Homework 17

(56 points)

Part 1

1. Given the task set shown below, can we guarantee schedulability at the given criticality levels with the Adaptive Mixed Criticality (AMC) approach? Show your work.

Priority	Criticality	T	C_{HI}	C_{LO}
1	HI	8	4	2
2	LO	20	9	3
3	LO	35	7	4
4	HI	49	12	10

(20 points)

2. Suppose that after an update to the system, the value of $C_{2,LO}$ changed. Which of the R^{HI}, R^{LO}, and R* computations would need to be updated, and for which tasks?

(4 points)

Part 2

We have explored EDF scheduling in this course. This homework assignment focuses on uniprocessor scheduling and will introduce another scheduling algorithm, called least-laxity-first (LLF) scheduling. LLF is also sometimes called least-slack-time-first. Like EDF, unless specified otherwise, LLF is preemptive.

First, you will look at the effects of context switch times on a task's execution. These are exaggerated for the sake of illustration. Then, LLF will be introduced, and you will compare a schedule produced by LLF to one produced by EDF. Finally, you will draw some conclusions based on these approaches and show that the non-preemptive version of LLF is not optimal.

1. Draw the EDF schedule of the task set (φ, T, C) = (1, 3, 1), (0, 9, 4.5) from t=0 to t=9.

(2 points)

2. Let's assume that the worst-case execution times that we were given only accounted for the initial load of the task and running its the code. Now we will account for context switch times. Each time a task is preempted and then begins executing again, it must first reload its state (data it was processing, etc.). Draw the EDF schedule of the task set (φ, T, C) = (1, 3, 1), (0, 4.5, 9) from t=0 to t=9, and assume that each context switch

takes 0.5 time units. Thus, after a preemption, the first 0.5 time units of execution are spent reloading the task, and any actual computation is done after that. In your schedule, indicate which portions of execution are reloading and which contribute to completing the task. Make sure that each task still does C units of actual work.

(6 points)

3. The laxity (or slack) referred to in LLF is a dynamic measure of the amount of time remaining before its deadline minus the amount of execution it must still perform before that deadline. Consider the task set (T, C) = (3, 1), (4, 2), (6, 1). At t=0, the laxity of $\tau_{1,1}$ is 3-1=2.

a. At t=0, what are the laxities of $\tau_{2,1}$ and $\tau_{3,1}$? (2 points)

b. Assume that $\tau_{1,1}$ executes for one time unit between t=0 and t=1. At t=1, what are the laxities of $\tau_{2,1}$ and $\tau_{3,1}$? (2 points)

4. To do LLF scheduling, at any given point in time, the laxity is computed for all released jobs, and the job with remaining computation and the lowest laxity is scheduled. This must be done with some granularity, so for this question, assume that the scheduler makes decisions at intervals of 1 time unit; at the end of each time unit, it reevaluates which job ought to be scheduled. Like with EDF, break any ties by task number. Draw the schedule produced by LLF for the task set (T, C) = (10, 5), (10, 5) from t=0 to t=10. At each point in time, indicate the laxity for each job. (10 points)

5. Draw the schedule produced by EDF for the task set (T, C) = (10, 5), (10, 5) from t=0 to t=10. (2 points)

6. Both preemptive EDF and preemptive LLF are optimal. How might context switch times affect the choice of which algorithm to use in practice? (2 points)

7. Like non-preemptive EDF, non-preemptive LLF is not optimal. Prove that non-preemptive LLF is not optimal. Clearly state your reasoning. (6 points)

Feedback

1. How much time did you spend completing this assignment (ignoring interruptions)?
2. How much time did you spend doing the assigned reading (ignoring interruptions)?
3. Any other feedback?