EDF Schedulability Bound

- An EDF schedule S has a job misses its deadline at t if and only if $\sum u_i > 1$
  - Don't need to worry about tasks with ready times after T_i and deadlines after T_i, since in EDF they won't compete for CPU time.

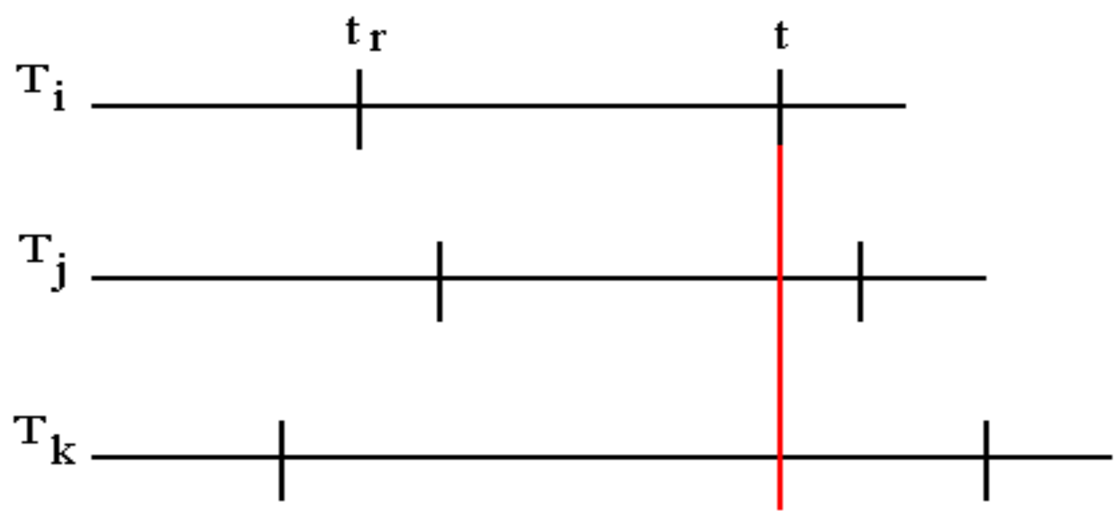
EDF Proof

- How much execution time is needed, total for T_i ?

$$\frac{t-\varphi_i}{P_i}e_i$$

- For the rest of the tasks?

$$\sum e_j \left\lfloor \frac{t-\varphi_j}{P_j} \right\rfloor$$



EDF Proof

- If the deadline is missed at *t*,

$$\frac{t-\varphi_i}{P_i}e_i + \sum e_j \left\lfloor \frac{t-\varphi_j}{P_j} \right\rfloor > t - \varphi_i$$

- Take away ϕ for convenience, and the floor function

$$\frac{t-\varphi_i}{P_i}e_i + \sum e_j \left\lfloor \frac{t-\varphi_j}{P_j} \right\rfloor$$

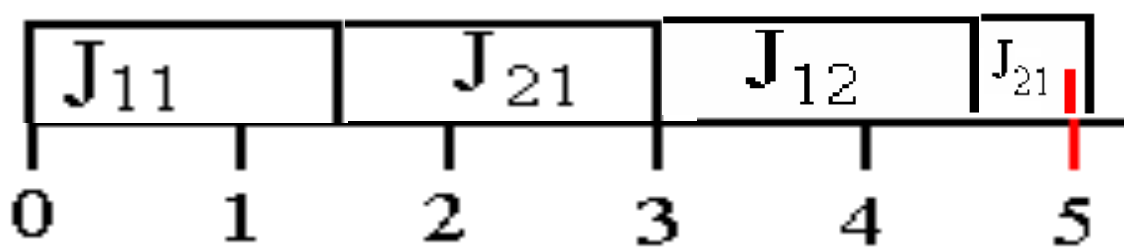
$$\leq \frac{te_i}{P_i} + \sum e_j \frac{t}{P_j} \leq t\sum_{i=1}^{n} \frac{e_i}{P_i} = t\sum u_i \boxed{\Rightarrow \sum u_i > 1}$$

Rate Monotonic Analysis Principle

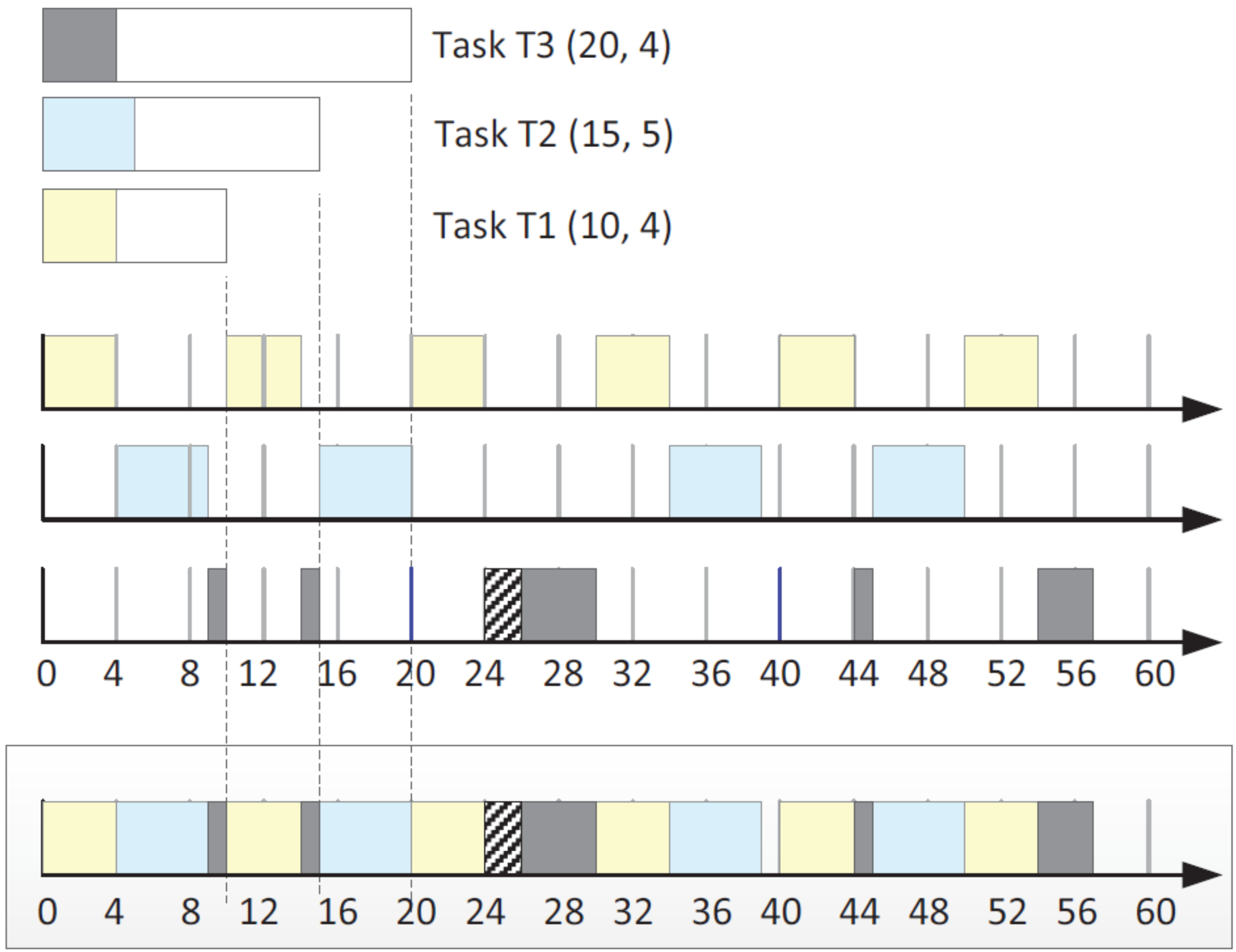
- Rate Monotonic Analysis (RMA) principle states that a task with the highest request rate (job release frequency) should take the highest priority.
- A system with a task set T is *schedulable* if all the tasks in T can be scheduled without breaking any deadlines.

Rate Monotonic: Schedulability

- Can Rate Monotonic guarantee a schedule if the total utilization is ≤ 1?
- See the following example
  - Let $\tau_1 = 1.51 / 3$, $\tau_2 = 2 / 5$, total utilization is 0.903
  - τ_1 has a higher priority than τ_2



RMA: a non-schedulable task set



The Optimality of Rate Monotonic

- RM is optimal among all fixed priority schedules
  - *Algorithm G* produces feasible *S with* $P_1 < P_2$ *but* $Pr_2 > Pr_1$
  - RM will produce a feasible *S'*

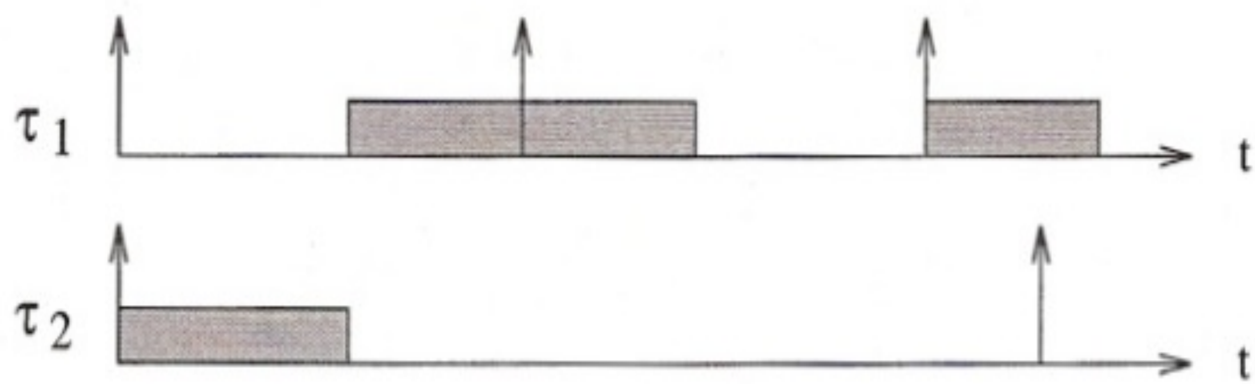


Figure 4.4 Tasks scheduled by an algorithm different than RM.

RM Optimality Idea

- RM is optimal in systems with fixed priorities
  - Suppose we have a schedule where:
    $P_1 < P_2$, but τ_2 is scheduled first, the opposite of RM
  - This implies that $C_1 + C_2 \leq P_1$.
  - But because P_1 is less that P_2, $C_1 + C_2 \leq P_2$
    - If you can finish in a shorter period, you can finish in the longer
  - Need to show RM can schedule both τ_1 *and* τ_2
    - *Before* $P_{2\,,}$ RM need to schedule τ_2 *and* many τ_1

RM Optimality Proof

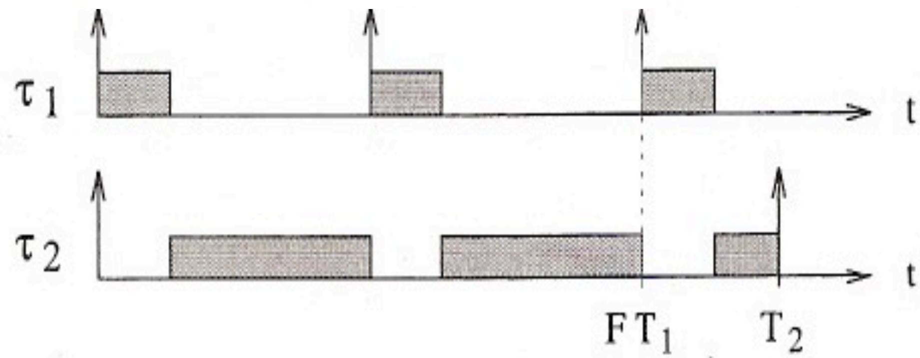
$$F = \left\lfloor \frac{T_2}{T_1} \right\rfloor$$

Since we assumed that $C_1 \leq T_2 - FT_1$, we have

$$(F+1)C_1 + C_2 \leq FT_1 + C_1 \leq T_2,$$

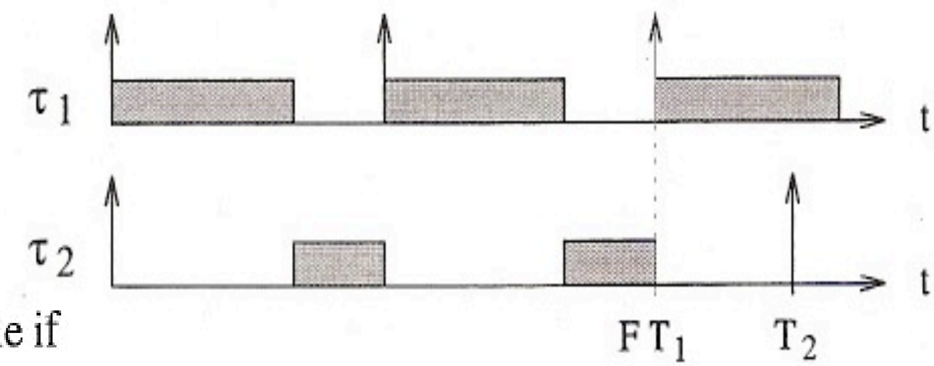
case (a)

$C_1 < T_2 - FT_1$



case (b)

$C_1 \geq T_2 - FT_1$



we can see that the task set is schedulable if

$$FC_1 + C_2 \leq FT_1.$$

Rate Monotonic: Schedulability

Calculate the total execution time (e) before a deadline

- How much time is needed by
  1) Itself
  2) All higher priority jobs

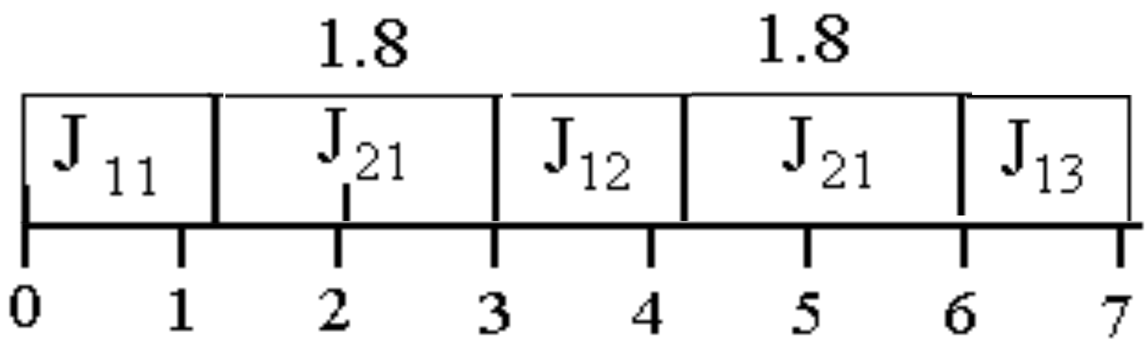
$$e_2 + e_1 \left\lceil \frac{T_2}{T_1} \right\rceil$$

- EDF doesn't have "higher priority" so the total time needed is only

$$e_2 + e_1 \left\lfloor \frac{T_2}{T_1} \right\rfloor$$

System Schedulability for RM

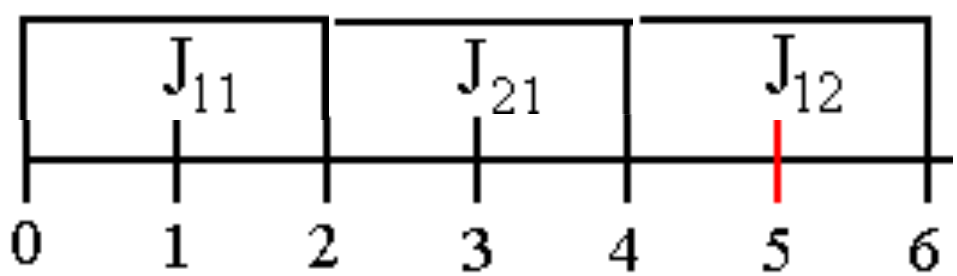
- Let a system be 1.2/3, 3.6/7.
- The total time needed by t=7 is 3(1.2) + 3.6 = 7.2, so we think it can't meet its deadline, but:



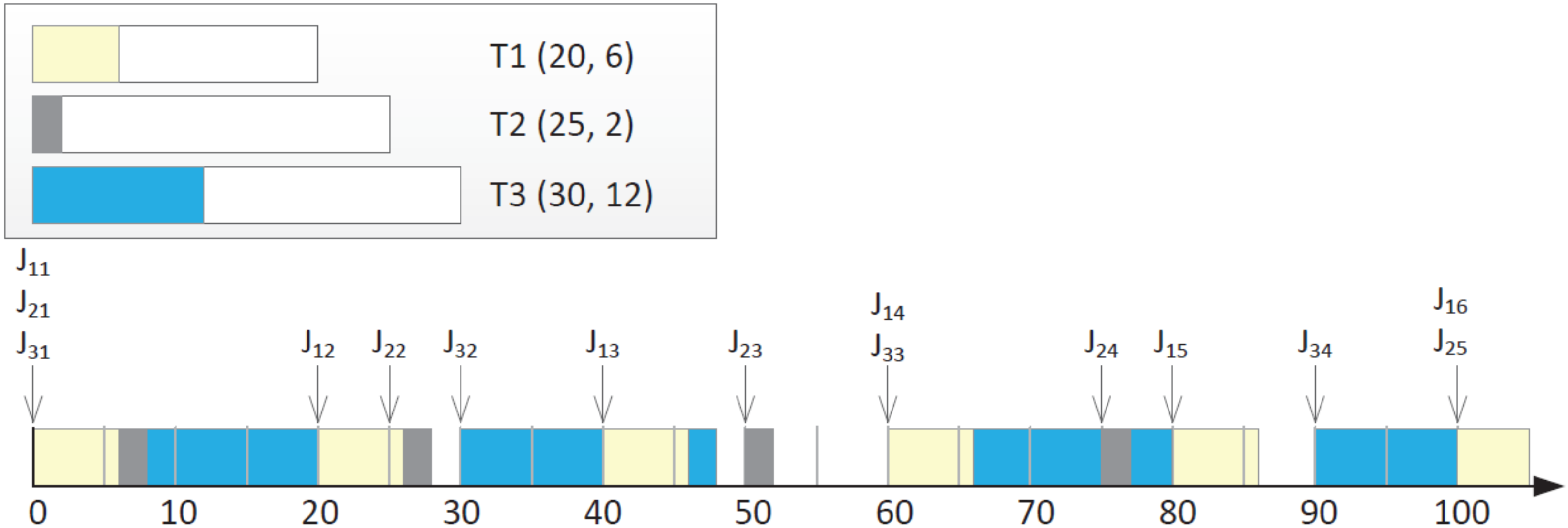
- But J_{21} completes on-time

RM Schedulability Condition

- Q: If $\sum u_i \leq$?, then RM can find a feasible schedule.
  - 1/4, 2/5 requires $\sum c_i = 4$, so RM can schedule and meet the deadline of 5.
  - 2/4, 2/5 requires $\sum c_i = 6$ by time 5 which is after the deadline



Critical Instant



Theorem 16.2 (Existence of Critical Instant). *Given a set $\mathbb{T}$ of independent, preemptable periodic tasks where every job completes before the next job of the same task is released, a critical instant of any task T_i occurs when one of its job J_{ix} is released at the same time with a job in every higher-priority task in $\mathbb{T}$.*

Corollary: *Every low priority task, if its first job can meet the deadline, then all its other jobs will as well.*

RM Theorem

- [Liu and Layland 1973]

  If n is the number of tasks, RM can find a feasible schedule if

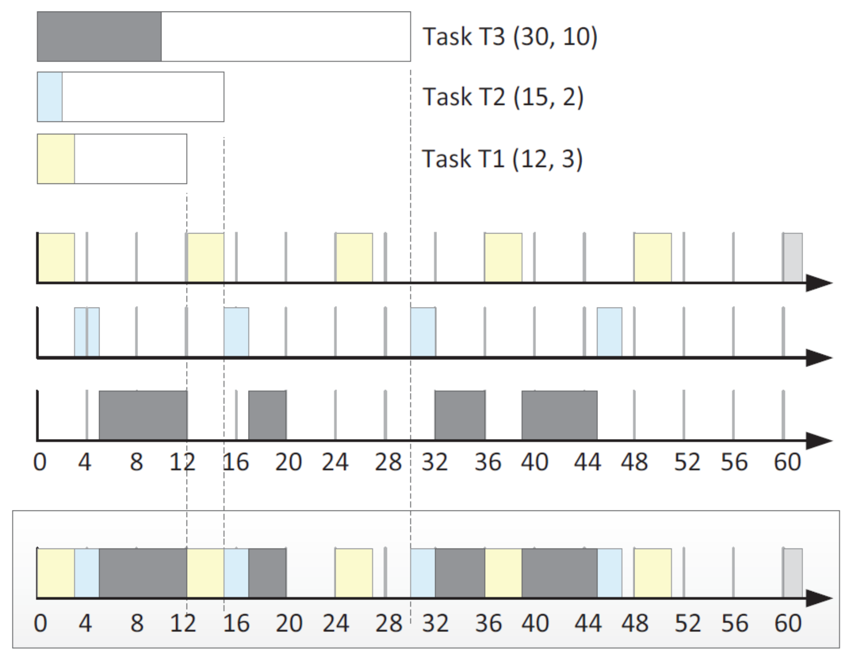
  $$\sum u_i \le n\left(2^{1/n} - 1\right)$$

  - *n=2, the value is 0.828427…*
  - As n approaches infinity, $\lim_{n \to \infty} n\left(2^{1/n} - 1\right) = \ln 2$
  - If the periods are always multiples, then we can always meet the deadline as long as the bound ≤ 1
    - Jobs with longer periods can utilize time not used by smaller period jobs

Rate-Monotonic Analysis: example

k	1	2	3	4	$\cdots$	∞
$k(2^{1/k} - 1)$	1.00	0.83	0.78	0.76	$\cdots$	$log_e(2) = 0.69$

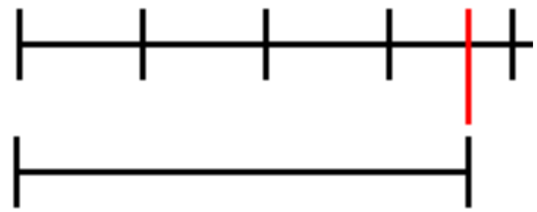
$$\mathbb{T}_4 = \begin{cases} T_1 = (12, 3), \\ T_2 = (15, 2), \\ T_3 = (30, 10). \end{cases}$$



RM Lower Bound Proof

- Assume we have two tasks, where $T_1 < T_2$.
- The total time we need to execute the tasks is

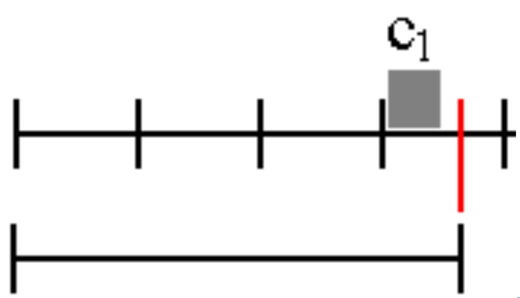
$$c_2 + \left\lceil \frac{T_2}{T_1} \right\rceil c_1 \le T_2$$

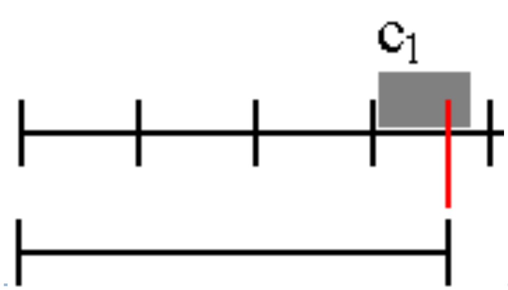


- Two possible cases:

$$c_1 \le T_2 - T_1 \left\lfloor \frac{T_2}{T_1} \right\rfloor$$

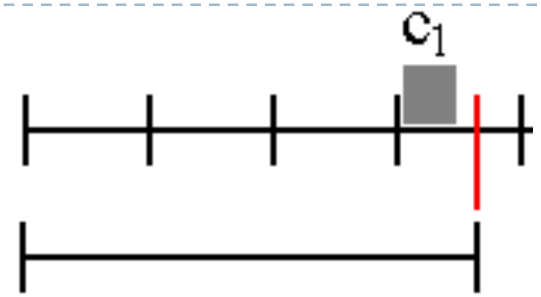
$$c_1 \ge T_2 - T_1 \left\lfloor \frac{T_2}{T_1} \right\rfloor$$





RM Lower Bound Proof

- Case 1: $c_1 \le T_2 - T_1 \left\lfloor \frac{T_2}{T_1} \right\rfloor$



  - This implies $c_2 = T_2 - c_1 \left\lceil \frac{T_2}{T_1} \right\rceil$
  - Calculating utilization as $c_2/T_2 + c_1/T_1$ and substitution yields…

RM Lower Bound Proof

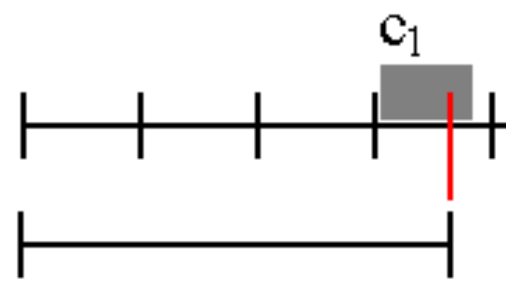
$$\frac{1}{T_2}\left(T_2 - c_1\left\lceil\frac{T_2}{T_1}\right\rceil\right) + \frac{c_1}{T_1} = 1 + c_1\left(\frac{1}{T_1} - \frac{1}{T_2}\left\lceil\frac{T_2}{T_1}\right\rceil\right)$$

- Note that $\frac{1}{T_1} - \frac{1}{T_2}\left\lceil\frac{T_2}{T_1}\right\rceil$ is negative, so

  utilization decreases as c_1 increases

RM Lower Bound Proof

- Case 2: $c_1 \geq T_2 - T_1 \left\lfloor \frac{T_2}{T_1} \right\rfloor$



  - Now, utilization is

$$\left\lfloor \frac{T_2}{T_1} \right\rfloor \left(\frac{T_1}{T_2} - \frac{c_1}{T_2} \right) + \frac{c_1}{T_1} = \frac{T_1}{T_2} \left\lfloor \frac{T_2}{T_1} \right\rfloor + c_1 \left(\frac{1}{T_1} - \frac{1}{T_2} \left\lfloor \frac{T_2}{T_1} \right\rfloor \right)$$

  - Utilization increases as c_1 increases

RM Lower Boundary Proof

- Conclusion: Boundary condition occurs at

$$c_1 = T_2 - T_1 \left\lfloor \frac{T_2}{T_1} \right\rfloor$$

I.e. $C_I = T_2 - FT_I$

In this situation, depicted in Figure 4.7, the largest possible value of C_2 is

$$C_2 = (T_1 - C_1)F,$$

and the corresponding upper bound U_{ub} is

$$\begin{aligned} U_{ub} &= \frac{C_1}{T_1} + \frac{C_2}{T_2} = \frac{C_1}{T_1} + \frac{(T_1 - C_1)F}{T_2} = \\ &= \frac{T_1}{T_2}F + \frac{C_1}{T_1} - \frac{C_1}{T_2}F = \\ &= \frac{T_1}{T_2}F + \frac{C_1}{T_2}\left[\frac{T_2}{T_1} - F\right]. \end{aligned} \tag{4.4}$$

Since the quantity in square brackets is positive, U_{ub} is monotonically increasing in C_1 and, being $C_1 \geq T_2 - FT_1$, the minimum of U_{ub} occurs for

$$C_1 = T_2 - FT_1.$$

$$
\begin{aligned}
U &= \frac{T_1}{T_2}F + \frac{C_1}{T_2}\left(\frac{T_2}{T_1} - F\right) = \\
&= \frac{T_1}{T_2}F + \frac{(T_2 - T_1F)}{T_2}\left(\frac{T_2}{T_1} - F\right) = \\
&= \frac{T_1}{T_2}\left[F + \left(\frac{T_2}{T_1} - F\right)\left(\frac{T_2}{T_1} - F\right)\right].
\end{aligned} \tag{4.5}
$$

To simplify the notation, let $G = T_2/T_1 - F$. Thus,

$$
\begin{aligned}
U &= \frac{T_1}{T_2}(F + G^2) = \frac{(F + G^2)}{T_2/T_1} = \\
&= \frac{(F + G^2)}{(T_2/T_1 - F) + F} = \frac{F + G^2}{F + G} = \\
&= \frac{(F + G) - (G - G^2)}{F + G} = 1 - \frac{G(1 - G)}{F + G}.
\end{aligned} \tag{4.6}
$$

Since $0 \leq G < 1$, the term $G(1 - G)$ is nonnegative. Hence, U is monotonically increasing with F. As a consequence, the minimum of U occurs for the minimum value of F; namely, $F = 1$. Thus,

$$
U = \frac{1 + G^2}{1 + G}. \tag{4.7}
$$

Value of U as a function of G

G	U
0	
0.1	0.918182
0.2	0.866667
0.3	0.838462
0.4	0.828571
0.5	0.833333
0.6	0.85
0.7	0.876471
0.8	0.911111
0.9	0.952632
1	1

$$G = T_2 / T_1 - F$$

0.405	0.828487544
0.41	0.828439716
0.415	0.828427562
0.42	0.828450704
0.425	0.828508772
0.43	0.828601399
0.435	0.828728223
0.44	0.828888889
0.445	0.829083045

Rate Monotonic Schedulability

- Question: Can we just check the lowest priority for the first period?
  - 3/7, 6/11, and 1/77, the total execution for the lowest priority:
    - 1+3*11 + 6*7 = 76
  - But 3/7, 6/11, can't be scheduled.
- The answer is "we need to check the schedulability of every task".

Testing if a system is schedulable

- Distinguish between a *sufficient* and a *necessary* condition. Which is more restrictive?

	sufficient	necessary
EDF	≤1	≤1
RM	0.69	$n(2^{1/n} - 1)$?

- $n\left(2^{1/n} - 1\right)$ is pessimistic