Embedded System Architecture - CSEN 701

Module 6: *Multi-tasking and Real-Time Systems*

Lecture 14: *RTOS Scheduling Algorithms - RM*

Dr. Eng. Catherine M. Elias

catherine.elias@guc.edu.eg

Lecturer, Computer Science and Engineering,
Faculty of Media Engineering and Technology, German University in Cairo

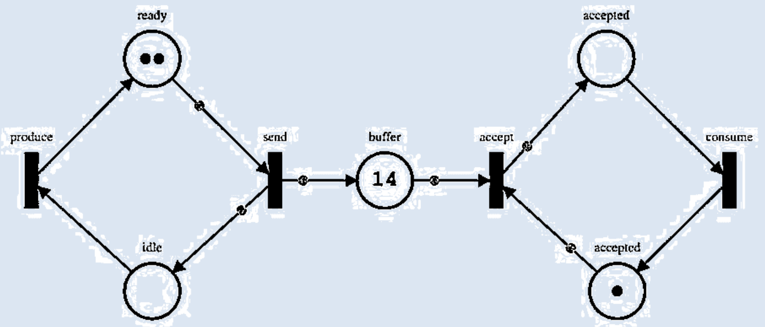
Outline

- Task Scheduling
- Rate-Monotonic Scheduling Algorithm

The big Picture

ES Modeling & Design
(2 Modules)

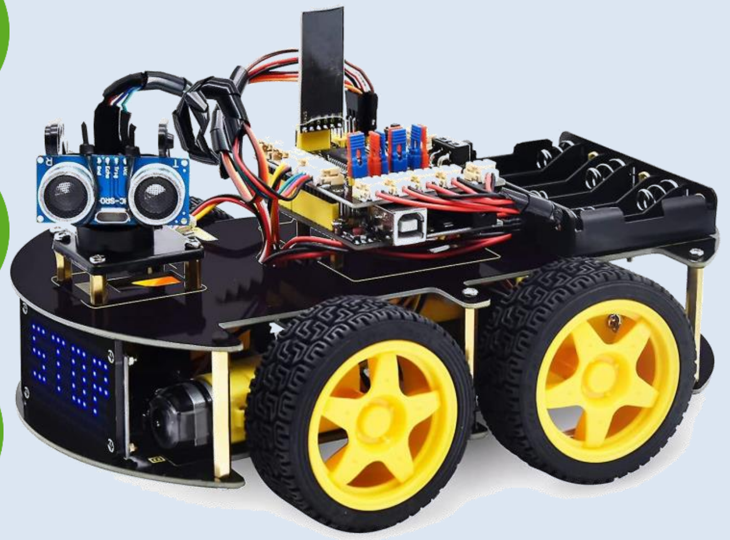
- System Modeling
- Design Considerations
- Power management and optimization
- Reliability and fault tolerance



The Real-time Embedded System

ES Hardware Components
(4 Modules)

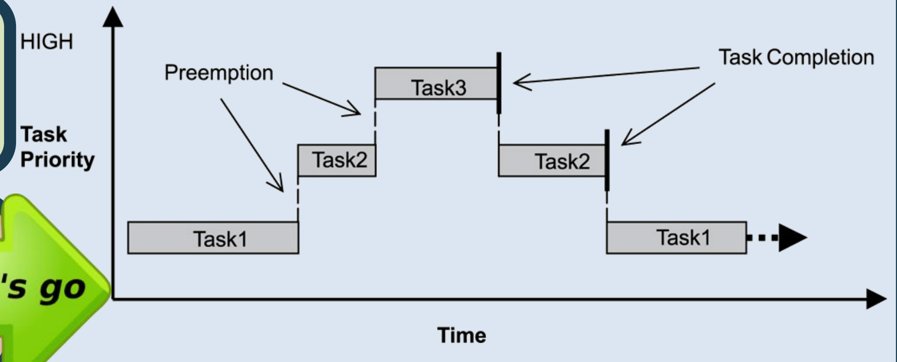
- Microcontroller Fundamentals
- Embedded programming languages
- Embedded Hardware
- Communication and Networking



ES Software Components
(1 Module)

- Real-Time Systems
- Multi-tasking
- Scheduling
- Resource Management

Let's go



Embedded System Tools & Software Development
(2 Modules)

- Debugging techniques
- Interrupts and exception handling
- Memory management

Let's go

Real-Time Constraints

- The **RTOS** is responsible for running concurrent tasks, while **maintaining their timing constraints** which are classified to:
  - **Hard timing constraints:** Missing its deadline is considered a fatal error (Example: inflating airbags after an accident)
  - **Soft timing constraints:** Missing the deadline or execute it a little bit late is not an issue (Example: opening mobile application)

- **Task scheduling** is used to run the concurrent tasks **while ensuring meeting the deadlines** (Feasible Schedule).

Task Scheduling

Tasks

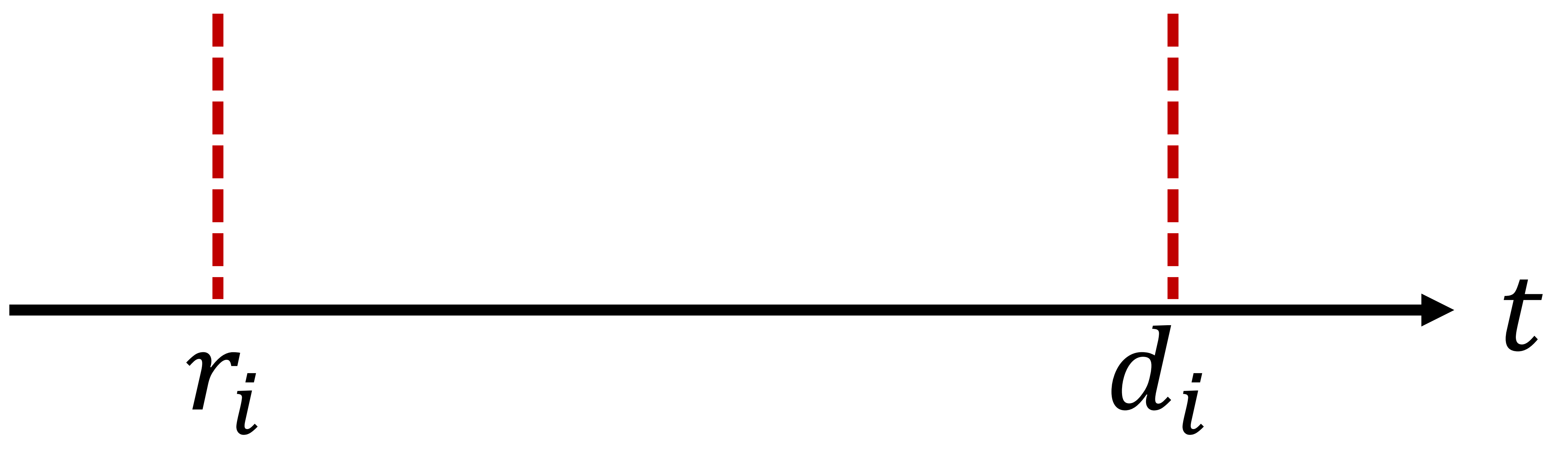
- A Task is the unit of work executed by the CPU
- A Task can be either:
  - **Periodic:**
    - Tasks **repeated** after a **period time** (p)
    - They have **hard deadlines** as the task must be completed before the end of the period (Ex: Reading sensory data)
  - **Aperiodic:**
    - unpredictable **one shot** tasks, having **soft or no deadlines**. (Ex: pushing a button in a vending machine)
  - **Sporadic:**
    - unpredictable **one shot** tasks, however they introduce a **hard deadline**. (Ex: inflating airbags after an accident).

Tasks Specification

- In real-time systems, a task (i) can be defined by the following timing parameters:

  - Release time ($\boldsymbol{r_i}$):
    - Time instant at which the task is ready to be executed

  - Deadline ($\boldsymbol{d_i}$):
    - Time instant where the execution of the task must be completed



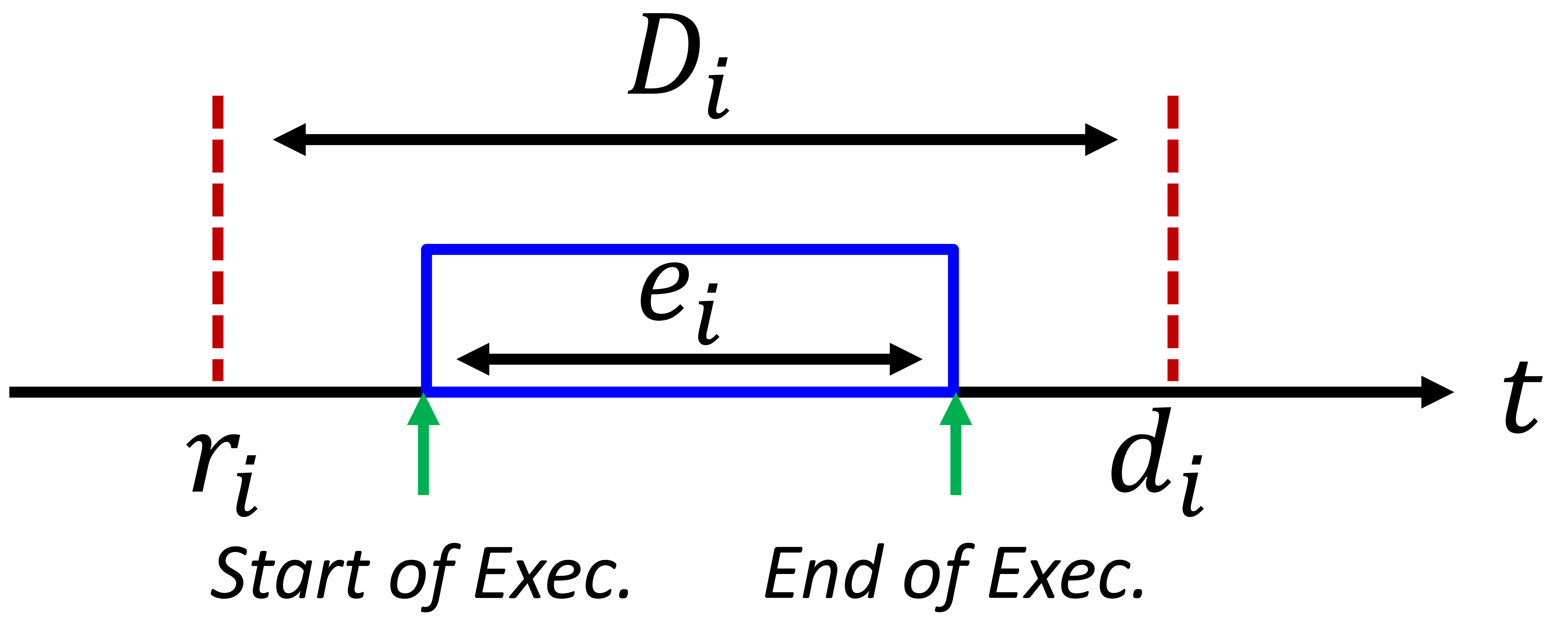
Tasks Specification

- In real-time systems, a task (i) can be defined by the following timing parameters:

  - **Relative deadline (D_i):**
    - Time difference between the release time and the deadline

$$D_i = d_i - r_i$$

  - **Execution time (e_i):**
    - The time required for the processor to execute the task



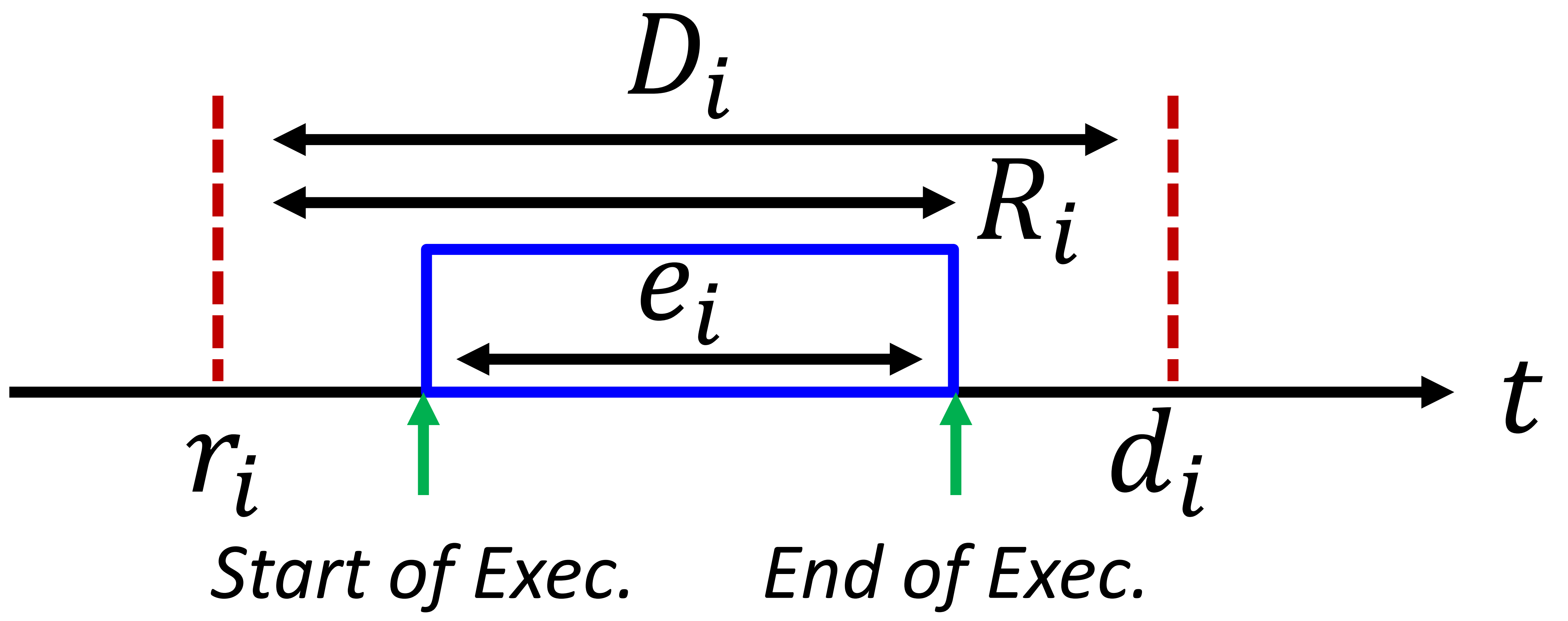
Tasks Specification

- In real-time systems, a task (i) can be defined by the following timing parameters:

  - **Response time (R_i):**
    - Time difference between the release time and the end of execution.

$$\boldsymbol{R_i = f_i - r_i}$$

  - The execution of task can start at any time after the release time, but the execution must end before the deadline

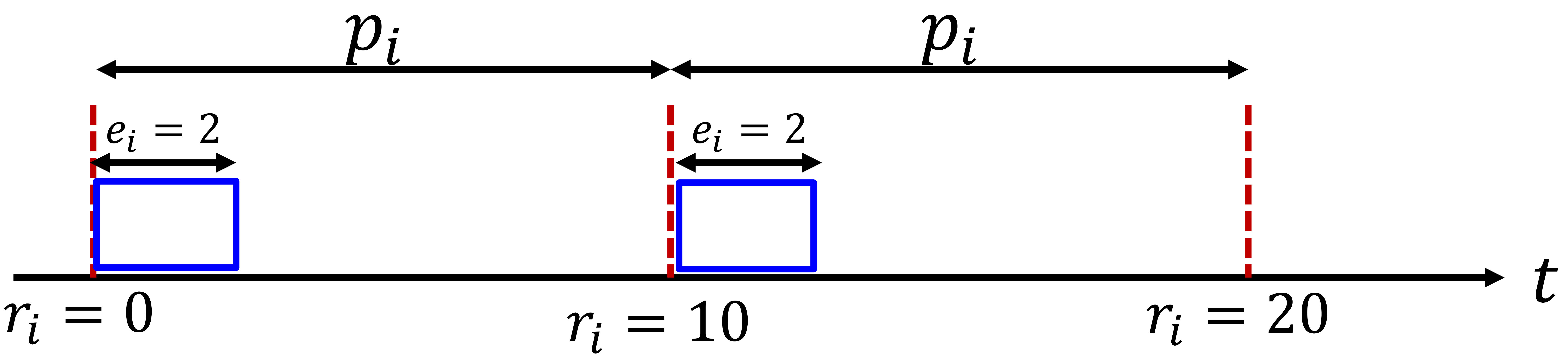


Periodic Tasks Specification

- **For periodic tasks:**
  - ➢The **relative deadline** is defined by the period of the task. Thus to schedule periodic task. A task is defined as:

$$\boldsymbol{T_i = (p_i, e_i)}$$

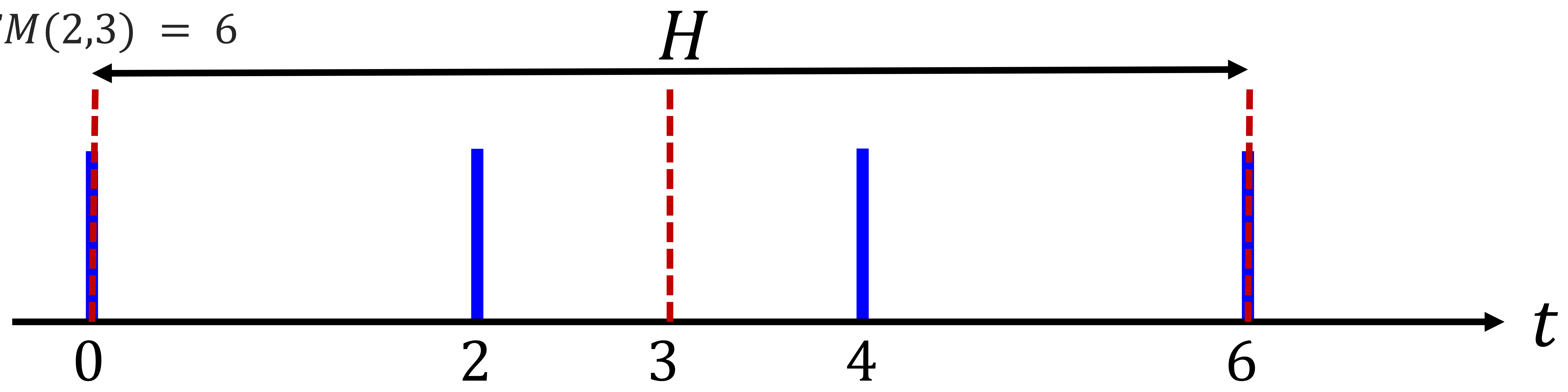
  - ➢Ex: A task T = (10,2), is released every 10 time units, have a relative deadline of 10 time units. And requires 2 time units on the processor to be completed.



Periodic Tasks

- For a set of periodic tasks having different periods (p_i).
  - A **hyperperiod** (H) defines the **repetition period of the whole set**.
  - A hyperperiod is defined as the **Least common multiple** of the different period values.

- Example:
  - $p_1 = 3,$
  - $p_2 = 2,$
  - $\therefore H = LCM(2,3) = 6$



Rate Monotonic Scheduling

- The most well-known static priority scheduling algorithm is Rate-Monotonic Scheduling (RM).

- It is used in scheduling periodic tasks.

- The **priority** is assigned based on the **period**.
  - Tasks having the **shortest period** has the **highest priority**.

Rate Monotonic Scheduling

Example

- 3 periodic tasks are scheduled using RM algorithm. The tasks are defined as:

$$T_1 = (4, 1),$$
$$T_2 = (5, 1),$$
$$T_3 = (10, 3)$$

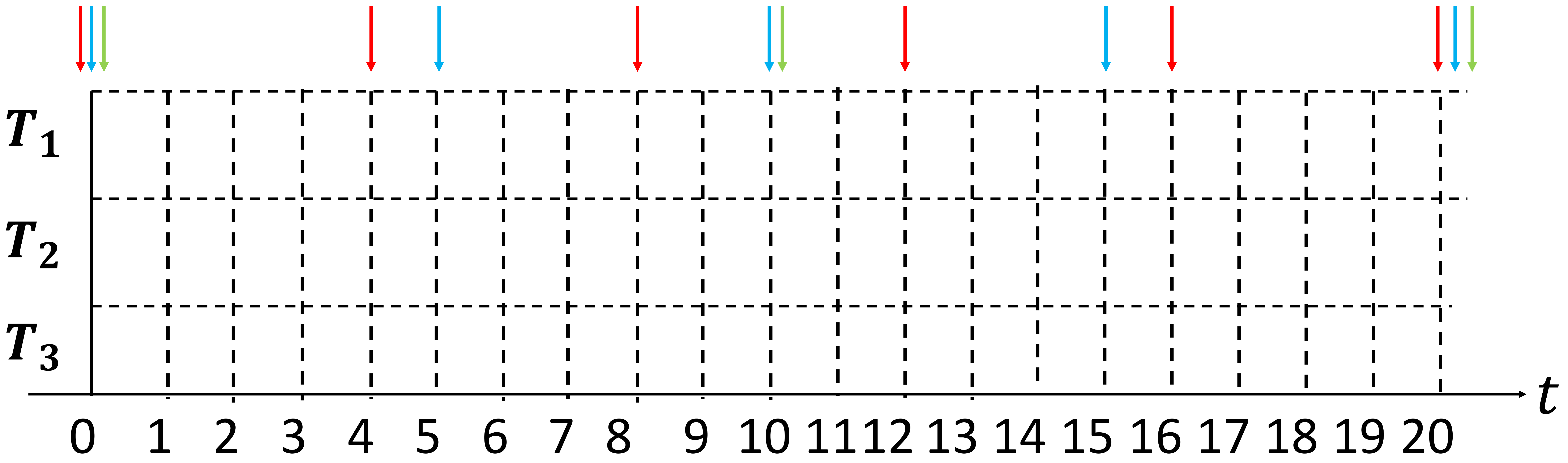
- Construct the schedule

Task	p	e	Priority
T_1	4	1	1
T_2	5	1	2
T_3	10	3	3

Example

Task	p	e	Priority
T_1	4	1	1
T_2	5	1	2
T_3	10	3	3

- 3 periodic tasks are scheduled using RM algorithm.
  - ***HyperPeriod:*** H = LCM(4, 5, 10) = 20

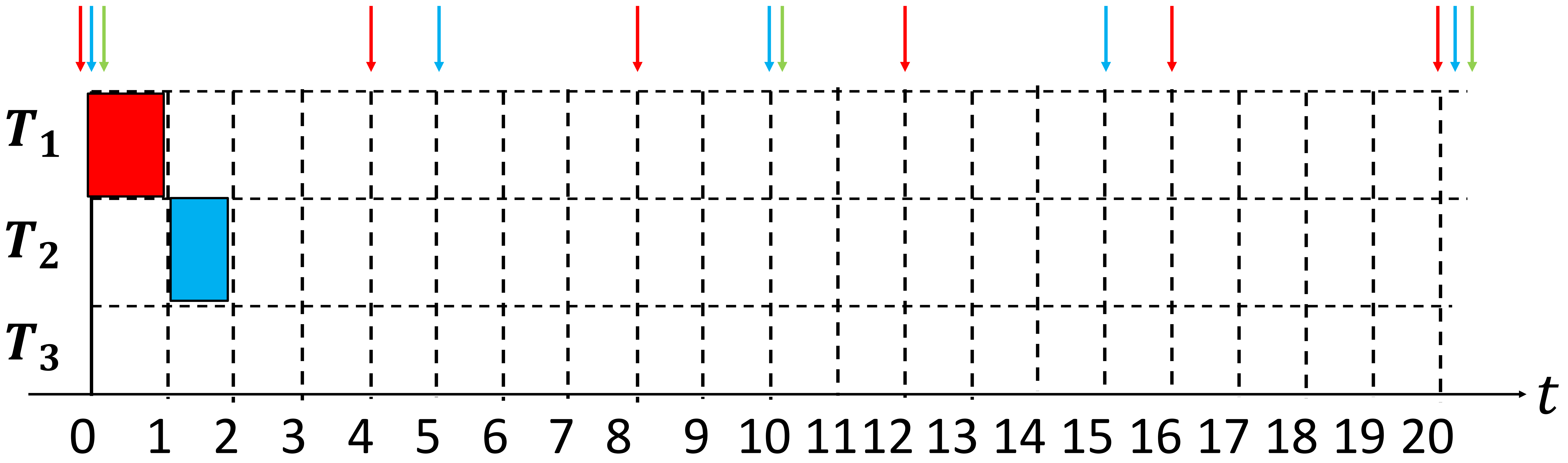


Rate Monotonic Scheduling

Example

Task	p	e	Priority
T_1	4	1	1
T_2	5	1	2
T_3	10	3	3

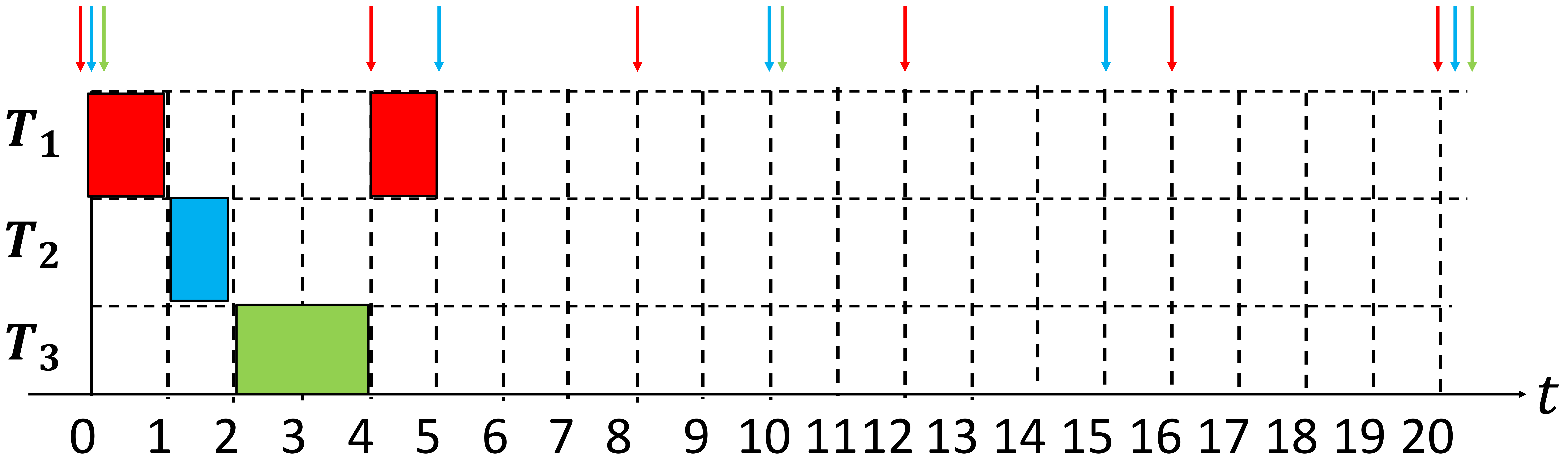
- 3 periodic tasks are scheduled using RM algorithm.
  - ➢At t=0, all tasks are ready. T1 has the shortest period, thus the highest priority. Thus T1 is executed.
  - ➢At t=1, the T1 instance is finished and the processor is idle. Thus the highest priority is chosen which is T2



Example

Task	p	e	Priority
T_1	4	1	1
T_2	5	1	2
T_3	10	3	3

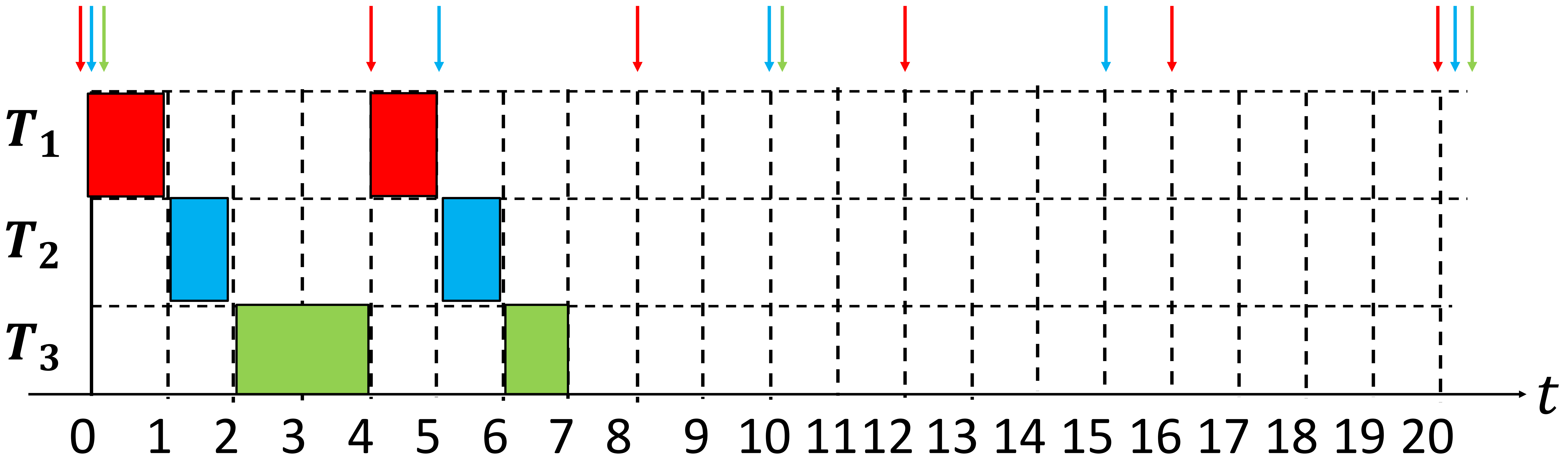
- 3 periodic tasks are scheduled using RM algorithm.
  - At t=2 and t=3, Task 3 is the only task ready to run, thus it is executed.
  - At t=4, T1 is ready, and it has a higher priority, thus T3 is preempted to execute T1



Example

- 3 periodic tasks are scheduled using RM algorithm.
  - ➢At t=5, T2 is ready and has higher priority than T3, thus T2 is executed.
  - ➢At t=6, T3 is resumed to finish the remaining 1 time unit

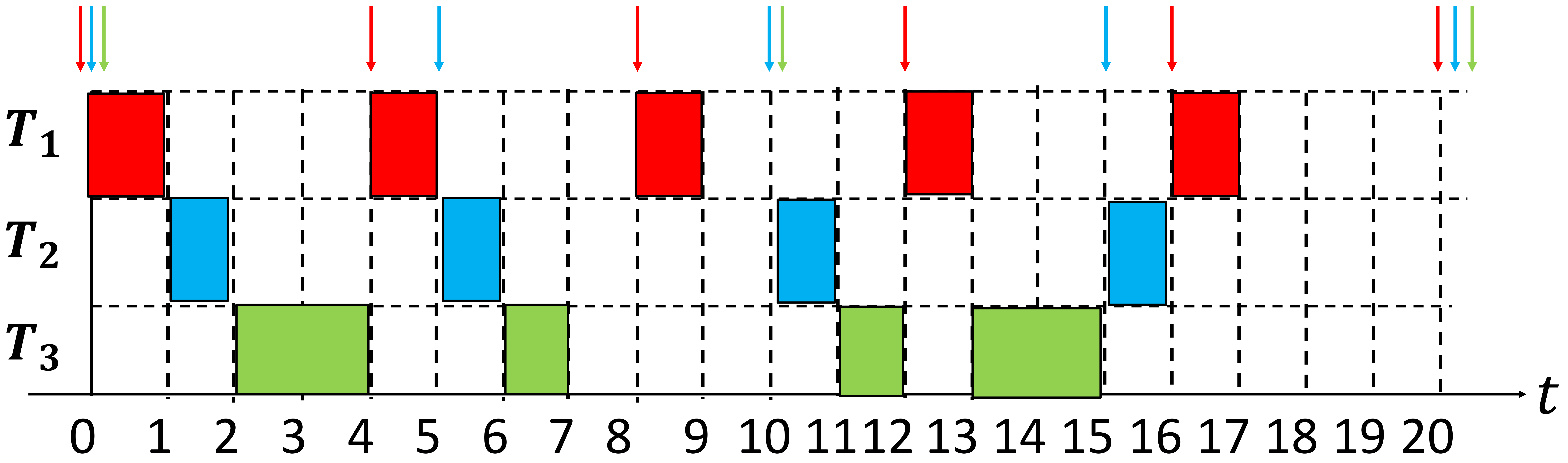
Task	p	e	Priority
T_1	4	1	1
T_2	5	1	2
T_3	10	3	3



Example

- 3 periodic tasks are scheduled using RM algorithm.
  - ➢The final schedule can be computed as shown

Task	p	e	Priority
T_1	4	1	1
T_2	5	1	2
T_3	10	3	3



RM Schedulability Check

- To check whether a set of tasks can be scheduled using RM. Two checks are used:
  - ➢Utilization Bound
  - ➢Response Time / Time Demand Analysis

RM Schedulability Check: *Utilization Bound*

- A ***Task utilization*** is defined as:

$$U_i = \frac{e_i}{p_i}$$

  - *For a task to be schedulable, its utilization value must be **less than or equal to 1***

- The ***total utilization*** of a set of tasks is defined as:

$$U_{total} = \sum \frac{e_i}{p_i}$$

RM Schedulability Check: *Utilization Bound*

- A ***sufficient condition for the RM scheduling*** to schedule a set of n tasks is:

$$U_{total} \leq U_{RM}$$

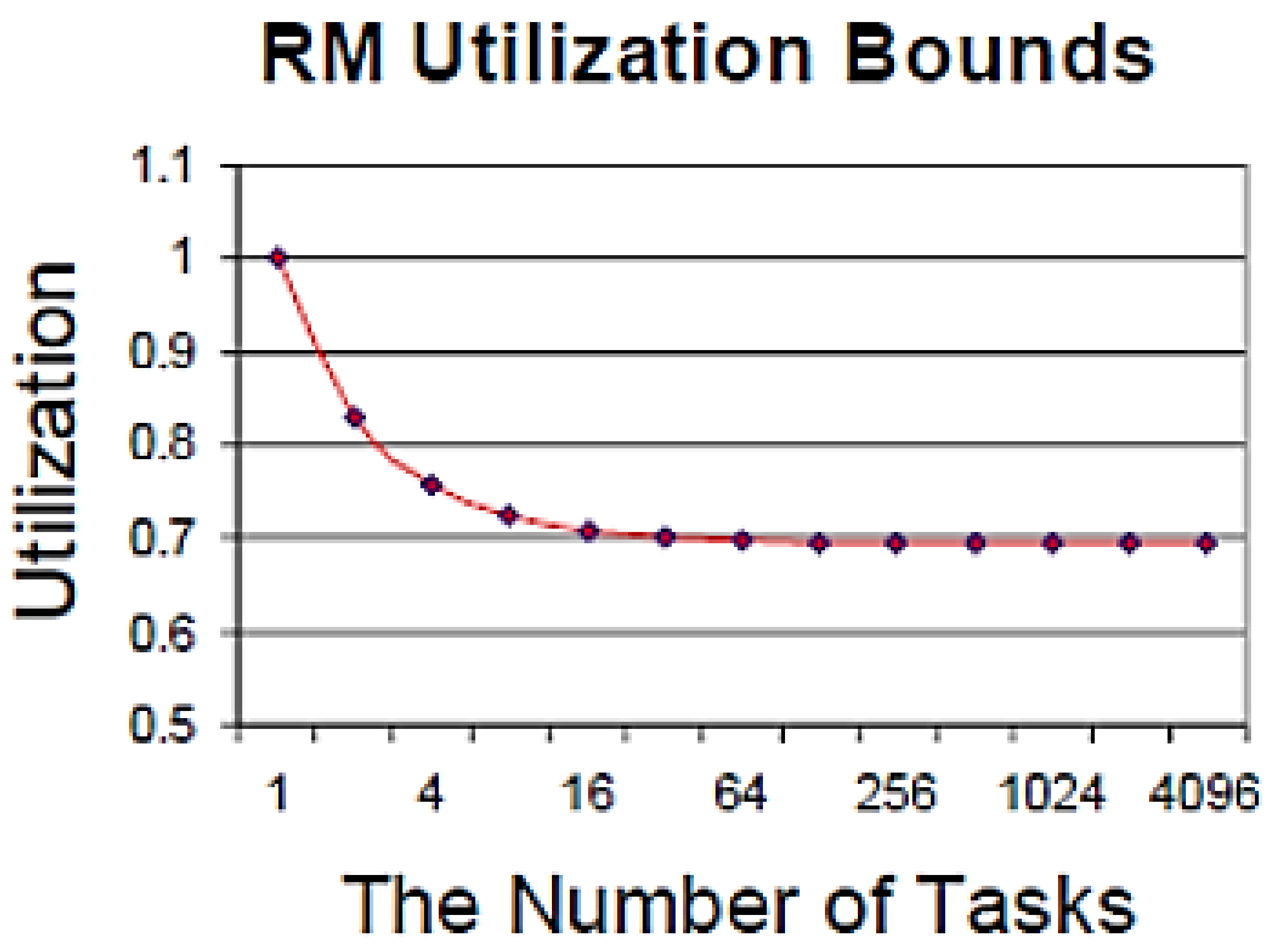
$$U_{RM} = n(2^{\frac{1}{n}} - 1)$$

- If the condition is **satisfied**, then this set of tasks **can be scheduled** using **RM**.
- If the condition is **not satisfied**, this set of tasks may or **may not be schedulable** by **RM**

RM Schedulability Check: *Utilization Bound*

B(1)=1.0	B(4)=0.756	B(7)=0.728
B(2)=0.828	B(5)=0.743	B(8)=0.724
B(3)=0.779	B(6)=0.734	$U(\infty)=0.693$

Note that $U(\infty)=0.693$!



RM Schedulability Check: *Utilization Bound* Example

Task	p	e	Priority
T_1	4	1	1
T_2	5	1	2
T_3	10	3	3

- The total utilization is:

$$U_{total} = \frac{1}{4} + \frac{1}{5} + \frac{3}{10} = 0.75$$

- The RM utilization upper bound

$$U_{RM} = 3 * \left(2^{1/3} - 1\right) = 0.779$$

$$\because U_{total} \leq U_{RM}$$

∴These tasks are schedulable using RM

RM Schedulability Check

- To check whether a set of tasks can be scheduled using RM. Two checks are used:
  - ➢Utilization Bound
  - ➢Response Time / Time Demand Analysis

RM Schedulability Check: *Response Time / Time Demand Analysis*

- TDA is a technique to check each Task schedulability using RM scheduling
- The TDA provides a mathematical way to prove that a task is schedulable at its **critical instant**. Thus proving the **schedulablitiy** of a set of tasks using **RM**.
- For each task (i), the TDA analysis is performed at its **critical instants**.
- A critical instant of a task (i) is defined where **tasks of higher priorities are released** and will **preempt** it, causing the **maximum response time** for the task (i).

RM Schedulability Check: *Response Time / Time Demand Analysis*

- Generally, The time demand of a Task (i) to be finished at time ***t*** is the **sum of all the execution times of higher priority tasks and the execution time task(*i*):**

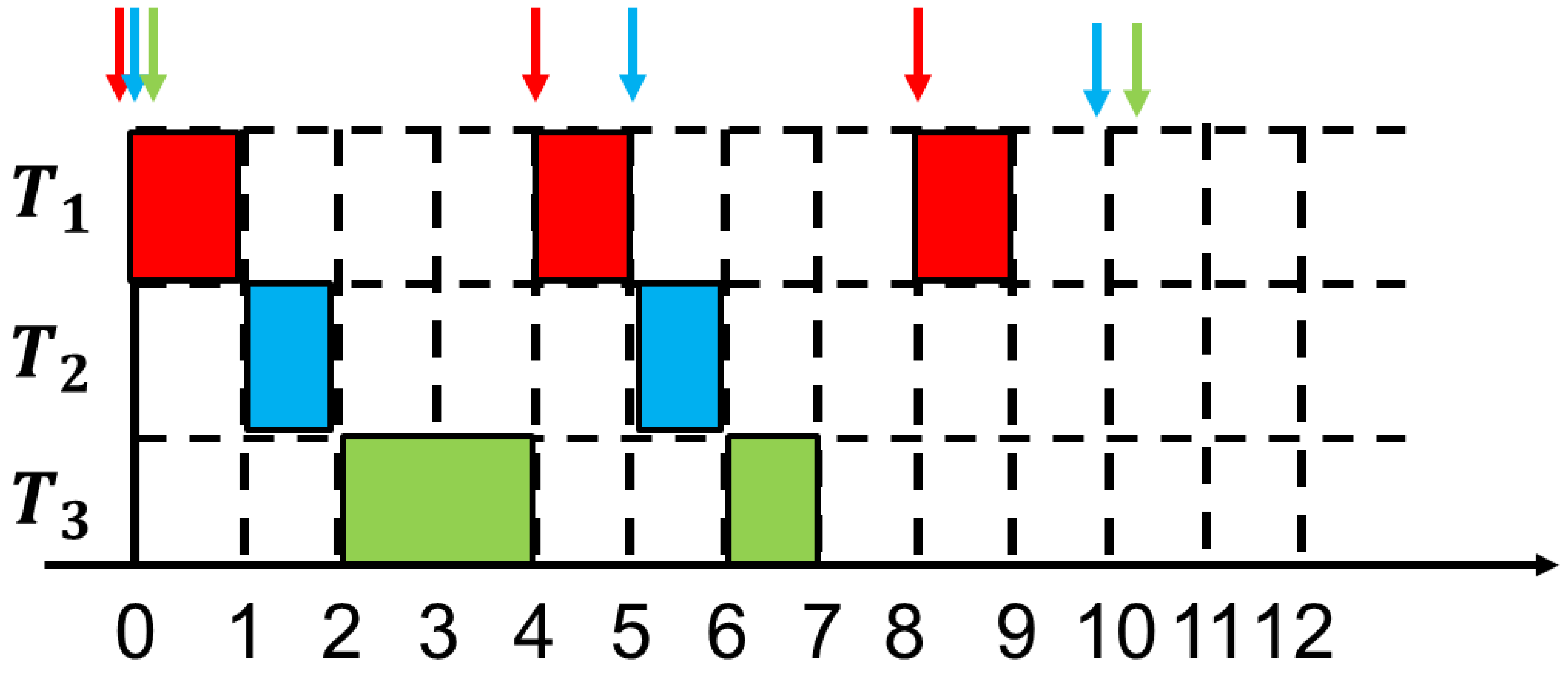
$$w_i(t) = \sum_{j=1}^{i-1} ceil\left(\frac{t}{p_j}\right) * e_j + e_i$$

- A task is schedulable if:
  - Its utilization $U = \frac{e_i}{p_i}$ is less than or equal 1
  - There exists a time instant ($t < p_i$) where:

$$w_i(t) \le t$$

RM Schedulability Check: *Response Time / Time Demand Analysis*

- The time required to finish T3 in the shown system is :



$$w_3(t) = ceil\left(\frac{t}{p_1}\right) * e_1 + ceil\left(\frac{t}{p_2}\right) * e_2 + e_3$$

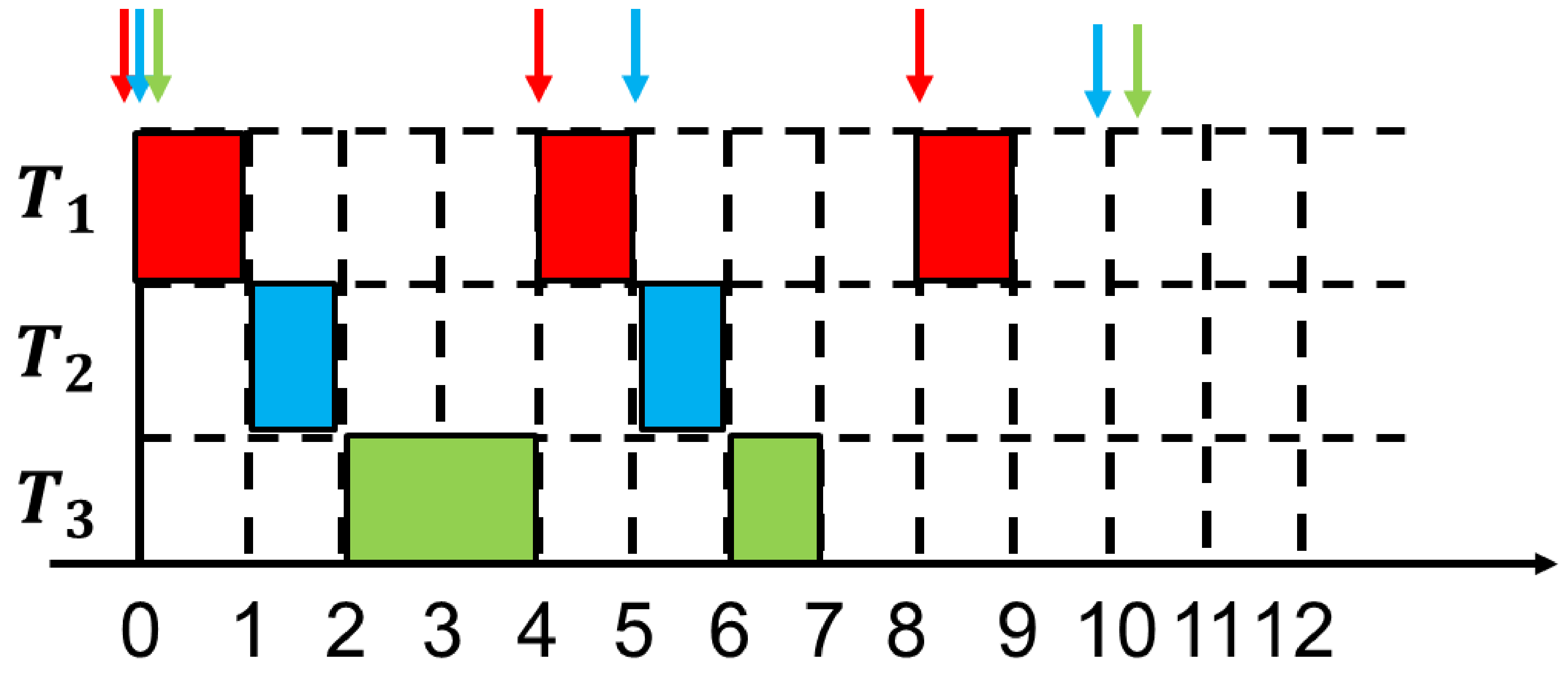
Num. of instances of T1 before instance (t)

Num. of instances of T2 before instance (t)

RM Schedulability Check: *Response Time / Time Demand Analysis* Example

- Check the RM schedulability of the following set of tasks using TDA:

$$T_1 = (4, 1),$$
$$T_2 = (5, 1),$$
$$T_3 = (10, 3)$$

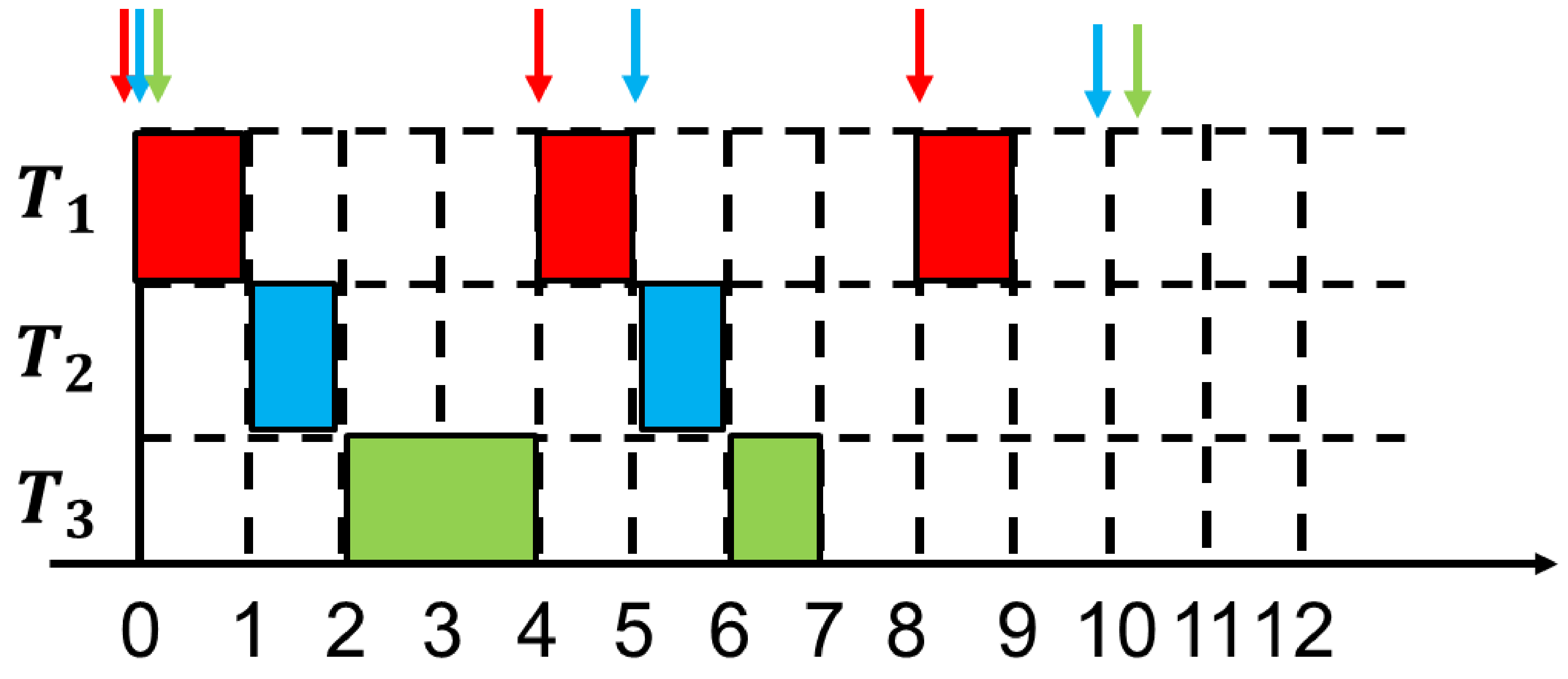


- **Schedulability of *Task1*:**
  - ➢ $U = \frac{1}{4} = 0.25 \leq 1$
  - ➢ Task 1 has the highest priority, thus it is certainly schedulable (It is never preempted)

RM Schedulability Check: *Response Time / Time Demand Analysis* Example

- Check the RM schedulability of the following set of tasks using TDA:

$$T_1 = (4, 1),$$
$$T_2 = (5, 1),$$
$$T_3 = (10, 3)$$



- **Schedulability of *Task2*:**
  - ➢ $U = \frac{1}{5} = 0.2 \leq 1$
  - ➢ To find the instance (t), the condition is checked at the **release times of T_1 during the period of T_2** as well as **the end of the period**. Thus, we will check the condition at **t = 4,5**
  - ➢ At t=4 :

$$w_2(4) = ceil\left(\frac{4}{p_1}\right) * e_1 + e_2$$

$$= (1 * 1) + 1 = 2 \leq 4 \text{ (Satisfied)}$$

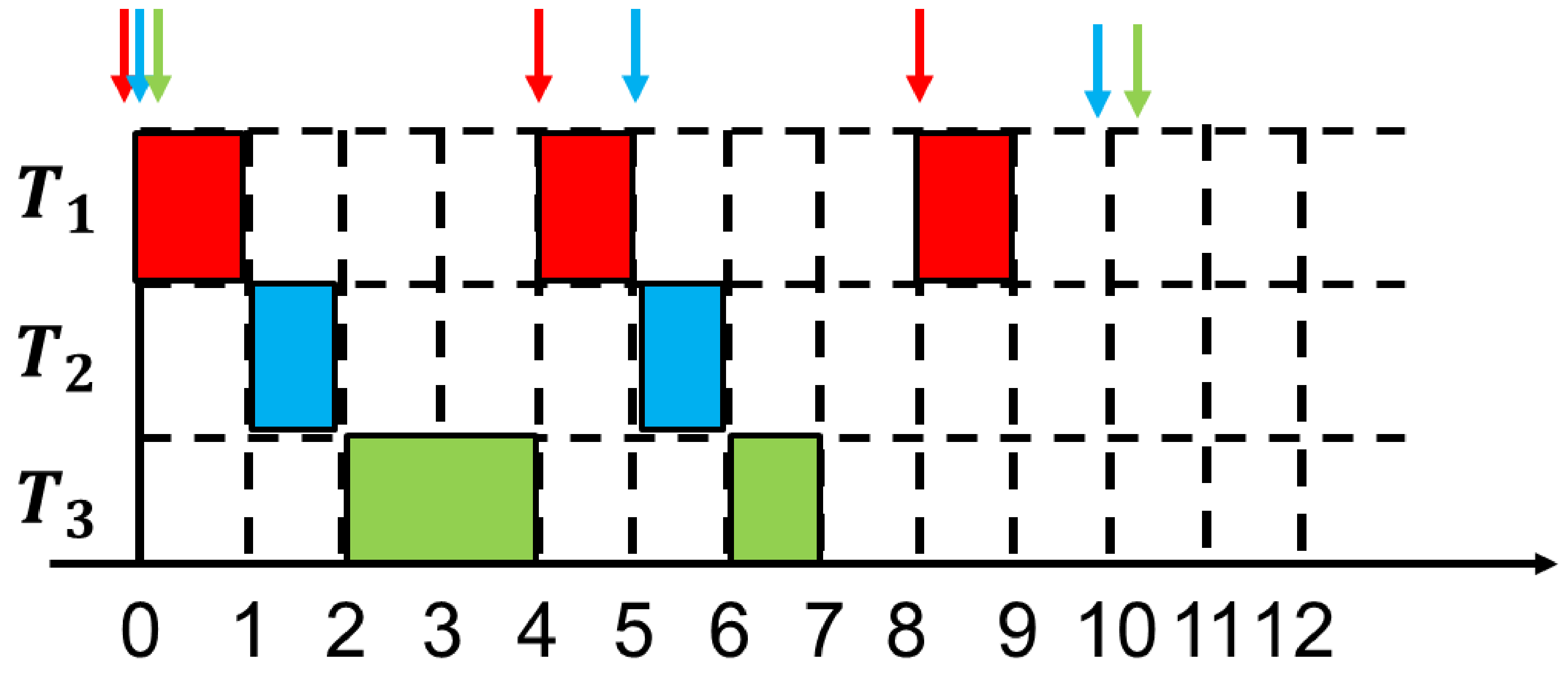
  - ➢ Task 2 is schedulable before t = 4

German University in Cairo
الجامعة الألمانية بالقاهرة

RM Schedulability Check: *Response Time / Time Demand Analysis* Example

- Check the RM schedulability of the following set of tasks using TDA:

$$T_1 = (4, 1),$$
$$T_2 = (5, 1),$$
$$T_3 = (10, 3)$$



- **Schedulability of *Task3*:**
  - $U = \frac{3}{10} = 0.3 \leq 1$
  - To find the instance (t), the condition is checked at the **release times of T_1, T_2 during the period of T_3** as well as **the end of the period**. Thus, we will check the condition at **t = 4, 5, 8, 10**
  - At t=4:

$$w_3(4) = ceil\left(\frac{4}{p_1}\right) * e_1 + ceil\left(\frac{4}{p_2}\right) * e_2 + e_3$$

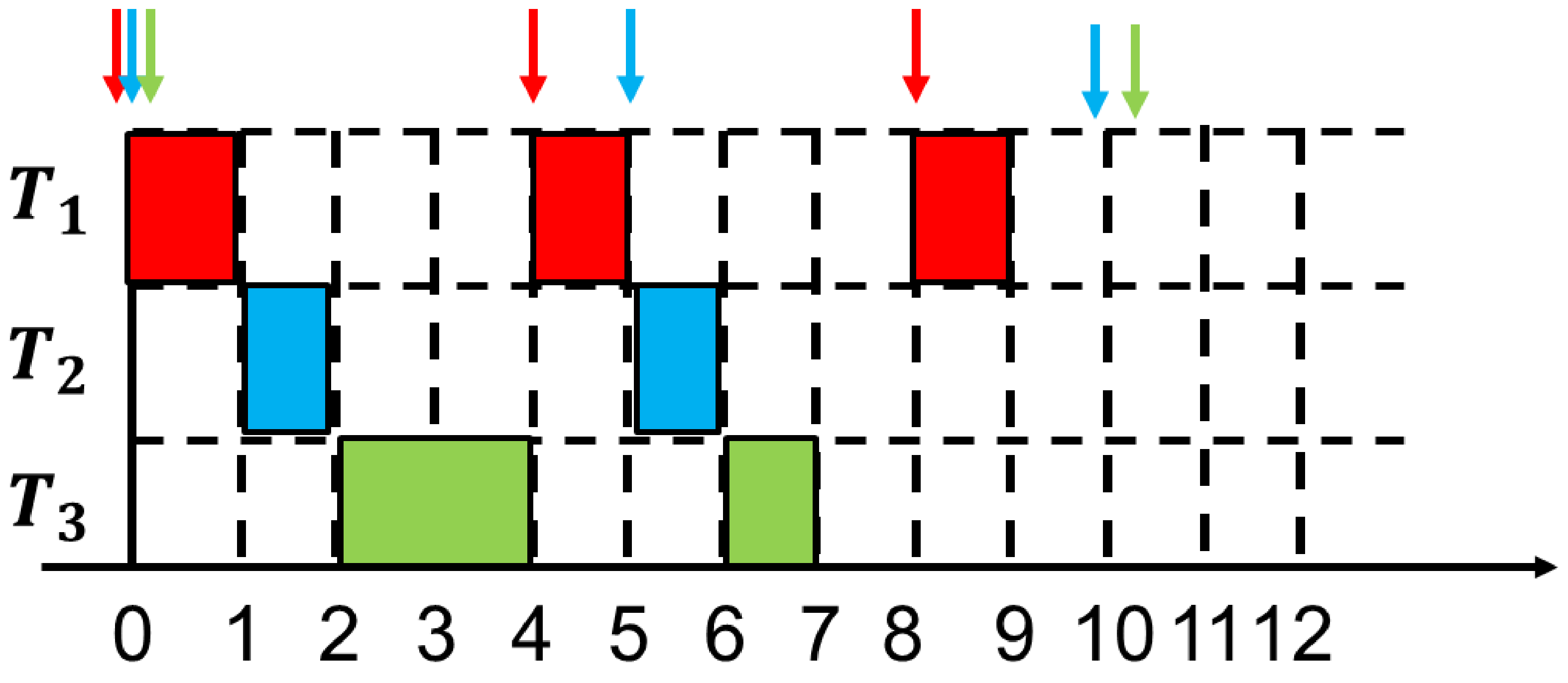
$$= (1 * 1) + (1 * 1) + 3 = 5 \geq 4 \text{ (Not Satisfied)}$$

GUC German University in Cairo الجامعة الألمانية بالقاهرة

RM Schedulability Check: *Response Time / Time Demand Analysis* Example

- Check the RM schedulability of the following set of tasks using TDA:

$$T_1 = (4, 1),$$
$$T_2 = (5, 1),$$
$$T_3 = (10, 3)$$



- **Schedulability of *Task3*:**
  - $U = \frac{3}{10} = 0.3 \leq 1$
  - To find the instance (t), the condition is checked at the **release times of T_1, T_2 during the period of T_3** as well as **the end of the period**. Thus, we will check the condition at **t = 4, 5, 8, 10**
  - At t=5 :

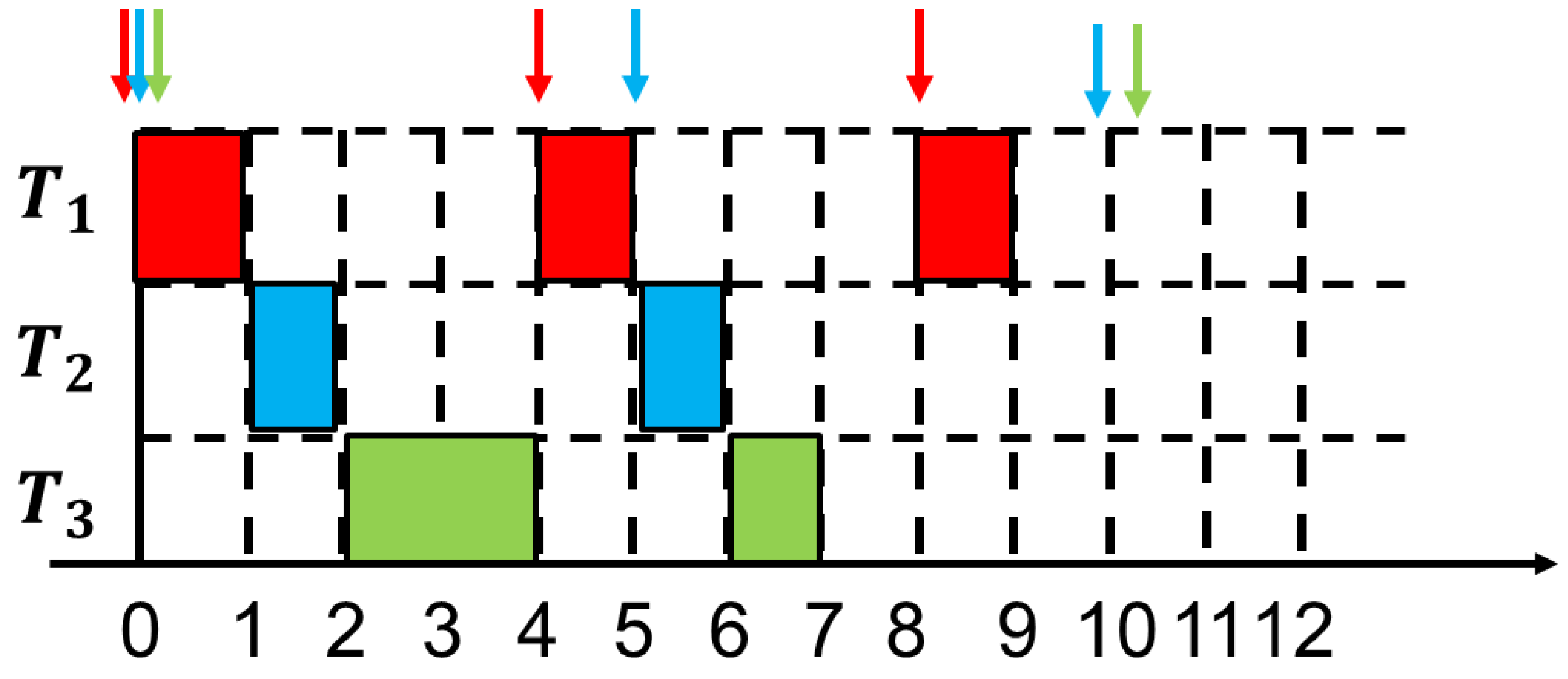
$$w_3(5) = ceil\left(\frac{5}{p_1}\right) * e_1 + ceil\left(\frac{5}{p_2}\right) * e_2 + e_3$$

$= (2 * 1) + (1 * 1) + 3 = 6 \geq 5$ (Not Satisfied)

RM Schedulability Check: *Response Time / Time Demand Analysis* Example

- Check the RM schedulability of the following set of tasks using TDA:

$$T_1 = (4, 1),$$
$$T_2 = (5, 1),$$
$$T_3 = (10, 3)$$



- **Schedulability of *Task3*:**
  - $U = \frac{3}{10} = 0.3 \leq 1$
  - To find the instance (t), the condition is checked at the **release times of T_1, T_2 during the period of T_3** as well as **the end of the period**. Thus, we will check the condition at **t = 4, 5, 8, 10**
  - At t = 8 :

$$w_3(8) = ceil\left(\frac{8}{p_1}\right) * e_1 + ceil\left(\frac{8}{p_2}\right) * e_2 + e_3$$

$$= (2 * 1) + (2 * 1) + 3 = 7 \leq 8 \text{ (Satisfied)}$$

  - Task 3 is schedulable before t = 8

German University in Cairo
الجامعة الألمانية بالقاهرة

RM Schedulability Check: *Response Time / Time Demand Analysis* Example

- Check the RM schedulability of the following set of tasks using TDA:

$$T_1 = (4, 1),$$
$$T_2 = (5, \mathbf{2}),$$
$$T_3 = (10, \mathbf{3.1})$$

- In this example, Task 3 misses the deadline by 0.1 time units
- Using TDA analysis for task 3 at t = 4,5,8,10
    - $w_3(4) = ceil\left(\frac{4}{p_1}\right) * e_1 + ceil\left(\frac{4}{p_2}\right) * e_2 + e_3 = 6.1 \geq 4$ **(Not Satisfied)**
    - $w_3(5) = ceil\left(\frac{5}{p_1}\right) * e_1 + ceil\left(\frac{5}{p_2}\right) * e_2 + e_3 = 7.1 \geq 5$ **(Not Satisfied)**
    - $w_3(8) = ceil\left(\frac{8}{p_1}\right) * e_1 + ceil\left(\frac{8}{p_2}\right) * e_2 + e_3 = 9.1 \geq 8$ **(Not Satisfied)**
    - $w_3(10) = ceil\left(\frac{10}{p_1}\right) * e_1 + ceil\left(\frac{10}{p_2}\right) * e_2 + e_3 = 10.1 \geq 10$ **(Not Satisfied)**

For Further Inquiries, Please send an email

Catherine.elias@guc.edu.eg,
Catherine.elias@ieee.org

Thank you for your attention!

See you next time ☺