THE UNIVERSITY OF KANSAS

Lecture 19: Real-Time Process Scheduling

EECS 388 – Spring 2023

Lecture notes are based on slides created by Prof. Mohammad Alian and Prof. Heechul Yun

Agenda

- Utilization Bound
- Exact Schedulability analysis

Liu & Layland Bound

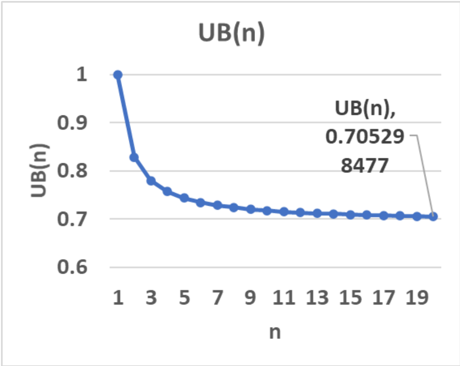
- A set of n periodic tasks is schedulable if

$$UB(n) = \frac{e_1}{p_1} + \frac{e_2}{p_2} + \ldots + \frac{e_n}{p_n} \leq n\left(2^{1/n} - 1\right)$$

- UB(1) = 1.0
- UB(2) = 0.828
- UB(3) = 0.779
- …
- UB(n) where n is large?

$$\lim_{n\to\infty} n(2^{1/n} - 1) = \ln(2) \approx 0.693.$$

What the theory says: *If you have n (say 3) periodic tasks to schedule , then your utilization should be less than or equal to UB(n) (that is 0.779) for these tasks to be schedulable using RM*



Q. If utilization is out of bound, does that mean the taskset is unschedulable?
A. A. Not necessarily. It’s a sufficient condition, but not necessary one.

Sample Problem

	e	p	U
Task $\tau 1$	20	100	0.200
Task $\tau 2$	40	150	0.267
Task $\tau 3$	100	350	0.286

L&L Bound

UB(1) = 1.0
UB(2) = 0.828
UB(3) = 0.779
UB(n) = 0.693

- Are all tasks schedulable?
  - $U_1 + U_2 + U_3 = 0.753 < U(3)$ ➔ **Schedulable!**
- What if we double the e of $\tau 1$
  - What if increase execution time of task1 to 40?
  - 0.2*2 + 0.267+ 0.286 = 0.953 > UB(3) = 0.779
  - See case 2 in next slide

4

Let's see if the theory works in practice:

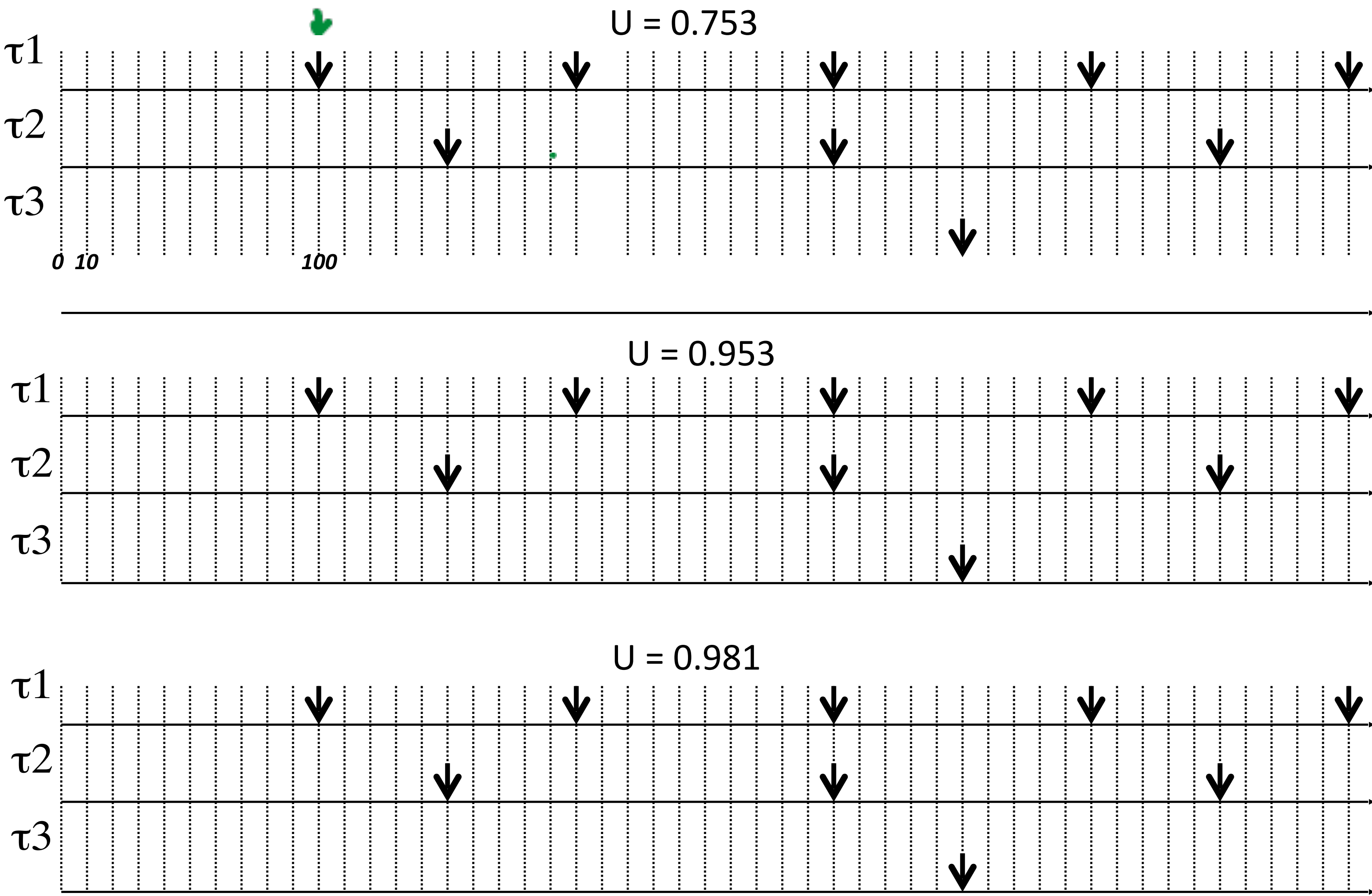
-Draw the timeline of the tasks using RM

-See if the theory assumption from last slide works

L&L Bound

UB(3) = 0.779

UB(n) = 0.693



e p

(20, 100)

(40, 150)

(100, 350)

(40, 100)

(40, 150)

(100, 350)

(40, 100)

(40, 150)

(**110**, 350)

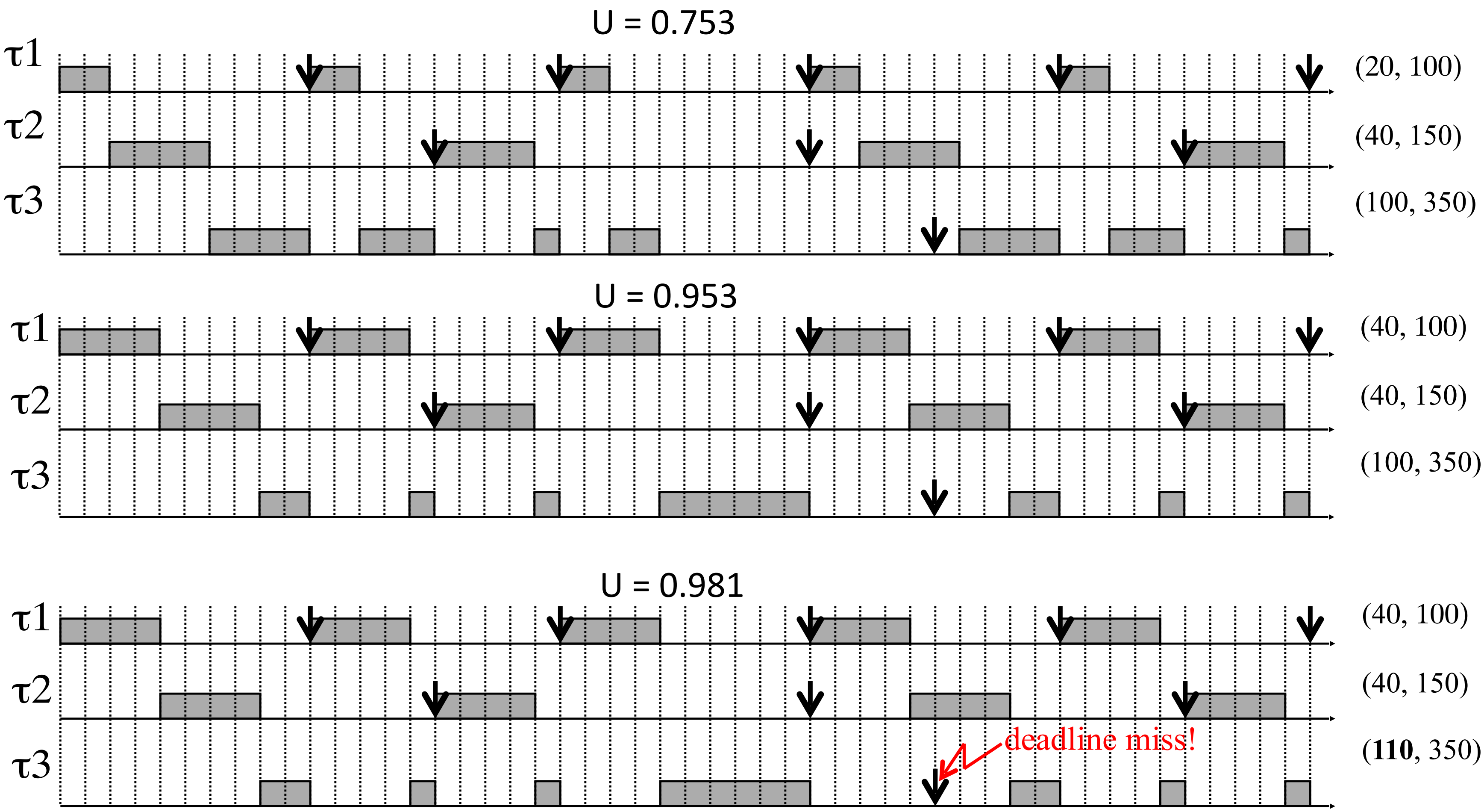
5

L&L Bound

$UB(3) = 0.779$

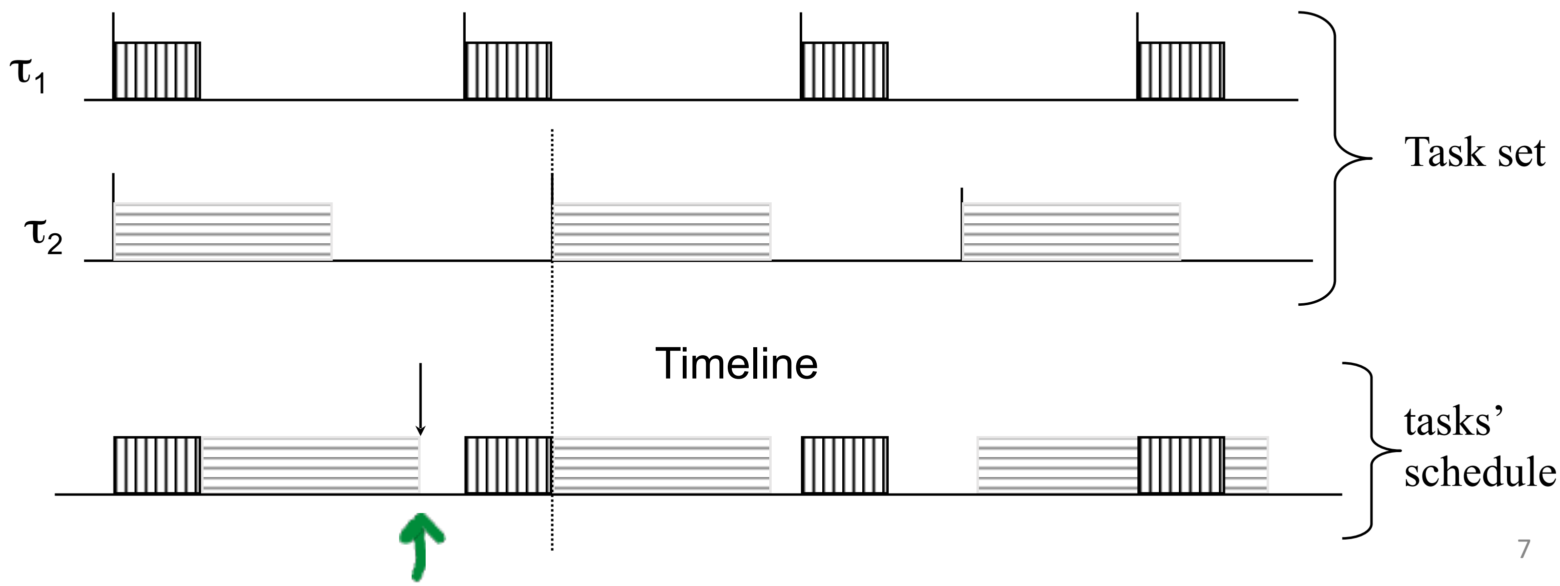
$UB(n) = 0.693$

Solution:



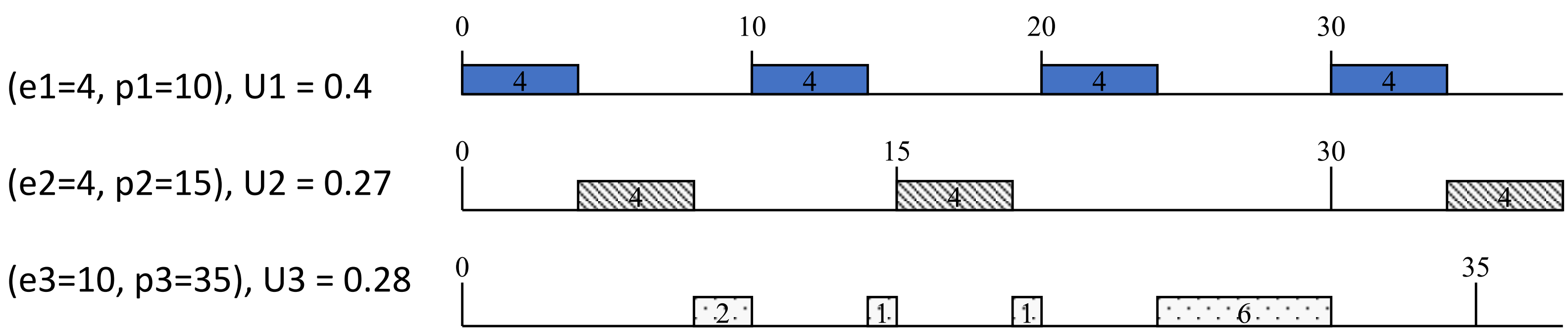
Critical Instant Theorem

- If a low priority task meets its first deadline when all higher priority tasks are started at the same time, then this task's future deadlines will always be met.
- Example: For the low priority task, $\tau_{2,}$ since the first instance met the deadline, all future deadlines will be met



Exact Schedulability Test

- For each task, checks if it can meet its first deadline (critical instant theorem in the last slide)
- *Estimating if the lowest priority task met deadline needs several iteration*



Exact Schedulability Test

- For lowest priority task, checks if it can meet its first deadline
- Iteratively calculates response time r_i^{k+1} until one of the two termination conditions are met
- Small equation for initial iteration (i.e., r_i^0), larger equation for subsequent iteration (i.e., r_i^{k+1}) starting with K=0
- The following equation calculates the response time r of the lowest priority task
- r^0 is the response time of the first iteration

$$r_i^{k+1} = e_i + \sum_{j=1}^{i-1} \left\lceil \frac{r_i^k}{p_j} \right\rceil e_j, \quad \text{where } r_i^0 = \sum_{j=1}^{i} e_j$$

Test terminates when $r_i^{k+1} > p_i$ (not schedulable)

or when $r_i^{k+1} = r_i^k \leq p_i$ (schedulable).

i→ task, k→ iteration

Assumption: task i has lower priority than task i-1

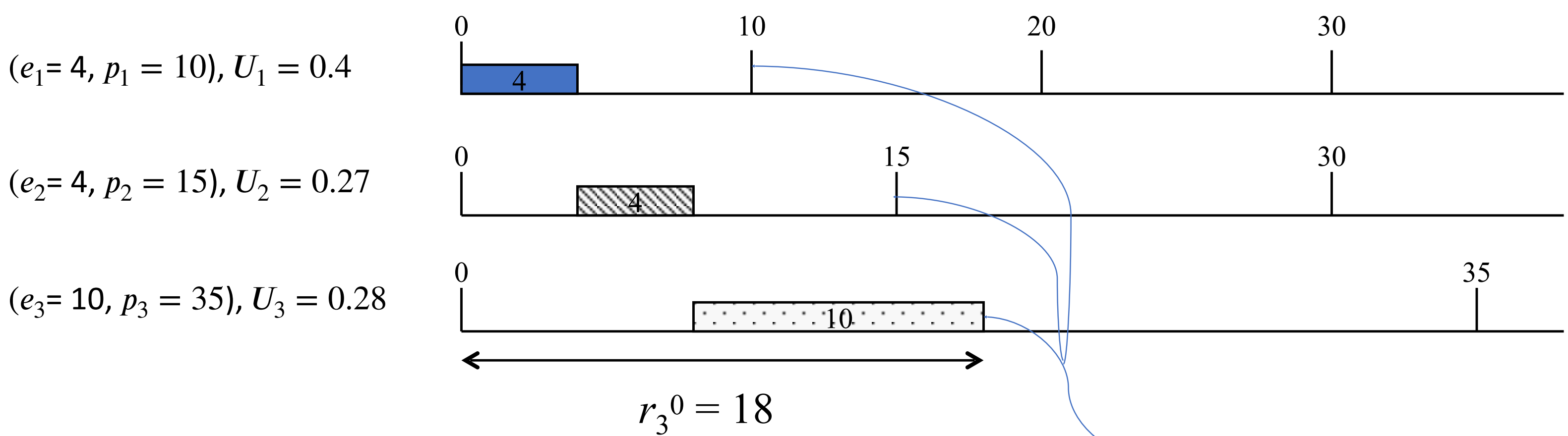
9

Exact Schedulability Test

- Let's calculate response times of task 3 (i=3)
  - **Initial iteration:** Find the execution time for the first occurrence of tasks according to their priority from time 0

$$r_i^{k+1} = e_i + \sum_{j=1}^{i-1} \left\lceil \frac{r_i^k}{p_j} \right\rceil e_j, \quad \text{where } r_i^0 = \sum_{j=1}^{i} e_j$$

$$r_3^0 = \sum_{j=1}^{3} e_j = e_1 + e_2 + e_3 = 4+4+10=18$$



Ignore the possible interference of the next occurrence of higher priority tasks

10

Exact Schedulability Test

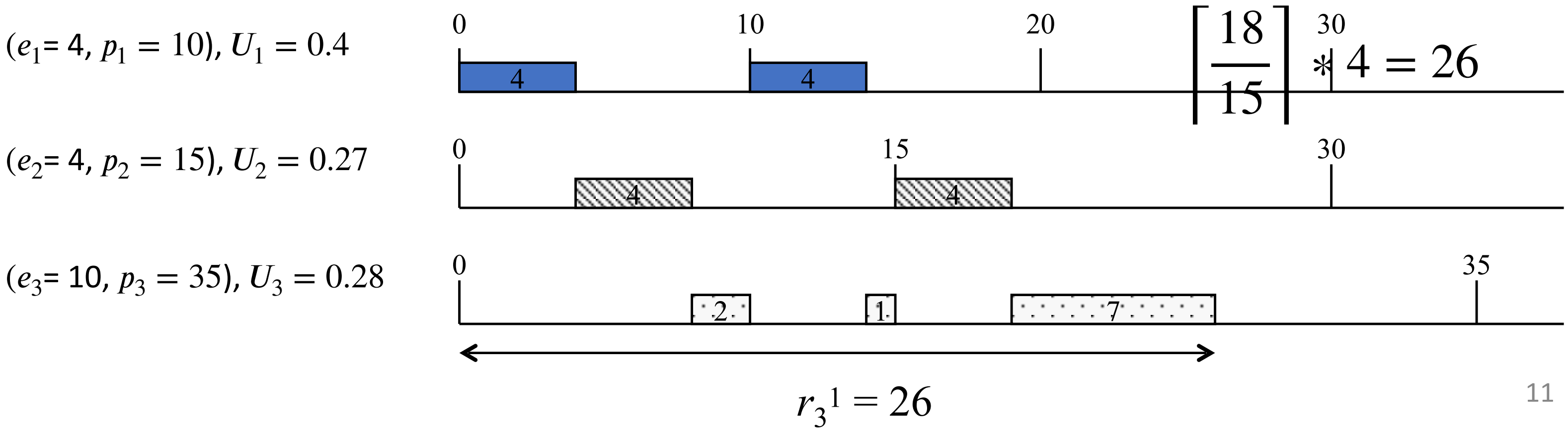
- For task 3 (i=3)
  - **First iteration, k=0:**
  - *Use equation for* r_i^{k+1}
  - Use r_3^0*=18 from last slide*
  - *Check* $r_i^{k+1} = r_i^k$
- $r_3^1 = 26$ and $r_3^0 = 18$, thus not equal
  - Need more iteration

$$r_i^{k+1} = e_i + \sum_{j=1}^{i-1} \left\lceil \frac{r_i^k}{p_j} \right\rceil e_j, \quad \text{where } r_i^0 = \sum_{j=1}^{i} e_j$$

$$r_3^1 = e_3 + \sum_{j=1}^{2} \left\lceil \frac{r_3^0}{p_j} \right\rceil e_j$$

$$= e_3 + \left\lceil \frac{r_3^0}{p_1} \right\rceil e_1 + \left\lceil \frac{r_3^0}{p_2} \right\rceil e_2$$

$$=10+\left\lceil \frac{18}{10} \right\rceil *4+ \left\lceil \frac{18}{15} \right\rceil * 4 = 26$$



Exact Schedulability Test

- For task (r=3)
  - Second iteration (k=1):
  - *Check if* $r_i^{k+1} = r_i^k$
  - $r_3^1 = 26$, $r_3^2 = 30$
  - Need more iteration

$$r_i^{k+1} = e_i + \sum_{j=1}^{i-1} \left\lceil \frac{r_i^k}{p_j} \right\rceil e_j, \quad \text{where } r_i^0 = \sum_{j=1}^{i} e_j$$

$$r_3^2 = e_3 + \sum_{j=1}^{2} \left\lceil \frac{r_3^1}{p_j} \right\rceil e_j$$

$$= e_3 + \left\lceil \frac{r_3^1}{p_1} \right\rceil e_1 + \left\lceil \frac{r_3^1}{p_2} \right\rceil e_2$$

$$=10+\left\lceil \frac{26}{10} \right\rceil *4+ \left\lceil \frac{26}{15} \right\rceil * 4$$

$$= 10 + 3 * 4 + 2 * 4 = 30$$

Exact Schedulability Test

- Find r_3^3

$$r_i^{k+1} = e_i + \sum_{j=1}^{i-1} \left\lceil \frac{r_i^k}{p_j} \right\rceil e_j, \quad \text{where } r_i^0 = \sum_{j=1}^{i} e_j$$

$$r_3^3 = e_3 + \sum_{j=1}^{2} \left\lceil \frac{r_3^2}{p_j} \right\rceil \cdot e_j = ?$$

Quiz!

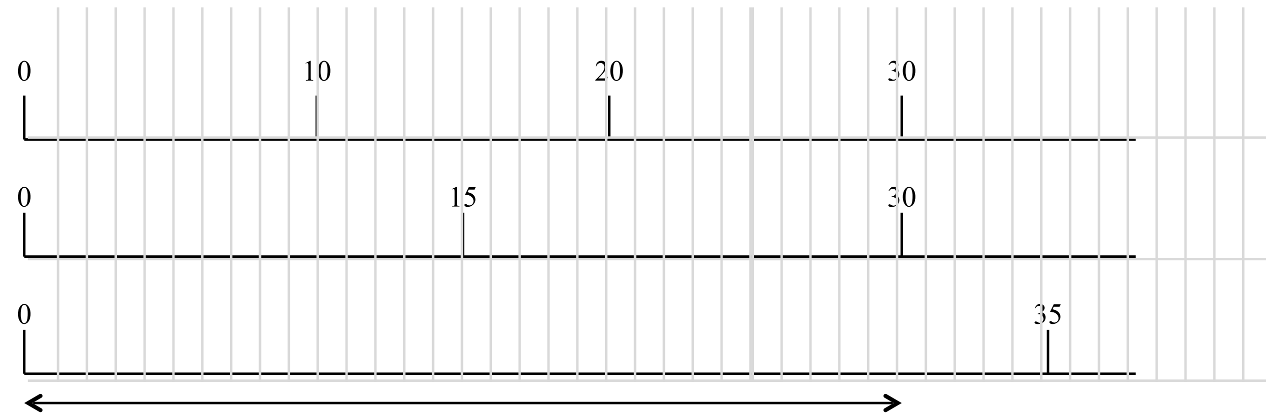
$$r_3^2 = e_3 + \sum_{j=1}^{2} \left\lceil \frac{r_3^1}{p_j} \right\rceil \cdot e_j = 10 + \left\lceil \frac{26}{10} \right\rceil 4 + \left\lceil \frac{26}{15} \right\rceil 4 = 30$$

- When we get same response time twice and its less than deadline, its schedulable

Test terminates when $r_i^{k+1} > p_i$ (not schedulable)
or when $r_i^{k+1} = r_i^k \leq p_i$ (schedulable).

Solve it manually and check

- When we get same response time twice and its less than deadline, its schedulable



$r_3^2 = r_3^3 = 30$

Assumptions

- So far the theories assume
  - All the tasks are periodic
  - Tasks are scheduled according to RMS
  - All tasks are independent and do not share resources (data)
  - Tasks do not self-suspend during their execution
  - Scheduler overhead (context-switch) is negligible

Sample Question

- Consider the following real-time taskset.

Task	C	P	U
t1	4	10	0.400
t2	4	15	0.267
t3	10	35	0.286

- Is this taskset schedulable under the rate monotonic scheduling? Use the exact Schedulability analysis for your answer.

Acknowledgements

- These slides draw on materials developed by
  - Lui Sha and Marco Caccamo (UIUC)
  - Rodolfo Pellizzoni (U. Waterloo)
  - Edward A. Lee and Prabal Dutta (UCB) for EECS149/249A