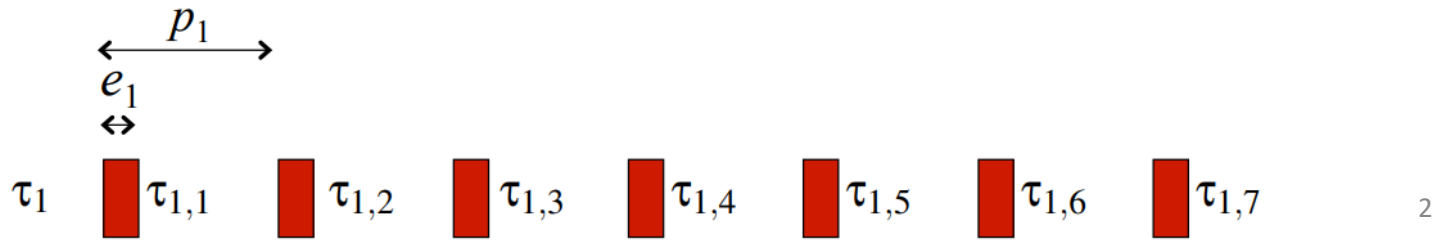
Lecture 18: Real-Time Process Scheduling

EECS 388 – Spring 2023

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

Recap: Modeling Periodic Tasks

Term	Definition	Symb.	Example
Tasks	Different tasks	τ_i	=Task 1, =Task 2
Task instance	Each task may have multiple instance		=occurrence 3 of Task 1
Period of the task	time interval after which a task repeats		See figure
Phase of the task	time from 0 till the occurrence of the first instance of the task	ϕ	Not shown here
Response time	the time a task takes for the task to produce its results (after starting)		Next slide
Execution time	the time a task takes for the task to produce its results (after releasing)		Next slide



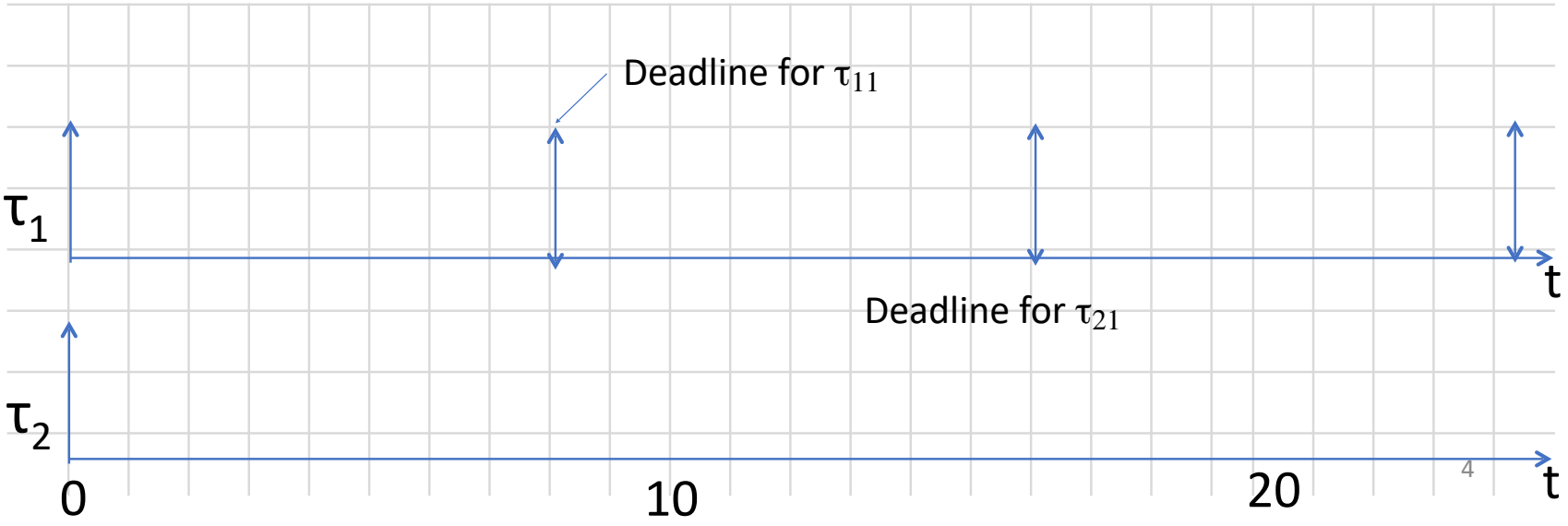
Rate Monotonic (RM)

- Fixed-priority scheduling algorithm
- Assigns priorities to tasks on the basis of their periods
- Only applies to periodic tasks
- **Shorter period = higher priority.**

- Assumptions:
  - All process runs on single CPU
  - Process execution time is constant
  - context switch/preemption time, is negligible
  - Deadline is at the end of a period

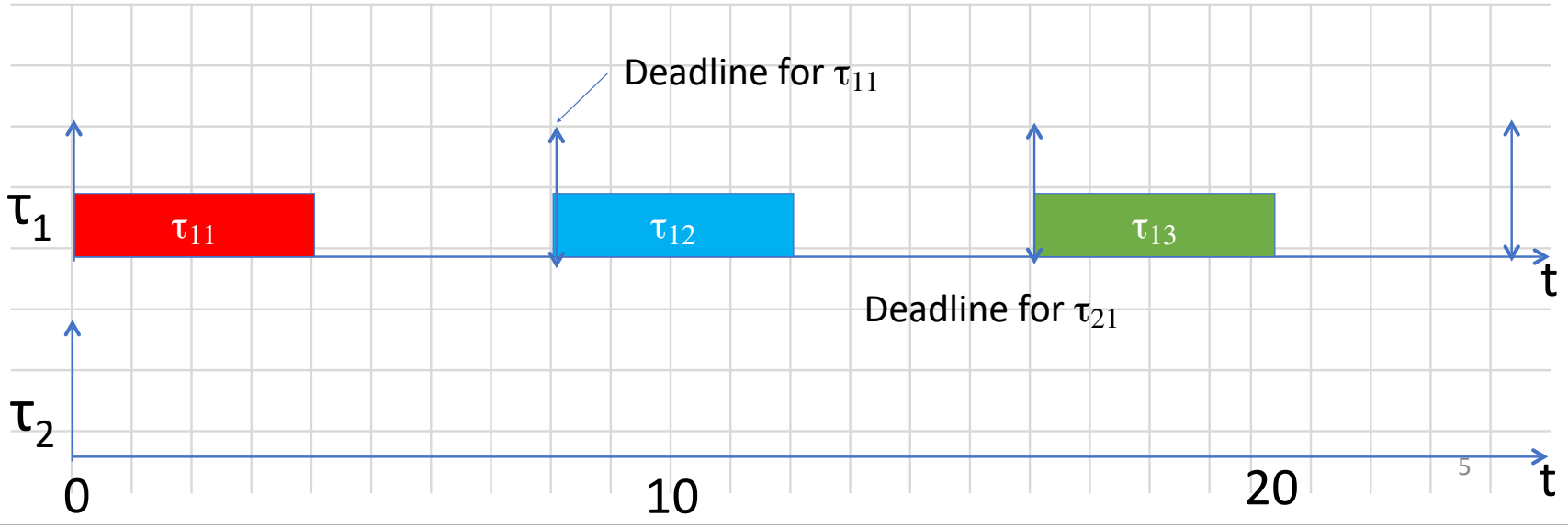
RM Example: Draw τ_2

- τ_1 (e1 = 4, p1 = 8), high prio
- τ_2 (e2 = 6, p2 = 12), low prio
- Utilization: U = 4/8 + 6/12 = 1
- Draw τ_2 Timeline:
  - 1) mark the deadlines 2) check the time blocks when CPU is free



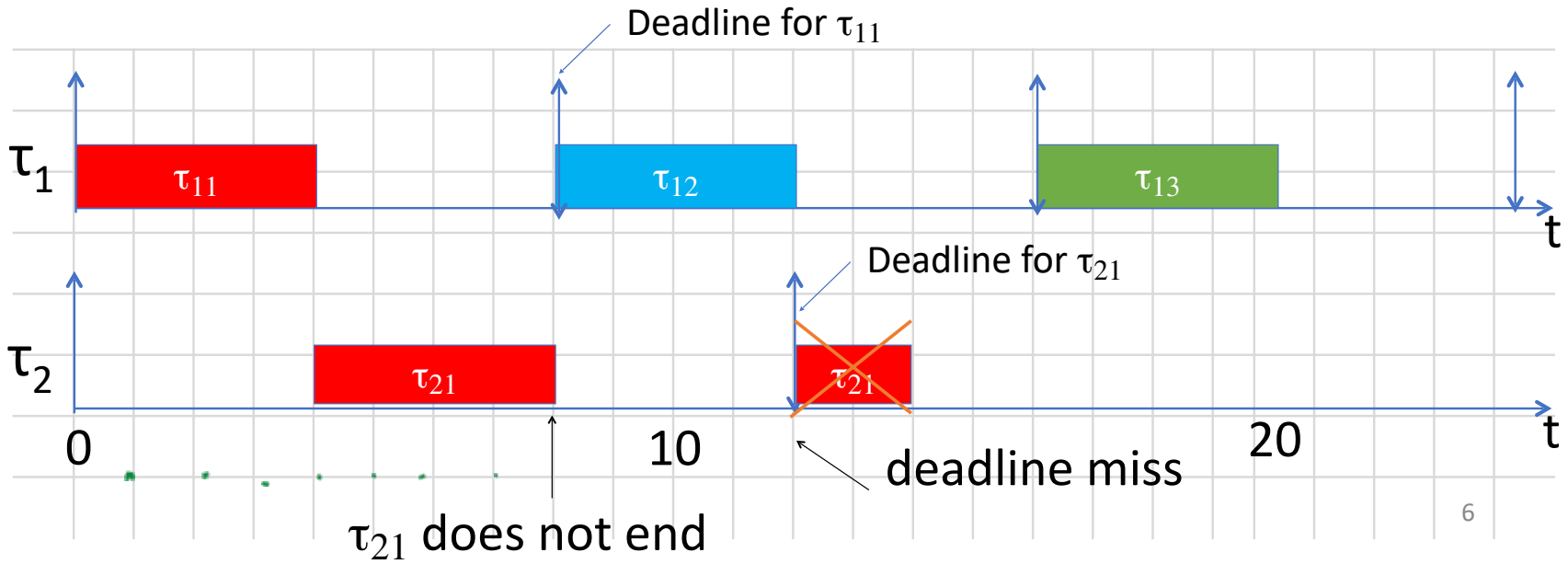
RM Example: Draw τ_2

- τ_1 (e1 = 4, p1 = 8), high prio
- τ_2 (e2 = 6, p2 = 12), low prio
- Utilization: U = 4/8 + 6/12 = 1
- Draw τ_2 Timeline:
  - 1) mark the deadlines 2) check the time blocks when CPU is free



RM Example

- τ_1 (e1 = 4, p1 = 8), high prio
- τ_2 (e2 = 6, p2 = 12), low prio
- Utilization: U = 4/8 + 6/12 = 1



Takeaway

- RM is not suitable for many cases
- Feasibility of using RM scheduling depends on the deadline and exaction time of all tasks

7

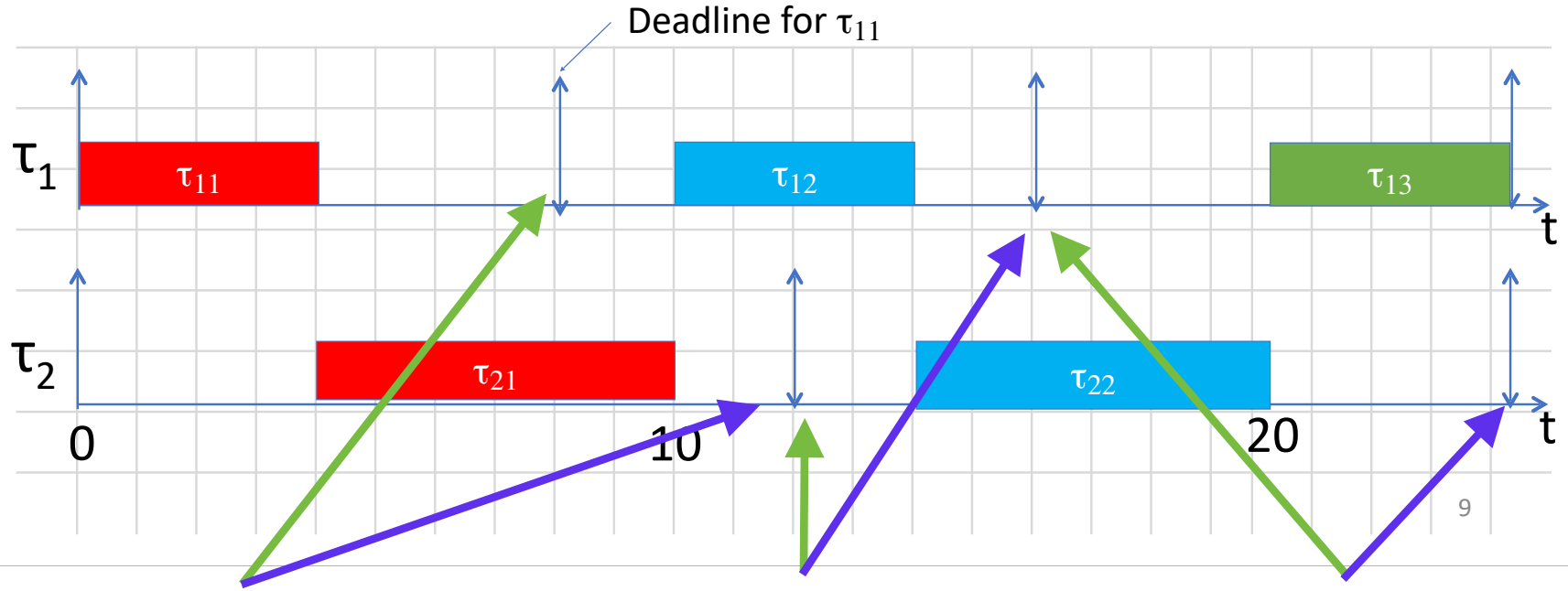
Earliest Deadline First (EDF)

- Dynamic-Priority Scheduling Algorithm
- **Earlier deadline = higher priority**

- *Task priority is **inversely proportional** to its current absolute deadline*
  - *Priority of a task→1/remaining time to deadline*
  - *E.g., say we gave 2 tasks currently with deadline at 6sec and 4sec respectively→ task 2 will have higher priority*

EDF Example

- Case#2: The case that didn't work with RM
  - τ_1 (e1 = 4, p1 = 8), τ_2 (e2 = 6, p2 = 12)
  - Utilization: U = 4/8 + 6/12 = U_b = 1
  - Finishing the tasks by the deadline is important
    - We can delay the start time of the tasks



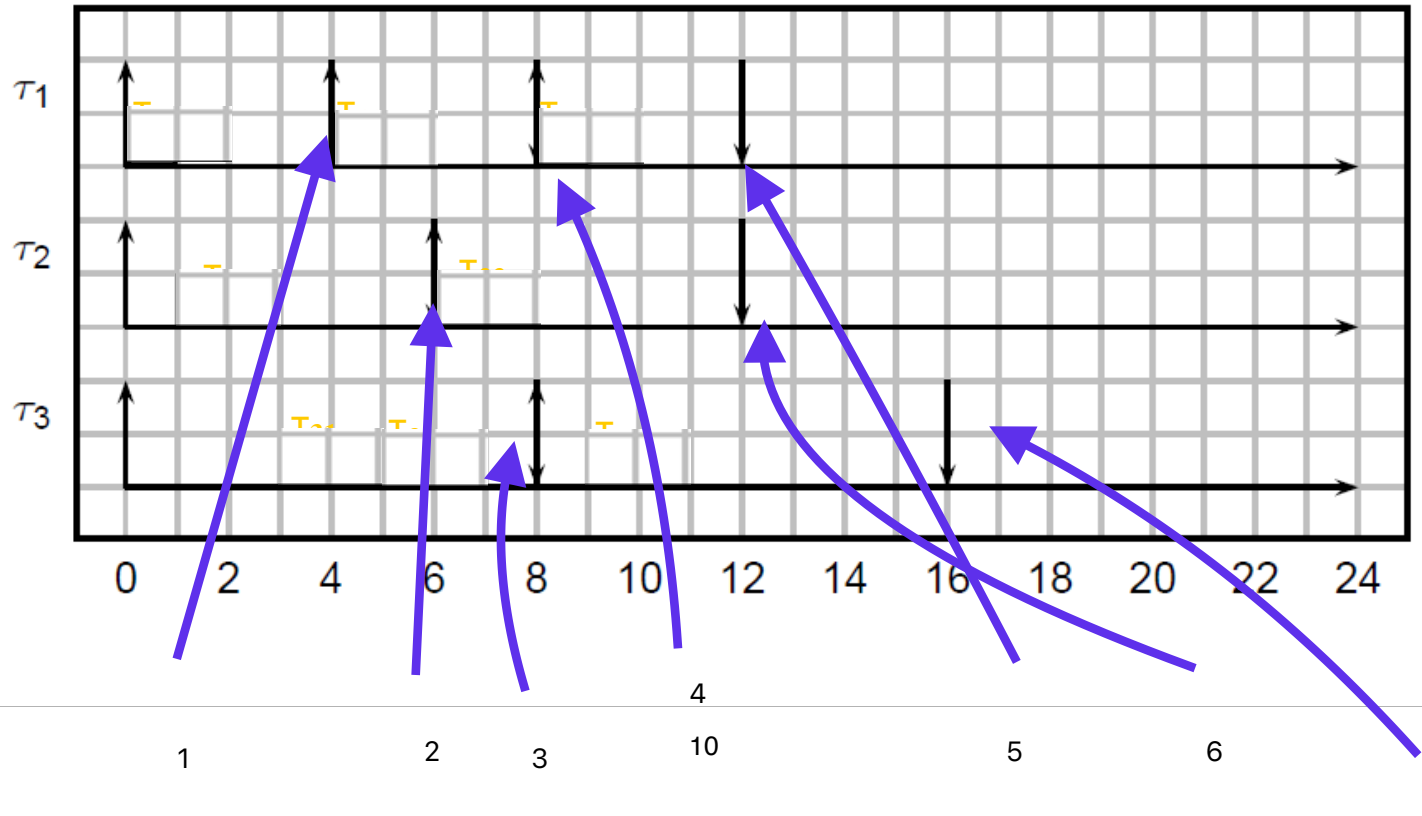
deadline for task 1 is earlier than deadline for task 2

deadline for task 2 is earlier than deadline for task 1

deadline for task 1 is earlier than deadline for task 2

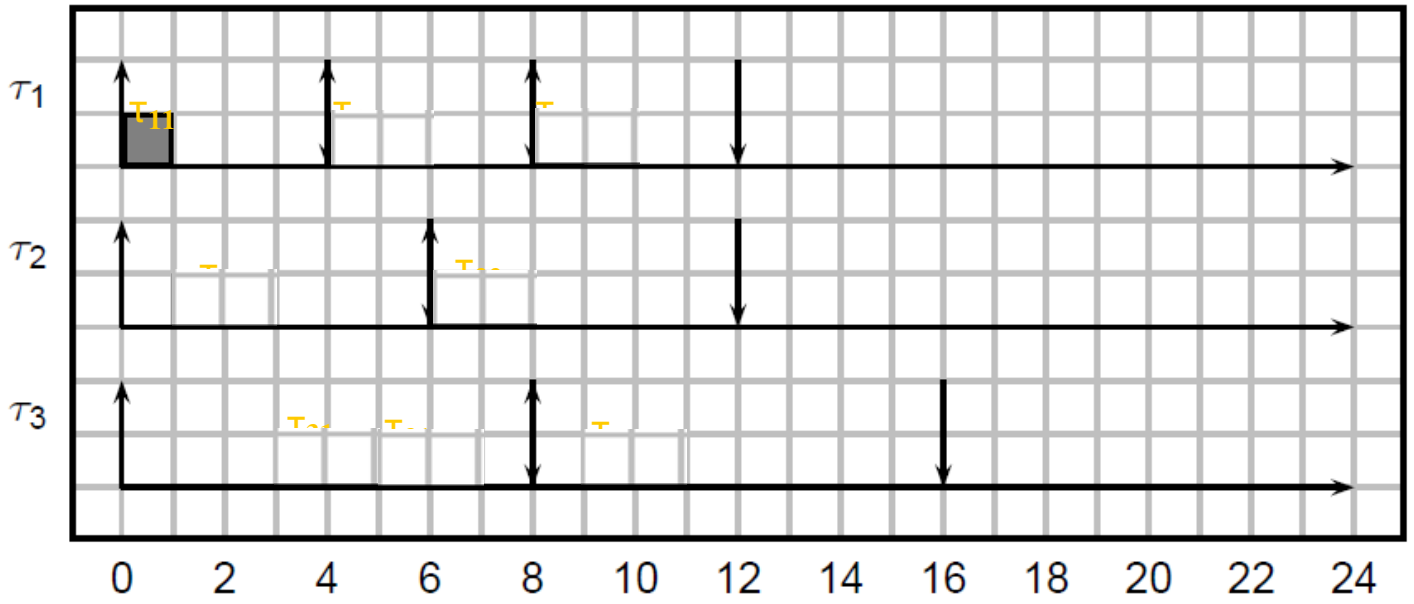
More examples: RM vs. EDF

- τ_1 (e1 = 1, p1 = 4), τ_2 (e2 = 2, p1 = 6), τ_3 (e3 = 3, p3 = 8)
- Utilization: U = 1/4 + 2/6 + 3/8 = 23/24
- **RM scheduling:**



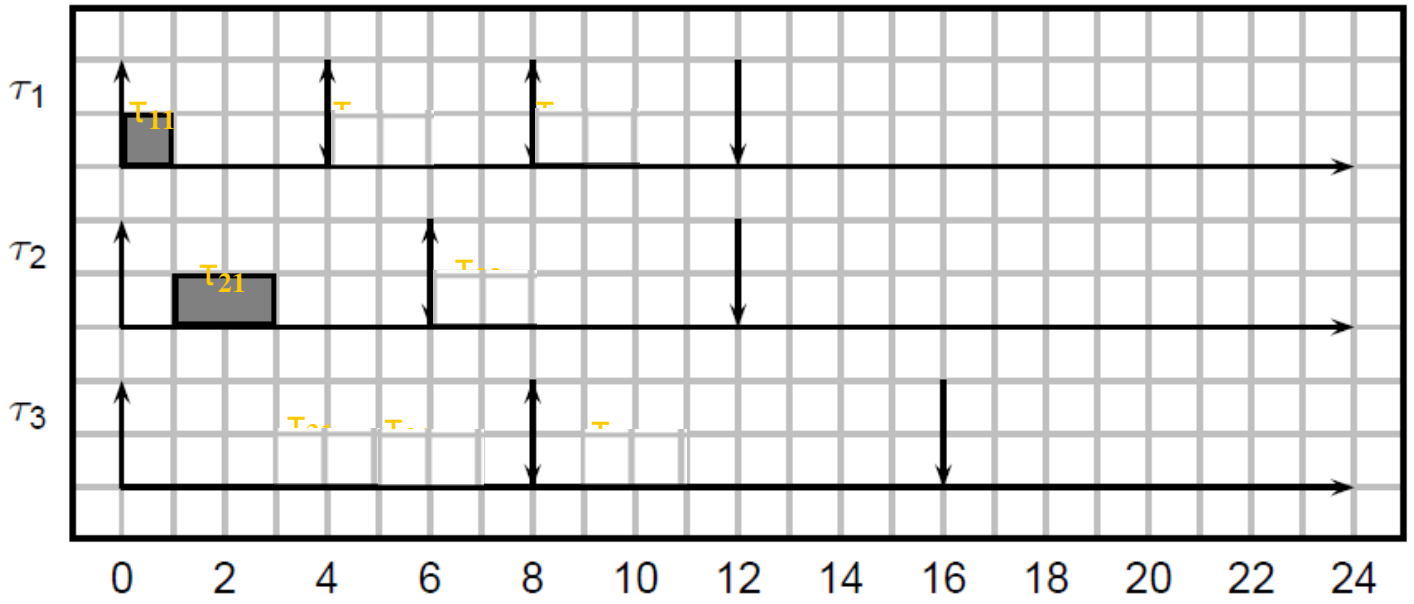
More examples: RM vs. EDF

- τ_1 (e1 = 1, p1 = 4), τ_2 (e2 = 2, p1 = 6), τ_3 (e3 = 3, p3 = 8)
- Utilization: U = 1/4 + 2/6 + 3/8 = 23/24
- **RM scheduling:**



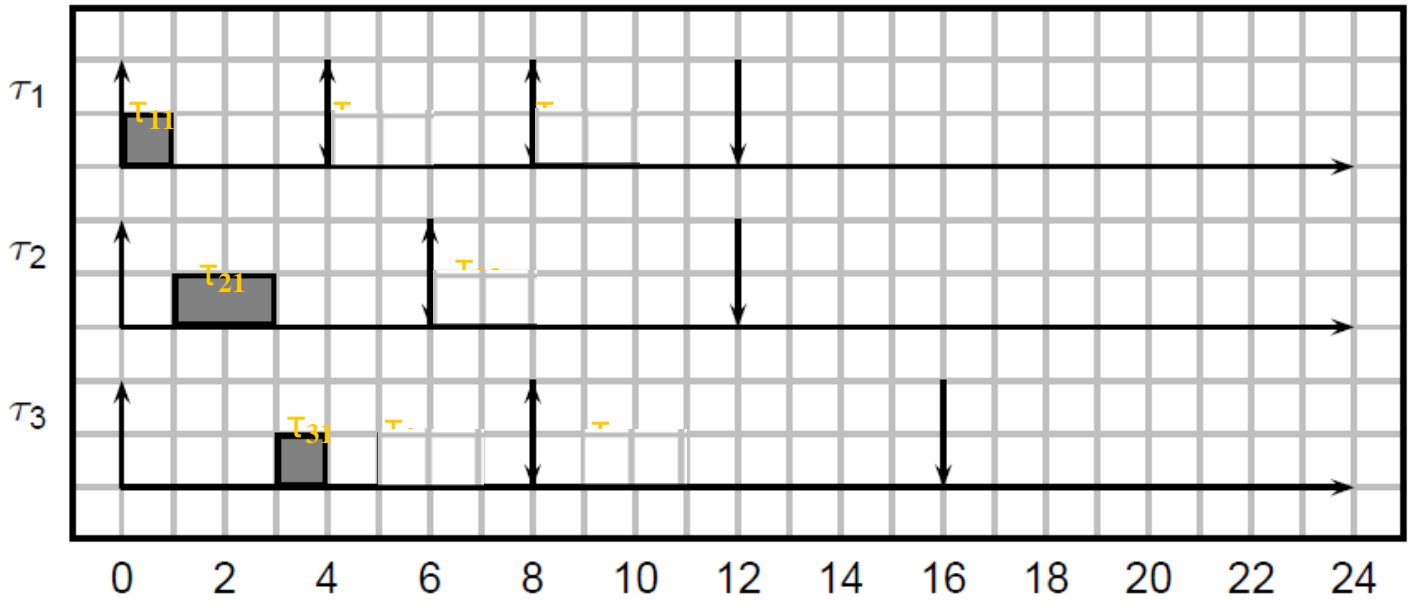
More examples: RM vs. EDF

- τ_1 (e1 = 1, p1 = 4), τ_2 (e2 = 2, p1 = 6), τ_3 (e3 = 3, p3 = 8)
- Utilization: U = 1/4 + 2/6 + 3/8 = 23/24
- **RM scheduling:**



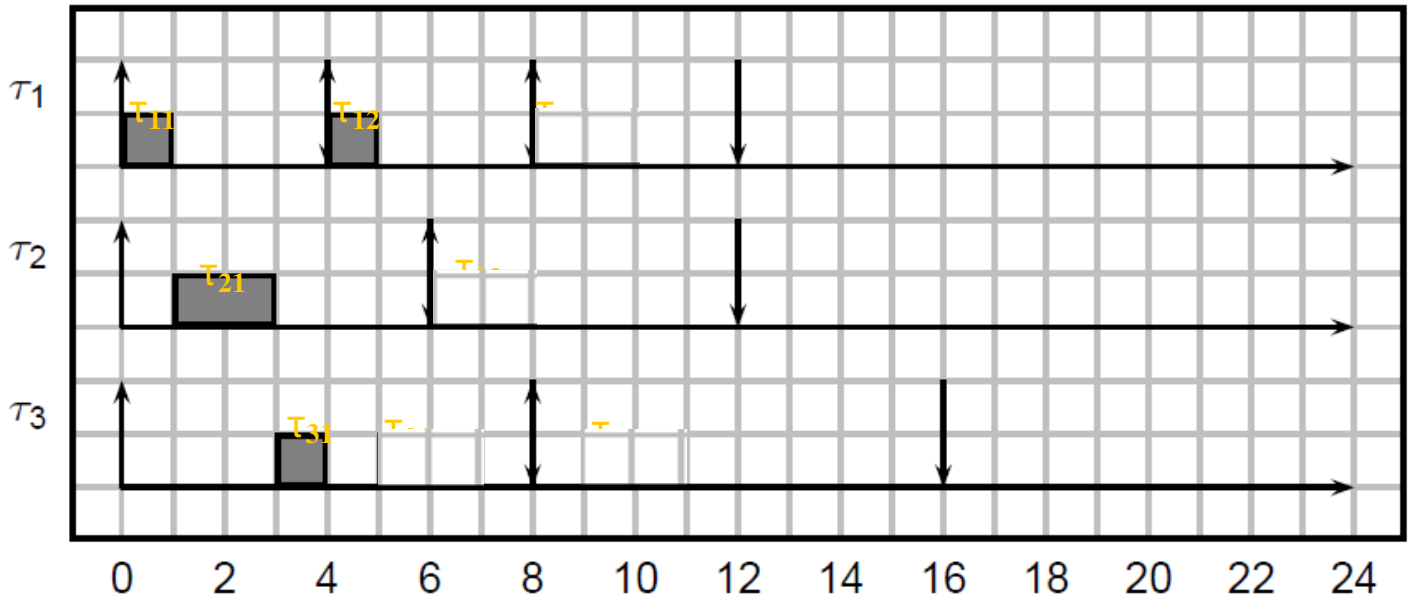
More examples: RM vs. EDF

- τ_1 (e1 = 1, p1 = 4), τ_2 (e2 = 2, p1 = 6), τ_3 (e3 = 3, p3 = 8)
- Utilization: U = 1/4 + 2/6 + 3/8 = 23/24
- **RM scheduling:**



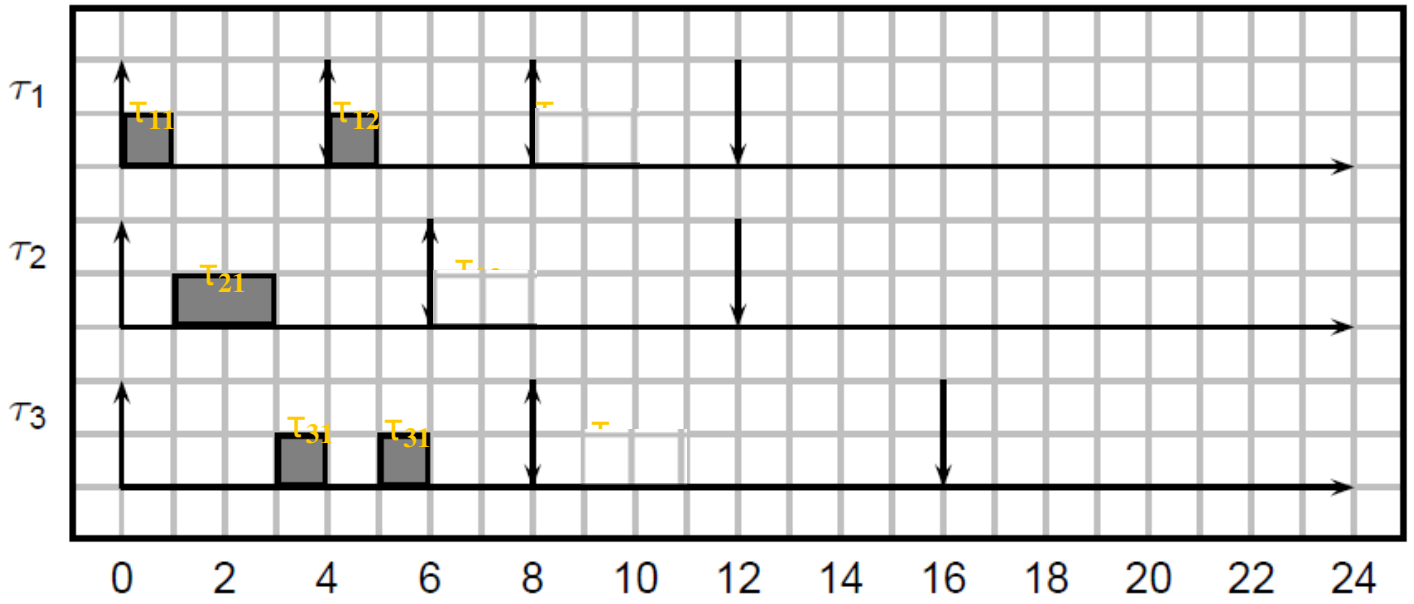
More examples: RM vs. EDF

- τ_1 (e1 = 1, p1 = 4), τ_2 (e2 = 2, p1 = 6), τ_3 (e3 = 3, p3 = 8)
- Utilization: U = 1/4 + 2/6 + 3/8 = 23/24
- **RM scheduling:**



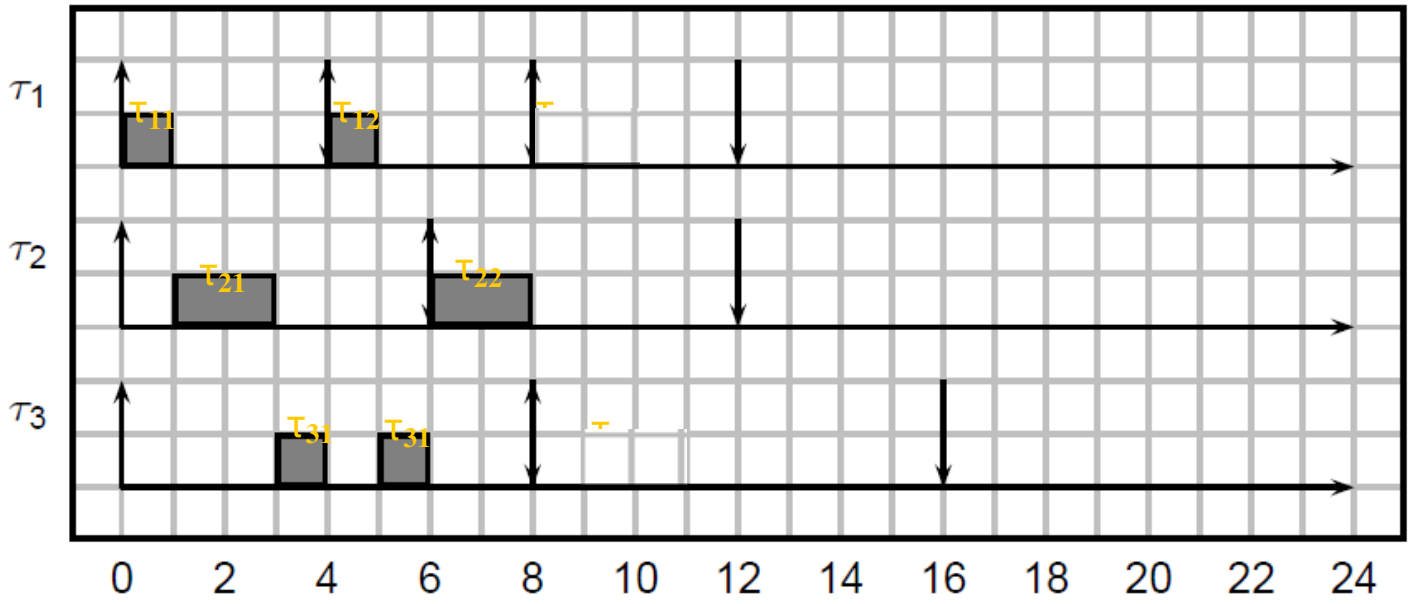
More examples: RM vs. EDF

- τ_1 (e1 = 1, p1 = 4), τ_2 (e2 = 2, p1 = 6), τ_3 (e3 = 3, p3 = 8)
- Utilization: U = 1/4 + 2/6 + 3/8 = 23/24
- **RM scheduling:**



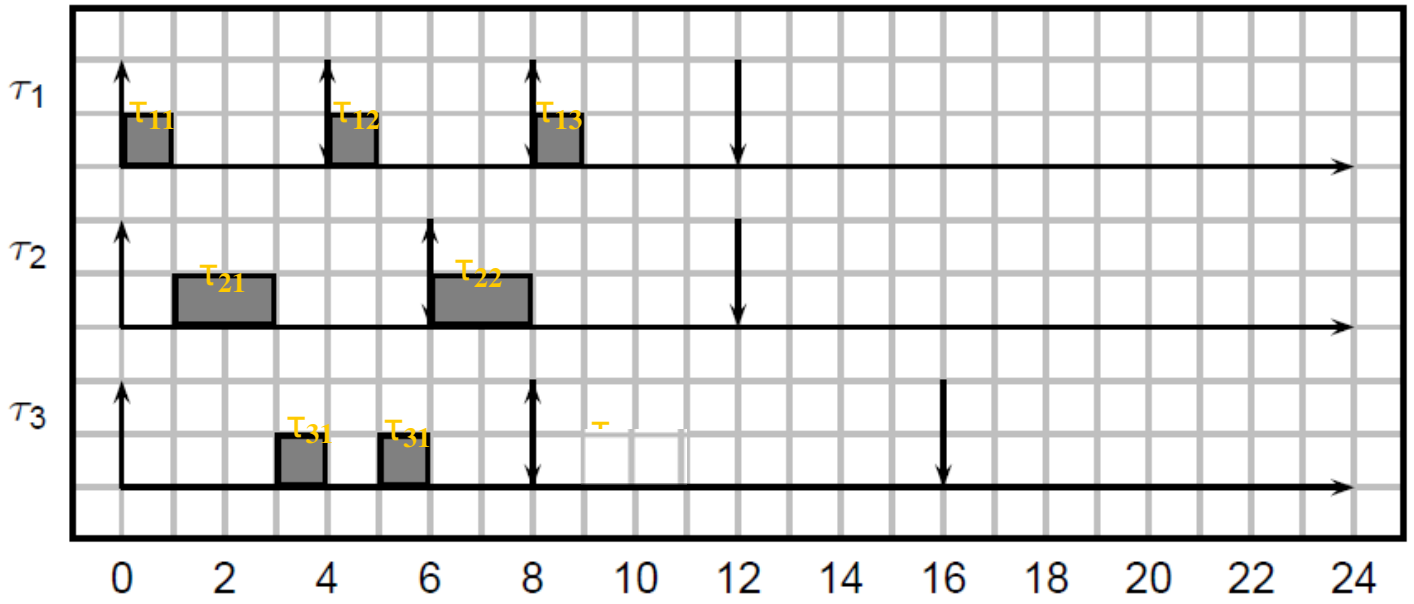
More examples: RM vs. EDF

- τ_1 (e1 = 1, p1 = 4), τ_2 (e2 = 2, p1 = 6), τ_3 (e3 = 3, p3 = 8)
- Utilization: U = 1/4 + 2/6 + 3/8 = 23/24
- **RM scheduling:**



More examples: RM vs. EDF

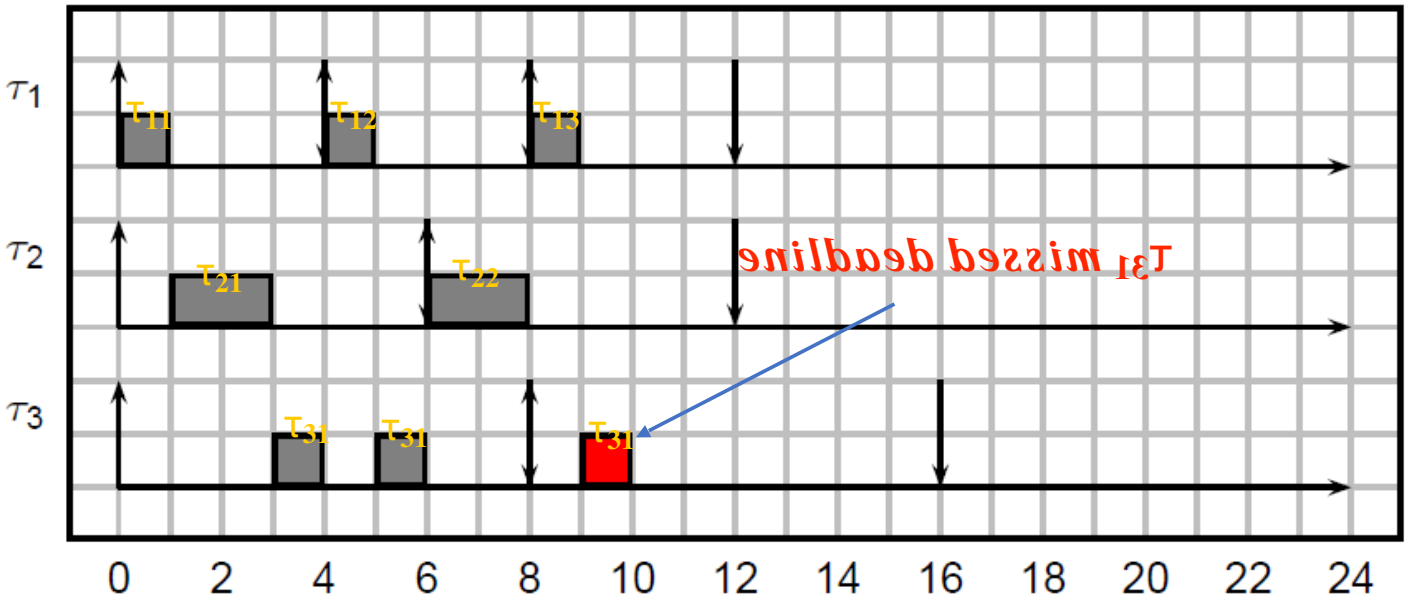
- τ_1 (e1 = 1, p1 = 4), τ_2 (e2 = 2, p1 = 6), τ_3 (e3 = 3, p3 = 8)
- Utilization: U = 1/4 + 2/6 + 3/8 = 23/24
- **RM scheduling:**



17

More examples: RM vs. EDF

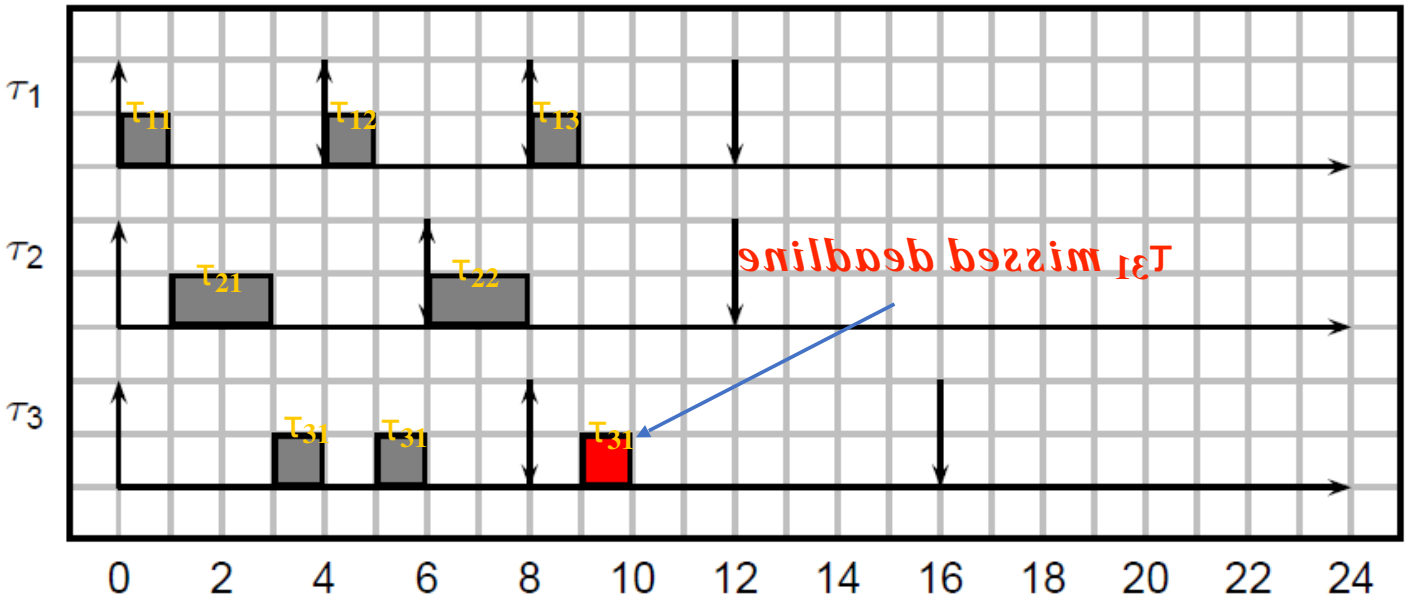
- τ_1 (e1 = 1, p1 = 4), τ_2 (e2 = 2, p1 = 6), τ_3 (e3 = 3, p3 = 8)
- Utilization: U = 1/4 + 2/6 + 3/8 = 23/24
- **RM scheduling:**



18

More examples: RM vs. EDF

- τ_1 (e1 = 1, p1 = 4), τ_2 (e2 = 2, p1 = 6), τ_3 (e3 = 3, p3 = 8)
- Utilization: U = 1/4 + 2/6 + 3/8 = 23/24
- **RM scheduling:**

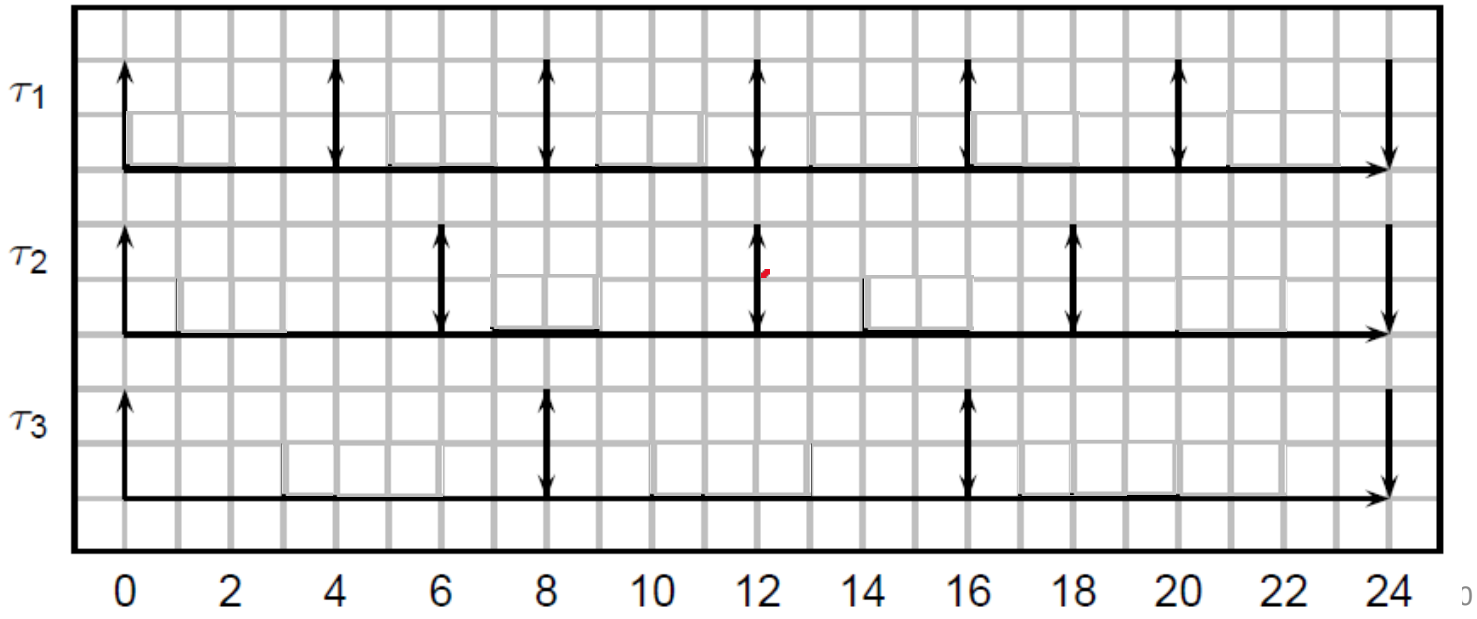


19

More Example: RM vs. EDF

Quiz 8!

- τ_1 (e1 = 1, p1 = 4), τ_2 (e2 = 2, p1 = 6), τ_3 (e3 = 3, p3 = 8)
- Utilization: U = 1/4 + 2/6 + 3/8 = 23/24
- **EDF scheduling:**
- What if equal deadlines for two tasks:
  - start the one with earliest start time, avoid context switch



Key Results

For periodic tasks with relative deadlines equal to their periods:

- Rate monotonic scheduling is the optimal fixed-priority priority policy
  - No other static priority assignment can do better
  - Yet, it might not achieve 100% CPU utilization
- Earliest deadline first scheduling is the optimal dynamic priority policy
  - EDF can achieve 100% CPU utilization

21

RM vs. EDF

- In practice, industrial systems heavily favor RM over EDF. Why?

RM vs. EDF

- In practice, industrial systems heavily favor RM over EDF. Why?

- RM is easier to implement
  - Task priority never changes.
- RM is more transparent and robust
  - easier to understand what is going on if something goes wrong (ex: overload).
  - if a task executes for longer than its prescribed worst-case time, higher priority tasks will be left untouched.

23

Next

- How can you know if a set of tasks are schedule using an algorithm?
- Schedulability analysis

Liu & Layland, JACM, Jan. 1973

Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment

C. L. LIU

Project MAC, Massachusetts Institute of Technology

AND

JAMES W. LAYLAND

Jet Propulsion Laboratory, California Institute of Technology

ABSTRACT. The problem of multiprogram scheduling on a single processor is studied from the viewpoint of the characteristics peculiar to the program functions that need guaranteed service. It is shown that an optimum fixed priority scheduler possesses an upper bound to processor utilization which may be as low as 70 percent for large task sets. It is also shown that full processor utilization can be achieved by dynamically assigning priorities on the basis of their current deadlines. A combination of these two scheduling techniques is also discussed.

KEY WORDS AND PHRASES: real-time multiprogramming, scheduling, multiprogram scheduling, dynamic scheduling, priority assignment, processor utilization, deadline driven scheduling

CR CATEGORIES: 3.80, 3.82, 3.83, 4.32

Liu & Layland Bound

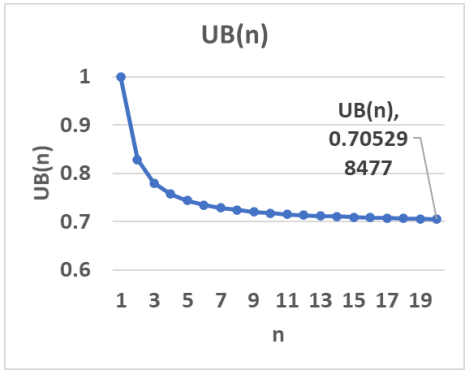
- A set of n periodic tasks is schedulable if

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

- 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

27

Let's see if the theory works in practice:

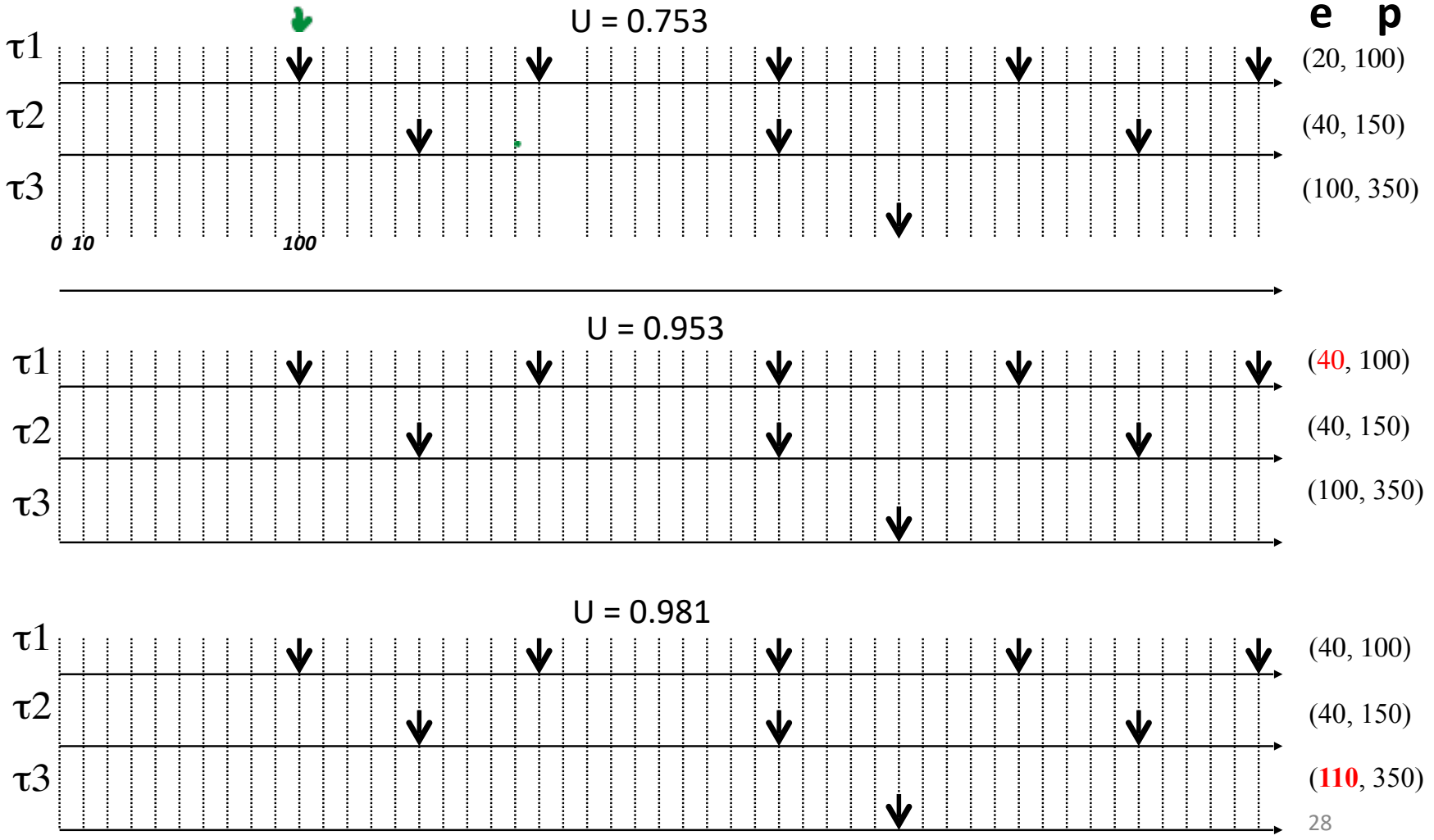
-Draw the timeline of the tasks using RM

-See if the theory assumption from last slide works

L&L Bound

UB(3) = 0.779

UB(n) = 0.693



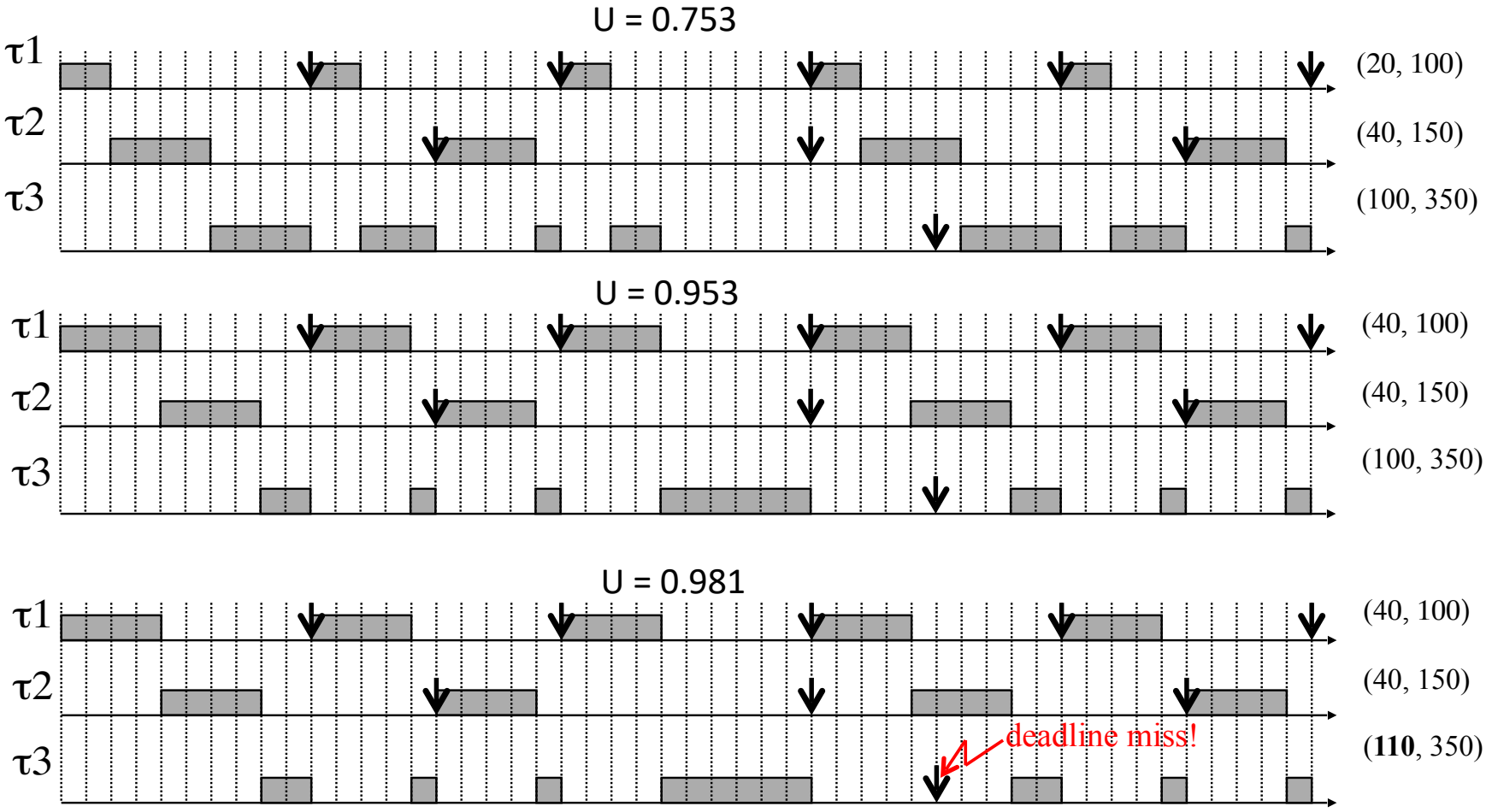
28

L&L Bound

UB(3) = 0.779

UB(n) = 0.693

Solution:

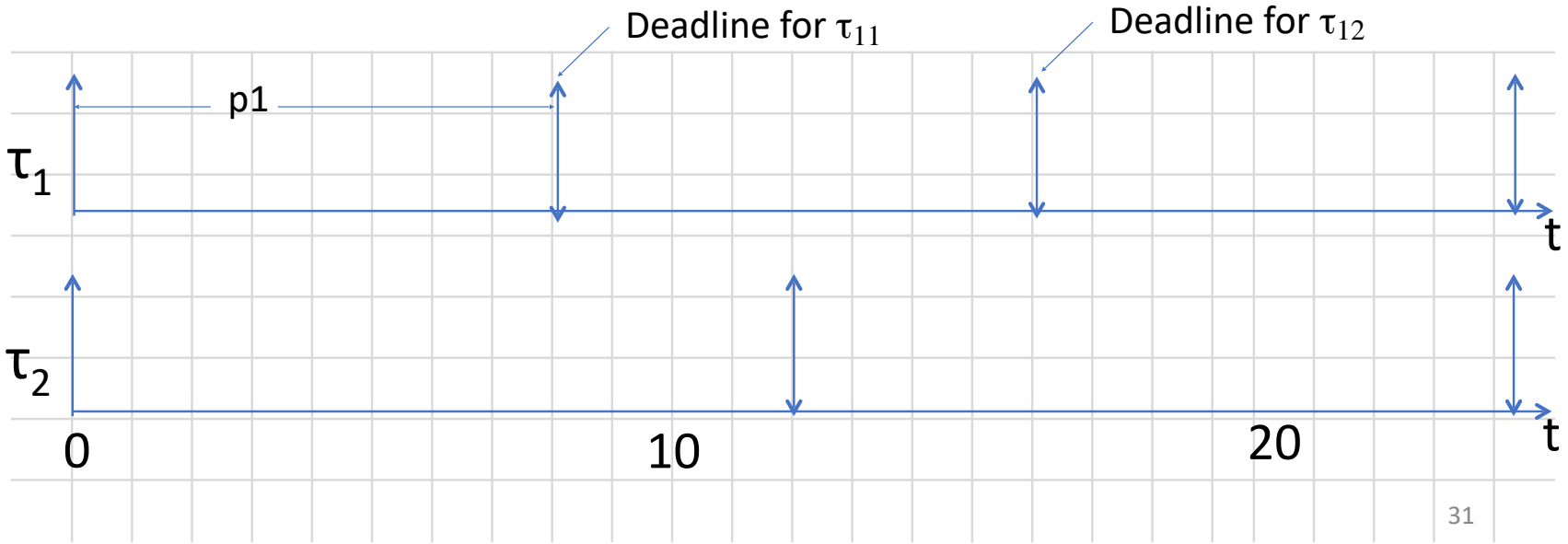


RM Scheduling Example for exam

30

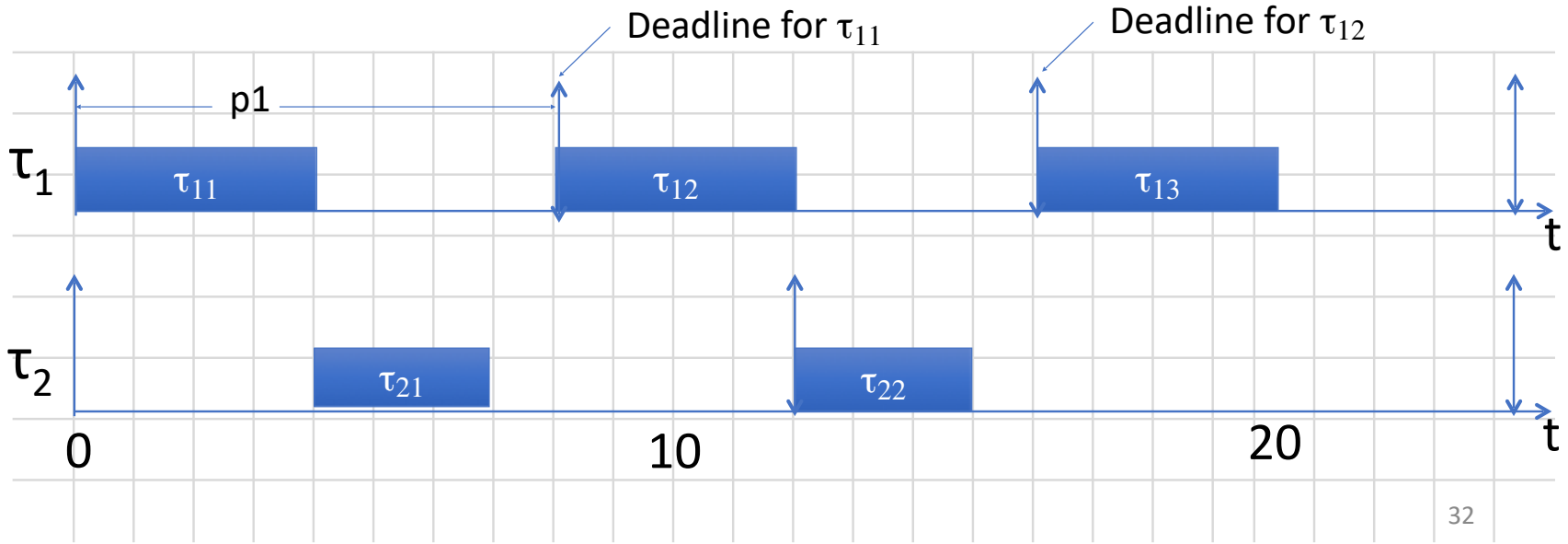
RM Example

- **Case#1**
  - τ_1 (e1 = 4, p1 = 8), high prio
  - τ_2 (e2 = 3, p2 = 12), low prio
  - Utilization: U = 4/8 + 3/12 = 0.75
  - Assumption: Deadline is at the end of a period



RM Example

- **Case#1**
  - τ_1 (e1 = 4, p1 = 8), high prio
  - τ_2 (e2 = 3, p2 = 12), low prio
  - Utilization: U = 4/8 + 3/12 = 0.75
  - Assumption: Deadline is at the end of a period



- **Case#2**

Consider the following real-time taskset:

Task	C	P	U
t1	2	10	0.200
t2	4	15	0.267
t3	10	35	0.286

Note1: C = worst-case execution time; P = period; U = utilization.
Assume a single core processor.

1- Compute the total utilization of the taskset.

2- Assign priorities of the tasks following the rate monotonic policy. You can use numbers, e.g. 1 (lowest priority) to 3 (highest).

3- Draw a timeline (from time 0 to 40) of the taskset under the rate monotonic scheduling (RMS).

33

Case#2 Solution

Consider the following real-time taskset:

Task	C	P	U
t1	2	10	0.200
t2	4	15	0.267
t3	10	35	0.286

Note1: C = worst-case execution time; P = period; U = utilization.
Assume a single core processor.

1- Compute the total utilization of the taskset. **0.753**

2- Assign priorities of the tasks following the rate monotonic policy. You can use numbers, e.g. 1 (lowest priority) to 3 (highest). t3 (1) < t2 (2) < t1(3)

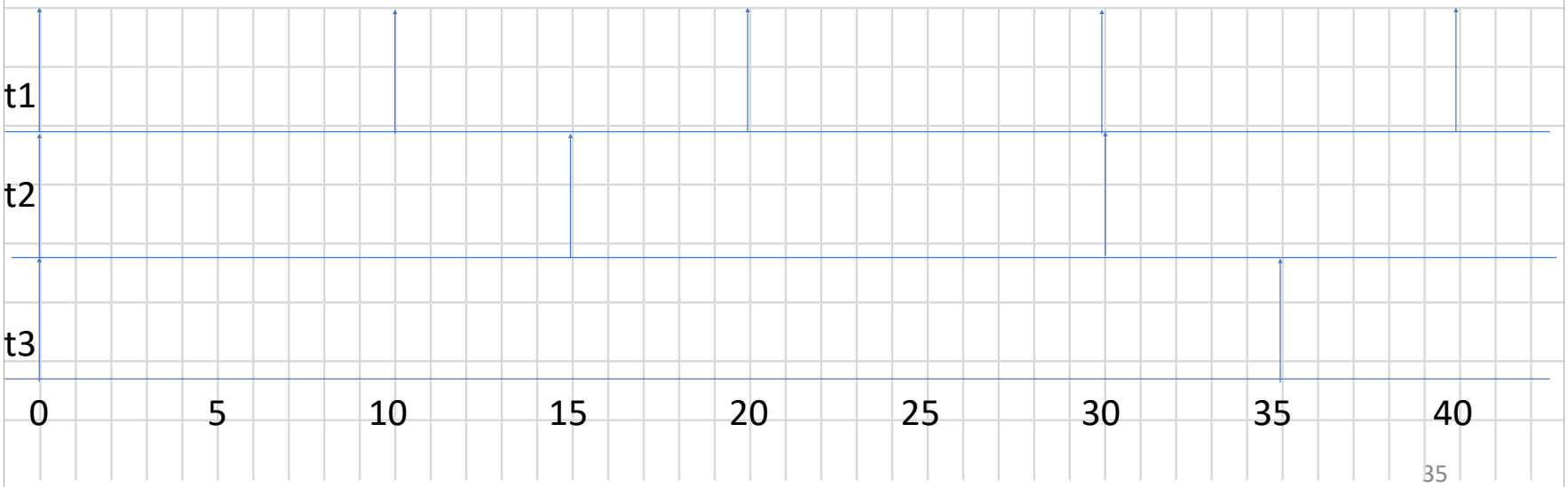
34

Case#2 Solution

3- Draw a timeline (from time 0 to 40) of the taskset under the rate monotonic scheduling (RMS

Task	C	P	U
t1	2	10	0.200
t2	4	15	0.267
t3	10	35	0.286

Note1: C = worst-case execution time; P = period; U = utilization.
Assume a single core processor.



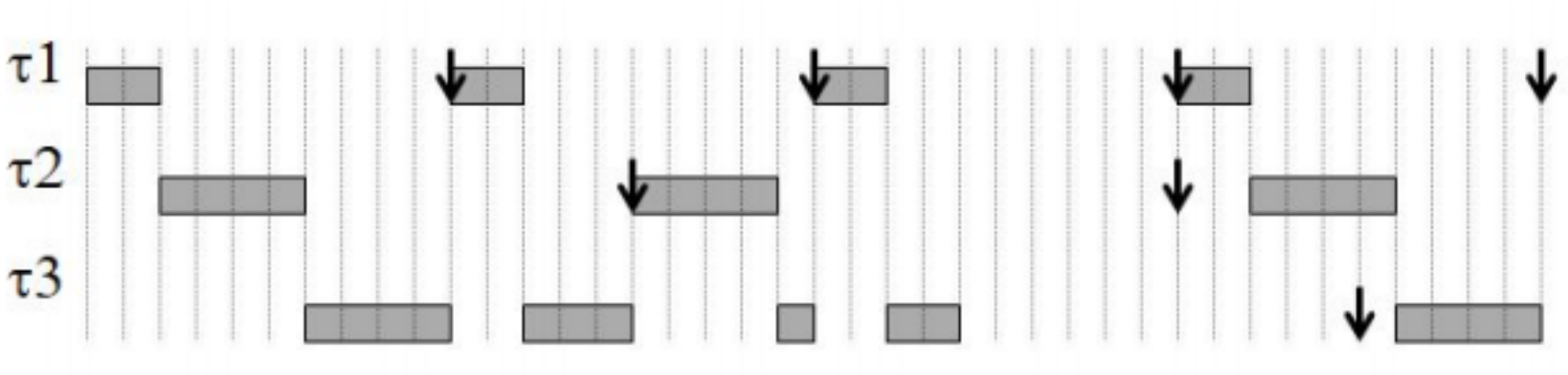
Case#2 Solution

Consider the following real-time taskset:

Task	C	P	U
t1	2	10	0.200
t2	4	15	0.267
t3	10	35	0.286

Note1: C = worst-case execution time; P = period; U = utilization.
Assume a single core processor.

3- Draw a timeline (from time 0 to 40) of the taskset under the rate monotonic scheduling (RMS).



36

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

37