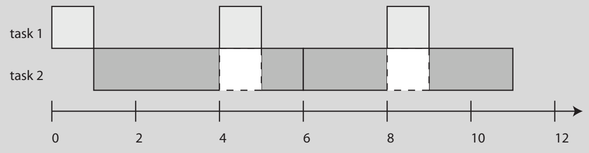
12 Scheduling — Exercises

1. This problem studies fixed-priority scheduling. Consider two tasks to be executed periodically on a single processor, where task 1 has period $p_1 = 4$ and task 2 has period $p_2 = 6$. p.314

   (a) Let the execution time of task 1 be $e_1 = 1$. Find the maximum value for the execution time e_2 of task 2 such that the RM schedule is feasible.

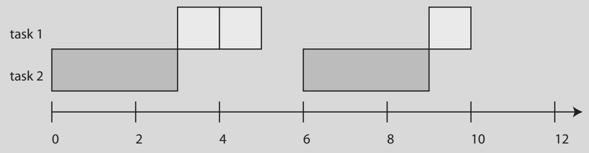
   Solution: The largest execution time for task 2 is $e_2 = 4$. The following figure shows the resulting schedule:



   The schedule repeats every 12 time units.

   (b) Again let the execution time of task 1 be $e_1 = 1$. Let non-RMS be a fixed-priority schedule that is not an RM schedule. Find the maximum value for the execution time e_2 of task 2 such that non-RMS is feasible.

   Solution: The largest execution time for task 2 is $e_2 = 3$. The following figure shows the resulting schedule:



   The schedule repeats every 12 time units.

   (c) For both your solutions to (a) and (b) above, find the processor utilization. Which is better?

Solution: From the figures above, we see that the RM schedule results in the machine being idle for 1 out of 12 time units, so the utilization is 11/12. The non-RM schedule results in the machine being idle for 3 out of 12 time units, so the utilization is 9/12 or 3/4. The RM schedule is better.

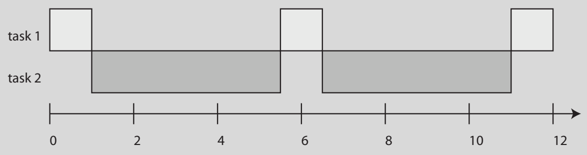
(d) For RM scheduling, are there any values for e_1 and e_2 that yield 100% utilization? If so, give an example.

Solution: Yes. For example, $e_1 = 4$ and $e_2 = 0$.

2. This problem studies dynamic-priority scheduling. Consider two tasks to be executed periodically on a single processor, where task 1 has period $p_1 = 4$ and task 2 has period $p_2 = 6$. Let the deadlines for each invocation of the tasks be the end of their period. That is, the first invocation of task 1 has deadline 4, the second invocation of task 1 has deadline 8, and so on. p.314

   (a) Let the execution time of task 1 be $e_1 = 1$. Find the maximum value for the execution time e_2 of task 2 such that EDF is feasible.

Solution: The largest execution time for task 2 is $e_2 = 4.5$. The following figure shows the resulting schedule:



The schedule repeats every 12 time units.

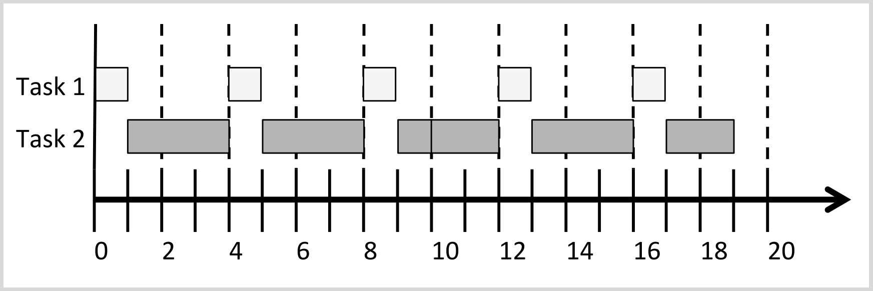
   (b) For the value of e_2 that you found in part (a), compare the EDF schedule against the RM schedule from Exercise 1 (a). Which schedule has less preemption? Which schedule has better utilization?

Solution: Comparing the schedule in (a) with the schedule in Exercise 1(a), we see that EDF has no preemption at all, while RM performs two preemptions every 12 time units. Moreover, EDF has 100% utilization, whereas RM has less.

4. This problem, formulated by Hokeun Kim, also compares RM and EDF schedules. Consider two tasks to be executed periodically on a single processor, where task 1 has period $p_1 = 4$ and task 2 has period $p_2 = 10$. Assume task 1 has execution time $e_1 = 1$, and task 2 has execution time $e_2 = 7$.

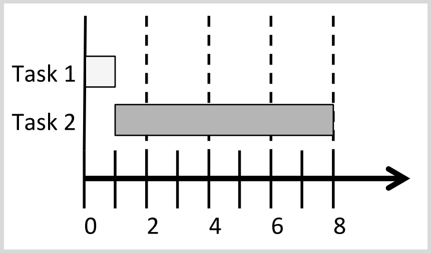
   (a) Sketch a rate-monotonic schedule (for 20 time units, the least common multiple of 4 and 10). Is the schedule feasible?

Solution: The rate monotonic schedule is feasible. The figure below shows the schedule.



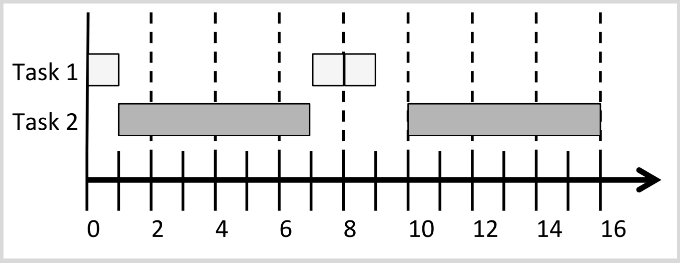
   (b) Now suppose task 1 and 2 contend for a mutex lock, assuming that the lock is acquired at the beginning of each execution and released at the end of each execution. Also, suppose that acquiring or releasing locks takes zero time and the priority inheritance protocol is used. Is the rate-monotonic schedule feasible?

Solution: No. Task 1 misses its deadline at time point 8 (the first deadline, which is met, is at time 4; the second deadline, which is not met, is at time 8).

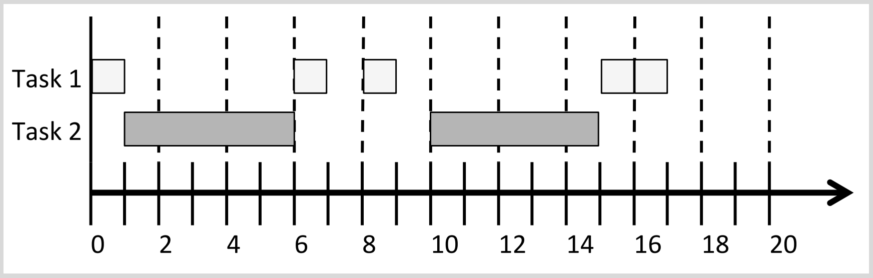


   (c) Assume still that tasks 1 and 2 contend for a mutex lock, as in part (b). Suppose that task 2 is running an **anytime algorithm**, which is an algorithm that can be terminated early and still deliver useful results. For example, it might be an image processing algorithm that will deliver a lower quality image when terminated early. Find the maximum value for the execution time e_2 of task 2 such that the rate-monotonic schedule is feasible. Construct the resulting schedule, with the reduced execution time for task 2, and sketch the schedule for 20 time units. You may assume that execution times are always positive integers.

Solution: If we reduce the execution time of task 2 to $e_2 = 6$, task 1 still misses its deadline at time point 16, as shown below:



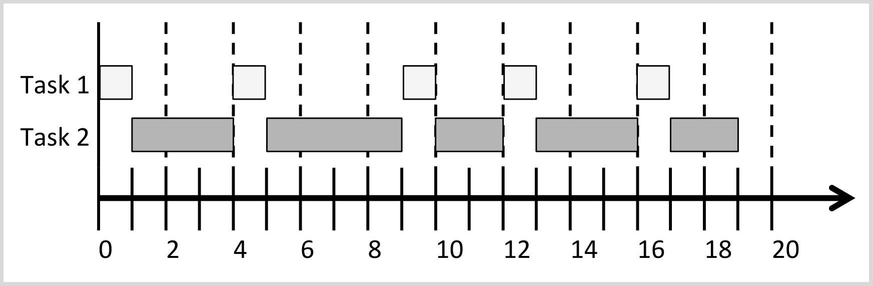
Reducing further to $e_2 = 5$ makes the schedule feasible, as shown below:



Therefore, the maximum value for the execution time of task 2 that makes the rate monotonic schedule feasible is $e_2 = 5$.

(d) For the original problem, where $e_1 = 1$ and $e_2 = 7$, and there is no mutex lock, sketch an EDF schedule for 20 time units. For tie-breaking among task executions with the same deadline, assume the execution of task 1 has higher priority than the execution of task 2. Is the schedule feasible?

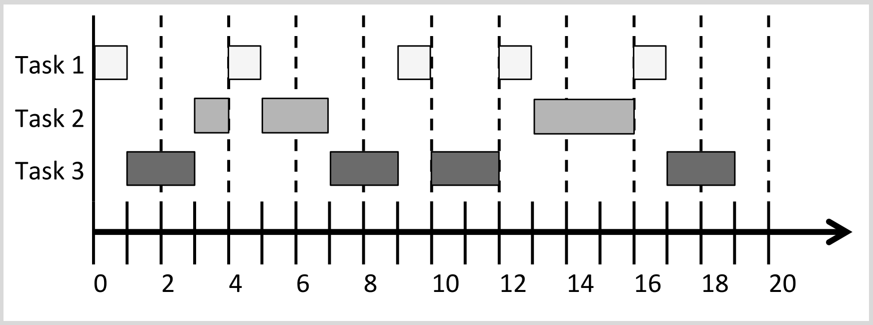
Solution: The EDF schedule is feasible. The figure below shows the schedule.



(e) Now consider adding a third task, task 3, which has period $p_3 = 5$ and execution time $e_3 = 2$. In addition, assume as in part (c) that we can adjust execution time of task 2.

Find the maximum value for the execution time e_2 of task 2 such that the EDF schedule is feasible and sketch the schedule for 20 time units. Again, you may assume that the execution times are always positive integers. For tie-breaking among task executions with the same deadline, assume task i has higher priority than task j if $i < j$.)

Solution: The total execution times of task 1 and 2 within the 20 time units window are 5 and 8, respectively. Thus, the total execution time of task 3 should be less than $20 - (5 + 8) = 7$ within the 20 time units window. Therefore, the EDF schedule becomes feasible when $e_2 = 3$. The figure below shows the schedule.


Task 1
Task 2
Task 3
0
2
4
6
8
10
12
14
16
18
20

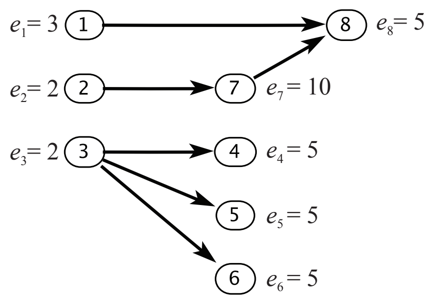
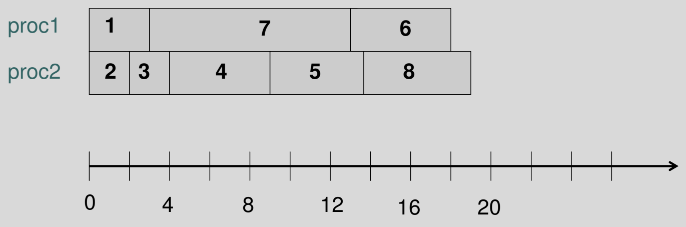


Figure 12.1: Precedence Graph for Exercise 6.

6. This problem studies scheduling anomalies. Consider the task precedence graph depicted in Figure 12.1 with eight tasks. In the figure, e_i denotes the execution time of task i. Assume task i has higher priority than task j if $i < j$. There is no preemption. The tasks must be scheduled respecting all precedence constraints and priorities. We assume that all tasks arrive at time $t = 0$.

   (a) Consider scheduling these tasks on two processors. Draw the schedule for these tasks and report the makespan.

Solution: The schedule is as shown below with a makespan of 19 units:

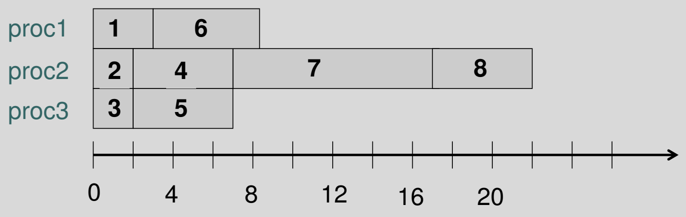


Schedule length = 19

   (b) Now consider scheduling these tasks on three processors. Draw the schedule for these tasks and report the makespan. Is the makespan bigger or smaller than that in part (a) above?

Solution: The schedule for 3 processors is as shown below with the makespan increasing to 22 units:

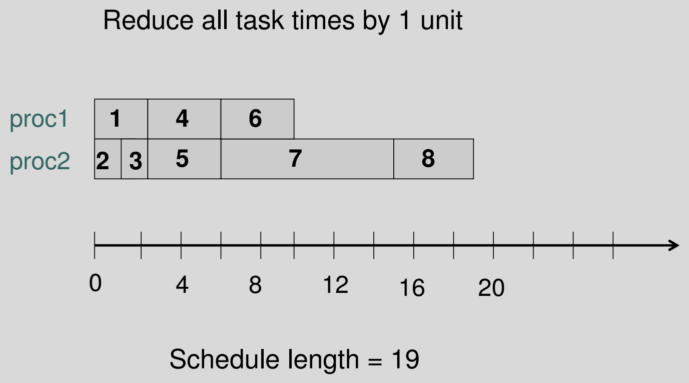
Add a processor



Schedule length = 22

(c) Now consider the case when the execution time of each task is reduced by 1 time unit. Consider scheduling these tasks on two processors. Draw the schedule for these tasks and report the makespan. Is the makespan bigger or smaller than that in part (a) above?

Solution: If each task time is reduced by 1 unit, the schedule is as shown with the makespan staying at 19 units below:


Reduce all task times by 1 unit
proc1
proc2
1
4
6
2
3
5
7
8
0
4
8
12
16
20
Schedule length = 19