Supplement of "Schedulability Analysis for a Mode Transition in Real-Time Multi-Core Systems"

Jinkyu Lee and Kang G. Shin

Department of Electrical Engineering and Computer Science

The University of Michigan, Ann Arbor, MI 48109-2121, U.S.A.

Appendix I: Detailed Proofs

A. Proof of Lemma 3.

R1. Since $\mathbf{W}_i^{g\Rightarrow h}(d_k^u)$ and $\mathbf{E}_i^{g\Rightarrow h}(d_k^u)$ are upper-bounds of $\mathbf{I}(\tau_k^u \leftarrow \tau_i^{g\Rightarrow h})$ with any release and execution patterns of tasks and any release pattern of the MTR, R1 holds.

R2. The schedulability analysis does not require any information from previous modes, such as response times of tasks with previous modes, and thus, R2 holds.

R3. If every task with M^g is identical to the task with M^h (i.e., $\tau_i^g = \tau_i^h$), $\mathbf{W}_i^{g\Rightarrow h}(\ell)$ and $\mathbf{E}_i^{g\Rightarrow h}(\ell)$ are the same as $W_i^g(\ell)$ and $E_i^g(\ell)$, respectively. The schedulability analysis in Theorem 1 is then equivalent to Lemma 1 with $W_i^g(\ell)$ and $E_i^g(\ell)$. Therefore, R3 holds.

B. Proof of Lemma 4.

We first look at a task τ_k in $\tau(1)$ (defined in Step 2 of Algorithm 1), and consider two aspects: (a) τ_k is schedulable or not; and (b) τ_k makes other tasks schedulable or not.

Placing τ_k's transition first is the best choice for τ_k's schedulability, because such an order maximizes the chance of τ_k^g's schedulability by Observation 4 and τ_k^h is schedulable with any sequential transition order by Observation 3.

Since $\tau_k^g \overset{I(\tau\setminus\tau^*)}{>} \tau_k^h$ holds, $\min({}^{\mathsf{S}}\mathbf{W}_k^{g\Rightarrow h}(d_i^g), d_i^g - e_i^g + 1) = \min(W_k^g(d_i^g), d_i^g - e_i^g + 1)$ holds, regardless of transition order. On the other hand, $\tau_k^{g\Rightarrow h} \prec \tau_i^{g\Rightarrow h}$ yields a smaller $\min({}^{\mathsf{S}}\mathbf{W}_k^{g\Rightarrow h}(d_i^h), d_i^h - e_i^h + 1)$, which equals $\min(W_k^h(d_i^h), d_i^h - e_i^h + 1)$. Therefore, placing τ_k's transition order in the earliest position minimizes the interference of τ_k on all other tasks in $\tau \setminus (\tau^* \cup \{\tau_k\})$. Note that we need not care for tasks in τ^* because the tasks with both modes are schedulable with any sequential transition, according to Observation 3.

In summary, placing the transition order of each task τ_k in $\tau(1)$ first maximizes the possibility of τ_k's schedulability, and minimizes its interferences on other tasks. This means that such a placement maximizes the chance of the schedulability of all tasks including τ_k itself.

The same reasoning holds for placing the transition order of tasks in $\tau(3)$ last. Therefore, the lemma holds.

C. Proof of Lemma 5.

Before proving this lemma, we introduce a property to be used, as stated in the following observation.

Observation 5. *Suppose that τ makes a sequential transition from M^g to M^h with a given order. Then, the left-hand side of Eq.* (18) *for a given task τ_k^u (u is either g or h) is not affected by the relative transition order of tasks in $\tau' \triangleq \{\tau_i | \tau_k^{g\Rightarrow h} \prec \tau_i^{g\Rightarrow h}\}$ and $\tau'' \triangleq \{\tau_i | \tau_k^{g\Rightarrow h} \succ \tau_i^{g\Rightarrow h}\}$, but affected by the elements of τ' and τ''.*

The observation holds because ${}^{\mathsf{S}}\mathbf{W}_i^{g\Rightarrow h}(d_k^u)$ depends only on whether $\tau_k^{g\Rightarrow h} \prec \tau_i^{g\Rightarrow h}$ or $\tau_k^{g\Rightarrow h} \succ \tau_i^{g\Rightarrow h}$ holds.

Suppose that τ is schedulable in the presence of a transition from τ^g to τ^h, with a given sequential transition order compliant with Algorithm 1. Now, we look at how transition order change of tasks in $\tau(1)$ affects the schedulability of two groups of tasks: (i) tasks in $\tau(1)$ and (ii) tasks in $\tau(2) \cup \tau(3)$.

For (i), the transition order of a task τ_k in $\tau(1)$ does not affect the schedulability of τ_i^g for all $\tau_i \in \tau(1) \setminus \{\tau_k\}$, since $\min({}^{\mathsf{S}}\mathbf{W}_k^{g\Rightarrow h}(d_i^g), d_i^g - e_i^g + 1) = \min(W_k^g(d_i^g), d_i^g - e_i^g + 1)$ holds regardless of the relative transition order of τ_k and τ_i (by the definition of $\tau_k^g \overset{I(\tau)}{>} \tau_k^h$). As to τ_i^h, it is also schedulable with any sequential transition by Observation 3, meaning that the transition order of a task τ_k in $\tau(1)$ does not affect the schedulability of τ_i^h for all $\tau_i \in \tau(1) \setminus \{\tau_k\}$.

The schedulability of tasks in $\tau(2) \cup \tau(3)$ is also not affected by the relative order of tasks in $\tau(1)$ according to Observation 5.

In summary, the relative transition order of tasks in $\tau(1)$ does not change the schedulability of every task in τ. This holds for $\tau(3)$ with the same reasoning. Therefore, the lemma follows.

Appendix II: Task set generation

To evaluate a variety of task sets in terms of task-set utilization, task utilization, the number of tasks, etc., we generate task sets as follows, based on a widely-used method [26].

For task parameters, p_i is uniformly distributed in $[1, 1000]$, d_i is set to p_i (i.e., implicit deadline tasks), and e_i is generated based on the exponential distribution of e_i/p_i, whose probability density function is $0.1 \cdot \exp(-0.1 \cdot x)$.

We focus on a situation where a task set τ switches from M^g to M^h. We generate τ^g and τ^h, both of which are schedulable by FP, and determine whether or not the task set is schedulable in the presence of the transition from M^g to M^h by our schedulability analyses. To achieve this, we generate 10,000 task sets (each of which has both modes M^g and M^h) for each $m = 2, 4, 8, 16$, by repeating the following procedure.

1) Initially, we generate a set of $m + 1$ tasks for τ^g because m tasks are trivially schedulable on m cores.
2) In order to exclude unschedulable sets, we check whether the generated task set τ^g can pass Lemma 1 with the upper-bound of $W_i(\ell)$ (for higher-priority tasks) and zero (for lower-priority tasks).
3) If τ^g fails to pass the test, we discard the generated task set and return Step 1. Otherwise, τ^g will be used for evaluation.
4) For each task τ_i^g in τ^g, we generate a new task τ_i^h for τ^h with probability 0.5; otherwise, we use the same task for τ^h, i.e., $\tau_i^h = \tau_i^g$.

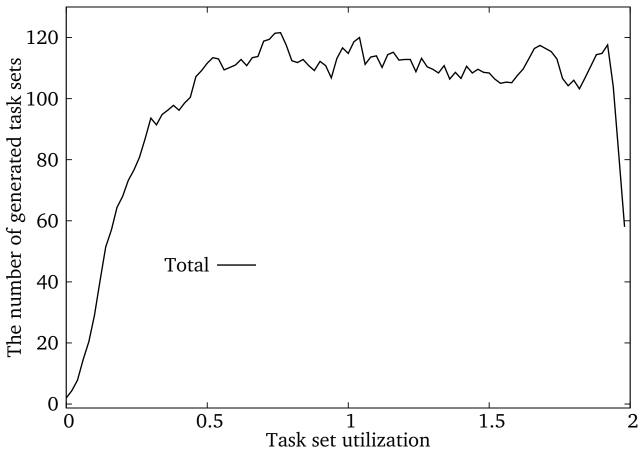

Fig. 5. Task-set utilization of generated task sets M^g for $m = 2$

5) We perform Step 2 for τ^h.
6) If τ^h fails to pass the test, we discard the generated task set and return Step 4. Otherwise, we include a pair of τ^g and τ^h for evaluation; we create a new set for τ^g by adding a new task into the current τ^g, and return Step 2.

For task sets for EDF, we apply $E_i(\ell)$ instead of $W_i(\ell)$. Note that we do not consider task deletion, since it cannot make a schedulable task set unschedulable. In future, we will evaluate addition of tasks.

We now present the statistics of generated sets M^g for $m = 2$; the trend for other m values is similar to $m = 2$. Fig. 5 shows the total number of generated task sets M^g with different task-set utilizations (i.e., $U_{sys} \triangleq \sum_{\tau_i \in \tau} e_i/p_i$) in $[U_{sys} - 0.01 \cdot m, U_{sys} + 0.01 \cdot m)$. Also, the generated task sets differ in the number of task sets, from 3 to 34.