. Anderson. Supporting soft real-time dag-based systems on multiprocessors with no utilization loss. In *Proc. of the 31st Real-Time Sys. Symp.*, pp. 3-13, 2010.