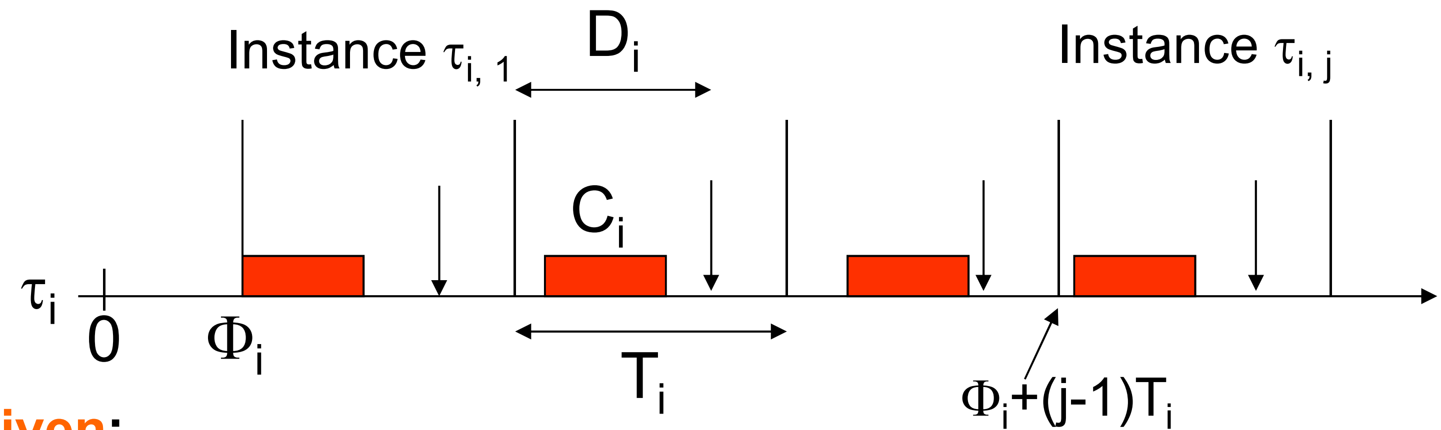
Embedded Systems 17

REVIEW: Periodic scheduling



- **Given:**
  - A set of periodic tasks $\Gamma = \{\tau_1, \ldots, \tau_n\}$ with
    - phases Φ_i (arrival times of first instances of tasks),
    - periods T_i (time difference between two consecutive activations)
    - relative deadlines D_i (deadline relative to arrival times of instances)
    - computation times C_i

  $\Rightarrow$ *j* th instance $\tau_{i,j}$ of task τ_i with
    - arrival time $a_{i,j} = \Phi_i + (j-1)\, T_i$,
    - deadline $d_{i,j} = \Phi_i + (j-1)\, T_i + D_i$,
- **Find a feasible schedule**
    - start time $s_{i,j}$ and
    - finishing time $f_{i,j}$

REVIEW: An example for periodic scheduling

	τ_1	τ_2
Φ_i	0	0
T_i	3	4
C_i	2	2
D_i	3	4

$T_1 \cdot T_2 = 12$

Within 12 units:

$\frac{12}{3} = 4$ executions of T_1

$\frac{12}{4} = 3$ executions of T_2

- No feasible schedule for single processor.

$4 \times 2 = 8$ units by T_1

$3 \times 2 = 6$ units by T_2

14 units computation within 12 units impossible!

REVIEW: Processor utilization

$$U = \sum_{i=1}^{n} \frac{C_i}{T_i}.$$

- Define $U_{bnd}(A) = \inf \{U(\Gamma) \mid \Gamma$ is not schedulable by algorithm $A\}$.
- If $U_{bnd}(A) > 0$ then a simple, sufficient criterion for schedulability by A can be based on processor utilization:
  - If $U(\Gamma) < U_{bnd}(A)$ then Γ is schedulable by A.
  - However, if $U_{bnd}(A) < U(\Gamma) \leq 1$, then Γ may or may not be schedulable by A.
- **Theorem:** A set of periodic tasks $\tau_1, \ldots, \tau_n$ with $D_i = T_i$ is schedulable with EDF iff $U \leq 1$.

EDF and processor utilization factor

- **Theorem:** A set of periodic tasks $\tau_1, ..., \tau_n$ with $D_i = T_i$ is schedulable with EDF iff $U \leq 1$.

"=>" . Let $T = T_1 \cdot \ldots \cdot T_n$

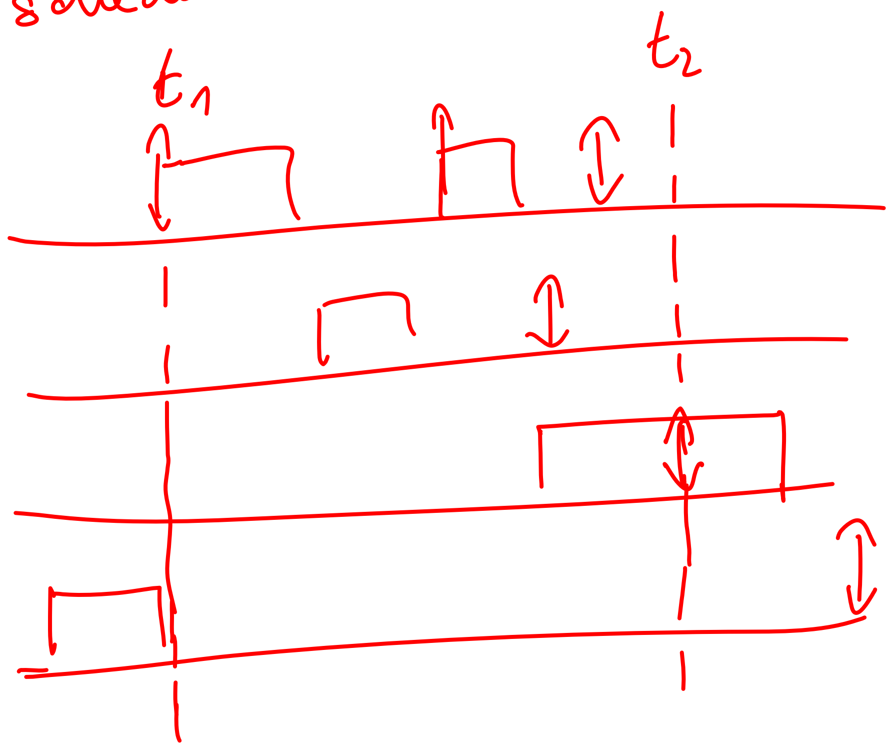
. $\sum_{i=1}^{n} \frac{T}{T_i} \cdot C_i$ time taken by task set within T

$= \sum_{i=1}^{n} \frac{C_i}{T_i} \cdot T = U \cdot T$

Assume $U > 1$. Then $UT > T$ and task set is not schedulable.

"$\Leftarrow$": Assume $U \le 1$ and the task set is <u>not</u> EDF-schedulable.

- Let t_2 be the earliest time in the EDF-schedule when a task misses its deadline.



Let $[t_1, t_2]$ be the largest interval s.t.

- no idle times
- only instances with deadline $\le t_2$ are executed.

Claim: The tasks executed in $[t_1, t_2]$ have arrival times $\geq t_1$

Case 1: The processor was idle directly before t_1

$\Rightarrow$ No unfinished tasks with arrival $< t_1$

Case 2: The task running directly before t_1 has a deadline $\leq t_2$

$\Rightarrow$ Contradiction to maximality of $[t_1, t_2]$

Case 3: The task running directly before t_1 has a deadline $> t_2$

Due to EDF, no task with arrival $< t_1$ and deadline $\leq t_2$ left since all tasks in $[t_1, t_2]$ have deadline $\le t_2$

$\Rightarrow$ all tasks in $[t_1, t_2]$ have arrived $\ge t_1$.

Time overflow at t_2:

$$(t_2 - t_1) < \sum_{\substack{a_{i,j} \ge t_1 \\ d_{i,j} \le t_2}} C_i$$

$$= \sum_{i=1}^{n} \left\lfloor \frac{t_2 - t_1}{T_i} \right\rfloor \cdot C_i$$

$$\le \sum_{i=1}^{n} \frac{t_2 - t_1}{T_i} \cdot C_i = (t_2 - t_1) \cdot \sum_{i=1}^{n} \frac{C_i}{T_i}$$

$$= (t_2 - t_1) \cdot U$$

$$\boxed{\Rightarrow U > 1}$$

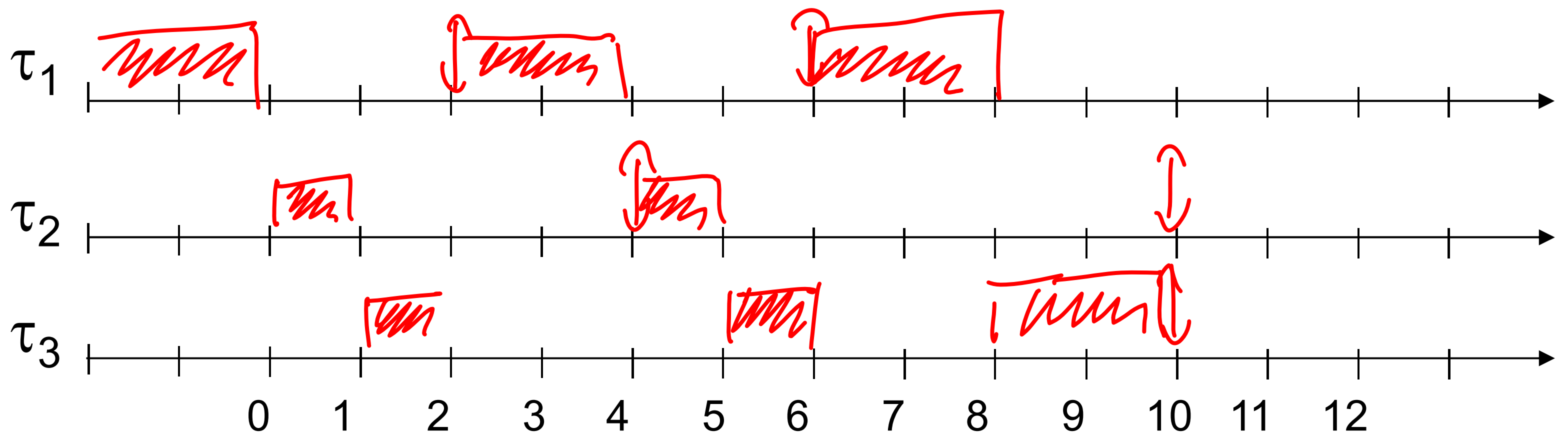
Rate monotonic scheduling (RM)

- Rate monotonic scheduling (RM) (Liu, Layland ’73):
  - Assign fixed priorities to tasks τ_i:
    - priority(τ_i) = $1/T_i$
    - I.e., priority reflects release rate
  - Always execute ready task with highest priority
  - Preemptive: currently executing task is preempted by newly arrived task with shorter period.

Example for RM (1)

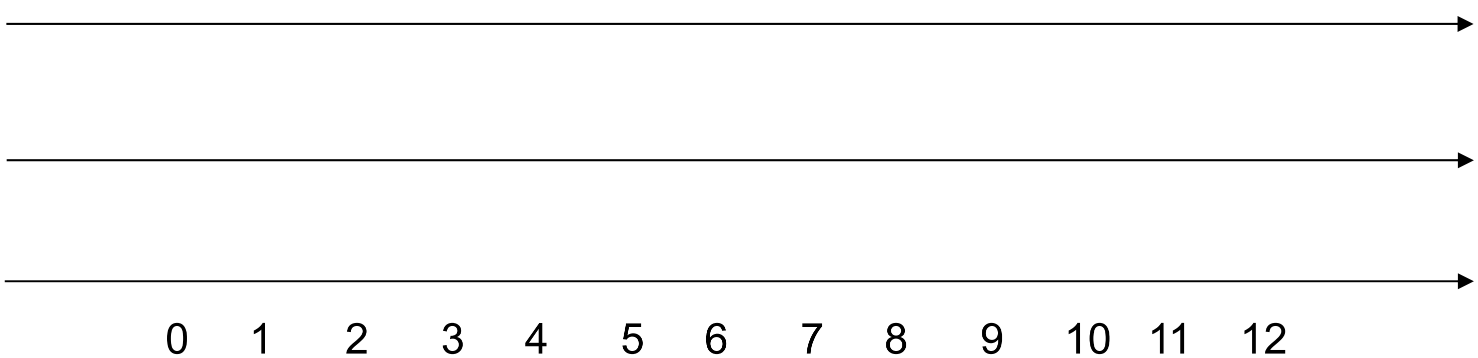
	τ_1	τ_2	τ_3
Φ_i	0	0	0
T_i	4	6	12
C_i	2	1	4
D_i	4	6	12

priority (τ_1) > priority (τ_2)
> priority (τ_3)



Example for RM (2)

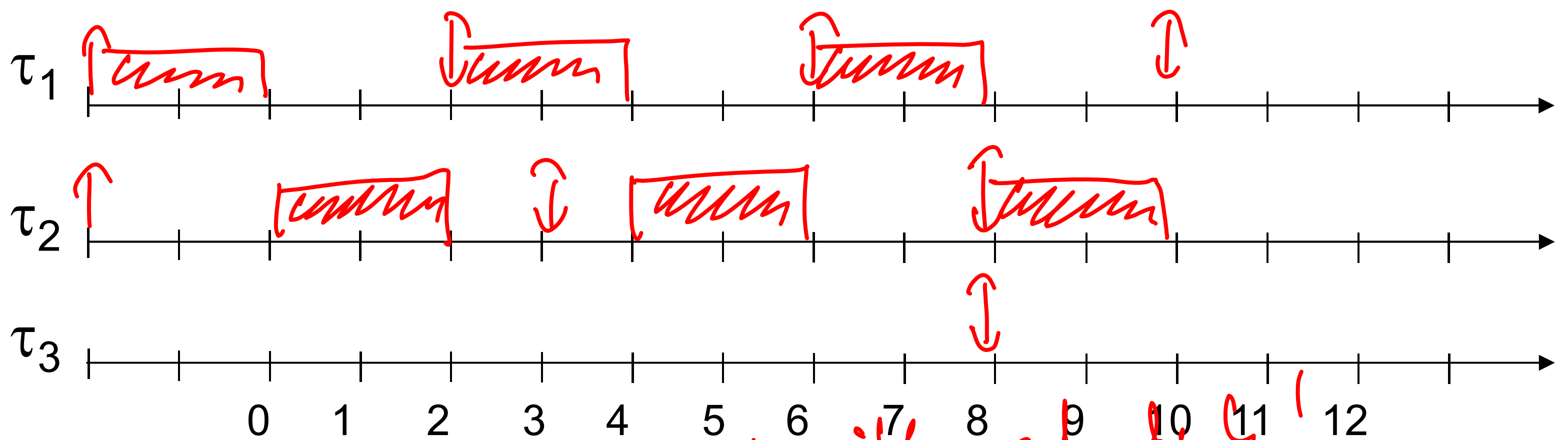
	τ_1	τ_2	τ_3
Φ_i	0	0	0
T_i	4	5	10
C_i	2	2	1
D_i	4	5	10



Example for RM (2)

	τ_1	τ_2	τ_3
Φ_i	0	0	0
T_i	4	5	10
C_i	2	2	1
D_i	4	5	10

$$U = \frac{2}{4} + \frac{2}{5} + \frac{1}{10} = 1$$



Non-feasible schedule!

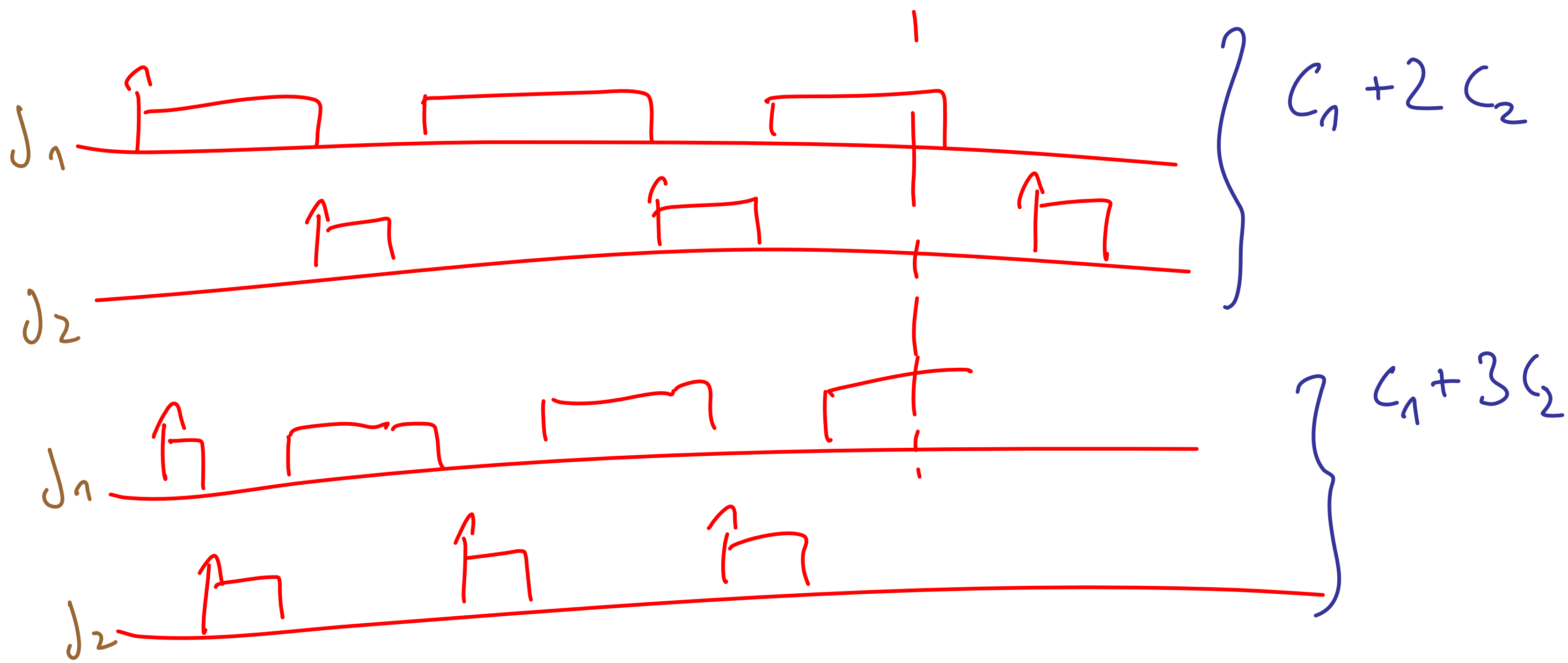
Optimality of Rate Monotonic Scheduling

- **Theorem (Liu, Layland, 1973):**
  RM is optimal among all fixed-priority scheduling algorithms.

- **Def.:** The **response time $R_{i,j}$** of an instance j of task i is the time (measured from the arrival time) at which the instance is finished: $R_{i,j} = f_{i,j} - a_{i,j}$.
- The critical instant of a task is the time at which the arrival of the task will produce the largest response time.

REVIEW: Response times and critical instants

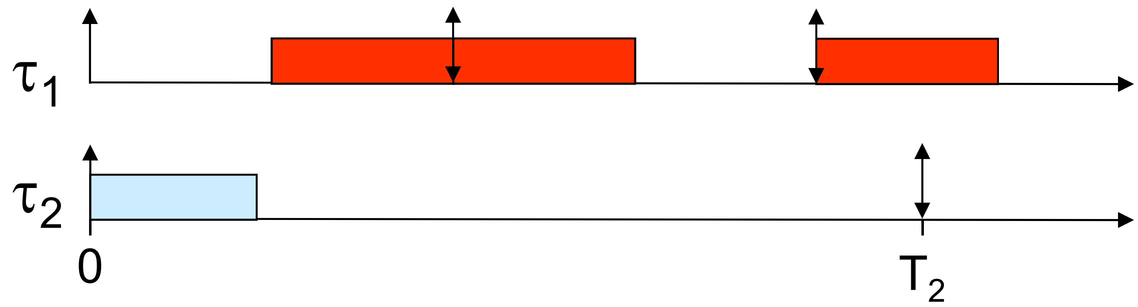
- **Observation**:
  For RM, the critical instant t of a task τ_i is given by the time when $\tau_{i,j}$ arrives together with all tasks $\tau_1, ..., \tau_{i-1}$ with higher priority.



Response times and critical instants

- For our “worst case task sets” we focus on the critical instants where an instance of a task arrives together with all higher priority tasks.
- A task set is schedulable, if the response time at these critical instants is not larger than the relative deadline.

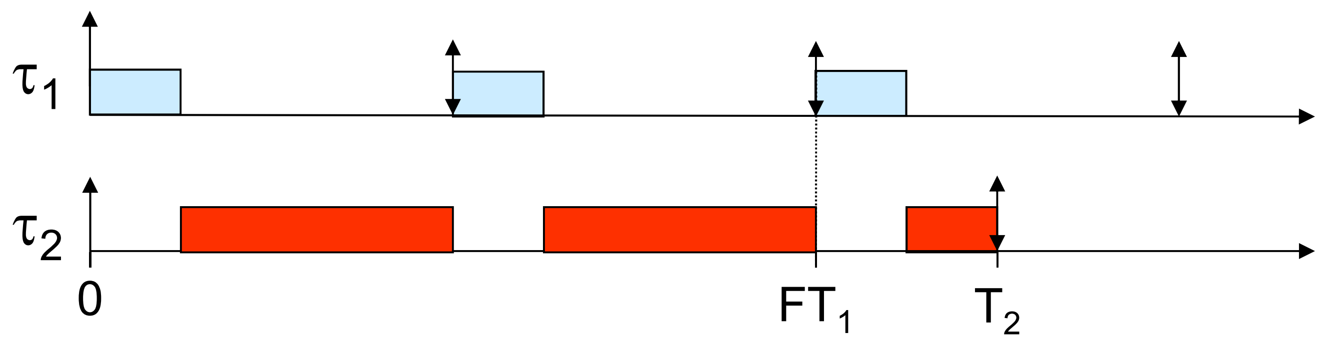
Non-RM Schedule



Schedule feasible iff $C_1 + C_2 \leq T_1$

RM-Schedule

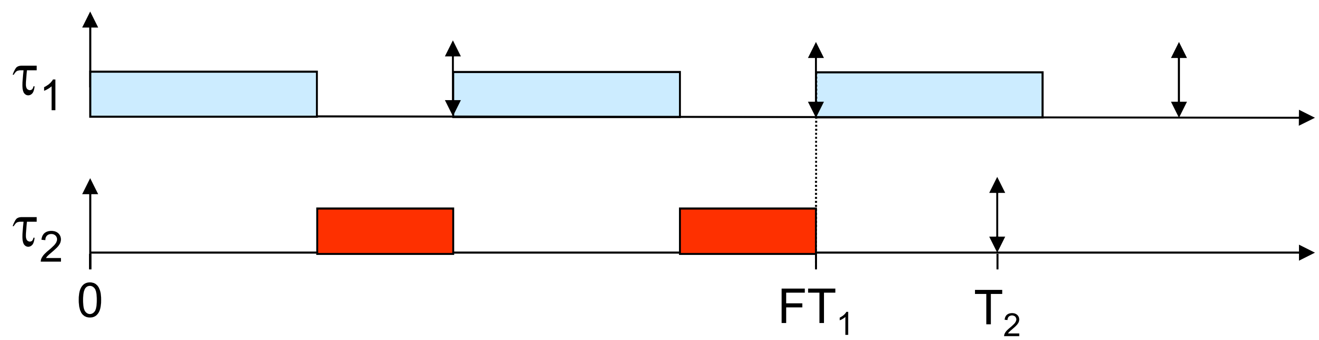
- Let $F = \lfloor T_2 / T_1 \rfloor$ be the number of periods of τ_1 entirely contained in T_2.
- **Case 1:**
  - The computation time C_1 is short enough, so that all requests of τ_1 within period of τ_2 are completed before second request of τ_2.
  - I.e. $C_1 \leq T_2 - F\,T_1$



Schedule feasible if $(F+1)C_1 + C_2 \leq T_2$

RM-Schedule

- **Case 2:**
  - The second request of τ_2 arrives when τ_1 is running.
  - I.e. $C_1 \geq T_2 - F\,T_1$



Schedule feasible if $FC_1 + C_2 \leq FT_1$

Proof of Liu/Layland

We show: If task set is schedulable by Non-RM
$\Rightarrow$ schedulable by RM

Case 1: $C_1 \le T_2 - F T_1$

$C_1 + C_2 \le T_1 \quad \overset{?}{\Longrightarrow} \quad (F+1) C_1 + C_2 \le T_2$

$\Rightarrow F \cdot C_1 + F C_2 \le F T_1$

$\Rightarrow F \cdot C_1 + C_2 \le F T_1$
$F \geq 1$

$\Rightarrow (F+1) C_1 + C_2 \le F T_1 + C_1$

$\Rightarrow (F+1) C_1 + C_2 \le T_2$

$\Rightarrow C_1 \le T_2 - F T_1$

Case 2: $C_1 \geq T_2 - F T_1$

$$C_1 + C_2 \leq T_1 \overset{?}{\Longrightarrow} F C_1 + C_2 \leq F T_1$$

$$\Rightarrow F C_1 + F C_2 \leq F T_1$$

$$\underset{F \geq 1}{\Rightarrow} F C_1 + C_2 \leq F T_1$$

REVIEW: Processor utilization as a schedulability criterion

- Given: a scheduling algorithm A
- Define $U_{bnd}(A) = \inf \{U(\Gamma) \mid \Gamma$ is not schedulable by algorithm A$\}$.

- If $U_{bnd}(A) > 0$ then a simple, sufficient criterion for schedulability by A can be based on processor utilization:
  - If $U(\Gamma) < U_{bnd}(A)$ then Γ is schedulable by A.
  - However, if $U_{bnd}(A) < U(\Gamma) \leq 1$, then Γ may or may not be schedulable by A.

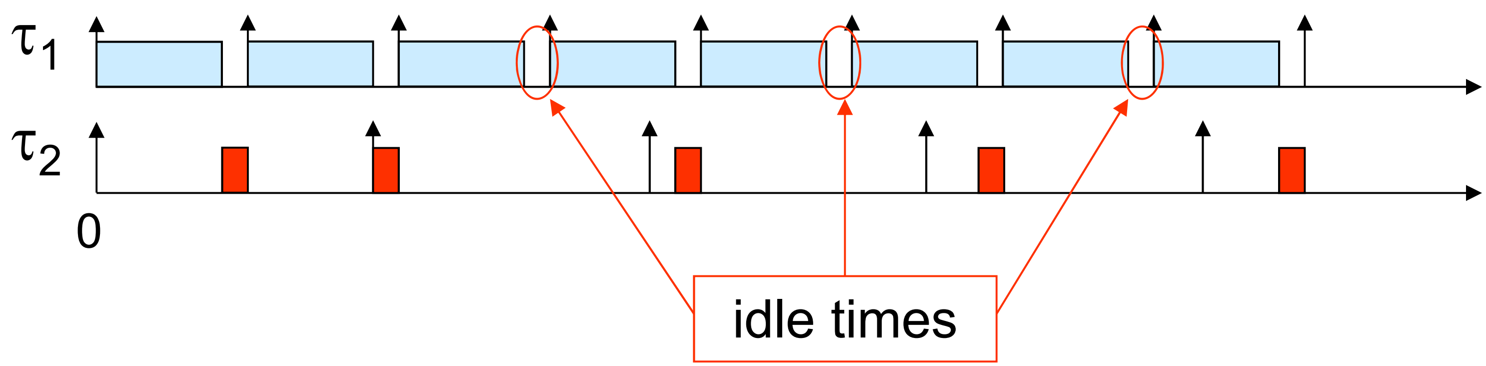
Computation of U_{bnd}(RM)

- We focus on task sets with 2 tasks (general case: n tasks)
- Computation of $U_{bnd}(RM, 2) = \inf \{U(\Gamma) \mid \Gamma \text{ is not schedulable by RM}, |\Gamma| = 2\}$.

- **Idea:**
  - Construct set of tasks with following properties:
    1. Set of tasks is schedulable by RM.
    2. Any increase of computation times makes the set of tasks non-schedulable.
    3. Processor utilization is minimal under properties 1. and 2.

Computation of U_{bnd}(RM, 2)

Worst case situation constructed for 2 processes:



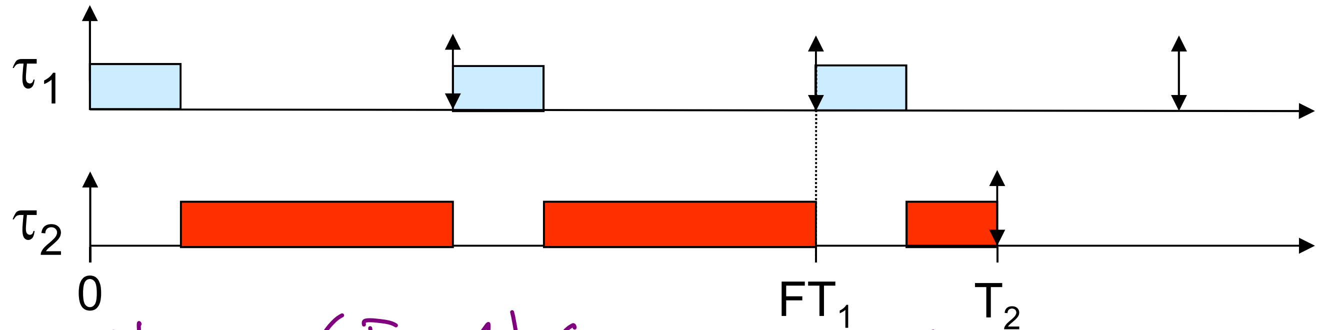
Computation of U_{bnd}(RM, 2)

- Consider a set of 2 periodic tasks τ_1 and τ_2 with $T_1 \leq T_2$ $\Rightarrow$ priority(τ_1) > priority(τ_2).
- We consider the critical instant when τ_1 and τ_2 arrive at the same time.
- We construct a worst case scenario where any increase of computation times destroys schedulability and minimize the processor utilization.

  This is done by manipulating

  - computation times C_1 and C_2 and
  - T_1 and T_2 (more precisely T_2 / T_1)

Case 1: $C_1 \leq T_2 - F\,T_1$



feasible if $(F+1)\,C_1 + C_2 \leq T_2$

max value of C_2: $C_2 = T_2 - (F+1)\,C_1$

minimize U by adjusting C_1:

$$U = \frac{C_1}{T_1} + \frac{C_2}{T_2} = \frac{C_1}{T_1} + \frac{T_2 - (F+1)\,C_1}{T_2} =$$

$$= 1 + \frac{C_1}{T_1} - \frac{C_1}{T_2}(F+1)$$

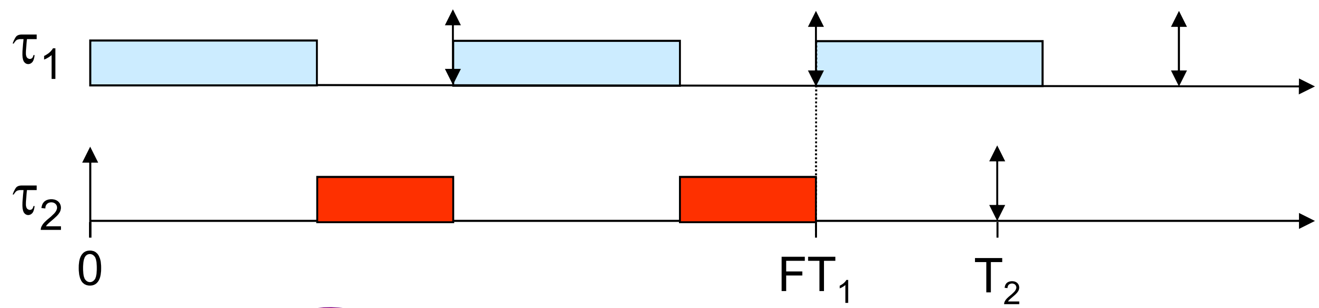
$$= 1 + \frac{C_1}{T_2}\underbrace{\left(\frac{T_2}{T_1} - \left(\left\lfloor \frac{T_2}{T_1} \right\rfloor + 1\right)\right)}_{<0}$$

$\Rightarrow$ U decreases monotonically with C_1

$\Rightarrow$ U minimal for $C_1 = T_2 - F T_1$

(in case 1)

Case 2: $C_1 \geq T_2 - F\,T_1$



feasible if $F C_1 + C_2 \leq F T_1$

$\Rightarrow$ Max value for C_2: $C_2 = (T_1 - C_1) \cdot F$

$$U = \frac{C_1}{T_1} + \frac{C_2}{T_2} = \frac{C_1}{T_1} + \frac{(T_1 - C_1) \cdot F}{T_2} =$$

$$= \frac{T_1}{T_2} F + \frac{C_1}{T_2} \underbrace{\left(\frac{T_2}{T_1} - F \right)}_{\geq 0} \qquad (*)$$

$\Rightarrow$ U increases monotonically with C_1

$\Rightarrow$ U is minimal for $C_1 = T_2 - FT_1$ (in case 2)

$\Rightarrow$ $\underline{C_1 = T_2 - FT_1}$ in both cases

Manipulating T_2/T_1

$$U = \frac{T_1}{T_2} F + \frac{c_1}{T_2}\left(\frac{T_2}{T_1} - F\right) \qquad (*)$$

$$= \frac{T_1}{T_2} F + \frac{T_2 - F T_1}{T_2}\left(\frac{T_2}{T_1} - F\right)$$

$c_1 = T_2 - F T_1$

$$= \frac{T_1}{T_2}\left[F + \left(\frac{T_2}{T_1} - F\right)\left(\frac{T_2}{T_1} - F\right)\right]$$

$\left(\text{Let } G = \frac{T_2}{T_1} - F\right)$

$$= \frac{T_1}{T_2}\left(F + G^2\right)$$

$$U = \frac{T_1}{T_2}\left(F + G^2\right) =$$

$$= \frac{F + G^2}{\frac{T_2}{T_1}} = \frac{F + G^2}{F + G} \qquad (**)$$

$$= \frac{F+G - \left(G - G^2\right)}{F+G} = 1 - \frac{G(1-G)}{F+G}$$

Since $G = \frac{T_2}{T_1} - \left\lfloor \frac{T_2}{T_1} \right\rfloor$, $0 \le G < 1$

$\Rightarrow G(1-G) \geq 0$

$\Rightarrow$ U increases monotonically with F

$\Rightarrow$ U minimal for minimal value of F

$$\left.\begin{array}{l} F = \left\lfloor \frac{T_2}{T_1} \right\rfloor \\ T_1 \le T_2 \end{array}\right\} \Rightarrow F = 1$$

$$U = \frac{F + G^2}{F + G} \quad (**), \quad F = 1$$

$$= \frac{1 + G^2}{1 + G}$$

Minimize U over G:

$$\frac{dU}{dG} = \frac{2G \cdot (1+G) - (1+G^2)}{(1+G)^2} = \frac{G^2 + 2G - 1}{(1+G)^2}$$

$$\frac{dU}{dG} = 0 \Rightarrow G^2 + 2G - 1 = 0$$

$$G \in \left\{ \frac{-2 + \sqrt{4+4}}{2}, \frac{-2 - \sqrt{4+4}}{2} \right\}$$

$$G \in \{ -1 + \sqrt{2}, -1 - \sqrt{2} \}$$

$$\Rightarrow G = -1 + \sqrt{2}, \text{ since } 0 \le G.$$

$$U = \frac{1+G^2}{1+G} = \frac{1+(-1+\sqrt{2})^2}{1+(-1+\sqrt{2})} =$$

$$= \frac{4-2\sqrt{2}}{\sqrt{2}} = 2(\sqrt{2}-1) \approx \underline{\underline{0.83}}$$

Computation of U_{bnd}(RM)

- Result for two processes:
  Any set of two periodic tasks with a processor utilization factor ≤ $\boxed{U_{bnd} = 2(2^{1/2} - 1)}$ can be scheduled by RM.

- Similarly, for the general case of **n** processes the following can be shown:
  Any set of n periodic tasks with a processor utilization factor ≤ $\boxed{U_{bnd} = n(2^{1/n} - 1)}$ can be scheduled by RM.

Computation of U_{bnd}(RM)

- Any set of n periodic tasks with a processor utilization factor ≤ $U_{bnd} = n(2^{1/n} - 1)$ can be scheduled by RM.

- U_{bnd} is decreasing with n and converges to ln 2 ≈ 0.69 for n → ∞