Supplement of "Beyond Implicit-Deadline Optimality: A Multiprocessor Scheduling Framework for Constrained-Deadline Tasks"

Hyeongboo Baek[1], Hoon Sung Chwa[2] and Jinkyu Lee[1]

[1]Department of Computer Science and Engineering, Sungkyunkwan University (SKKU), Republic of Korea.

[2]Department of Electrical Engineering and Computer Science, The University of Michigan, U.S.A.

A. Execution-time bound under TL(τ, Fluid, Fluid-FP)

In the following lemma, we prove that the amount of execution of jobs of τ_i^{Class} in an interval of length ℓ is upper-bounded by $R_i^{\text{Class}} \cdot L_i^{\text{Class}}(\ell)$ even if τ_i^{Class} executes with a rate lower than R_i^{Class} (i.e., executing up to R_i^{Class}).

Lemma 8: For given ℓ, the amount of execution of jobs of τ_i^{Class} in an interval of length ℓ is upper-bounded by $R_i^{\text{Class}} \cdot L_i^{\text{Class}}(\ell)$ if a job of τ_i^{Class} executes with a rate up to R_i^{Class} (which coincides with the scheduling policy of TL(τ, Fluid, Fluid-FP)).

Proof: Suppose that there exist job(s) of τ_i^{Class} executing with a rate lower than R_i^{Class} in an interval of length ℓ, and the amount of execution of jobs of τ_i^{Class} in the interval is larger than $R_i^{\text{Class}} \cdot L_i^{\text{Class}}(\ell)$. We focus on the base situation where the jobs of τ_i^{Class} execute with exactly R_i^{Class} rate shown in Fig. 5, resulting in $R_i^{\text{Class}} \cdot L_i^{\text{Class}}(\ell)$ amount of execution in an interval of length ℓ. Focusing on jobs of τ_i in the interval of interest (e.g., three jobs in Fig. 5), we will show that if (i) the earliest-released job, (ii) the latest-released job, and (iii) other jobs (i.e., body jobs) in the interval execute with less than R_i rate, we need at least ℓ interval length for jobs of τ_i^{Class} in the interval to have $R_i^{\text{Class}} \cdot L_i^{\text{Class}}(\ell)$ amount of execution, which contradicts the supposition.

(Case i and ii) If a job of τ_i^{Class} executes with a rate lower than R_i^{Class}, the execution length of the job increases compared with exactly R_i^{Class} rate. Therefore, in order to have $R_i^{\text{Class}} \cdot L_i^{\text{Class}}(\ell)$ amount of execution of jobs of τ_i^{Class} in the interval, the interval should be extended to the left (Case i) and right (Case ii) directions compared to the base situation.

(Case iii) Since body jobs fully execute anyway regardless of their execution rates, we need the same interval as the base situation, in order to have $R_i^{\text{Class}} \cdot L_i^{\text{Class}}(\ell)$ amount of execution of jobs of τ_i^{Class} in the interval.

In the three cases, in order to have $R_i^{\text{Class}} \cdot L_i^{\text{Class}}(\ell)$ amount of execution of jobs of τ_i^{Class} in the interval, the interval length should be larger than or equal to that of the base situation. This contradicts the supposition. ■

B. Property of NC-ORA

We prove that binary search in Line 11 of Algo. 3 can be done without backtracking in the following lemma.

Lemma 9: Consider two execution rate assignments for τ_i^{LO}: $R_k^{\text{LO}} = A/C_k$ and $R_k^{\text{LO}'} = A'/C_k$, where $A > A'$. Then, if Lemma 5 deems τ_k^{LO} schedulable (i) with given R_k^{LO} and the execution rate and task priority assignment by Lemma 6, the same holds (ii) for τ_k^{LO} with $R_k^{\text{LO}'}$.

Proof: We show that if Eqs. (5) and (6) hold for (i), those also hold for (ii). The execution rate and task priority assignment in Lemma 6 guarantees that Eq. (5) is always satisfied for τ_k^{LO} (regardless of R_k^{LO}) since the left term of the LHS of Eq. (5) is 0, and the right term of that cannot be larger then $m \cdot (D_k - X_k^{\text{LO}})$ due to $\delta_{sum}(\tau^{\text{HI}}) = m$. When it comes to Eq. (6), the reduction from R_k^{LO} to $R_k^{\text{LO}'}$ increases the RHS of Eq. (6) exactly by $R_k^{\text{LO}} - R_k^{\text{LO}'}$ while it increases the LHS of Eq. (6) by less than $R_k^{\text{LO}} - R_k^{\text{LO}'}$, meaning the same amount of execution is extracted from τ_k^{LO} and added to τ_k^{HI} but such execution is performed in τ^{HI} with an execution rate no larger than that in τ^{LO} since $R_i^{\text{HI}} \leq R_i^{\text{LO}}$ holds for $C_i^{\text{HI}} = C_i^{\text{LO}}$. Thus the lemma holds. ■

C. Example of NC-ORA

We describe how NC-ORA Algo. 3 works with the following example.

Example 4: Recall the same task set as Example 3. After Lines 1–8 in Algo. 3 are executed, for given C_k^{HI} and C_k^{LO} of each task τ_k, the first execution rate and priority assignment illustrated in Lemma 6 constructs the followings:

for τ_1 : $\tau^{\text{HI}} = \{\tau_2, \tau_3, \tau_4, \tau_1^{\text{HI}}(T_1{=}10, C_1^{\text{HI}}{=}1.8, D_1{=}6)\}$, $\tau^{\text{LO}} = \{\tau_1^{\text{LO}}(10, 1.2, 6)\}$.

for τ_2 : $\tau^{\text{HI}} = \{\tau_3, \tau_1, \tau_4, \tau_2^{\text{HI}}(12, 2, 5)\}$, $\tau^{\text{LO}} = \{\tau_2^{\text{LO}}(12, 1, 5)\}$.

for τ_3 : $\tau^{\text{HI}} = \{\tau_1, \tau_2, \tau_4, \tau_3^{\text{HI}}(12, 2, 5)\}$, $\tau^{\text{LO}} = \{\tau_3^{\text{LO}}(12, 1, 5)\}$.

for τ_4 : $\tau^{\text{HI}} = \{\tau_1, \tau_2, \tau_3, \tau_4^{\text{HI}}(15, 3, 10)\}$, $\tau^{\text{LO}} = \{\tau_4^{\text{LO}}(15, 2, 10)\}$.

For τ_4, it is deemed schedulable by Lemma 5 with above execution rate and priority assignment, indicating that the execution rate of $\tau_4^{\text{HI}} = (15, 3, 10)$ is the minimum one to avoid deadline miss for τ_4. On the other hand, for τ_1, τ_2 and τ_3, NC-ORA cannot find any rate of C_k^{LO} that makes τ_k^{LO} deemed schedulable using binary search (Lines 11–12), meaning whole amount of C_k should be assigned to τ^{HI}. Since the summation of minimum execution rates that should be assigned to τ^{HI} is 3/6+3/5+3/5+3/10 = 2, the example task set τ satisfies the necessary condition in Lemma 7.

D. Task set generation method

We randomly generate 100,000 constrained-deadline task sets for each $m \in \{2, 4, 8, 16\}$, based on a technique proposed in [27] used in many studies, e.g., [13, 23]. For task set generation, we consider two input parameters: the number of processor (m = 4 or 8) and task utilization parameter. For a task τ_i, T_i is uniformly chosen in [1, 1000], and C_i is determined by a bimodal and exponential task utilization parameter. For a given bimodal parameter p, a value for C_i/T_i is uniformly chosen in [0, 0.5) with probability p, and for a given exponential parameter $1/\lambda$, that is chosen according to the exponential distribution whose probability density function is $\lambda \cdot exp(-\lambda \cdot x)$, where each parameter can be a value of 0.1, 0.3, 0.5, 0.7 or 0.9. Then, D_i is uniformly chosen in (C_i, T_i) as we consider constrained deadline task model.

For each task utilization, we conduct the following steps until 10,000 task sets are generated.

1) We generate a task set containing $m + 1$ tasks.
2) We check whether the generated task set passes a necessary feasibility condition [28].
3) If it passes the necessary feasibility condition, we include the task for evaluation. Then, we generate a new task set by adding a new task into the old task set and return to Step 2). Otherwise, we discard the task and return to Step 1).

We create 10,000 task sets for each task utilization model (bimodal or exponential model with a given parameter value chosen among 0.1, 0.3, 0.5, 0.7 and 0.9), in total 100,000 task sets for a given m.