

Paper Review

CBEDF

-Taeju Park and Soontae Kim

This Paper

Proposed CBEDF (Criticality Based Earliest Deadline First), based on EDF.

Shows CBEDF is better than OCBP (by S. Baruah).

Based on Dual Criticality Models (LO criticality and HI criticality task sets

CA vs Developers

Developers care about the entire systems as a whole. So they test all the task with simple method (by Experiments). With their WCET, they try to certify the system.

Certification Authorities care only about safety critical tasks, and they test only those (HI criticality) tasks with complex methods (ex. Worst case path analysis). As a result WCET measured by CA are more than that measured by Developers.

Dual Criticality Job Model

A_i : Release time of J_i

D_i : Deadline of J_i

X_i : Criticality Level of J_i (HI/LO)

C_i = {C_i(LO), C_i(HI)}

E_{Si} = Empty Slack of J_i. (E_{Si} = 0 for a Lo criticality job.)

Important Terms

- Behaviour = $(y_1, y_2, y_3, \dots, y_n)$. Collection of running time of each job J_i .
- Criticality Level $L(I) = LO$ if $y_i \leq C_i(LO)$ for all job

HI if $y_i > C_i(LO)$ for any job

- Correctness of Mixed Criticality Scheduling :

$L(I) = LO \Rightarrow$ Execute every job within deadline

$L(I) = HI \Rightarrow$ Execute all HI critical job within its deadline

- Load : Doubt :(

CBEDF : Basic Idea

All HI Critical Tasks reserve their $C_i(HI)$ units of time .

Then LO Tasks can use :

- 1) Remaining Slack : Time between $C_i(LO)$ and $C_i(HI)$ if a high critical job finishes early.
- 2) Empty Slack : Time left when all the HI critical task have reserved $C_i(HI)$ time for themselves.

Relative Location of remaining slack cannot be changed, while it can be changed for empty slack.

Offline discovery of empty slack

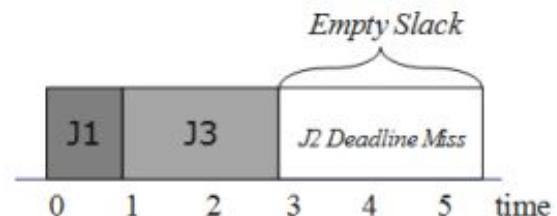
Ex :

$J_1 = (0, 2, HI, \{1, 1\})$

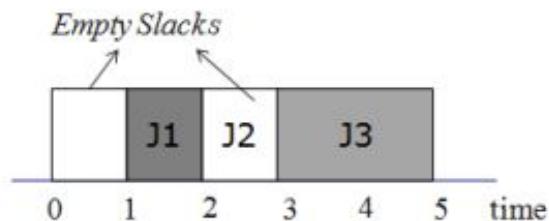
$J_2 = (2, 3, LO, \{1, 1\})$

$J_3 = (0, 5, HI, \{2, 2\})$

If you schedule based on criticality, you'd have :



But the empty slack can be realigned to get a valid schedule :

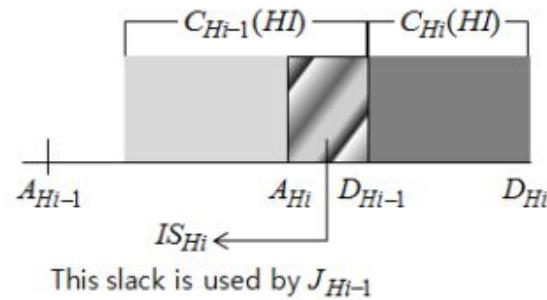


So, we can align all the HI critical jobs as close to their deadline as possible to maximize the benefit of empty slack for LO critical jobs.

If $I_{HI} = \{J_i \mid X_i = HI\}$ and I_{HI} is sorted according to the deadlines,

Then every HI critical job has $D_{Hi} - C_{Hi}(HI) - A_{Hi}$ time available for its empty slack.

This empty slack can be used by other HI critical job as shown here:



So empty slack available for a HI critical job J_{Hi} is :

$$ES_{Hi} = (D_{Hi} - D_{Hi-1}) - IS_{Hi} - C_{Hi}(HI), \text{ if +ve}$$
$$0, \text{ if -ve.}$$

And the units of time used by other HI critical jobs for J_{Hi} is :

$$IS_{Hi} = (D_{Hi} - D_{Hi+1}) + IS_{Hi+1} + C_{Hi+1}(HI); \text{ if +ve}$$
$$0; \text{ if -ve or if } (i == n) \quad (\text{DOUBT})$$

Scheduling at Runtime

Two runqueues : For LO, HI

At any time, We have four cases :

- 1) There is no ready job
- 2) There are only LO critical ready jobs
- 3) There are only HI critical ready jobs
- 4) There are both LO and HI critical ready jobs

For 1,2,3 CBEDF dispatches any ready jobs. For 4, it first checks whether slack available is free slack or not. If yes (empty or remaining slack) , then a LO critical job is dispatched.

Algorithm 1 CBEDF(RunQueue HI, RunQueue LO)

```
/* RunQueue HI : runqueue for HI criticality jobs. Jobs  
   are dispatched by EDF order */  
/* RunQueue LO : runqueue for LO criticality jobs. Jobs  
   are dispatched by EDF order */  
1: WHILE TRUE DO  
2:   IF HI.head = NULL and LO.head=NULL THEN  
3:     CONTINUE  
4:   ELSE IF HI.head=NULL THEN  
5:     DISPATCH(LO)  
6:   ELSE IF LO.head=NULL THEN  
7:     DISPATCH(HI)  
8:   ELSE  
9:     IF (isRemainingSlack or isEmptySlack) THEN  
10:      DISPATCH(LO)  
11:    ELSE  
12:      DISPATCH(HI)  
13:    END IF  
14:  END IF  
15: END WHILE
```

Example

J2 = (0, 5, H1, {2,2}, 1)

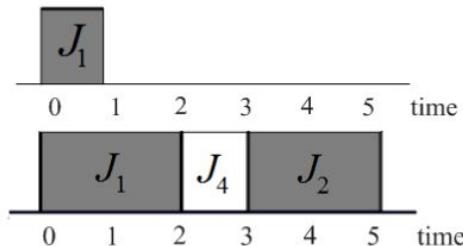
J3 = (1, 2, LO, {1,1}, 0)

J4 = (0, 5, LO, {1,1}, 0)

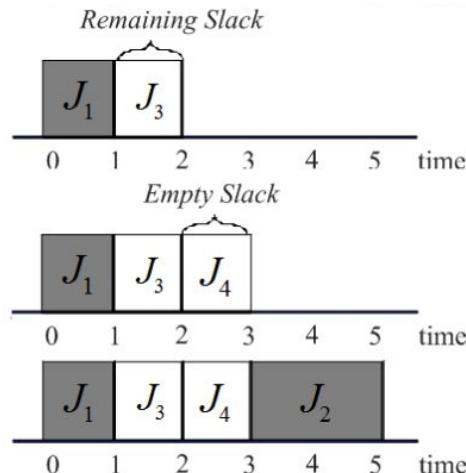
At time $t = 0$: Both LO, HI jobs ready (J_1, J_2, J_4). Since current slack is not a free slack (Check with top of HI queue : 0 for J_1) , it dispatches HI job according to EDF.

Possibilities at time 1:

- 1) J1 takes more time than $C_1(\text{LO})$. So $L(\text{I})$ becomes HI. Therefore all HI jobs reserve their time. So, Worst Case J1 finishes by 2. Now since we have an empty slack (cause top of HI is J2 having $E_s=1$), J4 executes and finishes by 3. And we then execute J2 from 3 to 5. Here, J3 missed deadline, but it doesn't matter since $L(\text{I})$ is LO.



2) J1 finishes by C_1 (LO). So we have a remaining slack (since a HI job finished earlier) between time 1 and 2. Available LO criticality job J3 (Head of Queue) finishes by 2. Then we have an empty slack for J4 since head of HI queue has J2 with 1 as Es. Therefore J4 finishes followed by J2.



Thank You

OCBP & Comparison of CBEDF & OCBP in next paper