1_2_Timing Analysis

1. Static Scheduling Policies

Fixed Priority Preemptive

- Each task has a **fixed priority**; The task dispatcher selects the task with the **highest priority**;
- All tasks are **preemptive**

Rate Monotonic Way

- a task with a **shorter period** is assigned a **higher priority** and preemptive
- $D_i = p_i$, i.e., **Deadline = period** (implicit deadline)
- implicit deadline means the requirements do not explicitely clarify the intended deadline of each task
- Schedulability test (**sufficient** condition)

$$U \leq n\left(2^{\frac{1}{n}} - 1\right)$$

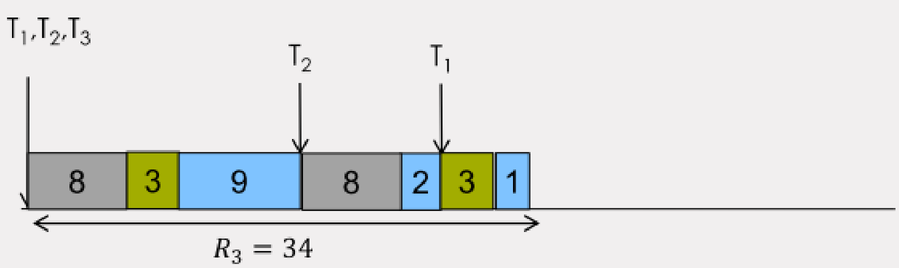
  - sufficient means "yes" is useful but a "No" says nothing.
  - if the test not hold, we should still try to find a schedule way

Consider the given fixed priority and preemptive task set running on a processor. The priorities are assigned using rate monotonic scheme. Perform a schedulability test if all the tasks are going to meet deadline under rate monotonic scheme.

Tasks	p_i (ms)	D_i (ms)	e_i (ms)	priority
T_1	30	30	3	2
T_2	20	20	8	1 (highest)
T_3	40	40	12	3

$$
\begin{aligned}
U &= \sum_{i=1}^{i=n} \frac{e_i}{p_i} \\
&= \frac{3}{30} + \frac{8}{20} + \frac{12}{40} \\
&= 0.8 \\
&> 3 \times (2^{1/3} - 1) \\
&> 0.78
\end{aligned}
\quad \Longrightarrow \quad \text{Not schedulable!}
$$

Timing at the critical instant



schedulable! we have a sufficient condition – a Yes is useful, but a No says nothing!

Deadline Monotonic Way

- a task with a **shorter relative deadline** is assigned **higher priority** and preemptive;
- $D_i = p_i$, i.e., **Deadline = period** (implicit deadline)
- Schedulability test (**sufficient** condition)

$$\sum_{i=1}^{n} \frac{e_i}{D_i} \leq n\left(2^{\frac{1}{n}} - 1\right)$$

- Deadline Monotonic Way is **optimal priority** assignment scheme, that is, if a taskset is schedulable under **any other policy**, it will certainly be schedulable under **deadline monotonic policy**

Response Timing Analysis

Preemptive case

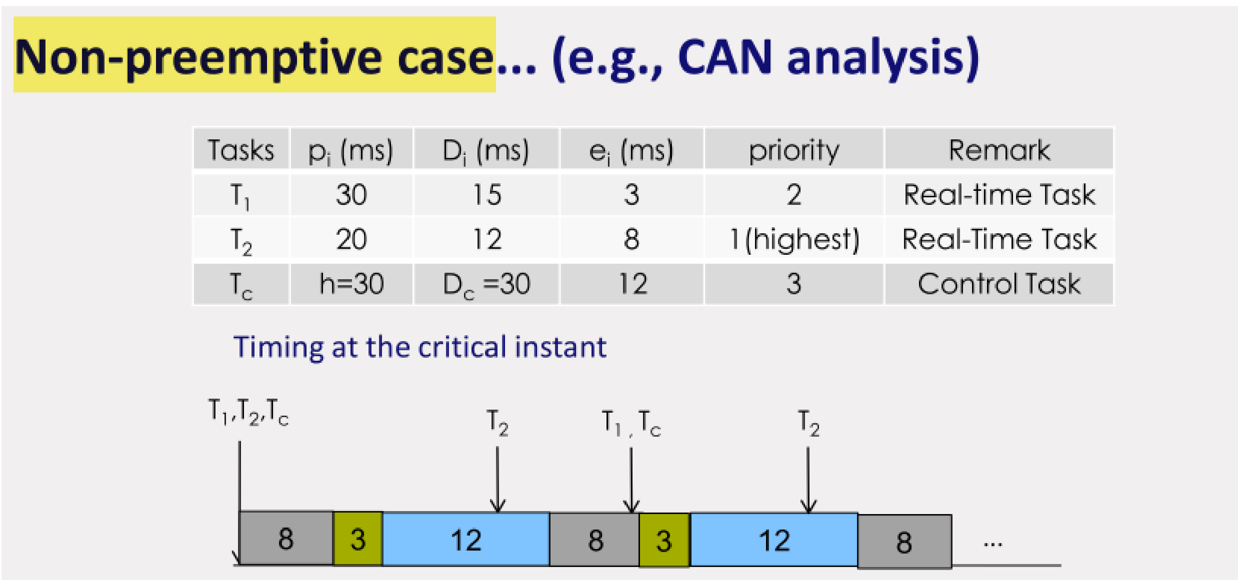
- Consider the critical instant
- Response time with fixed priority preemptive scheduling for the given task set is given by

$$R_i = e_i + \sum_{\forall j \in hp(i)} \lceil \frac{R_i}{p_j} \rceil e_j$$

- Always we needs recurrently find the response time by iteration start with $R_i^0 = 0$, we stop when $R_i^{n+1} = R_i^n$

$$R_i^{n+1} = e_i + \sum_{\forall j \in hp(i)} \lceil \frac{R_i^n}{p_j} \rceil e_j$$

Non-Preemptive Case



Non-preemptive case... (e.g., CAN analysis)

Tasks	p_i (ms)	D_i (ms)	e_i (ms)	priority	Remark
T_1	30	15	3	2	Real-time Task
T_2	20	12	8	1 (highest)	Real-Time Task
T_c	h=30	D_c =30	12	3	Control Task

2. Dynamic Scheduling Policies

Earliest Deadline First (EDF)

Dynamic Dispatcher:

- all scheduling decisions are **made online**.
- Optimal schedule and 100 utilization are possible when $D_i = p_i$ (implicit deadline)

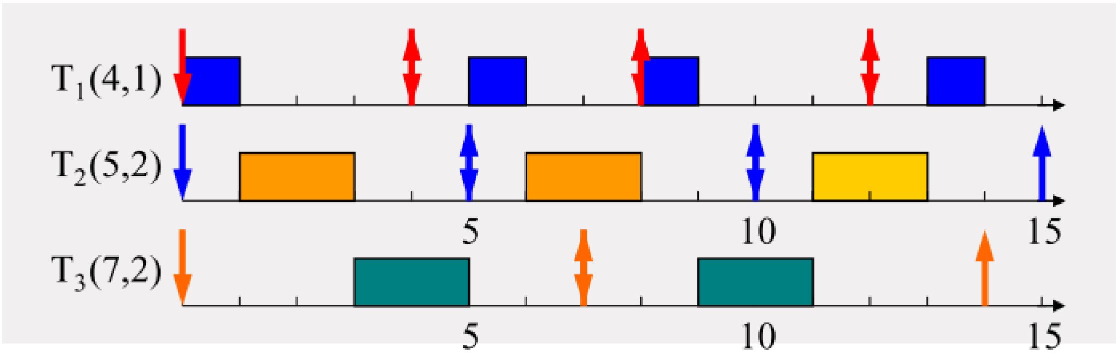
Rules:

- The task with the **shortest deadline** runs
- Preemptive

Schedulability Test (necessary and sufficient test)

$$U = \sum_i \frac{e_i}{p_i} \leq 1$$

<u>Example</u>



Property

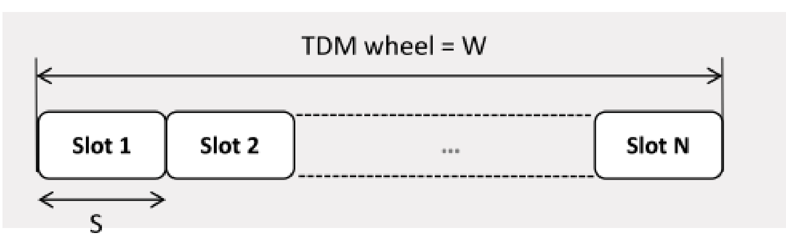
Optimal scheduling algorithm – if there is a schedule (i.e., all tasks meet deadline) for a set of real-time tasks, EDF can schedule it

3. Time-Triggered Way: Time-division multiplex access (TDMA)

3.1. Basic: Without Context Switch Overhead

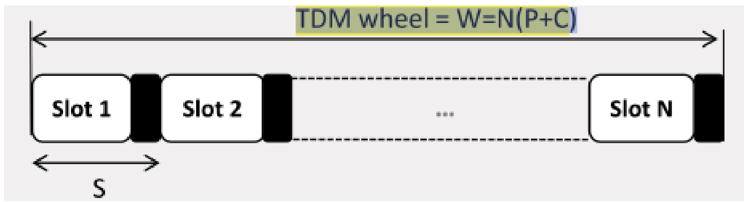
- Each slot can be assigned to **one and only one task** and the task executes in the assigned slot(s) **without any interference** from other tasks
- Predefined **time windows** for access; time windows are called **slots**
- The length of the slots **can be equal or different**
- The **order** of execution repeats and the **length of the period** is called **TDM wheel**
  - when all slots are equal:

    $$\text{TDMA wheel W} = \text{number of slots} \cdot \text{slot size} = N \cdot S$$



3.2. With Context Switch Overhead

- Switching between tasks from one slot to another **involves some time for context switching (context switch overhead)**. Over this period, operating system or micro-kernel, saves the states of the leaving task and brings the states of the new task
- It is a **preemptive scheme**
- **No application task is allowed** to run in the in this period
- now the TDM whell is:



Summary

- Static Scheduling Policies
  - Fixed Priority Preemptive: Rate Monotonic, Deadline Monotonic
  - Response Time Analysis: preemptive case + iteration
- Dynamic Scheduling Policies
  - EDF
- TDMA
  - Slots
  - Context Switch Overhead