Mhd Feras Aljoudi

200412916

ENSE-452-Assignment 3

Date: Oct. 14, 2024

1. **(10 marks)** Consider a system that has three tasks with periods: 10 millisecond, 39 millisecond, and 1 second. If the WCETs have been estimated at 4 milliseconds, 12 milliseconds, and 98 milliseconds, respectively, what is the total time-loading of the system? (We are ignoring context switch time)

   Is the task set guaranteed to have a feasible schedule, by the RMS criterion? If not, what would be the *easiest* rewrite that would make the three tasks schedulable? Explain your answer

1) $P_1 = 10$ millisecond $\qquad WCET_1 = 4$ milisecond

$P_2 = 39$ millisecond $\qquad WCET_2 = 12$ millisecond

$P_3 = 1$ second $= 1000$ millisecond $\qquad WCET_3 = 98$ millisecond

total time-loading of the system = ?

$$U = \sum_{i=1}^{3} \frac{WCET_i}{P_i} = \frac{4}{10} + \frac{12}{39} + \frac{98}{1000} = 0.806$$

task set guaranteed to have a feasible schedule by RMS iff utilization $U \leq n(2^{\frac{1}{n}} - 1)$

Where n tasks is 3

$$n(2^{1/n} - 1) \Rightarrow 3(2^{\frac{1}{3}} - 1) = 0.779$$

$\Rightarrow 0.806 > 0.779$, thus the task set is not guaranteed to have a feasible schedule by RMS criterion.

To make the three tasks schedulable we need to have U less or equal to $3(2^{1/3} - 1)$, we can do that by increasing one of the period:

$$\frac{4}{x} + \frac{12}{39} + \frac{98}{1000} \leq 0.779 \Rightarrow \frac{4}{x} \leq 0.3733$$

$\Rightarrow x \geq 10.72$

So, if we changed P_1 from 10 millisecond to 10.72 millisecond or more, the task set will be guaranteed to have a feasible schedule.

test: $\frac{4}{10.72} + \frac{12}{39} + \frac{98}{1000} = 0.7788$

$$3(2^{\frac{1}{3}} - 1) = 0.779$$

$$0.7788 \leq 0.779$$

2. **(20 marks)** A preemptive system has three concurrent tasks, described by the table below (context switch time is ignored). The background, or idle task is assumed to be nonessential and is fully preemptable by all higher priority tasks.

Task	Cycle	Execution Time	Priority
TaskA	10ms	4ms	3 (highest)
TaskB	20ms	5ms	1
TaskC	40ms	10ms	2
Idle	(continuous)	5ms	—

(a) Answer the following:

i. What is the system utilization?

ii. Is this task set RMS scheduled?

iii. What is the response time for each task?

iv. Do all the tasks meet their deadlines? By how much does each task beat, or miss, its deadline.

v. Draw an execution time line for this system.

2) a) i) ET: for Execution Time
C : for Cycle

$$U = \sum_{i=1}^{3} \frac{ET_i}{C_i} = \frac{4}{10} + \frac{5}{20} + \frac{10}{40} = 0.9 \times 100\% = 90\%$$

ii) RMS says that shorter period task gets the highest priority.
In the table we see Task C has more priority (2) than Task B,
thus, the task set is NOT RMS scheduled.

iii) Task A has highest priority:
$\boxed{\text{Task A}}$ response time is: $\boxed{4ms}$
Task C priority comes after Task A:
$\boxed{\text{Task C}}$ response time is: Task A + 10ms = $\boxed{14ms}$
Task B priorit comes after Task C:
$\boxed{\text{Task B}}$ response time is: Task A + Task C + 5ms = $\boxed{19ms}$

iv) Task A: Period is 10ms and response time is 4ms

⇒ Task A meets its deadline, and it beats deadline by 10 - 4 = 6ms

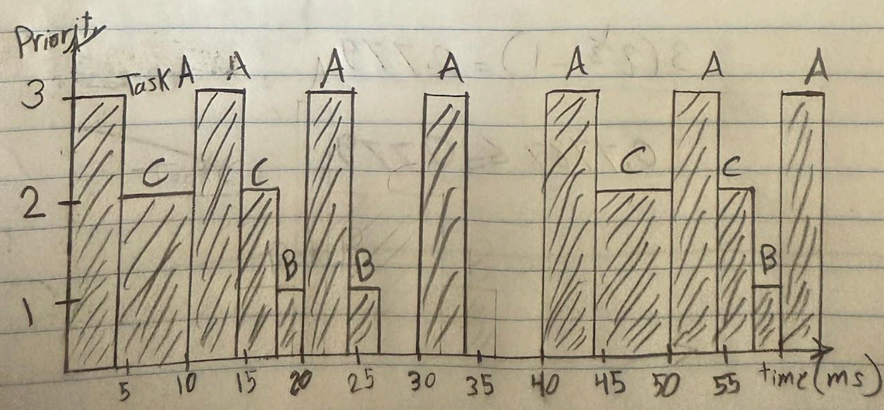
Task B: Period is 20ms & response time is 19ms

⇒ Task B meets its deadline, and it beats deadline by 20 - 19 = 1ms

Task C: Period is 40ms & response time is 14 ms

⇒ Task C meets its deadline & it beats deadline by 40 - 14 = 26ms

v)



(b) Now suppose the priorities of Task B and C are interchanged, that is, TaskB has priority 2 and TaskC has priority 1. Answer the following:

i. What is the system utilization?

ii. What is the response time for each task?

iii. Do all the tasks meet their deadlines? By how much does each task beat, or miss, its deadline.

iv. Draw an execution time line for this system.

b) i) The system utilization still same 90%

ii) Now TaskA has priority of 3, Task B is 2 & TaskC is 1

⇒ Task A response time is 4 ms

Task B response time is: TaskA + 5ms = 9 ms

Task C response time is: TaskA + TaskB + 10ms = 19ms

iii) Task A meets and beats deadline by 10 − 4 = 6 ms

TaskB meets & beats deadline by 20 − 9 = 11 ms

TaskC meets & beats deadline by 40 − 19 = 21ms

iv)