RATE MONOTONIC (RM) SCHEDULING ALGORITHM

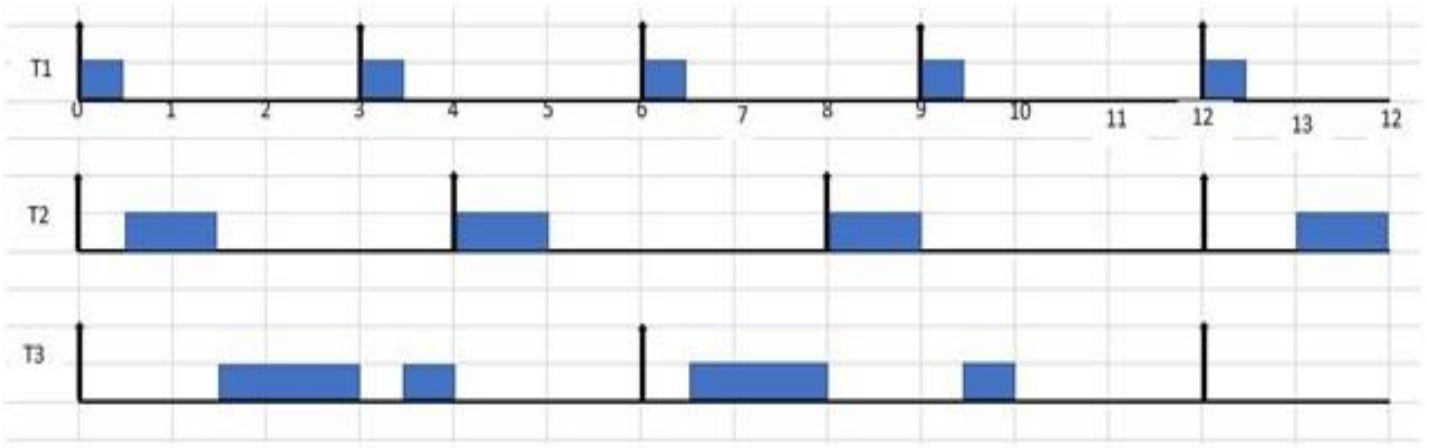
The shortest period = the highest priority

Tasks	Release time	Execution time	Time period
T1	**0**	**0.5**	**3**
T2	**0**	**1**	**4**
T3	**0**	**2**	**6**

The CPU utilization = 0.5/3 +1/4 +2/6 = 0.167+ 0.25 + 0.333 = 0.75 -> 75%

As processor utilization is less than 1 or 100% so task set is schedulable.

The timing diagram



Steps:

1. According to RM scheduling algorithm task with shorter period has higher priority so T1 has high priority, T2 has intermediate priority and T3 has lowest priority.
2. At t=0 all the tasks are released. Now T1 has highest priority so it executes first till t=0.5.
3. At t=0.5, task T2 has higher priority than T3 so it executes first for 1-time unit till t=1.5. After its completion only one task is remained in the system that is T3, so it starts its execution and executes till t=3.

4. At t=3, T1 releases, as it has higher priority than T3 so it preempts or blocks T3 and starts it execution till t=3.5. After that, the remaining part of T3 executes.
5. At t=4, T2 releases and completes it execution as there is no task running in the system at this time.
6. At t=6, both T1 and T3 are released at the same time but T1 has higher priority due to shorter period so it preempts T3 and executes till t=6.5, after that T3 starts running and executes till t=8.
7. At t=8, T2 with higher priority than T3 releases so it preempts T3 and starts its execution.
8. At t=9, T1 is released again, and it preempts T3 and executes first and at t=9.5 T3 executes its remaining part. Similarly, the execution goes on.

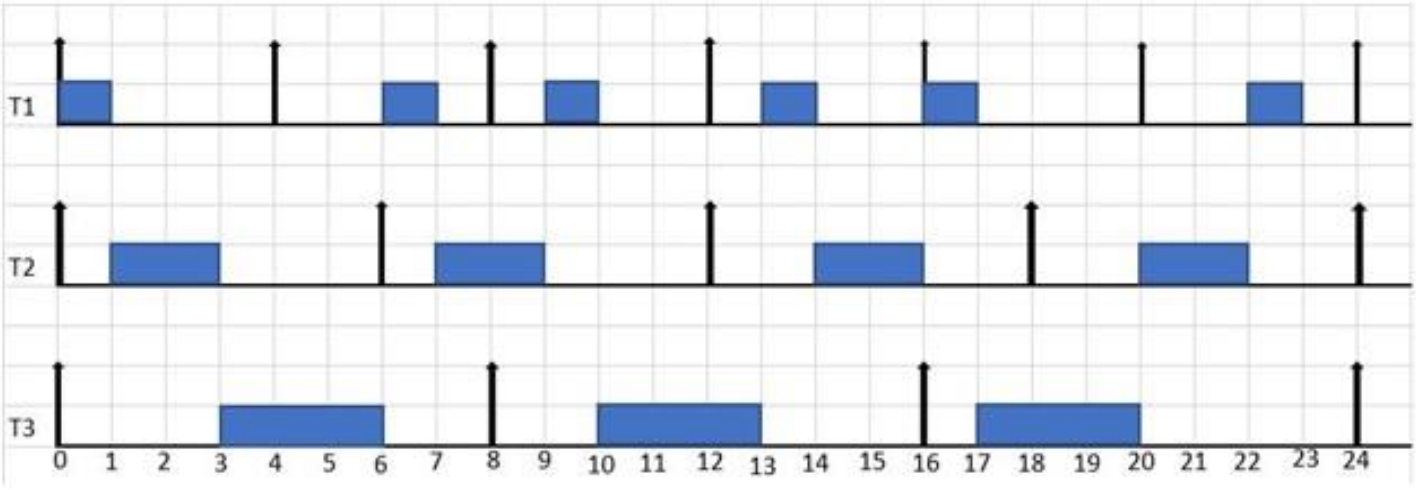
EARLIEST DEADLINE FIRST (EDF) SCHEDULING ALGORITHM

earlier deadline = highest priority

Task	Release time	Execution Time	Deadline
T1	**0**	**1**	**4**
T2	**0**	**2**	**6**
T3	**0**	**3**	**8**

The CPU utilization= 1/4 +2/6 +3/8 = 0.25 + 0.333 +0.375 = 0.95 → 95%

The timing diagram.



1. At t=0, all the tasks are released, but priorities are decided according to their deadlines so T1 has higher priority as its deadline is 4 earlier than T2 whose deadline is 6 and T3 whose deadline is 8, that's why it executes first.
2. At t=1, again deadline is compared and T2 has shorter deadline, so it executes and after that T3 starts execution but at t=4 T1 comes in the system and deadlines are compared, at this instant both T1 and T3 has same deadlines so continue to execute T3.

3. At t=6, T2 is released, now deadline of T1 is earliest than T2 so it starts execution and after that T2 begins to execute.
4. At t=8 again T1 and T2 have same deadlines i.e. t=16, so ties are broken randomly an T2 continues its execution and then T1 completes. Now at t=12 T1 and T2 come in the system simultaneously so by comparing deadlines, T1 and T2 has same deadline, and we continue to execute T3.
5. At t=13, T1 begins it execution and ends at t=14. Now T2 is the only task in the system so it completes it execution.
6. At t=16, T1 and T2 are released together, priorities are decided according to deadlines so T1 execute first as its deadline is t=20 and T3's deadline is t=24. After T1 completion T3 starts and reaches at t=17 where T2 comes in the system now by deadline comparison both have same deadline t=24 so continue to execute T3.
7. At t=20, both T1 and T2 are in the system and both have same deadline t=24 so and T2 executes. After that T1 completes it execution.

Earliest Deadline First Scheduling Algorithm

earlier deadline = highest priority.

Consider the following set of tasks in a real-time system:

Task	Arrival time	Execution time	Deadline
T1	0	4	40
T2	2	7	15
T3	5	10	20

Perform EDF scheduling on this task set.

Solution

T1 is the first task to arrive, so it starts executing.

After 2 time units, T2 arrives, whose deadline is shorter than T1. Therefore, it pre-empts T1 and T2 starts executing.

After 5 time-units, T3 arrives but T2 continues, as deadline of T3 is more than T2. Therefore T2 continues and completes its execution.

After this, now there are T1 and T3 tasks that need the processor. Since T3 has a less deadline, it starts executing and completes execution.

Finally, T1 resumes and completes its execution.

The tasks on the timing diagram are as follows: