Technische Universität Braunschweig

INSTITUTE OF COMPUTER AND NETWORK ENGINEERING

Rechnerstrukturen 2 - Übung 11

SCHEDULING

Today‘s Goal…

- **Closer look at these topics:**
  - Simulation and Analysis
  - Modelling
  - TDMA Scheduling
  - Round Robin Scheduling
  - Rate Monotonic Rate Scheduling
  - Static Priority Preemptive Scheduling

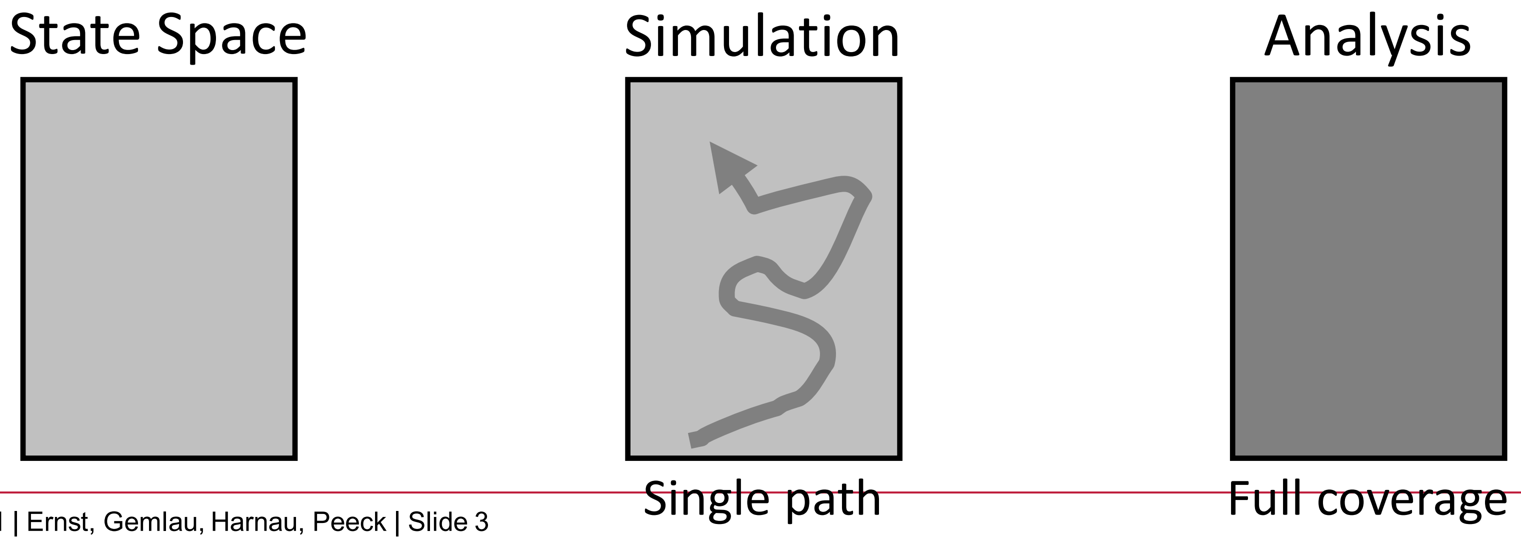
Simulation vs. Analysis

The problem with simulation

- Only covers one particular execution path
- Hard to simulate corner cases
- Does not scale with problem size

Analysis guarantees correctness

- Utilizes abstract models to describe system properties
- Ongoing research (SymTA/S tool developed at IDA)

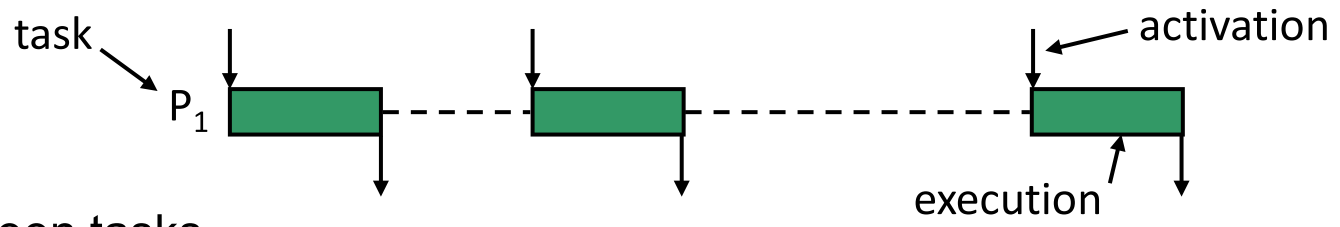


Modelling - Event Streams

Task activation can be time triggered or event triggered

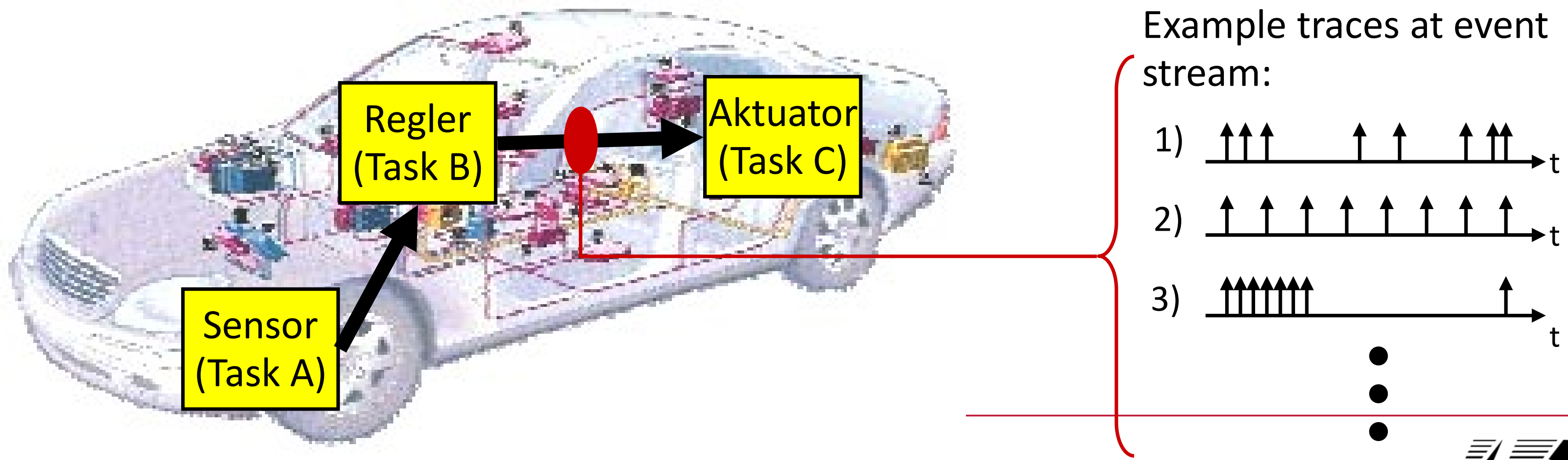
- In both ways modeled by event streams

- May trigger successor tasks by generating new events



Event streams

- Communication between tasks
- Properties depending on task behavior → multiple traces possible



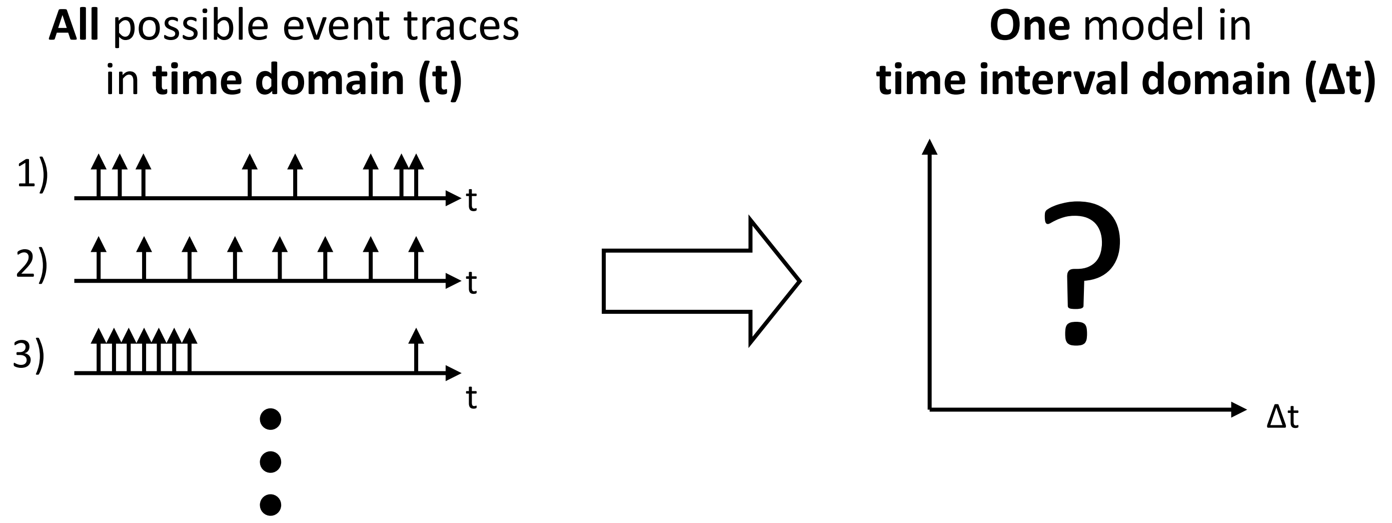
Modelling - Event Streams - Characterization

Characterization of event streams

- Instead of investigating *all* individual event traces and their timing, formal schedulability analysis performs *one* analysis on the **event bounds**

All activating events within a time window of given size Δt

- $\eta^+(\Delta t)$: Maximum number of events in window Δt
- $\eta^-(\Delta t)$: Minimum number of events in window Δt

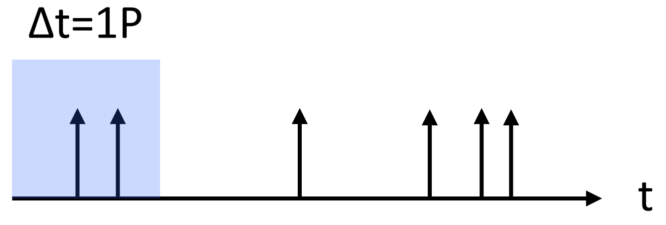

All possible event traces in **time domain (t)**
1)
2)
3)
t
t
t
One model in **time interval domain (Δt)**
?
Δt

Modelling - Time Domain to Time Interval Domain Transformation (1/2)

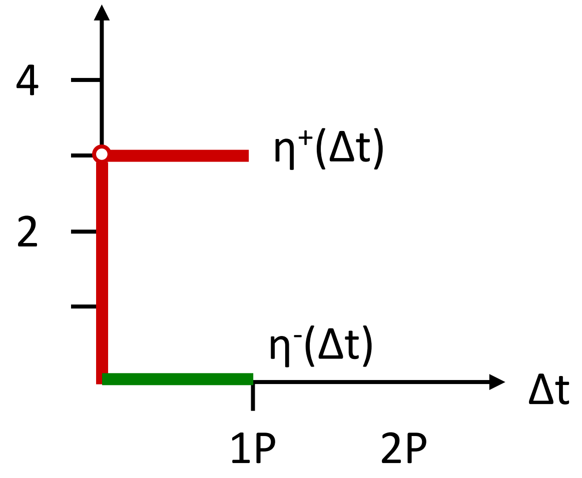
Arrival Curves in time interval domain (Δt)

- η+(Δt): Maximum number of events in window Δt
- η-(Δt): Minimum number of events in window Δt

Have to be done for every possible event trace!



Δt	Min. #Events	Max. #Events
1P	0	3

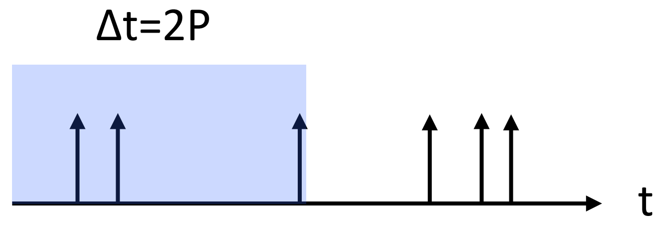


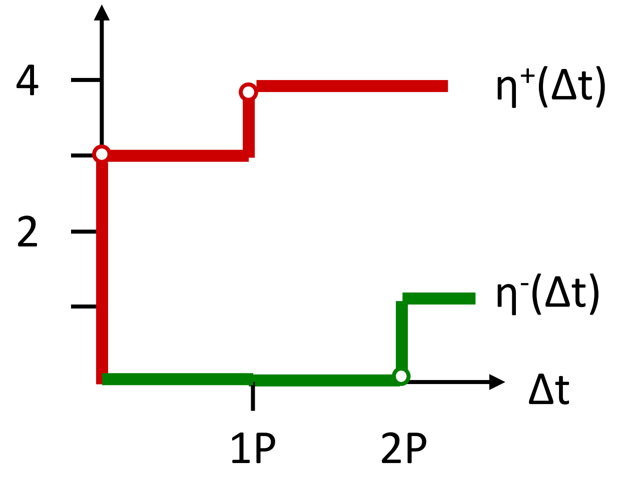
Modelling - Time Domain to Time Interval Domain Transformation (2/2)

Arrival Curves in time interval domain (Δt)

- η+(Δt): Maximum number of events in window Δt
- η-(Δt): Minimum number of events in window Δt

Have to be done for every possible event trace!





Δt	Min. #Events	Max. #Events
1P	0	3
2P	1	4

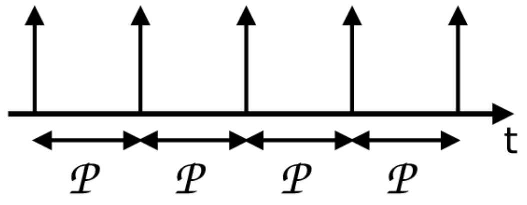
Modelling - Event Models

Abstraction in time domain (t)

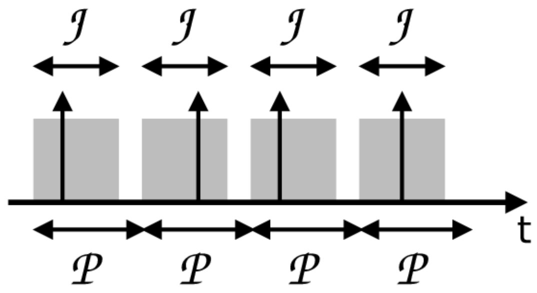
- Describe all traces on one event stream through a limited parameter set

Perform formal analysis on this parameter set: (P,J,dmin)

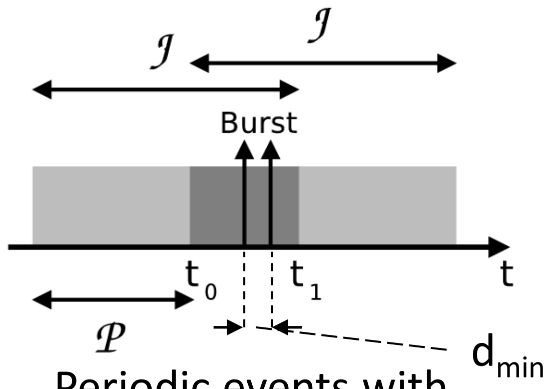
P…Period
J…Jitter
dmin…minimum distance between two consecutive events



Periodic events



Periodic events with jitter



Periodic events with jitter and burst

Allows conservative transformation from the actual event timing

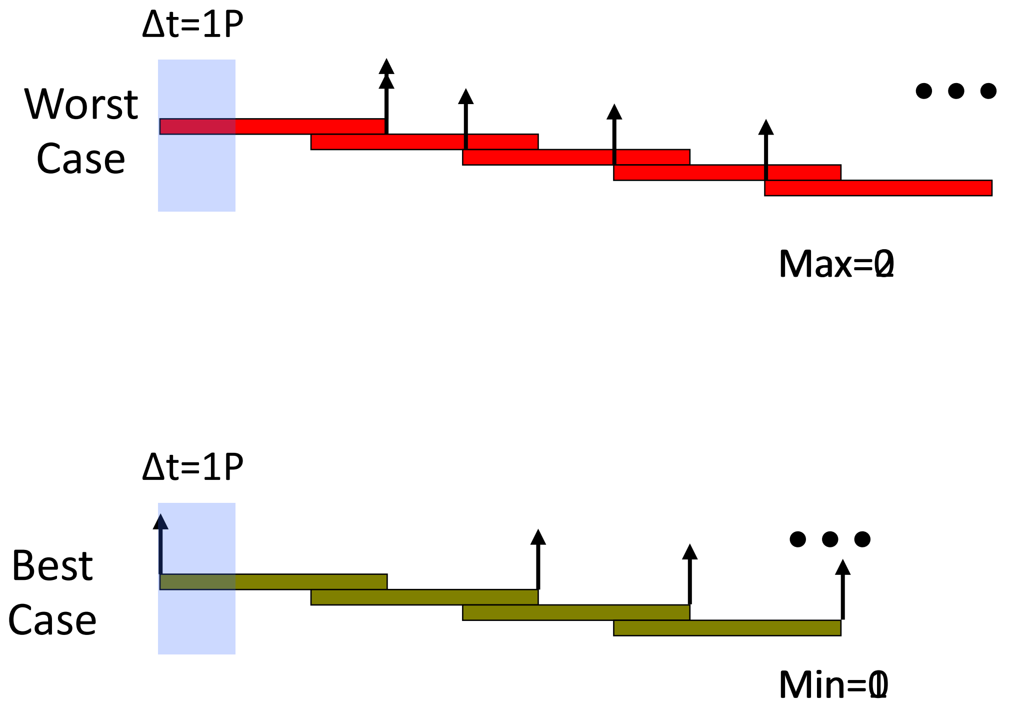
- Only one analysis necessary instead of checking all event correlations

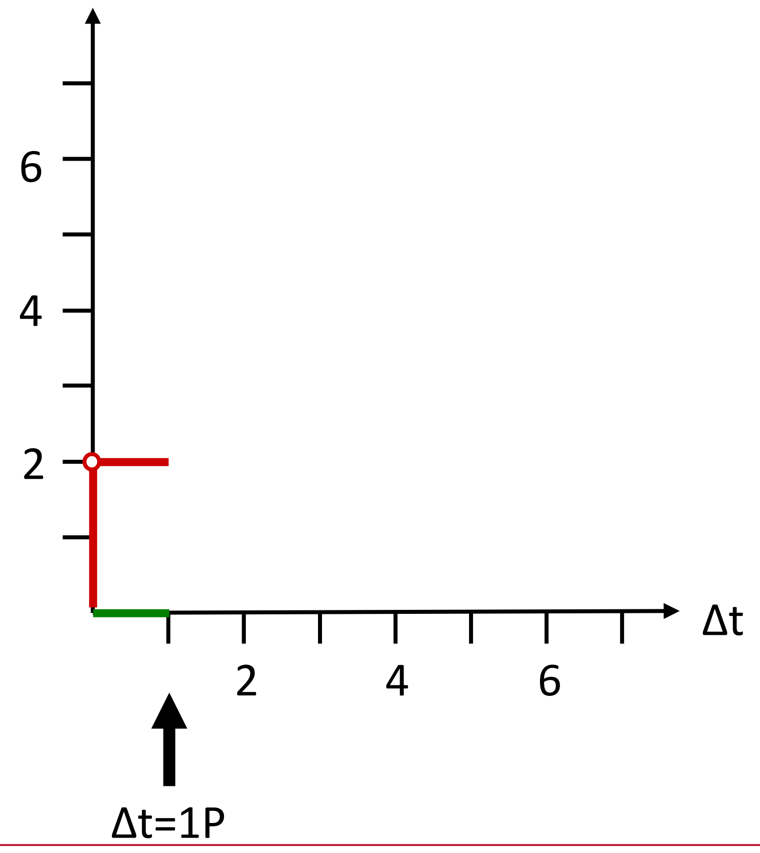
Modelling - Event Model: Periodic with Jitter (Δt=1P)

Event model: (2,3,0) ⟵ (Periode, Jitter, min. Distanz)

Analyze all possible event traces with Δt=1P

- Due to abstraction (2,3,0), we know the best and worst case streams



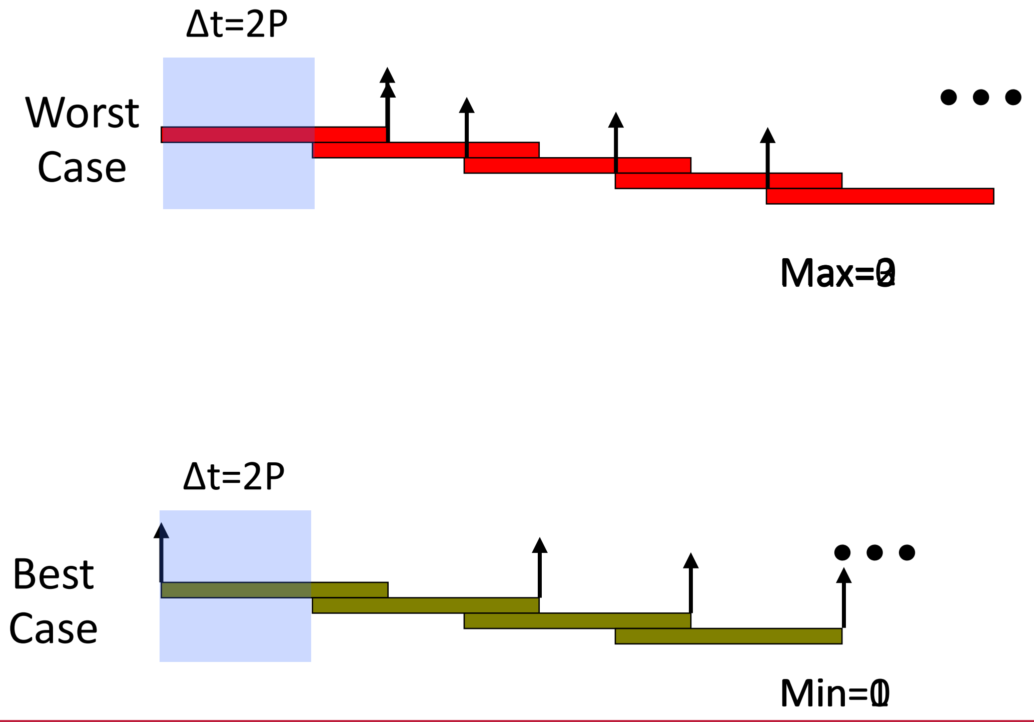


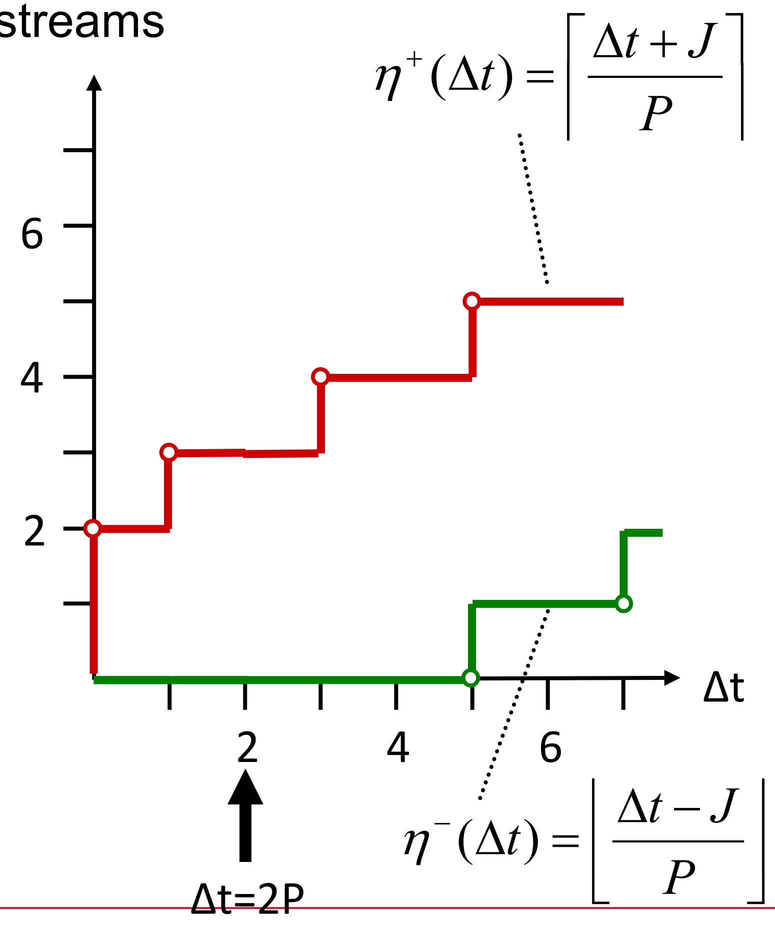
Modelling - Event Model: Periodic with Jitter (Δt=2P)

Event model: (2,3,0)

Analyze all possible event streams with Δt=2P

- Due to abstraction (2,3,0), we know the best and worst case streams





Reminder: Scheduling

Task

- Set of instructions
- (Sub)program

Scheduling

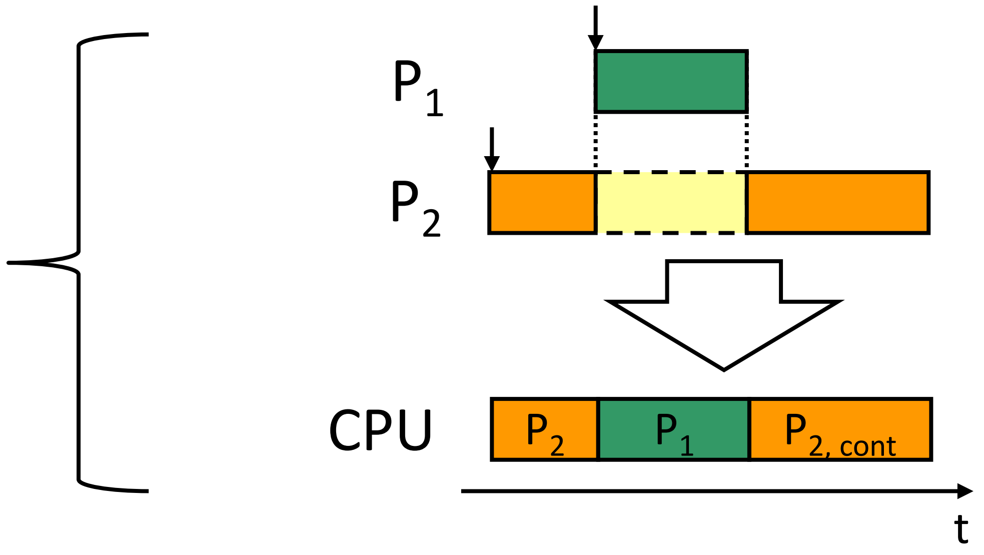
- Temporal partitioning of processing or communication resources
- Sequenced task execution

Scheduler

- Assigns tasks to available (usually limited) resources

Preemption

- Temporary interruption of a task
- No cooperation required by the interrupted task
- Interrupted task is resumed eventually



Scheduling Analysis

What guarantees can be made given a certain environment?

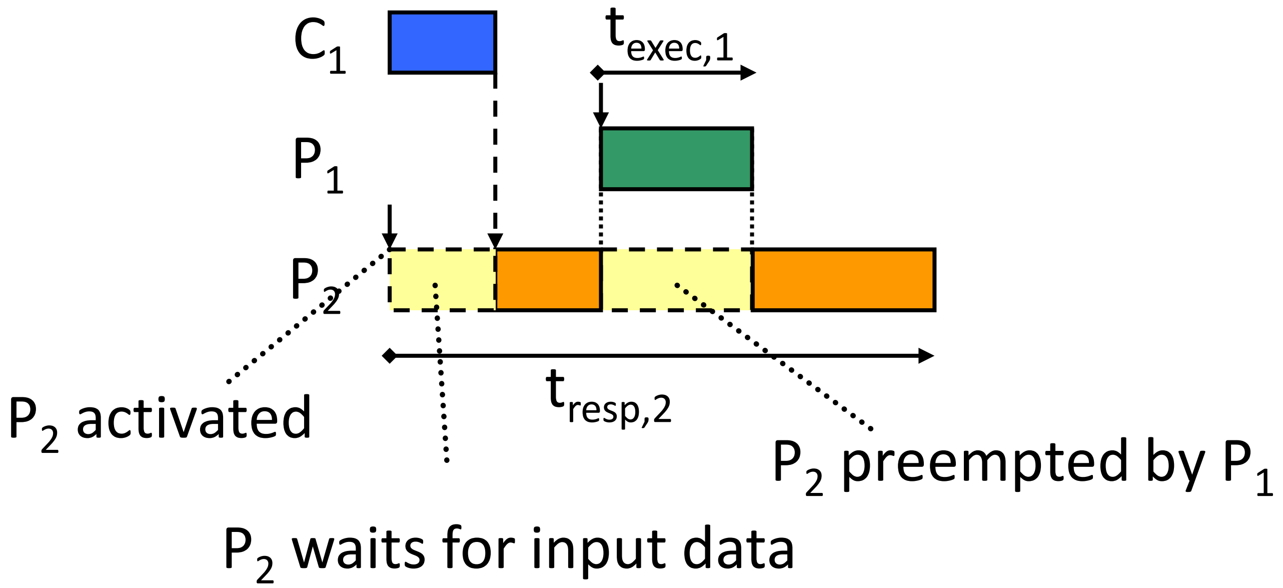
- **Worst-case**
  - E.g. highest resource usage/bus congestion a certain task experiences

By Response time analysis

- Time from task activation to end of execution
- $t_{resp} \geq t_{exec}$

By Gantt-Charts

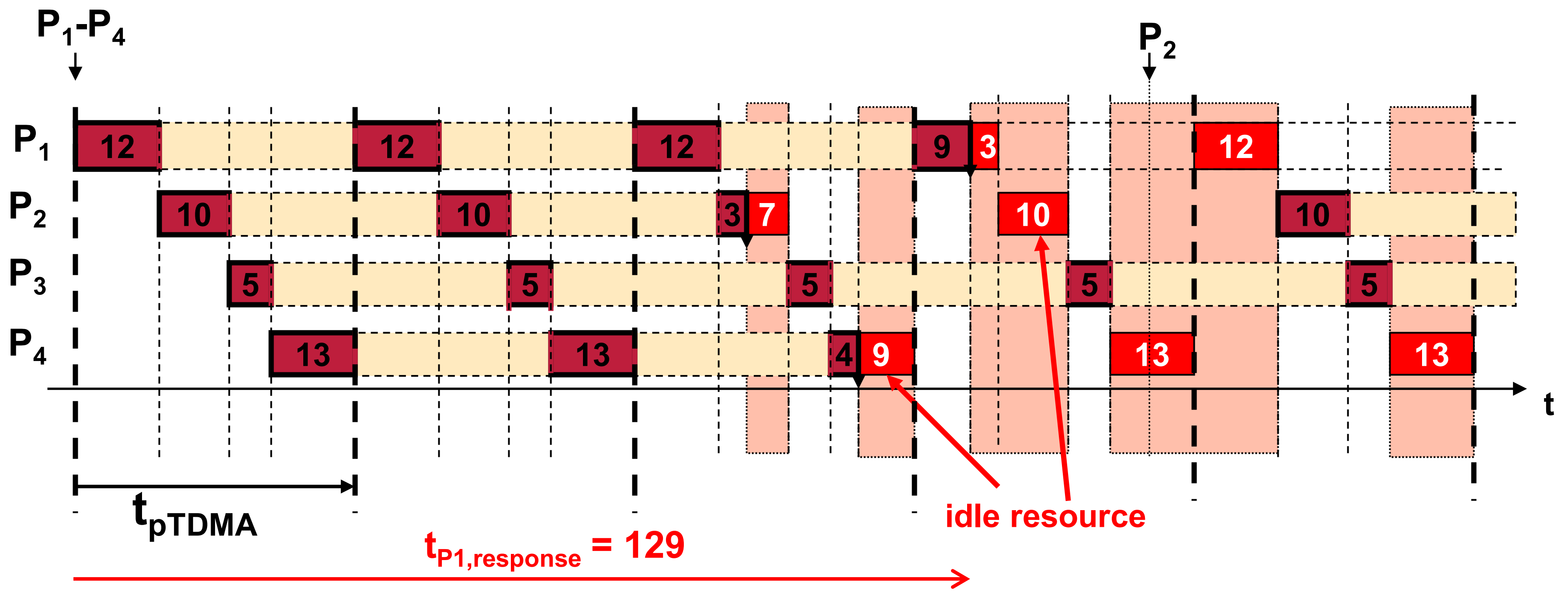
- Illustrate schedules
- Start and finish times
- Dependencies



TDMA Scheduling

Static time-driven scheduling

- Each task assigned to a specific time slot in reoccurring TDMA round
- Good analytical properties
- May lead to inefficient resource utilization



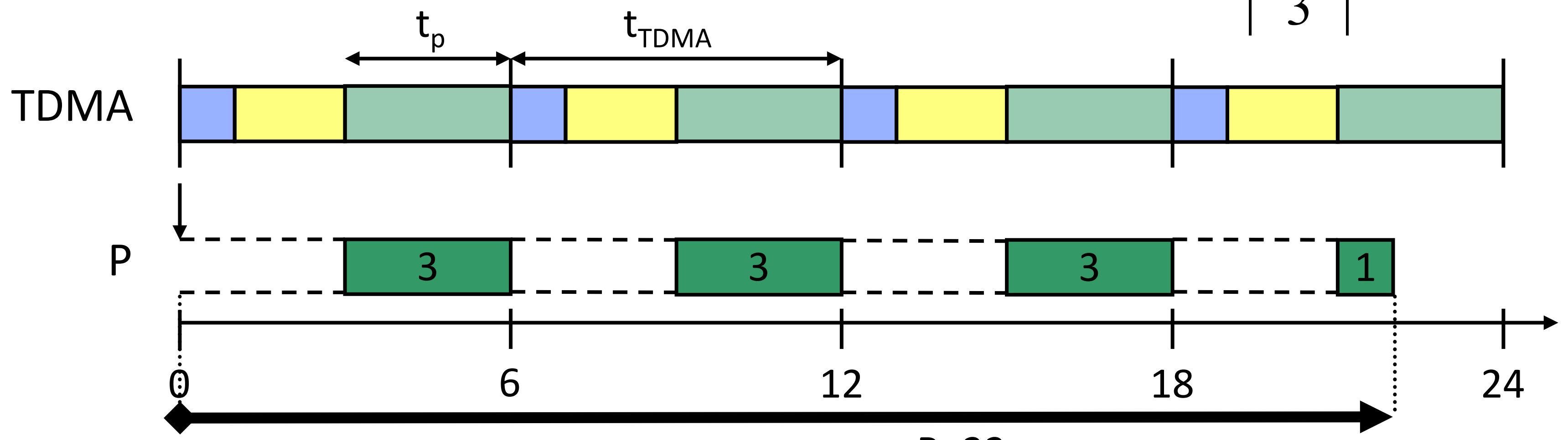
TDMA Analysis – Worst Case Response Time

$$R_i = \underbrace{C_i}_{\text{Core execution time}} + \underbrace{\underbrace{\left(t_{TDMA} - t_{P_i}\right)}_{\text{Preemption by other tasks}} \cdot \underbrace{\left\lceil \frac{C_i}{t_{P_i}} \right\rceil}_{\text{Max. number of TDMA rounds needed for one Execution}}}_{\text{Accumulated time of preemption until task finishes}}$$

Example:

C=10, t_P=3, t_{TDMA}=6

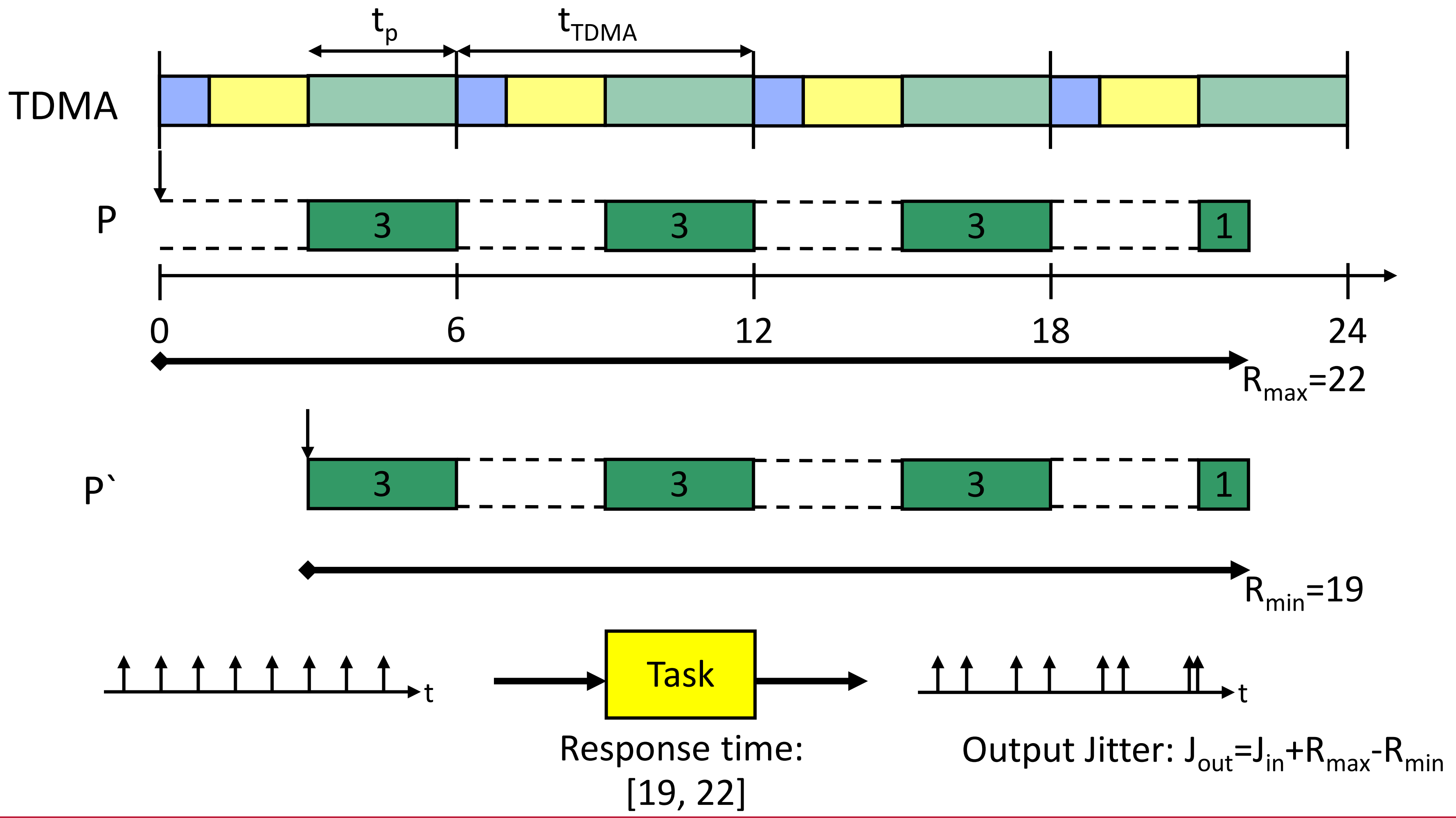
$$R = 10 + (6-3) \cdot \left\lceil \frac{10}{3} \right\rceil = 22$$



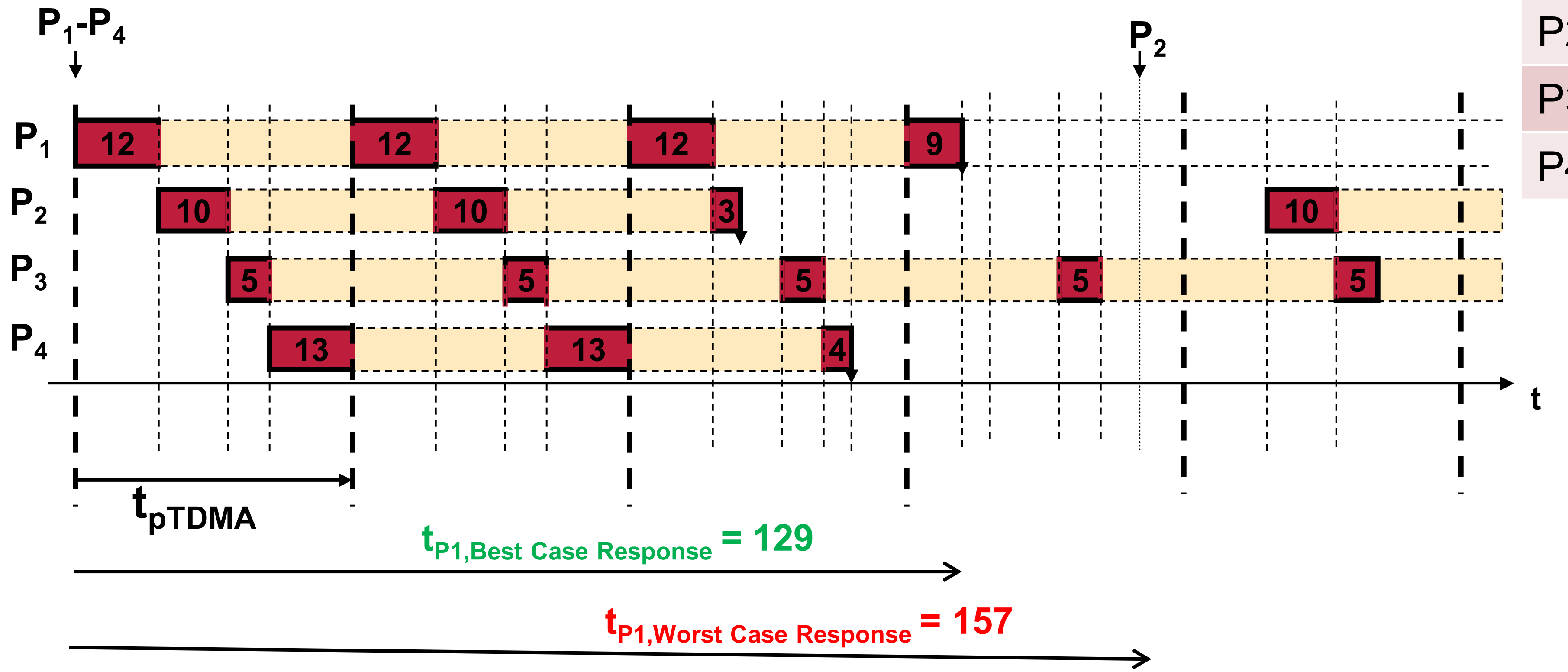
R=22

TDMA Output Jitter

- **Different response times cause output jitter**



TDMA Scheduling - Example



task	C	t_P	WCRT
P1	45	12	157
P2	23	10	113
P3	25	5	200
P4	30	13	111

TDMA – Practical Issues

Clock synchronization in distributed embedded systems

- All task have to be synchronized
  - Phase and frequency adjustment

Synchronization Methods

- Every task knows the entire TDMA schedule
- "Bus Sniffing"

After Synchronization

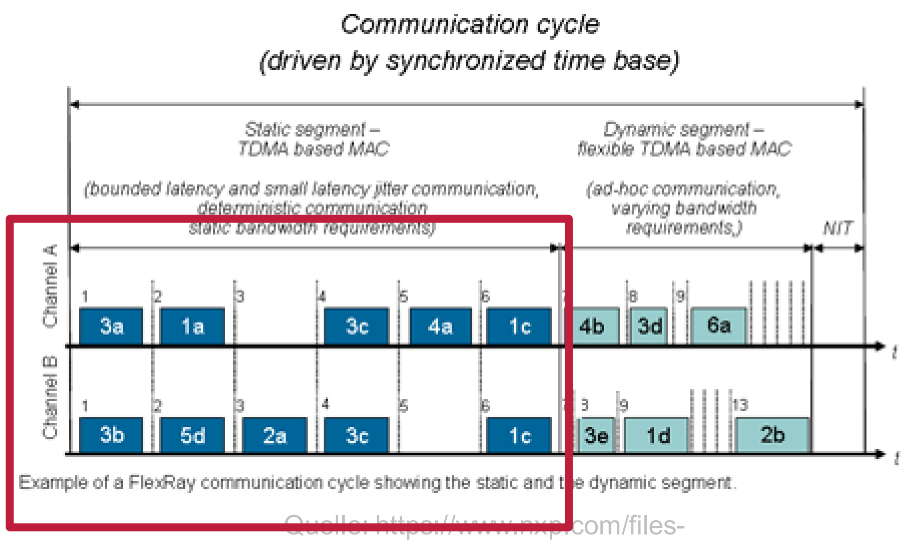
- No need for dedicated arbitration

Output event model

- May create output jitter due to differing Ri

FlexRay™

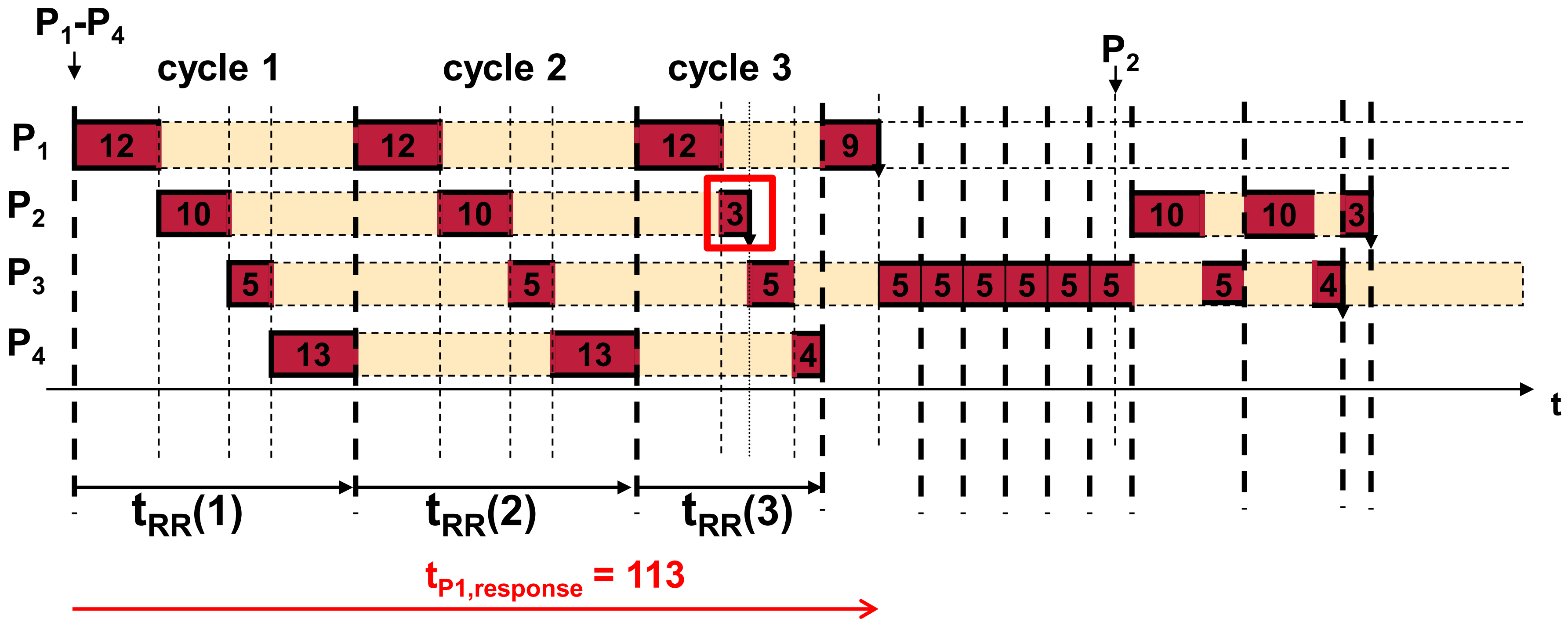
Quelle: https://de.wikipedia.org/wiki/FlexRay


Communication cycle
(driven by synchronized time base)
Static segment – TDMA based MAC
(bounded latency and small latency jitter communication, deterministic communication static bandwidth requirements)
Dynamic segment – flexible TDMA based MAC
(ad-hoc communication, varying bandwidth requirements,)
NIT
Channel A
Channel B
3a
1a
3c
4a
1c
4b
3d
6a
3b
5d
2a
3c
1c
3e
1d
2b
t
Example of a FlexRay communication cycle showing the static and the dynamic segment.

Quelle: https://www.nxp.com/files-static/abstract/overview_applications/FRWORKS.html

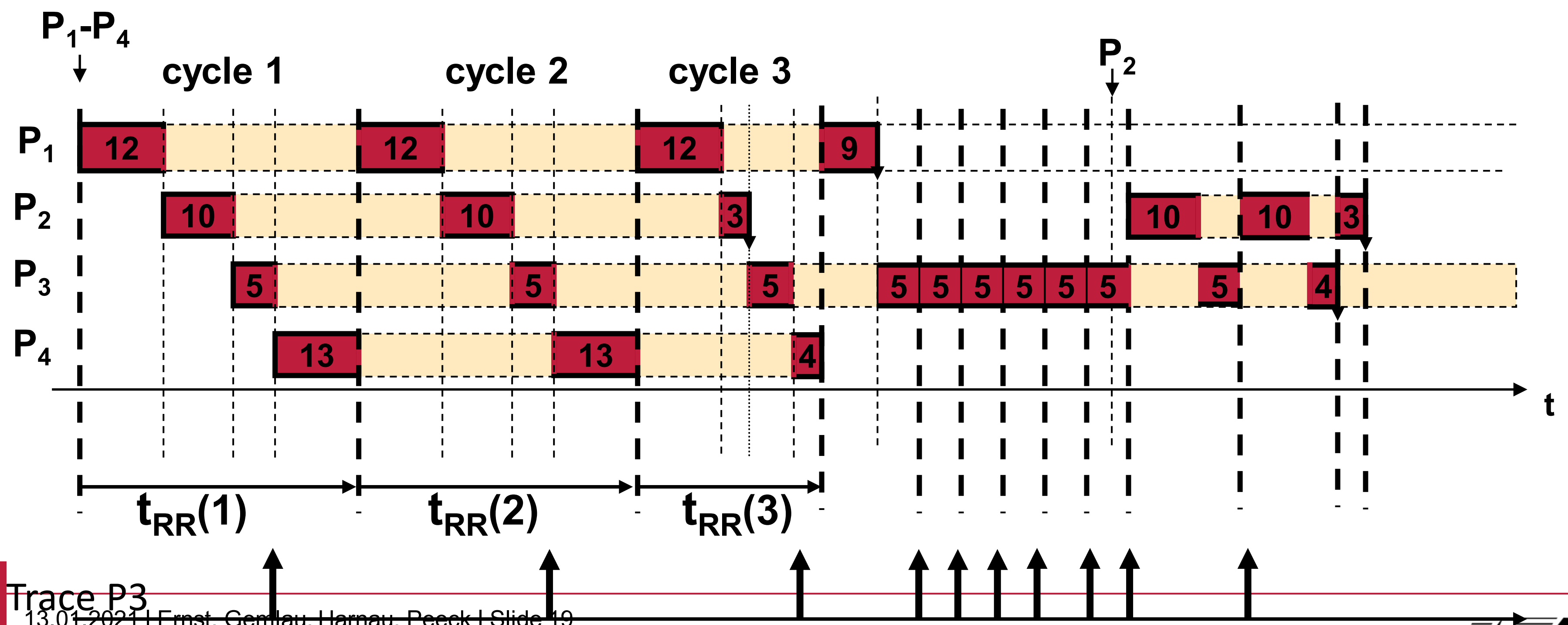
Round Robin Scheduling

- **Dynamic time-driven scheduling**
- Cyclic task execution
- Task release resources when execution finishes (no forced idle times)
  - Better resource utilization than TDMA



Round Robin - Practical Issues (1/2)

- **Dynamic behavior makes analysis more difficult than TDMA**
- **Output event model**
- May create output jitter and bursts
  - Example: During execution P_3 generates an event every 5 clock cycles



Round Robin - Practical Issues (2/2)

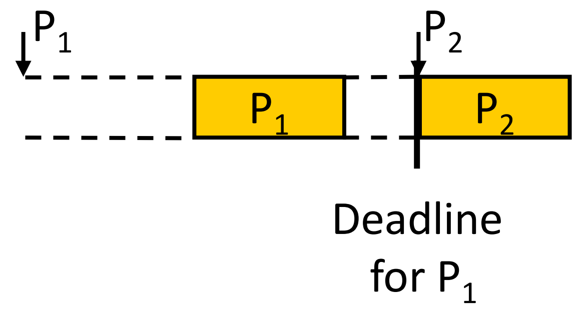
Clock synchronization in distributed embedded systems

- Synchronization requirements (like TDMA)
- Implementation more challenging than TDMA
  - Does a task want to send data?
  - Next task to schedule?
- Control Overhead

Teil 2

Rate Monotonic Scheduling (RMS)

- **Priority-driven scheduling approach**
- **Deadlines at the end of each task's period**
- **Fixed priorities**
  - The shorter the period, the higher the priority
- **Optimal with regard to single processor scheduling**
- **Commonly used**
- **Little cost**



Technische Universität Braunschweig

INSTITUTE OF COMPUTER AND NETWORK ENGINEERING

RMS - Analysis

Processor utilization: U(n)

A system of n independent RMS scheduled processes always meets its deadlines if (sufficient):

$$\sum_{i=1}^{n} \underbrace{\frac{C_i}{T_i}}_{\text{Utilization of process i}} = U(n) \leq n(2^{\frac{1}{n}} - 1) \qquad (1)$$

(Liu/Layland ’73)

Where:

C_i: Core execution time of process i

T_i: Period of process i

→ **Sufficient** condition, but not necessary

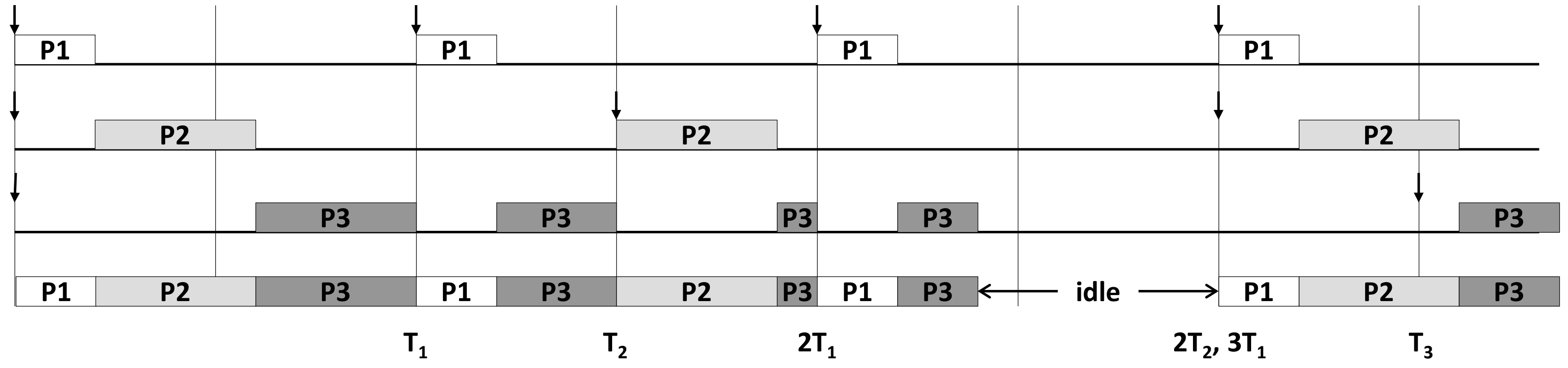
→If not met, no conclusion with respect to schedulability can be drawn

RMS – Example 1

$T_1=100$	$C_1=20$
$T_2=150$	$C_2=40$
$T_3=350$	$C_3=100$

$$\sum_{i=1}^{3}\frac{C_i}{T_i}=75{,}24\% \le 77{,}98\% = 3(2^{1/3}-1)$$

Equation (1) met, system schedulable



RMS – Example 2 (1/2)

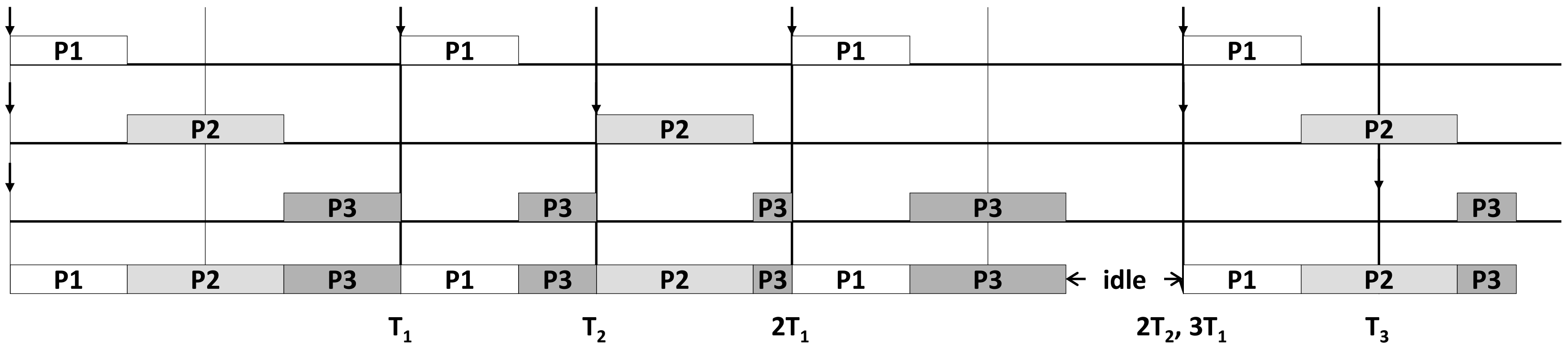
$T_1=100$ $C_1=30$

$T_2=150$ $C_2=40$

$T_3=350$ $C_3=100$

$$\sum_{i=1}^{3}\frac{C_i}{T_i}=85{,}24\%>77{,}98\%=3(2^{1/3}-1)$$

Equation (1) not met, schedule not guaranteed



- How to prove that all deadlines are still met in this case?

RMS - Workload

At each time t the accumulated requested workload is given by

$$W_n(t) = C_1\left\lceil \frac{t}{T_1} \right\rceil + C_2\left\lceil \frac{t}{T_2} \right\rceil + \ldots + C_n\left\lceil \frac{t}{T_n} \right\rceil = \sum_{i=1}^{n} C_i \left\lceil \frac{t}{T_i} \right\rceil$$

- Evaluate at $t=nT_i$ for every task i
- If the workload $W_n(t)$ at time t is less than or equal to t, sufficient resources are available

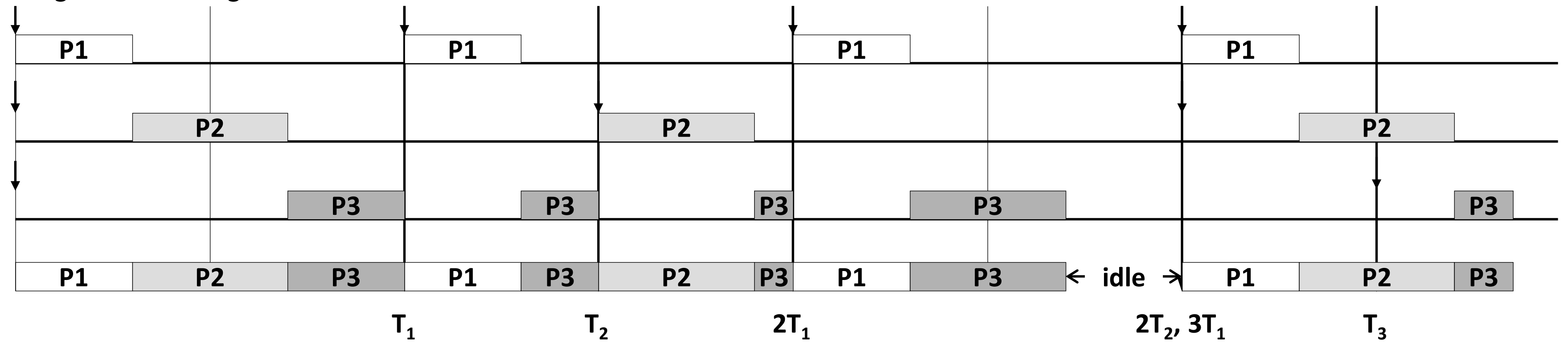
RMS – Example 2 (2/2)

$T_1=100$ $C_1=30$

$T_2=150$ $C_2=40$

$T_3=350$ $C_3=100$

$$\sum_{i=1}^{3}\frac{C_i}{T_i} = 85{,}24\% > 77{,}98\% = 3(2^{1/3}-1)$$



Workload for n=3:

$$W_3(t) = C_1\left\lceil \frac{t}{T_1}\right\rceil + C_2\left\lceil \frac{t}{T_2}\right\rceil + C_3\left\lceil \frac{t}{T_3}\right\rceil$$

(with t=350)

$$= 30\left\lceil \frac{350}{100}\right\rceil + 40\left\lceil \frac{350}{150}\right\rceil + 100\left\lceil \frac{350}{350}\right\rceil = 340 \le 350$$

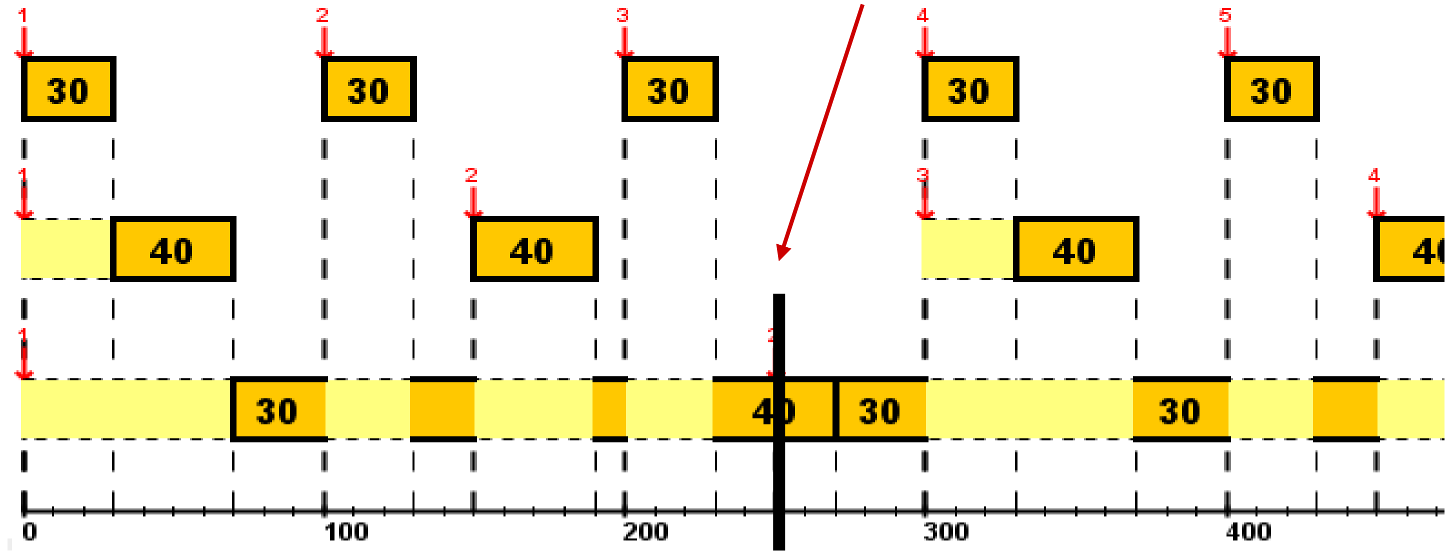
RMS – Example 3

$T_1=100$	$C_1=30$
$T_2=150$	$C_2=40$
$T_3=250$	$C_3=100$

$$\sum_{i=1}^{3}\frac{C_i}{T_i} = 96{,}67\% > 77{,}98\% = 3(2^{1/3}-1)$$

Equation (1) not met, schedule not guaranteed

DEADLINE MISS!



$$W_3(t) = C_1\left\lceil\frac{t}{T_1}\right\rceil + C_2\left\lceil\frac{t}{T_2}\right\rceil + C_3\left\lceil\frac{t}{T_3}\right\rceil = 30\left\lceil\frac{250}{100}\right\rceil + 40\left\lceil\frac{250}{150}\right\rceil + 100\left\lceil\frac{250}{250}\right\rceil = 270 \geq 250$$

(with t=250)

RMS - Schedulability

Requirement: Worst case response time has to be smaller than deadline

How to trigger worst case behavior?

Critical instant = Situation in which a certain task experiences its worst case response time

- In case of RMS: A task is activated together with all higher priority tasks

Given a critical instant, if a task in a RMS scheduled system meets its first deadline then it will meet all deadlines

Recursive approach:

$$R_i^n = C_i + \underbrace{\sum_{j \in hp(i)} C_j \cdot \left\lceil \frac{R_i^{n-1}}{T_j} \right\rceil}_{\text{Preemption of task i by all higher priority tasks j in time window given by } R_i^{n-1}} \le D_i, \quad R_i^0 = 0$$

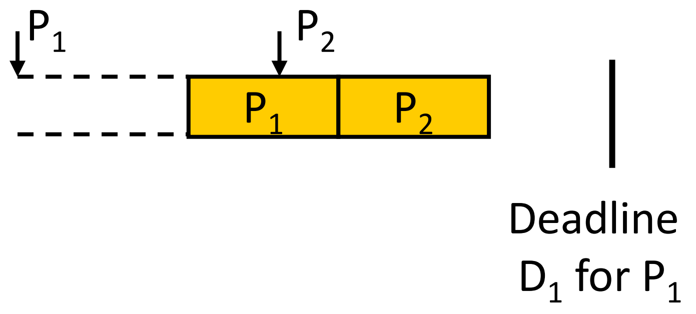
Core execution time of task i

Recur until fixed point R_i reached or (due to monotocity) $R_i^n > D_i$

Analysis with Arbitrary Deadlines

Arbitrary deadlines

- Now additional task activations during task execution/preemption possible



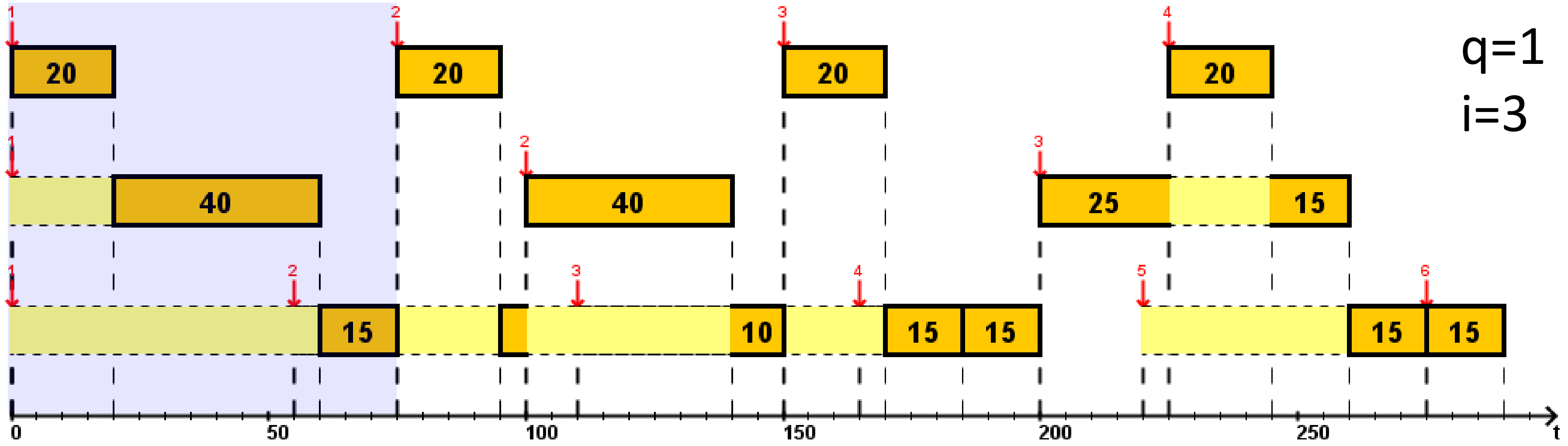
Goal: Find worst case response time and check Ri ≤ Di

Solution: Windowing technique

```
iterate over q=1,… {
    $w_i(q) = q \cdot C_i + \sum_{j \in hp(i)} C_j \cdot \left\lceil \frac{w_i(q)}{T_j} \right\rceil$        } Workload for q activations of task i (Time to finish q activations)
    $R_i(q) = w_i(q) - (q-1)T_i$        } Response time of the q'th activation of task i
} until ( $w_i(q) \le q \cdot T_i$ )        } Iterate until q'th activation finishes before the q+1'th
```

$$R_{i,WorstCase} = \max_q \left\{ R_i(q) \right\}$$

Static Priority Preemptive Scheduling (1/4)



q=1
i=3

$$w_i(1) = C_i + \sum_{j \in hp(i)} C_j \cdot \left\lceil \frac{w_i(1)}{T_j} \right\rceil = 75$$

$$R_i\,(1) = w_i(1) - (1-1)T_i = 75 - 0 = 75$$

$w_i(1) = 75 > 55 = 1 \cdot T_i$ →CONTINUE

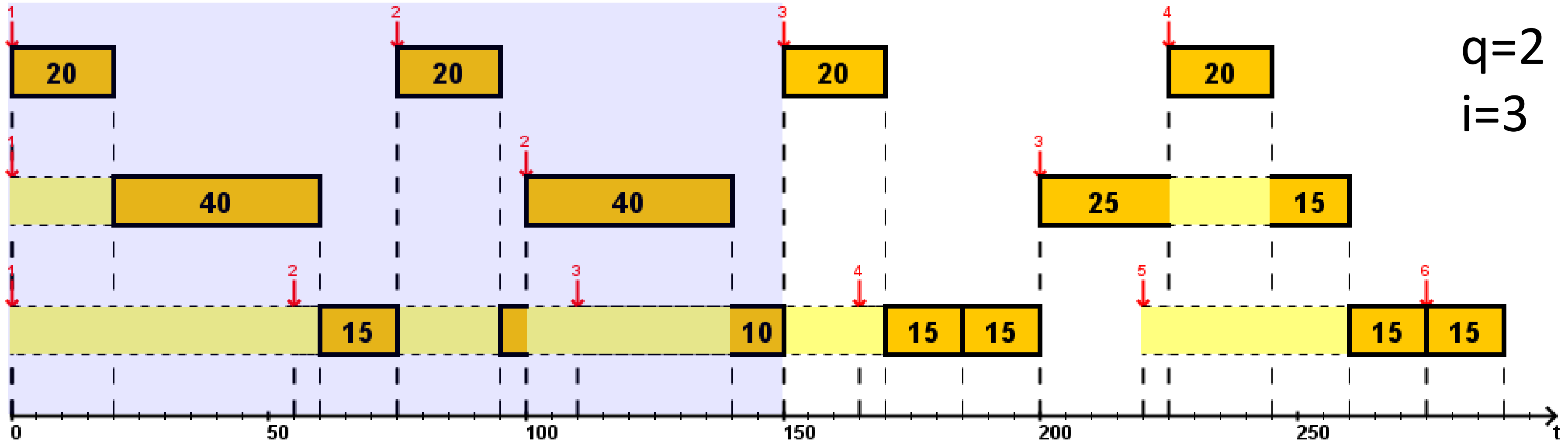
$$\begin{cases} w_3^0(1) = 0 \\ w_3^1(1) = 15 + 20\left\lceil \frac{0}{75} \right\rceil + 40\left\lceil \frac{0}{100} \right\rceil = 15 \\ w_3^2(1) = 15 + 20\left\lceil \frac{15}{75} \right\rceil + 40\left\lceil \frac{15}{100} \right\rceil = 75 \\ w_3^3(1) = 15 + 20\left\lceil \frac{75}{75} \right\rceil + 40\left\lceil \frac{75}{100} \right\rceil = 75 \end{cases}$$

Fixed point reached

Technische Universität Braunschweig

INSTITUTE OF COMPUTER AND NETWORK ENGINEERING

Static Priority Preemptive Scheduling (2/4)



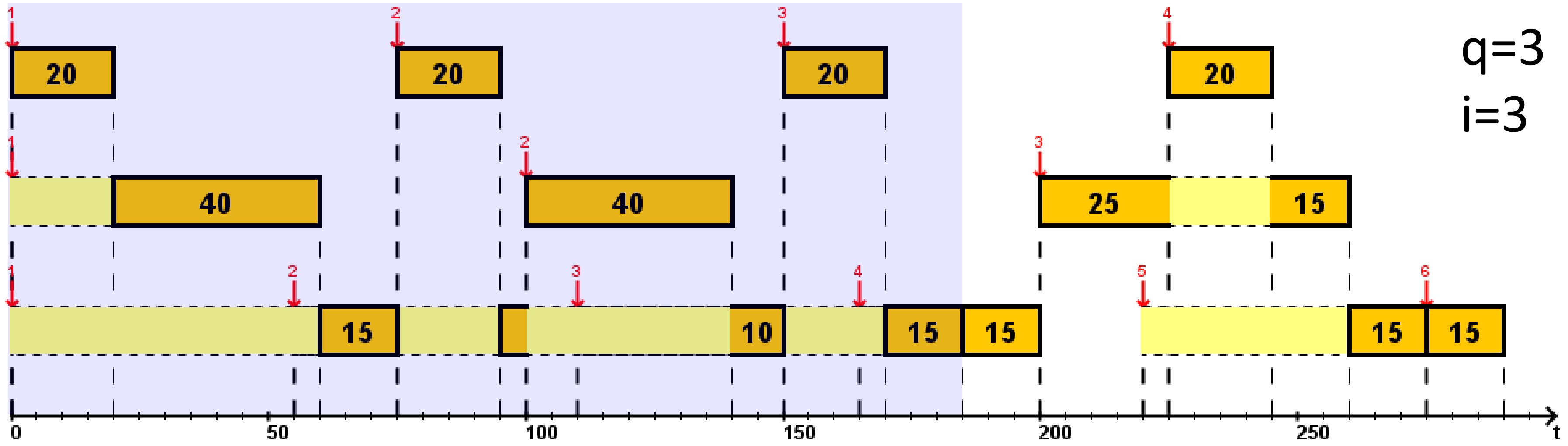
q=2
i=3

$$w_i(2) = 2C_i + \sum_{j \in hp(i)} C_j \cdot \left\lceil \frac{w_i(2)}{T_j} \right\rceil = 150$$

$$R_i\,(2) = w_i(2) - (2-1)T_i = 150 - 55 = 95$$

$$w_i(2) = 150 > 110 = 2 \cdot T_i$$ →CONTINUE

Static Priority Preemptive Scheduling (3/4)



q=3

i=3

$$w_i(3) = 3C_i + \sum_{j \in hp(i)} C_j \cdot \left\lceil \frac{w_i(3)}{T_j} \right\rceil = 185$$

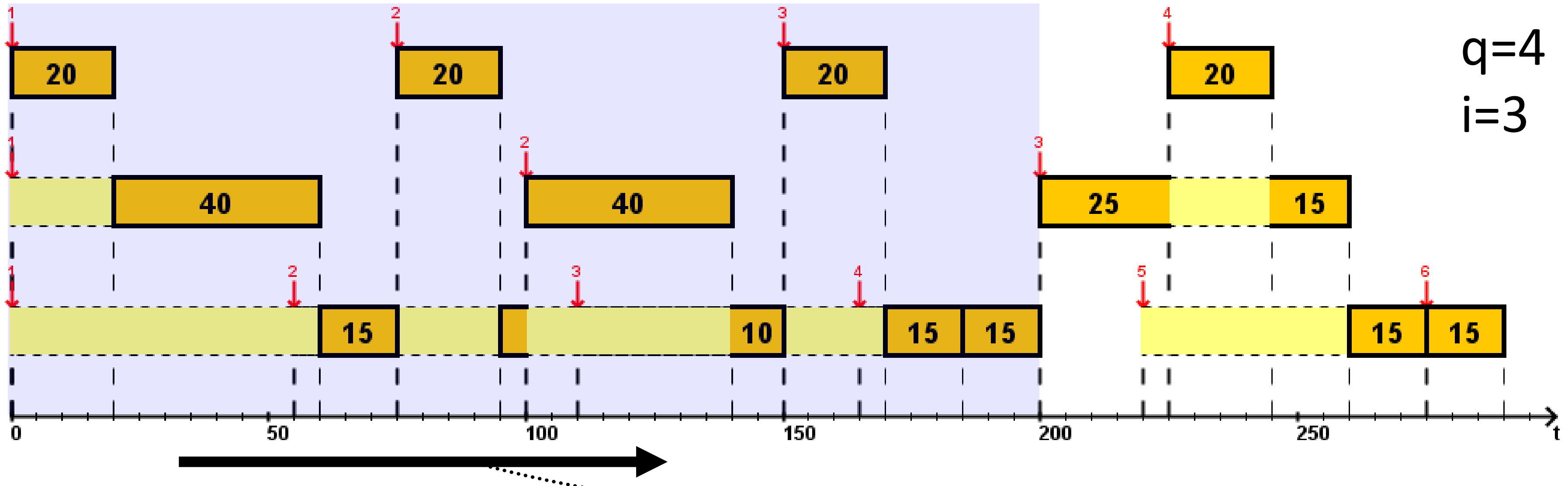
$$R_i\ (3) = w_i(3) - (3-1)T_i = 185 - 2 \cdot 55 = 75$$

$w_i(3) = 185 > 165 = 3 \cdot T_i$ →**CONTINUE**

Technische Universität Braunschweig

INSTITUTE OF COMPUTER AND NETWORK ENGINEERING

Static Priority Preemptive Scheduling (4/4)



q=4

i=3

$$w_i(4) = 4C_i + \sum_{j \in hp(i)} C_j \cdot \left\lceil \frac{w_i(4)}{T_j} \right\rceil = 200$$

$$R_i\ (4) = w_i(4) - (4-1)T_i = 200 - 3 \cdot 55 = 35$$

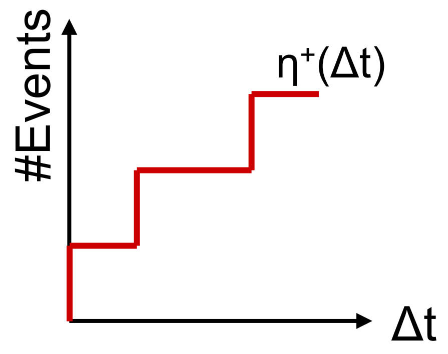
$w_i(4) = 200 < 220 = 4 \cdot T_i$ **STOP!** $R_{i,WorstCase} = \max\{75, 95, 35\} = 95$

Generalization

- Arbitrary priority assignment
- Arbitrary event models

$$R_i^n = C_i + \sum_{j=1}^{i-1} C_j \cdot \eta_j^+\left(R_i^{n-1}\right) \leq D_i$$

$$R_i^0 = 0$$



Reminder $\eta^+(\Delta t)$:
Max. number of events that can occur in time interval Δt.

Generalizing the Windowing Technique

Arbitrary event models and arbitrary deadlines

$$w_i(q) = q \cdot C_i + \sum_{j \in hp(i)} C_j \cdot \eta_j^+\left(w_i(q)\right)$$

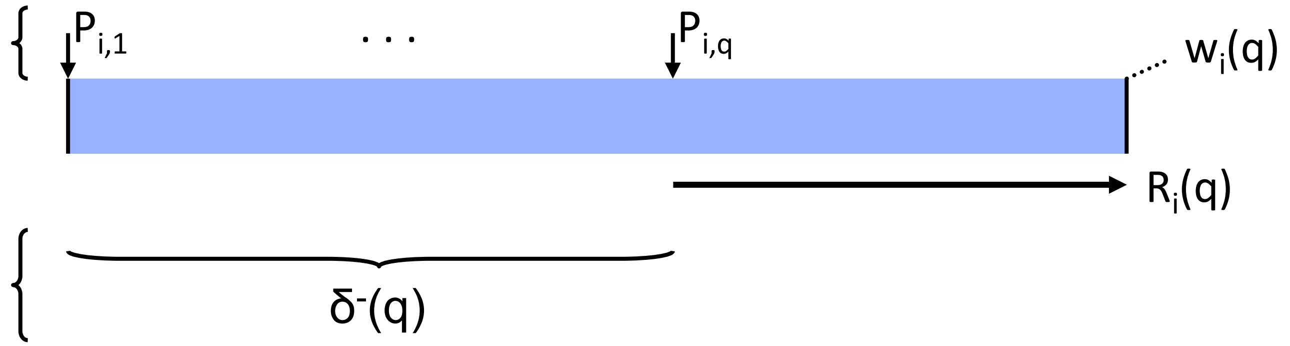
$$R_i\,(q) = w_i(q) - \delta_i^-(q)$$

$$w_i(q) \le \delta_i^-(q+1)$$

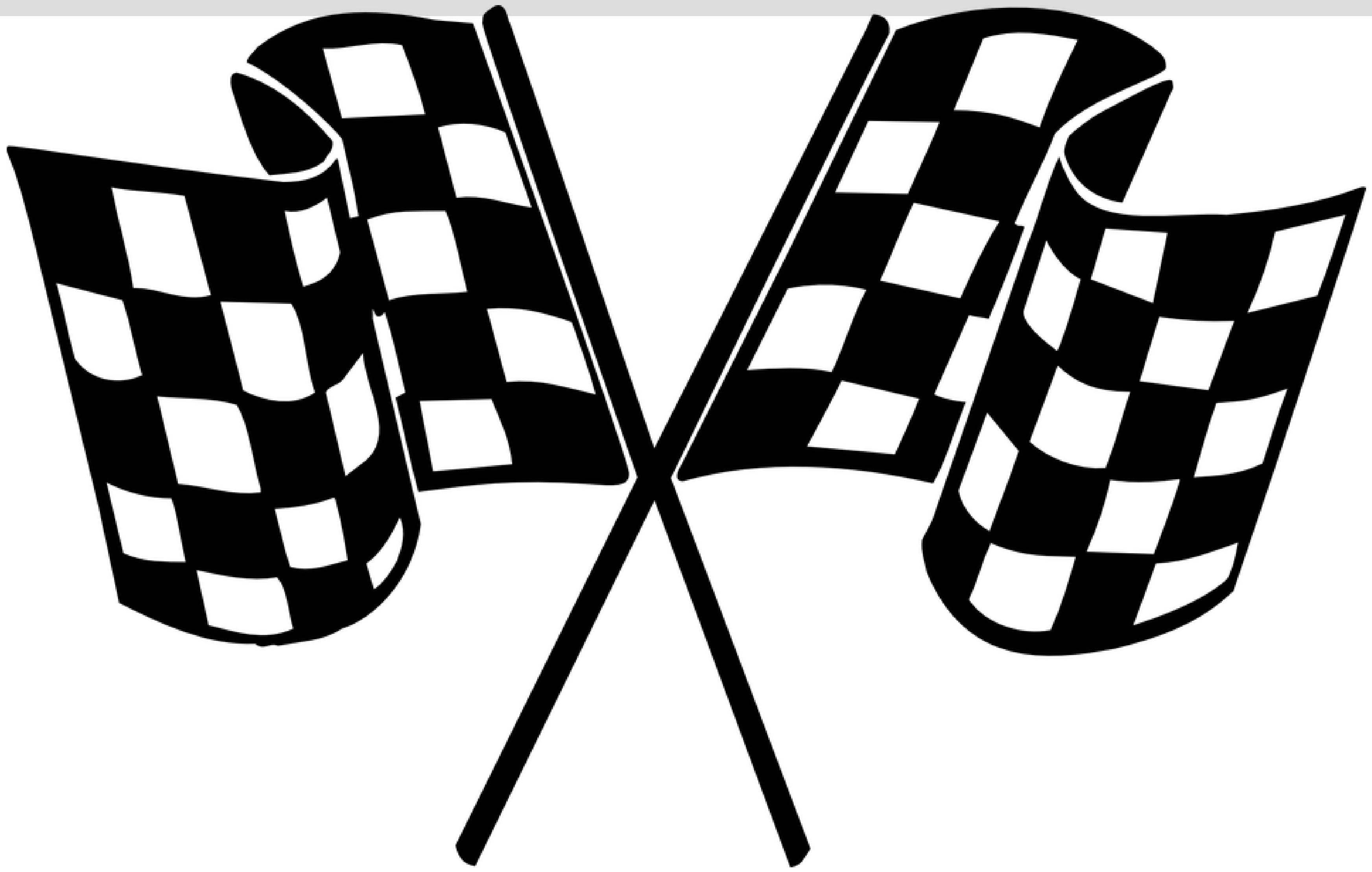
$\delta^-(n)$ is the inverse of $\eta^+(\Delta t)$: Smallest interval in which any n events may occur

Concrete event models: Each P_i is known

Arbitrary event models: Only bounds ($\delta^-(q)$) for P_i are known



END



Technische Universität Braunschweig

INSTITUTE OF COMPUTER AND NETWORK ENGINEERING