7 Written Problems

Complete the following two written problems using PCP. You may complete this question digitally or on paper. Upload a picture (.png or .jpg) or a scan (.pdf) of your solution in the base directory of your repository. Please name this file `written.<png/jpg/pdf>`.

7.1 Question 1

Assume we want to schedule the following task set {C,T} with RMS where all the tasks are independent. Assume all tasks finish within their allotted C.

$\tau_1 = (4, 11)$
$\tau_2 = (5, 16)$
$\tau_3 = (3, 31)$

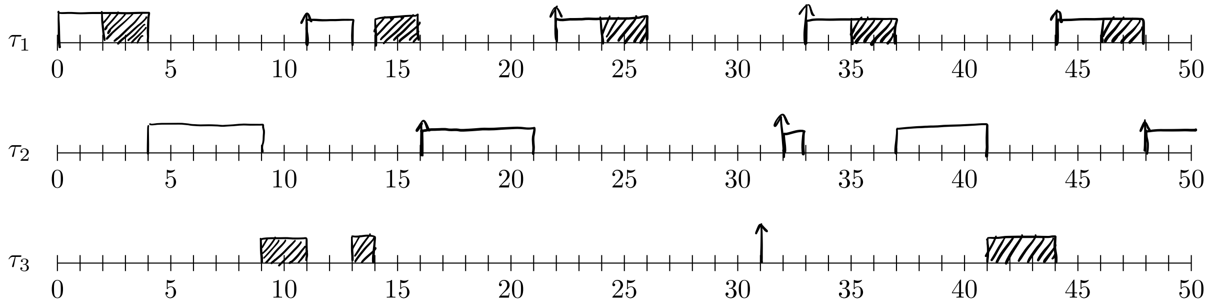
Prove that the above task set is schedulable.

$U = 4/11 + 5/16 + 3/31 = 0.7729 < U(3) = 0.77976$ ✓

Unfortunately, we can't always run our tasks in isolation. We need to share resources between each of them. We protect this data from corruption with the following scheme:

- τ_1 uses mutex m_1 for the **last two ticks** of its computation time.
- τ_3 uses mutex m_1 for all three ticks of its computation time.

Please draw the expected scheduling using PCP of this task set with the mutexes on the timelines below, until $t = 40$. Assume that locks and unlocks can be done instantaneously (0 ticks). Indicate in which ticks each task is holding a mutex by shading that tick in. Does any task miss its deadline?



7.2 Question 2

Once again, assume we want to schedule the following task set {C,T} with RMS where all the tasks are independent. Assume all tasks finish within their allotted C.

$\tau_1 = (3, 7)$
$\tau_2 = (1, 9)$
$\tau_3 = (6, 26)$

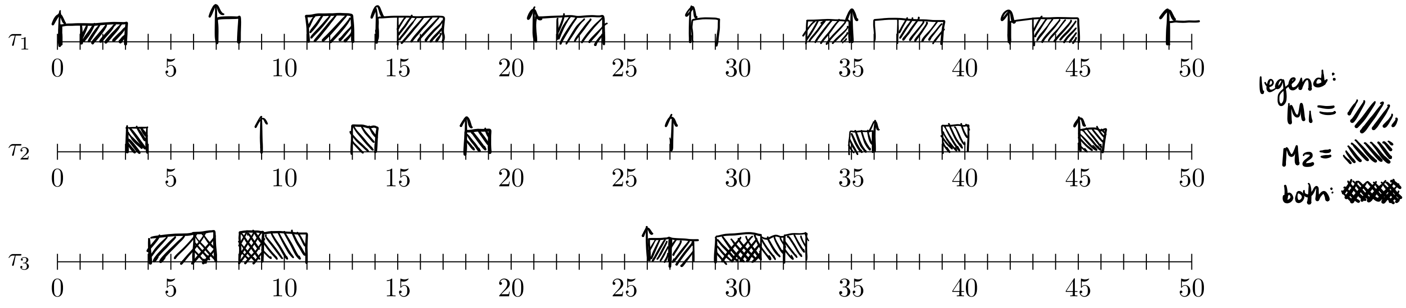
Prove that the above task set is schedulable.

$u = 3/7 + 1/9 + 6/26 = 0.770 < u(3) = 0.77976$ ✓

Once again, consider now that the threads share resources as follows:

- τ_1 uses mutex m_1 for the **last two ticks** of its computation time.
- τ_2 uses mutex m_2 for all of its computation time (lock at the beginning of the tick, unlock at the end of the tick).
- τ_3 uses mutex m_1 for the **first four ticks** of its computation time, and mutex m_2 for the **last four ticks** of its computation time. (There will be two ticks when both mutexes are locked.)

Please draw the expected scheduling using PCP of this task set with the mutexes on the timelines below until $t = 40$. Assume that locks and unlocks can be done instantaneously (0 ticks). Indicate in which ticks each task is holding a mutex by shading that tick in. Does any task miss its deadline?



8 Checkpoint

Congratulations, you have completed the checkpoint. Instructions on submitting to GitHub are in Section 13. Remember to submit the written question!