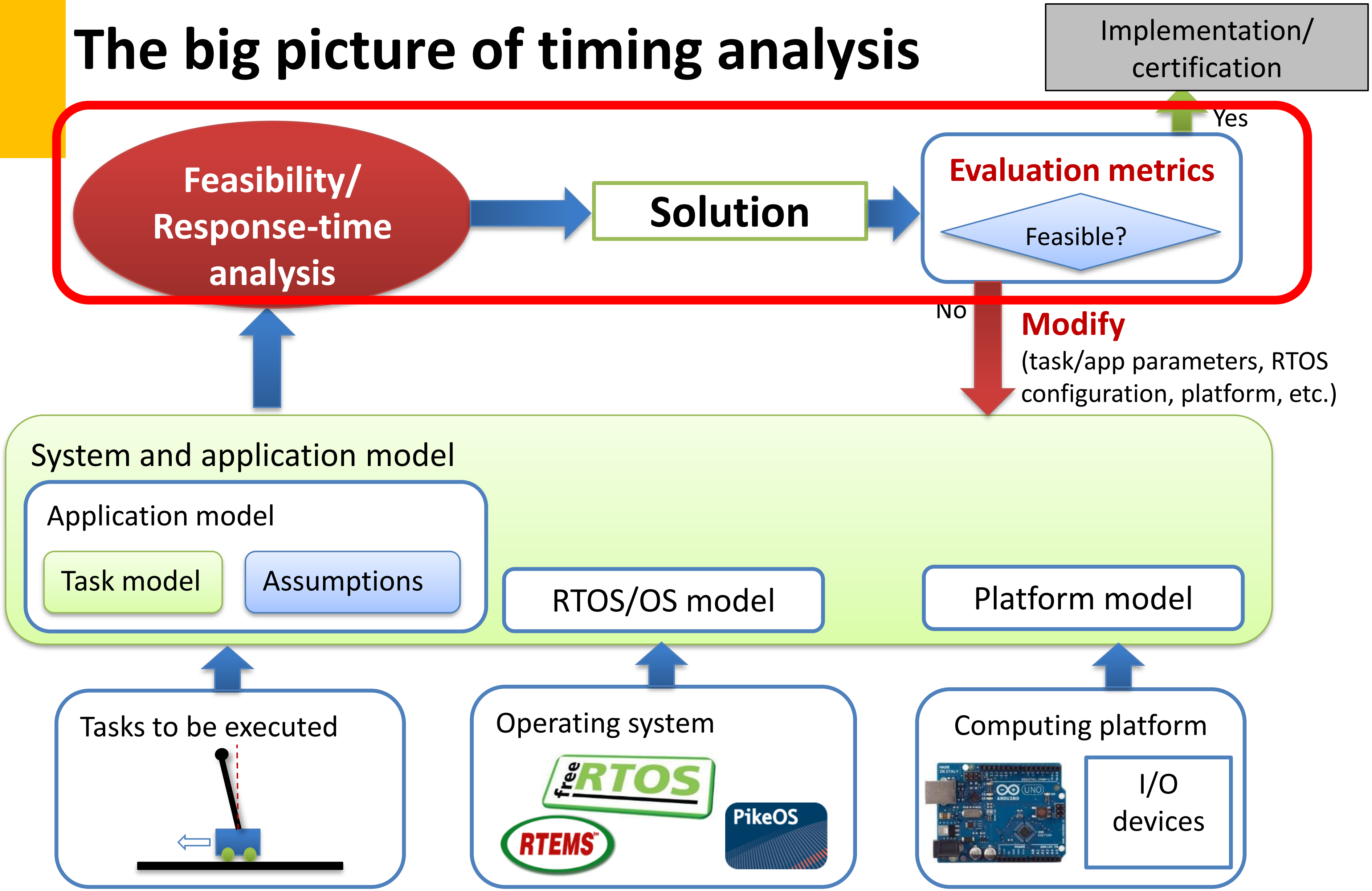
2IMN20 - Real-Time Systems

Schedulability Tests for Periodic (and Sporadic) Tasks

Geoffrey Nelissen

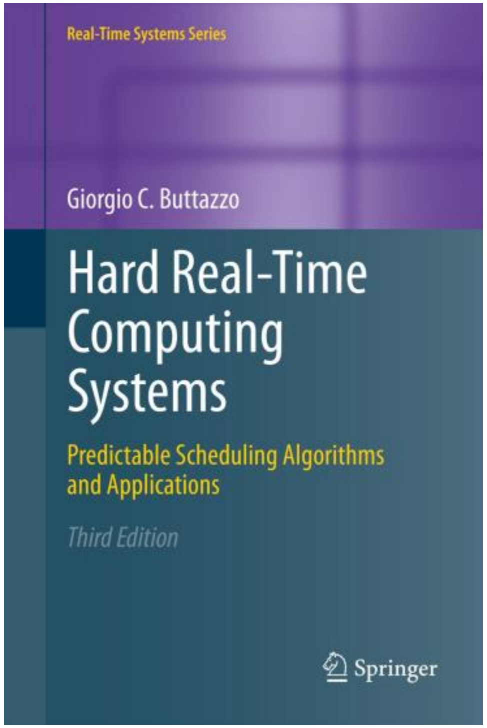
2023-2024

The big picture of timing analysis



Online scheduling of periodic tasks

Buttazzo's book, chapter 4



Disclaimer:
Many slides were provided by Dr. Mitra Nasri
Some slides have been taken from [Giorgio Buttazzo]

Agenda

- **Necessary vs sufficient schedulability tests**
- **EDF schedulability test**
- **Priority assignment for task-level fixed-priority scheduling**
  - Rate monotonic (RM)
  - Deadline monotonic (DM)
  - Audsley's Optimal priority assignment algorithm (OPA)
- **RM schedulability tests**
  - Liu and Layland's test [1973]
  - Hyperbolic bound [2000]
  - A utilization-based test for harmonic tasks

Recall: notations

We consider a computing system that has to execute a set τ of $\boldsymbol{n}$ periodic real-time tasks:

$$\tau = \{\tau_1, \tau_2, \tau_3, \dots, \tau_n\}$$

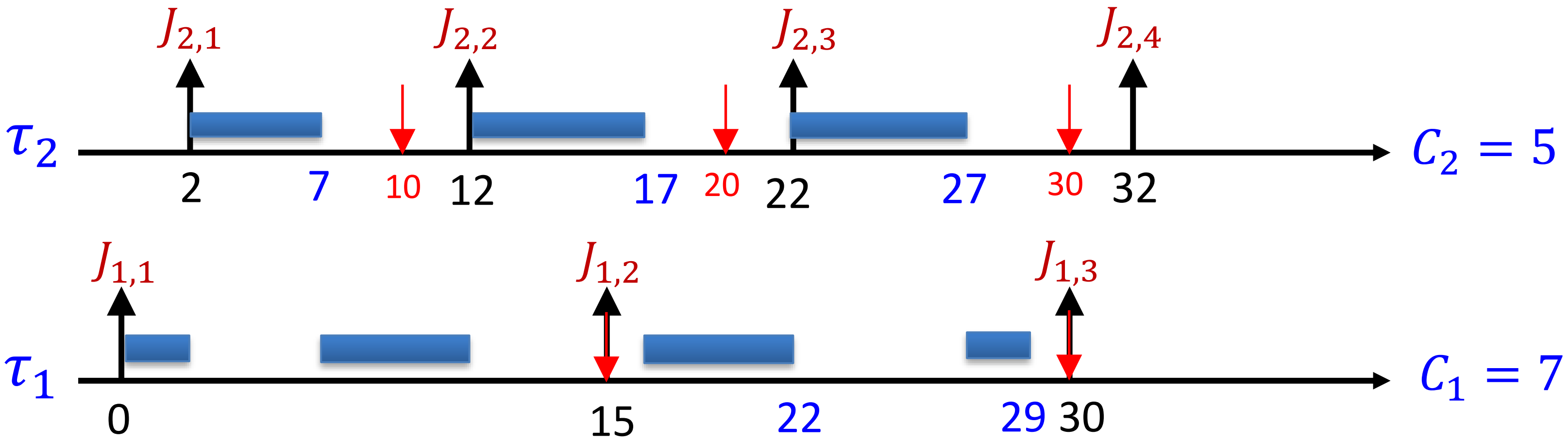
T_i = period

C_i = worst-case execution time (WCET)

D_i = relative deadline

ϕ_i = offset (or phase)

σ_i = release jitter



Definition: utilization

- Each task uses the processor for a fraction of time: $$U_i = \frac{C_i}{T_i}$$

- Hence the total **processor utilization** is: $$U = \sum_{i=1}^{n} \frac{C_i}{T_i}$$

- U is a measure of the **processor load**

Assumptions in this lecture

- We assume

  A1. Tasks are fully preemptive

  A2. Context switch, preemption, and scheduling overheads are zero

  A3. Tasks are independent:

  - no precedence relations
  - no resource constraints
  - No shared resource accesses
  - …

  A4. No self-suspension

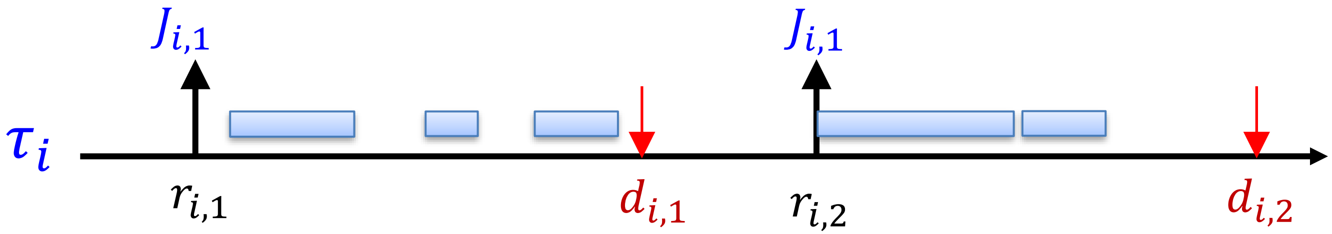
  - no blocking on I/O operations
  - …

Feasibility of a task set

A task set τ is **feasible** **if and only if** there always exists a schedule in which all tasks meet their timing constraints.

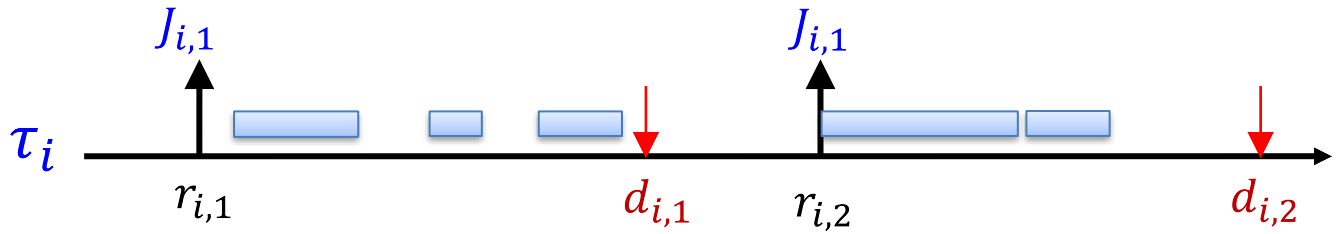
Feasibility of a task set

A task set τ is **feasible if and only if** there always exists a schedule in which each task $\tau_i \in \tau$ can execute for C_i units of time within <u>every interval</u> $[r_{i,k}, d_{i,k})$ for all $k \in \mathbb{N}$.



Feasible schedule

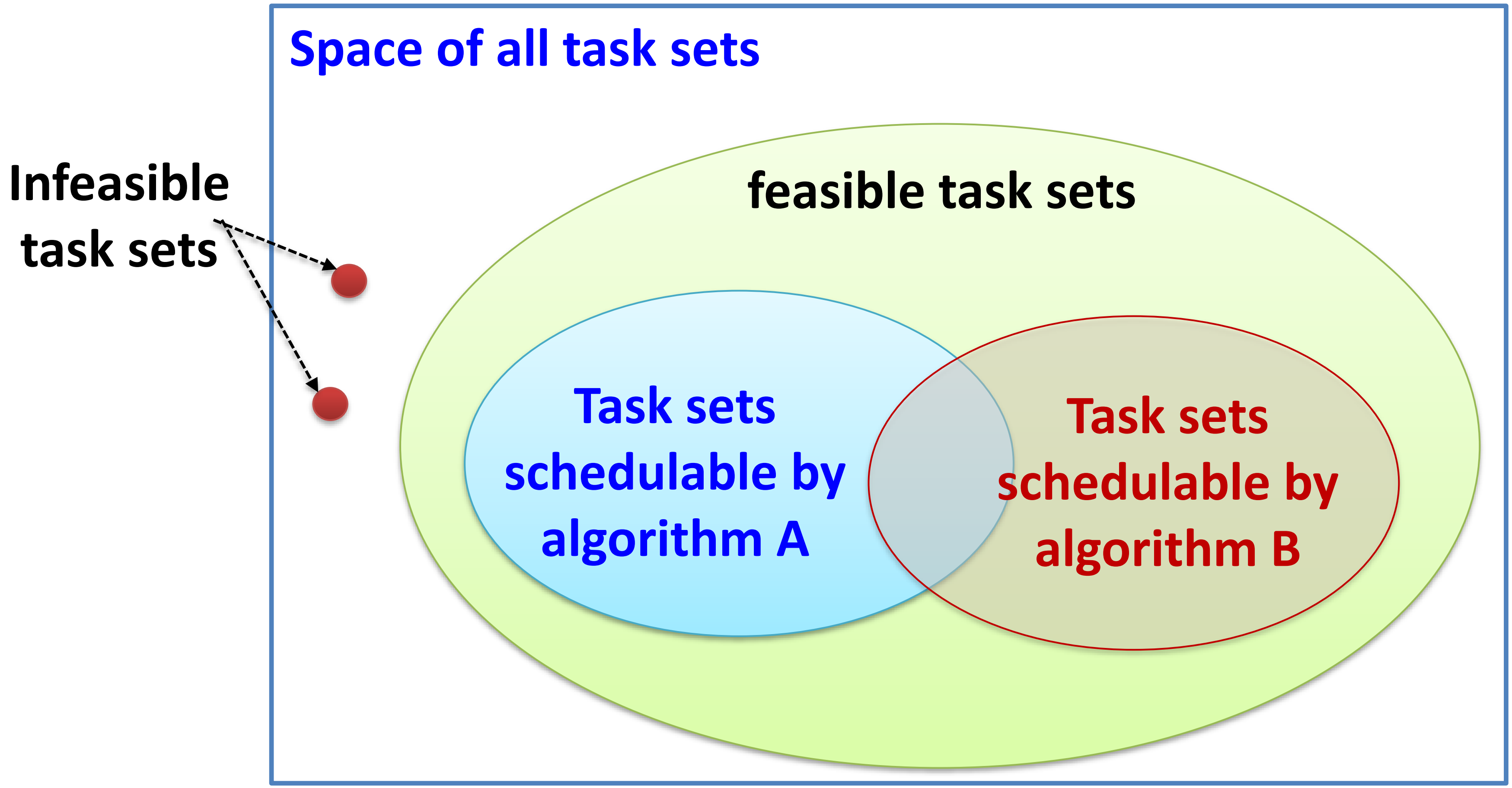
A schedule of task set τ is **feasible** **if and only if** it respects all timing constraints of τ.



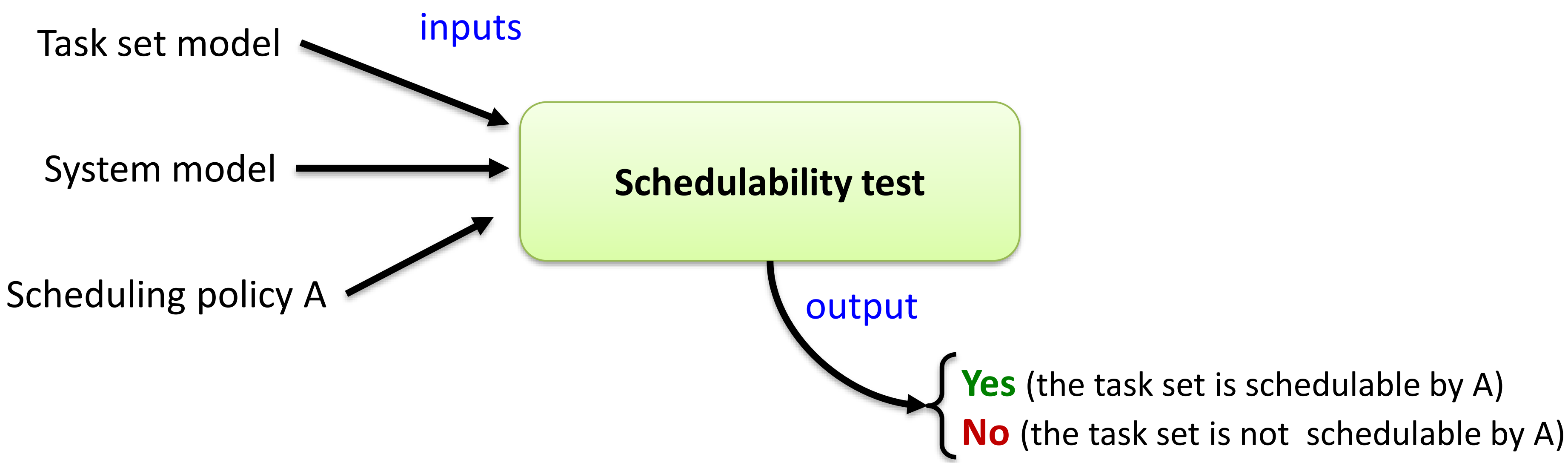
Schedulability of a task set with algorithm A

A task set τ is **schedulable with algorithm A** **if and only if** algorithm A always generates a feasible schedule for τ.

Reminder: Feasibility vs. schedulability



Schedulability test



Necessary vs sufficient schedulability tests

Necessary schedulability test for a scheduling algorithm A:
If a task set is schedulable by the scheduling algorithm A
then it certainly passes the test

Sufficient schedulability test for a scheduling algorithm A:
if a task set that passes the test, then **it is certainly schedulable** by the scheduling algorithm **A**

Exact schedulability test for a scheduling algorithm A:
A task set that **passes** the test **if and only if** it is **schedulable** by **A**

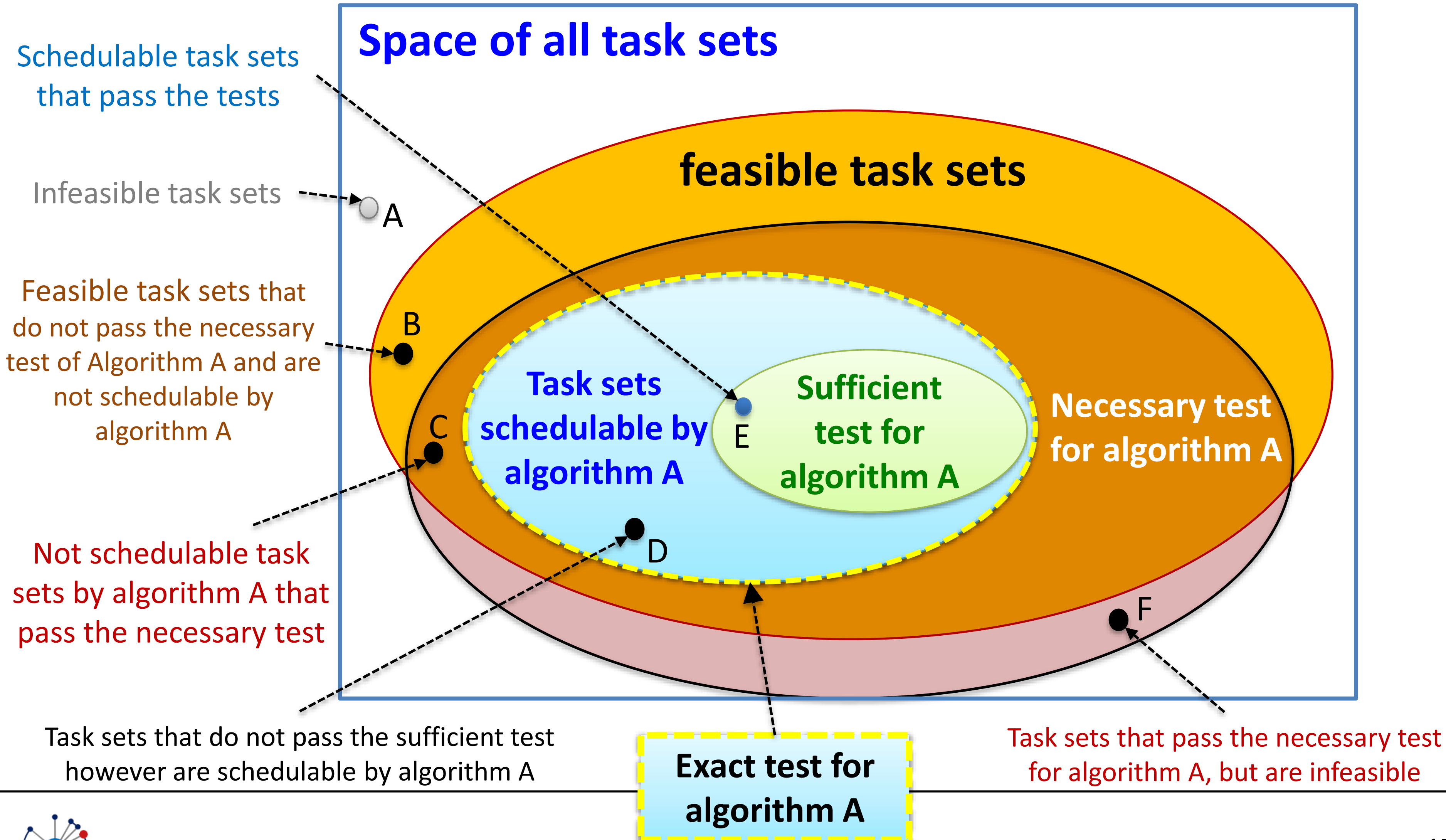
Necessary vs sufficient schedulability tests

A **schedulability test** for a scheduling algorithm **A** is a **necessary test** if any task set that **does NOT pass** the test is certainly **not schedulable** by the scheduling algorithm **A**

A **schedulability test** for a scheduling algorithm **A** is a **sufficient test** if any task set that **passes** the test is certainly **schedulable** by the scheduling algorithm **A**

A **schedulability test** for a scheduling algorithm **A** is an **exact test** if any task set that **passes** the test is certainly **schedulable** and any task set that **does not pass** the test is certainly **not schedulable** by the scheduling algorithm **A**

Necessary vs sufficient schedulability tests



Utilization-based schedulability tests

A **utilization-based schedulability test** for an **algorithm A** is a test that checks whether a function f_A on the utilizations of the tasks in the task set holds.

A necessary utilization-based schedulability test for uniprocessor scheduling

A necessary test for uniprocessor

$$U = \sum_{i=1}^{n} \frac{C_i}{T_i}$$

Can we find a feasible schedule for a task set whose utilization is larger than 1? ($U > 1$)

A: yes

B: depends on the scheduling policy

C: depends on the task set

D: no

A necessary test for uniprocessor

$$U = \sum_{i=1}^{n} \frac{C_i}{T_i}$$

Can we find a feasible schedule for a task set whose utilization is larger than 1? ($U > 1$)

A: yes

B: depends on the scheduling policy

C: depends on the task set

D: no

If $U > 1$, then the amount of work to be done per unit of time **is larger than the unit of time itself!**

A utilization-based schedulability test for EDF

EDF

Schedulability test

This **test** is both **necessary and sufficient for EDF** (for implicit deadline periodic or sporadic tasks executed on a single core platform)

- **Liu and Layland 1973**

  A preemptive task set τ with $\forall i, D_i = T_i$, is schedulable by EDF **if and only if:**

$$\sum_{i=1}^{n} \frac{C_i}{T_i} \leq 1$$

Any feasible periodic or sporadic task set with $\forall i, D_i = T_i$ and $U \leq 1$ can be scheduled by EDF

How is such a task called?

EDF Optimality

EDF is **optimal** among all algorithms:

If there exists a feasible schedule for Γ, then EDF will generate a feasible schedule.

$\Updownarrow$

If Γ is not schedulable by EDF, then it cannot be scheduled by any algorithm.

Task-level fixed priority scheduling

Fixed-priority scheduling

Two steps to design a timing-predictable periodic task system under fixed-priority scheduling:

1. Assign priorities to each task based on its timing constraints.
    1. Rate monotonic
    2. Deadline monotonic
    3. Optimal priority assignment algorithm (OPA)

2. Verify the schedulability of the task set using a schedulability test.

In this lecture we limit ourselves to presenting tests for RM. We will cover all priority assignment policies in the next lecture.

Priority assignment for fixed-priority scheduling

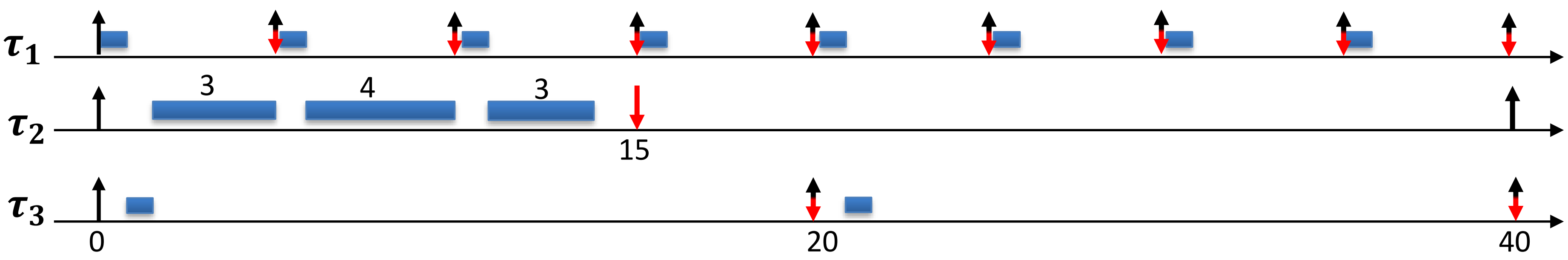
Rate monotonic

- Assign priorities monotonically with the activation frequency, a.k.a., *rate* ($\sim 1 / T$) such that a task with a smaller period gets a higher priority
- Example: $T = 10 \Rightarrow rate = \frac{1}{10}$

τ_i	C_i	T_i	D_i	U_i
τ_1	1	5	5	0.2
τ_2	10	40	15	0.25
τ_3	1	20	20	0.05

Priority ordering?

P1 > P3 > P2



Priority assignment for fixed-priority scheduling

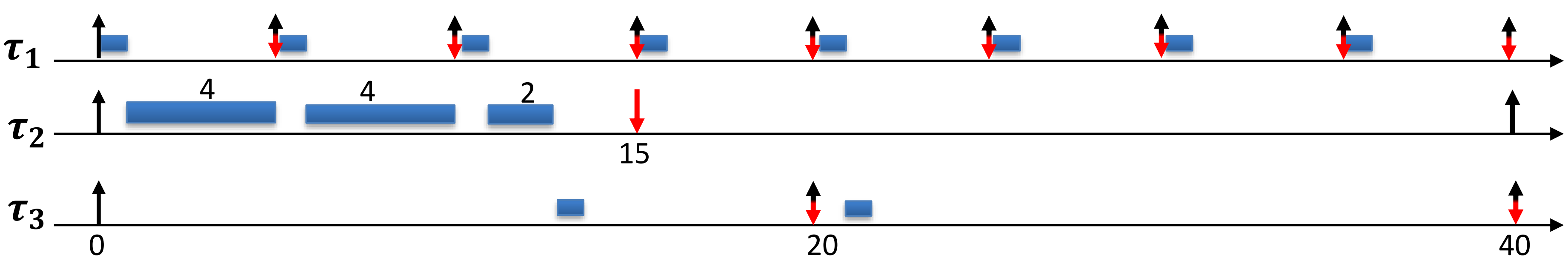
Deadline monotonic

- Assign priorities monotonically with the relative deadline of the task, ($\sim 1 / D$) such that a task with a smaller relative deadline gets a higher priority

τ_i	C_i	T_i	D_i	U_i
τ_1	1	5	5	0.2
τ_2	10	40	15	0.25
τ_3	1	20	20	0.05

Priority ordering?

P1 > P2 > P3



Audsley's optimal priority assignment algorithm (OPA)

for (each priority level k, lowest first)

{

for (each unassigned task τ_i)

{

}

return unschedulable;

}

return schedulable (return priorities);

Audsley's optimal priority assignment algorithm (OPA)

for (each priority level k, lowest first)

{

for (each unassigned task τ_i)

{

if (τ_i is schedulable at priority k according to an **exact schedulability test S** with all unassigned tasks assumed to have higher priorities) **then**

{

assign τ_i to priority k;

break and **continue** the outer loop;

}

}

return unschedulable;

}

return schedulable (return priorities);

Is FP an optimal scheduling policy with RM, DM or Audsley's priority assignment?

What do you think?

The limitation of fixed-priority scheduling

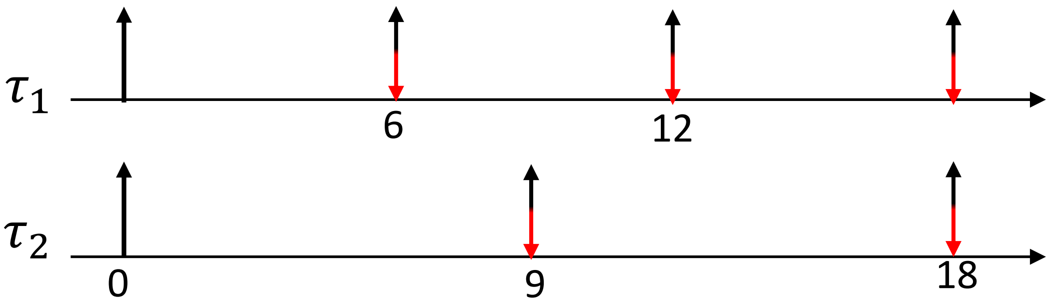
Fixed-priority scheduling with **RM** priorities is **not** an **optimal** scheduling policy (in the sense of feasibility)

Neither is it with DM or Audsley's priority assignment

RM-priority assignment is not optimal

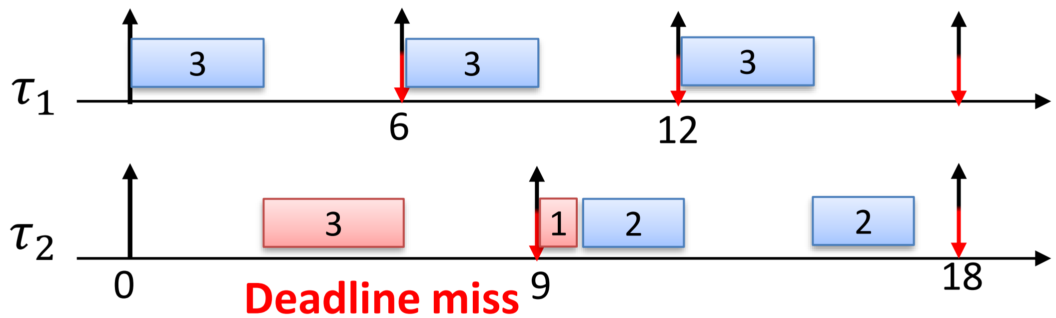
Build a feasible task set with $U \leq 1$ that is not schedulable by RM (rate monotonic)

Hint:
What execution times will lead to U<1 and will result in a deadline miss for one of the tasks?



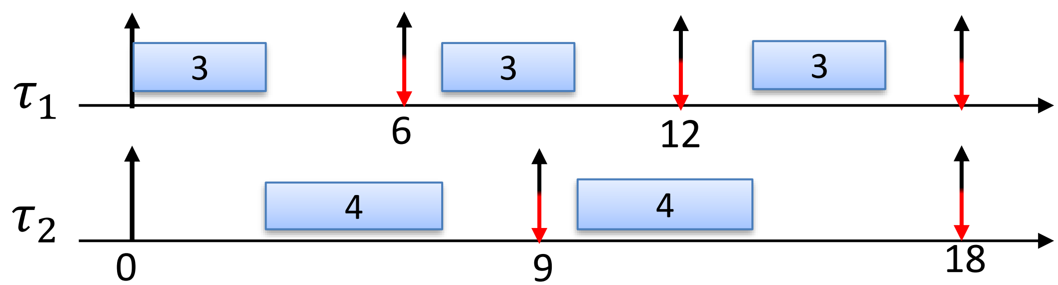
RM-priority assignment is not optimal

Build a feasible task set with $U \leq 1$ that is not schedulable by RM (rate monotonic)



τ_i	C_i	T_i	D_i	U_i
τ_1	3	6	6	0.5
τ_2	4	9	9	0.44

$$U = \sum_{i=1}^{n} U_i = 0.5 + 0.44 = 0.94$$



A feasible schedule

Fixed-priority scheduling with **RM** priorities is **not** an **optimal** scheduling policy (in the sense of feasibility)

However, if $\forall i, D_i = T_i$ then

the **rate-monotonic priority assignment** is

an **optimal priority assignment** among all *other fixed priority assignments*

Rate Monotonic is optimal

RM is **optimal** among all fixed priority algorithms (if $D_i = T_i$):

If there exists a fixed priority assignment which leads to a feasible schedule for Γ, then the RM assignment is feasible for Γ.

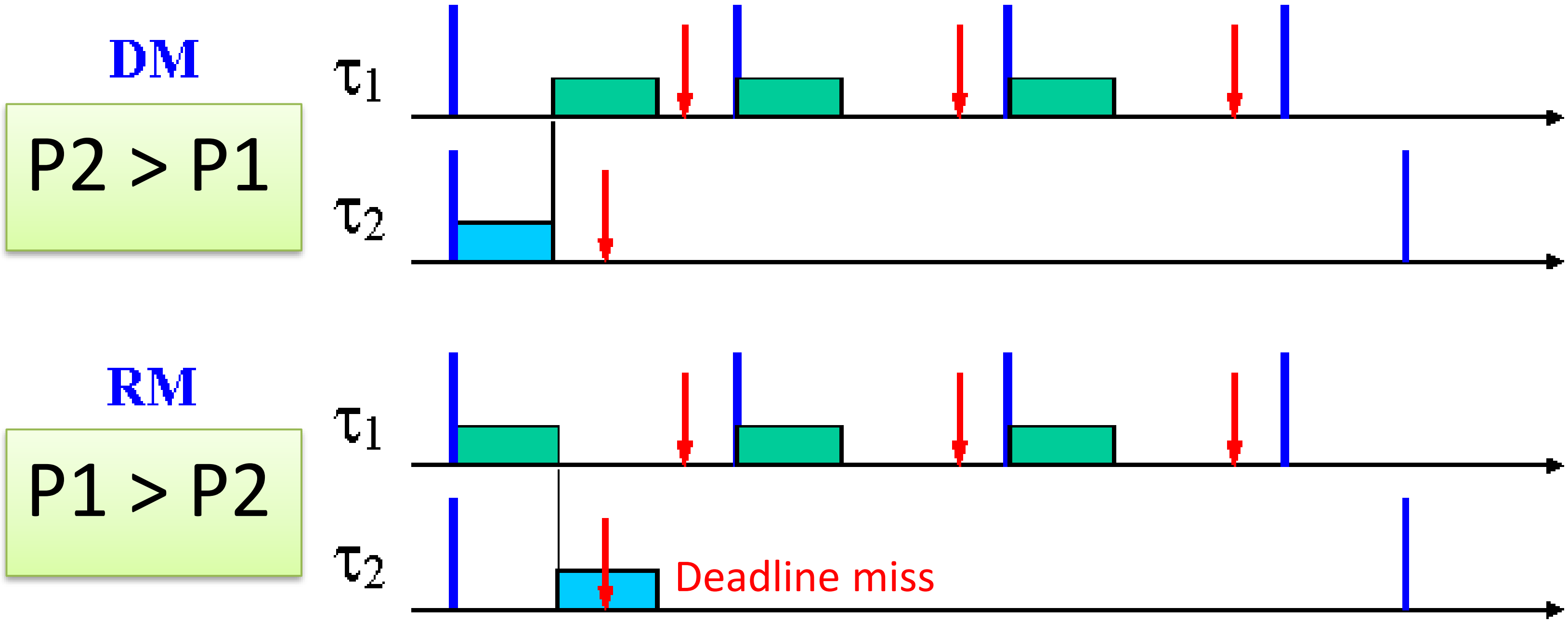
$\Updownarrow$

If Γ is not schedulable by RM, then it cannot be scheduled by any fixed priority assignment.

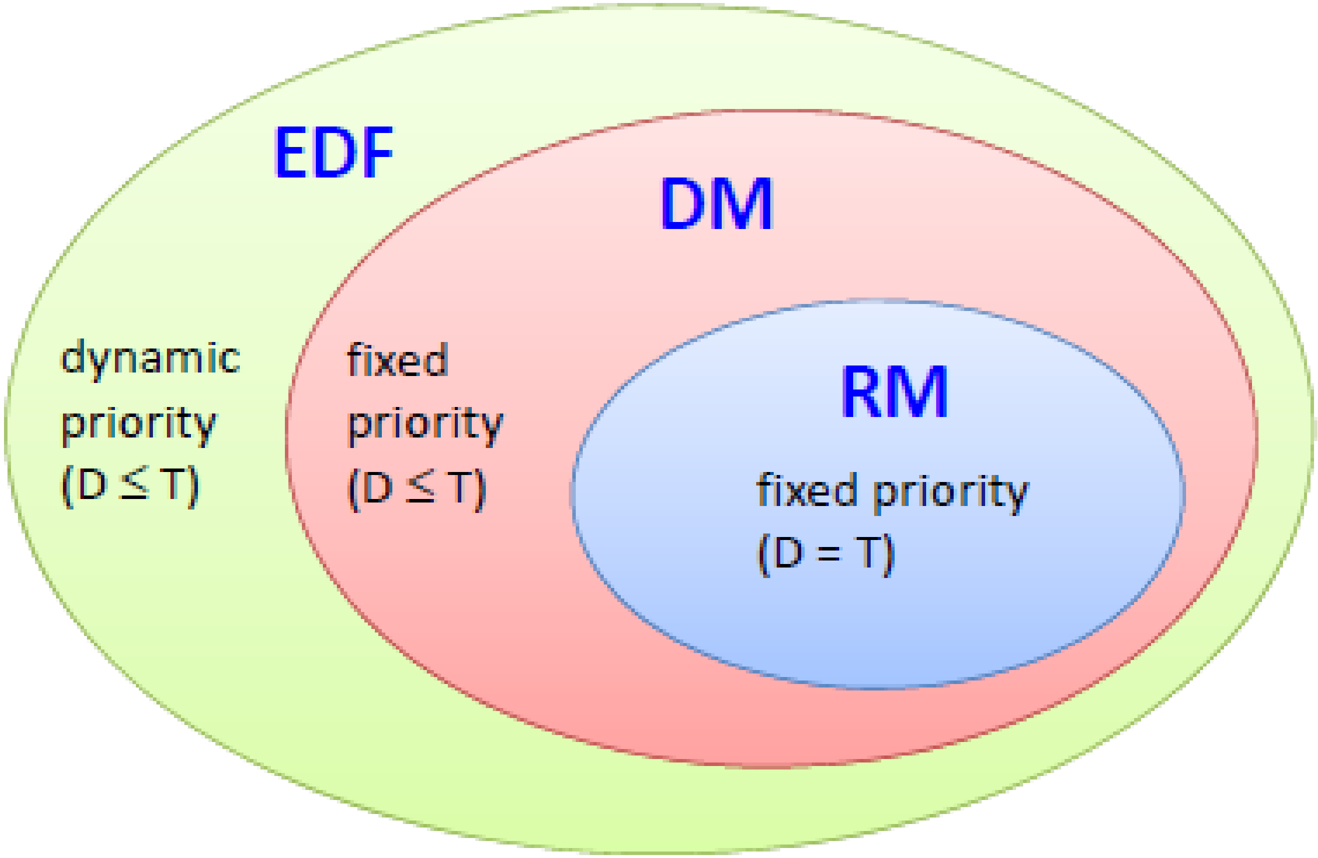
How is such a task called?

Deadline Monotonic is optimal

If $D_i \le T_i$ then the **optimal** priority assignment is given by **Deadline Monotonic (DM)**:



Optimality



EDF
DM
RM
dynamic priority (D ≤ T)
fixed priority (D ≤ T)
fixed priority (D = T)

Utilization-based schedulability tests for RM

Building a first utilization-based test for RM

Find a utilization threshold (i.e, a bound) such that **ANY task set** with **utilization lower than that bound** is **CERTAINLY schedulable by RM**

The simplest utilization-based test

For a given task set, check whether or not

$$\boldsymbol{U \leq U_{th}}$$

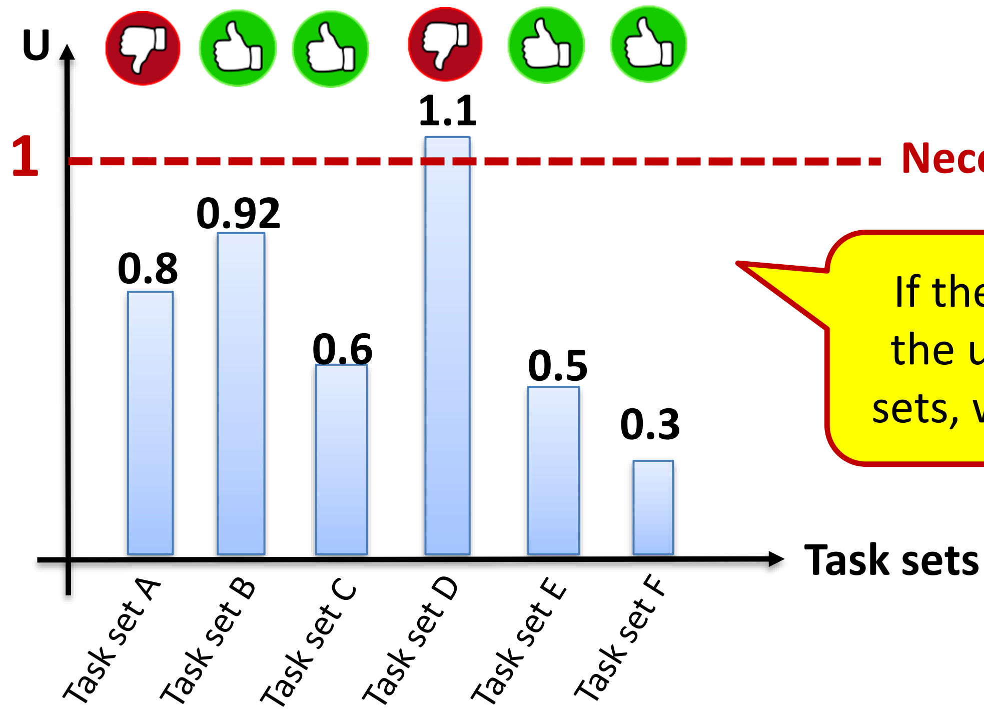
where U_{th} is the largest utilization such that any task set with $\boldsymbol{U \leq U_{th}}$ is **always schedulable by RM.**

The simplest utilization-based test

For a given task set, check whether or not

$$U \leq U_{th}$$

where U_{th} is the largest utilization such that any task set with $U \leq U_{th}$ is **always schedulable by RM.**



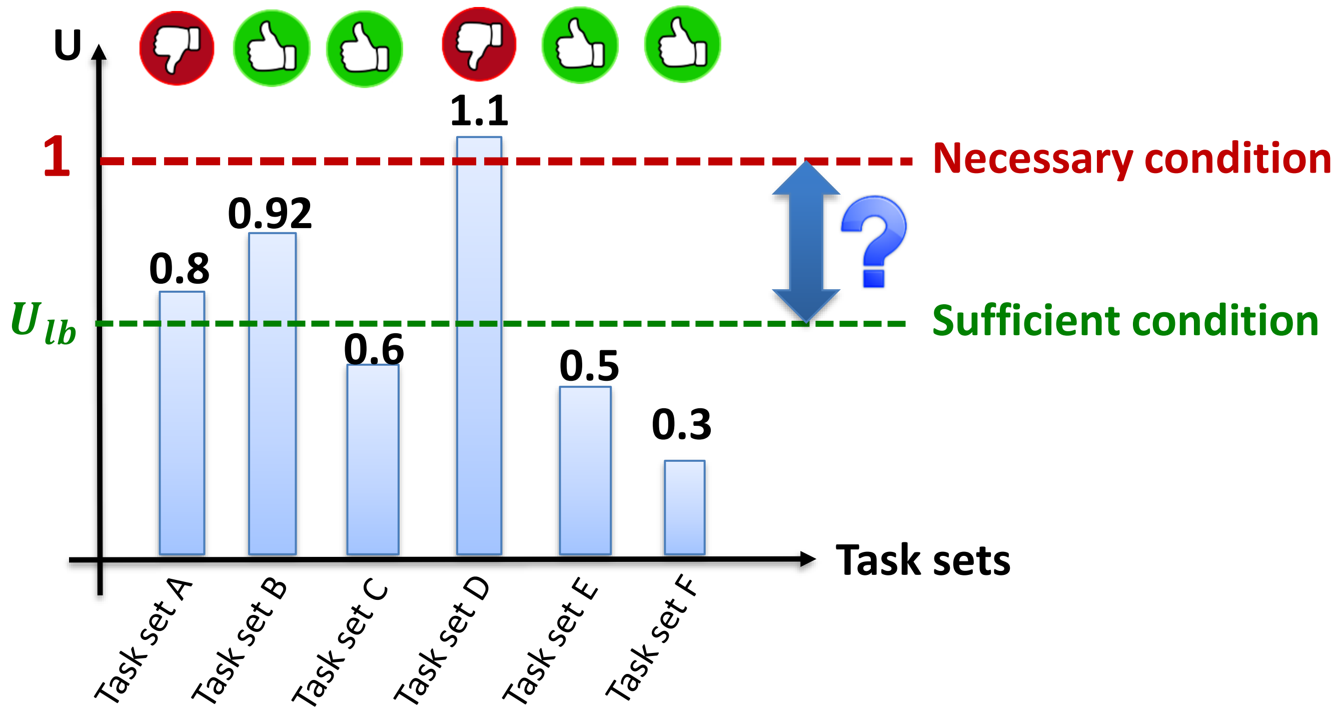
If these task sets were the universe of all task sets, what could be U_{lb}?

The simplest utilization-based test

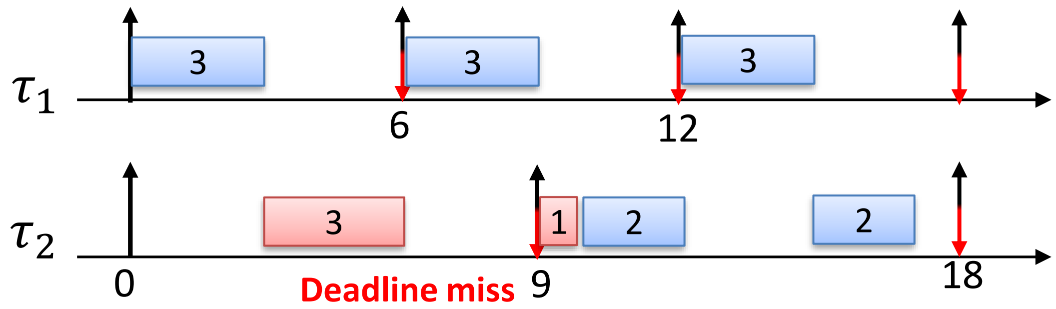
For a given task set, check whether or not

$$U \le U_{th}$$

where U_{th} is the largest utilization such that any task set with $U \le U_{th}$ is **always schedulable by RM.**



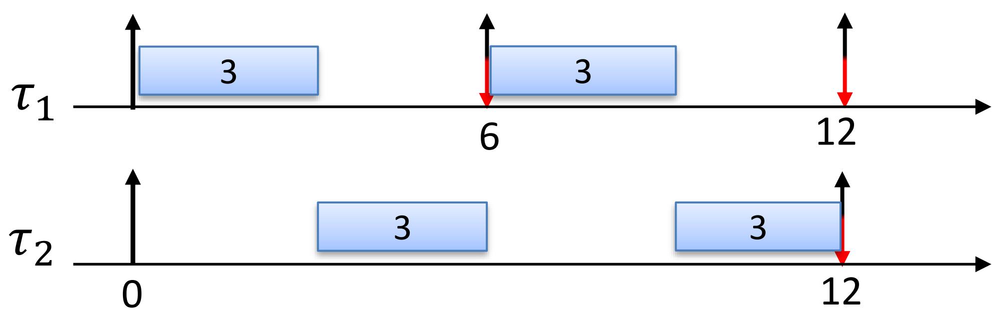
Schedulability of a FP policy is not monotonic with the utilization!



τ_i	C_i	T_i	D_i	U_i
τ_1	3	6	6	0.5
τ_2	4	9	9	0.44

$U = 0.94$

$U = 0.94$ (smaller than 1) but not schedulable!



τ_i	C_i	T_i	D_i	U_i
τ_1	3	6	6	0.5
τ_2	6	12	12	0.5

$U = 1$

$U = 1$, but it is schedulable by RM!

In this task set, periods are harmonic. Each period divides all smaller ones.

Liu and Layland test for RM

Liu and Layland [1973] derived a value for U_{lb} for the **rate monotonic** scheduling under certain **assumptions**:

A1. Every job of τ_i executes for its WCET $\boldsymbol{C_i}$

A2. For each task, $\boldsymbol{T_i = D_i}$

A3. Tasks are fully preemptive

A4. Context switch, preemption, and scheduling overheads are zero

A5. Tasks are sequential and independent:

- no precedence relations
- no resource constraints
- no blocking on I/O operations
- no self suspension
- No shared resource accesses

Hint for the exam: remember the assumptions

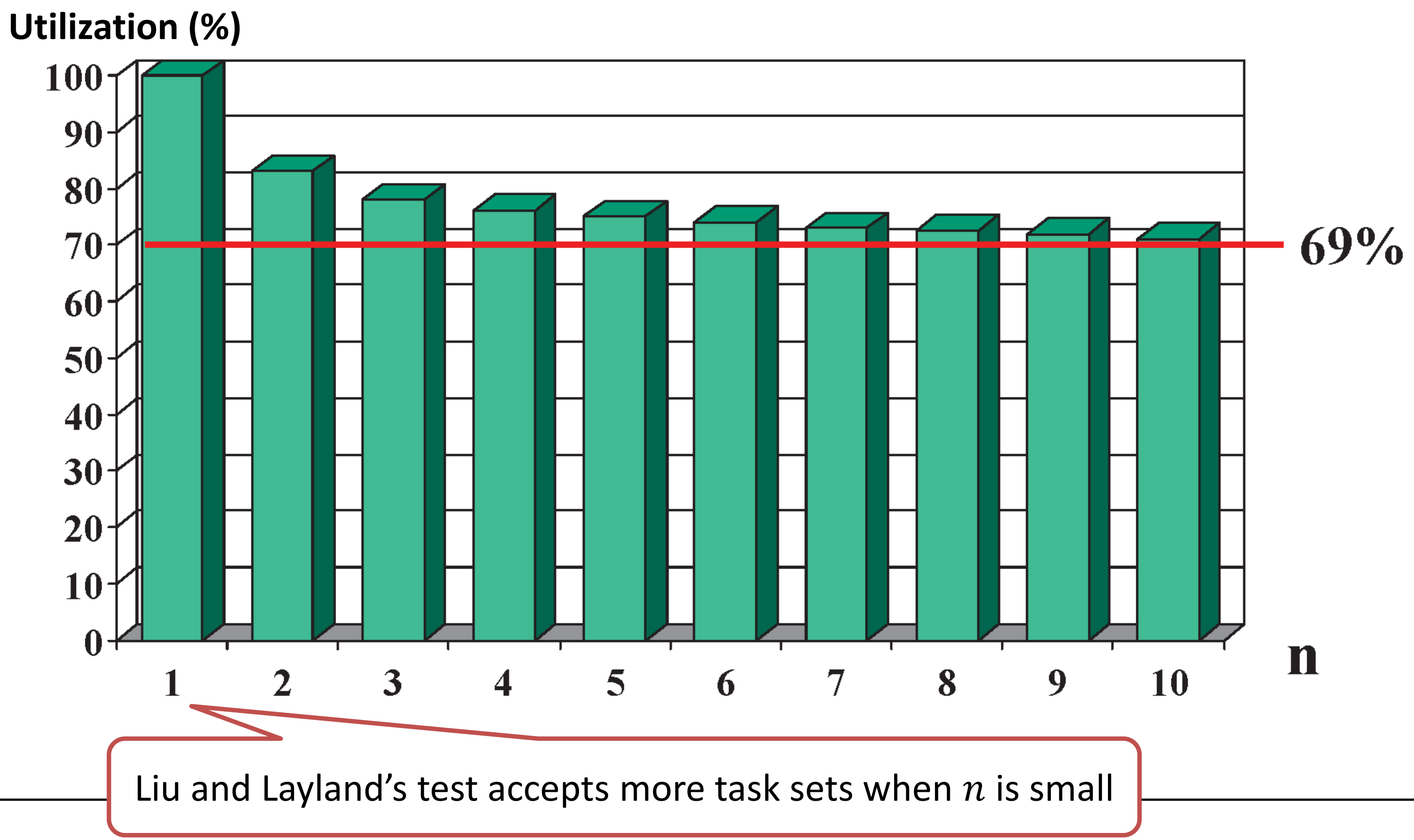
Liu and Layland's test for RM

n is the number of tasks in the task set

$$U \leq n \cdot \left(2^{1/n} - 1\right)$$

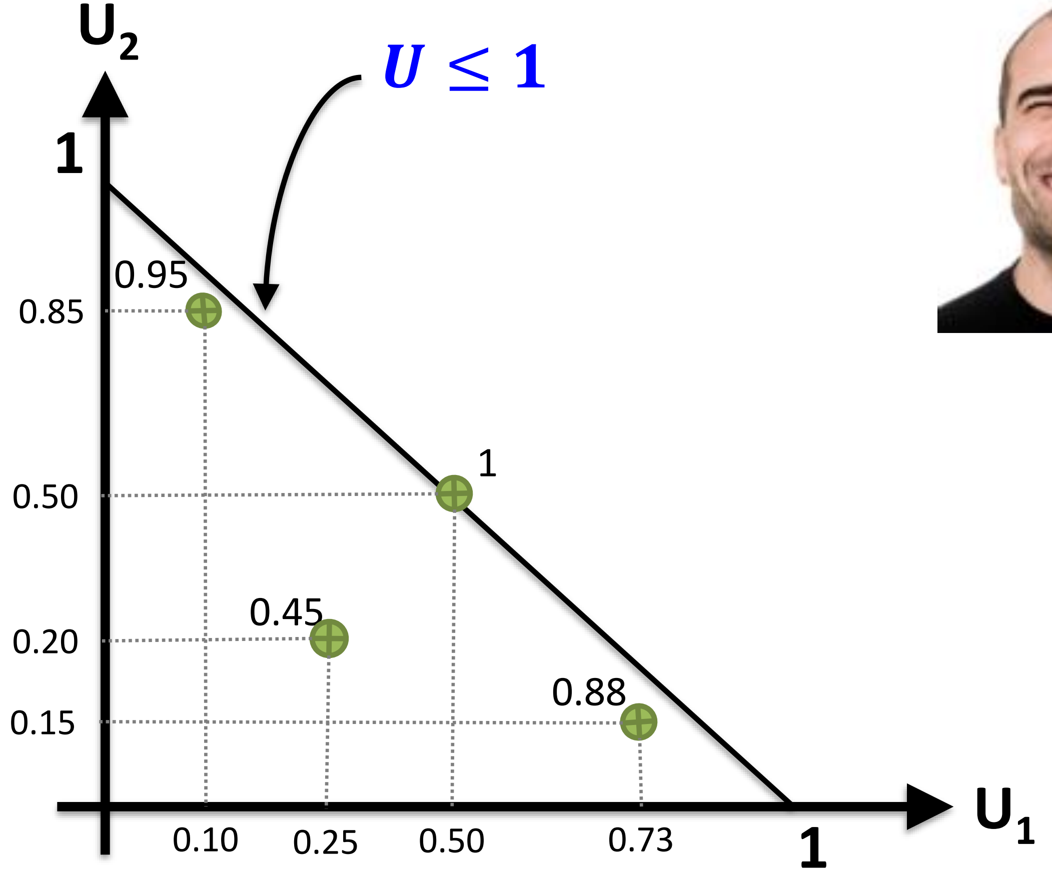
$n \to \infty$ $U_{lb} \to ?$	⇒	$n \to \infty$ $U_{lb} \to \ln 2 \sim 0.691$
$n \to 2$ $U_{lb} \to ?$	⇒	$n \to 2$ $U_{lb} \to 0.83$

Liu and Layland (L&L) test for RM



How happy are we with the L&L RM test?

$U \leq 1$ is necessary and sufficient for EDF schedulability



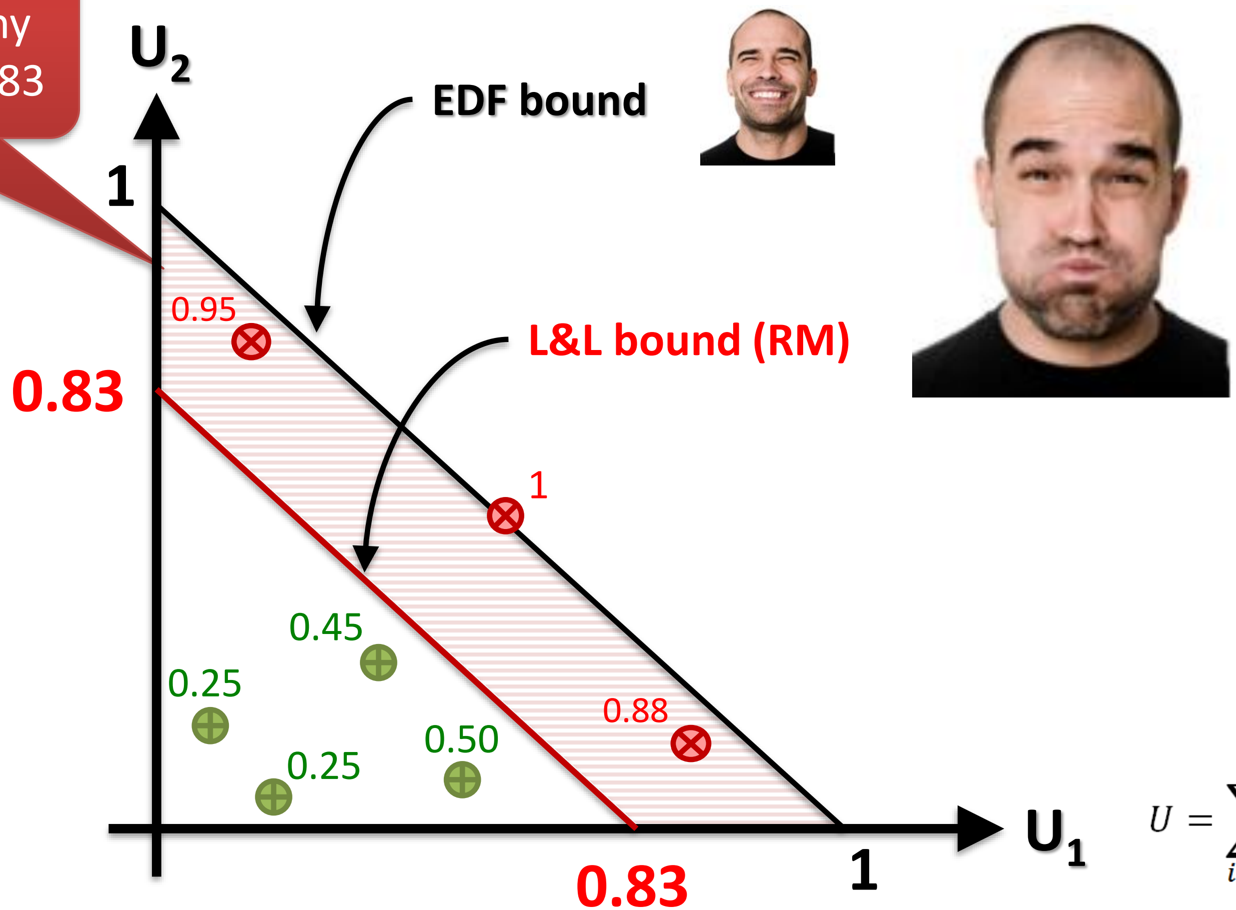
$$U = \sum_{i=1}^{2} U_i = U_1 + U_2$$

How happy are we with the L&L RM test?

The test rejects any task set with U> 0.83

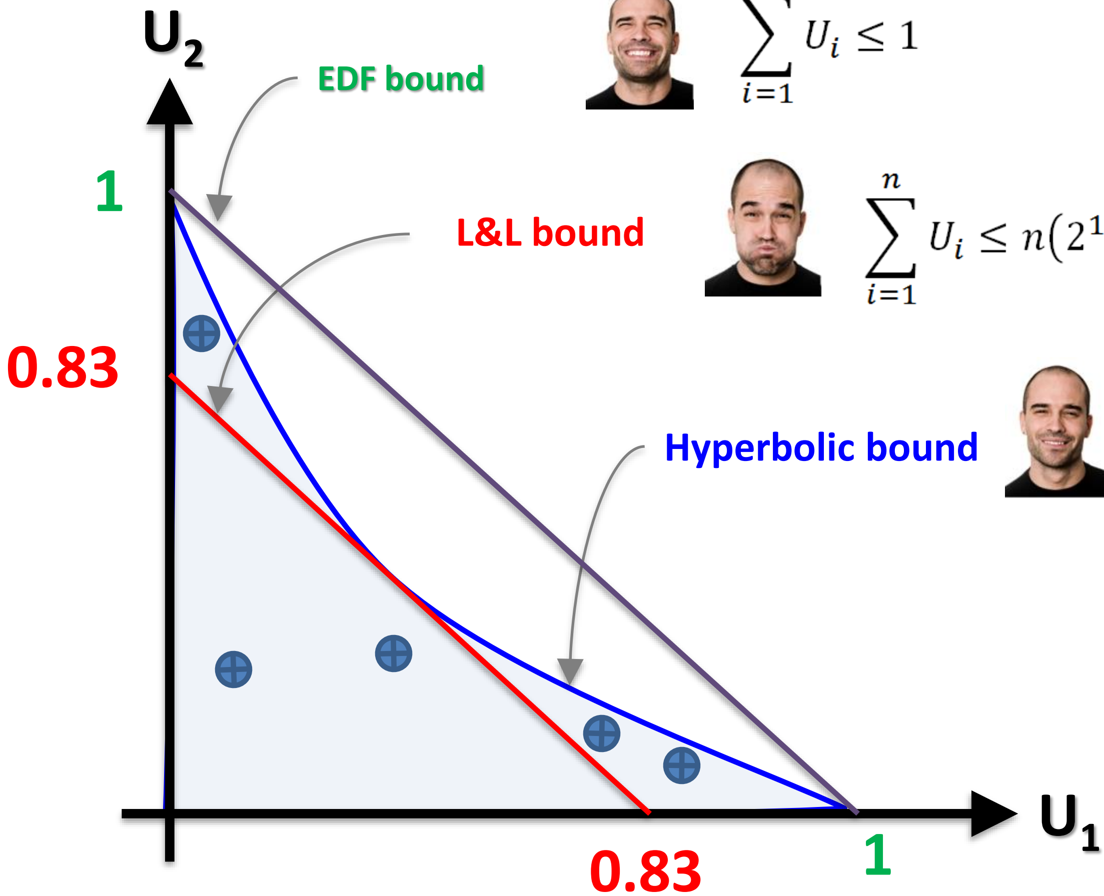
L&L Test for RM:

$$\sum_{i=1}^{n} U_i \leq n\left(2^{1/n} - 1\right)$$



$$U = \sum_{i=1}^{2} U_i = U_1 + U_2$$

Hyperbolic bound



$$\sum_{i=1}^{n} U_i \leq 1$$

$$\sum_{i=1}^{n} U_i \leq n\left(2^{1/n} - 1\right)$$

$$\prod_{i=1}^{n} (U_i + 1) \leq 2$$

The Hyperbolic Bound

- In 2000, **Bini et al.** proved that a set of n periodic tasks is schedulable with RM if:

$$\boxed{\prod_{i=1}^{n}(U_i+1) \leq 2}$$

Example

τ_i	C_i	T_i	D_i	U_i
τ_1	8	10	10	0.8
τ_2	0.9	18	18	0.05

$U = 0.85$

Is the task set feasible?

Yes $U \leq 1$

Does the task set pass the Liu & Layland's test?

No because $U = 0.85 > 2(2^{1/2} - 1) \sim 0.83$

Does the task set pass the hyperbolic-bound test?

Yes because $(0.8 + 1) \times (0.05 + 1) = 1.89 < 2$

Is the task set schedulable by RM?

Yes! Because it passed the hyperbolic bound test!

$$\sum_{i=1}^{n} U_i \leq 1$$

necessary

$$\sum_{i=1}^{n} U_i \leq n\left(2^{1/n} - 1\right)$$

Liu and Layland test

$$\prod_{i=1}^{n} (U_i + 1) \leq 2$$

Hyperbolic bound

IRiS

Example

τ_i	C_i	T_i	D_i	U_i
τ_1	3	5	5	0.6
τ_2	4	10	10	0.4

$U = 1$

Is the task set feasible?

Yes $U \leq 1$

Does the task set pass the Liu & Layland's test?

No because $U > 2(2^{1/2} - 1) \sim 0.83$

Does the task set pass the hyperbolic-bound test?

No because $(0.4 + 1) \times (0.6 + 1) = 2.24 > 2$

Is the task set schedulable by RM?

Inconclusive. Those utilization bounds are only sufficient tests, they are not exact.
(Note: the task set is in fact schedulable with RM)

$$\sum_{i=1}^{n} U_i \leq 1$$

necessary

$$\sum_{i=1}^{n} U_i \leq n\left(2^{1/n} - 1\right)$$

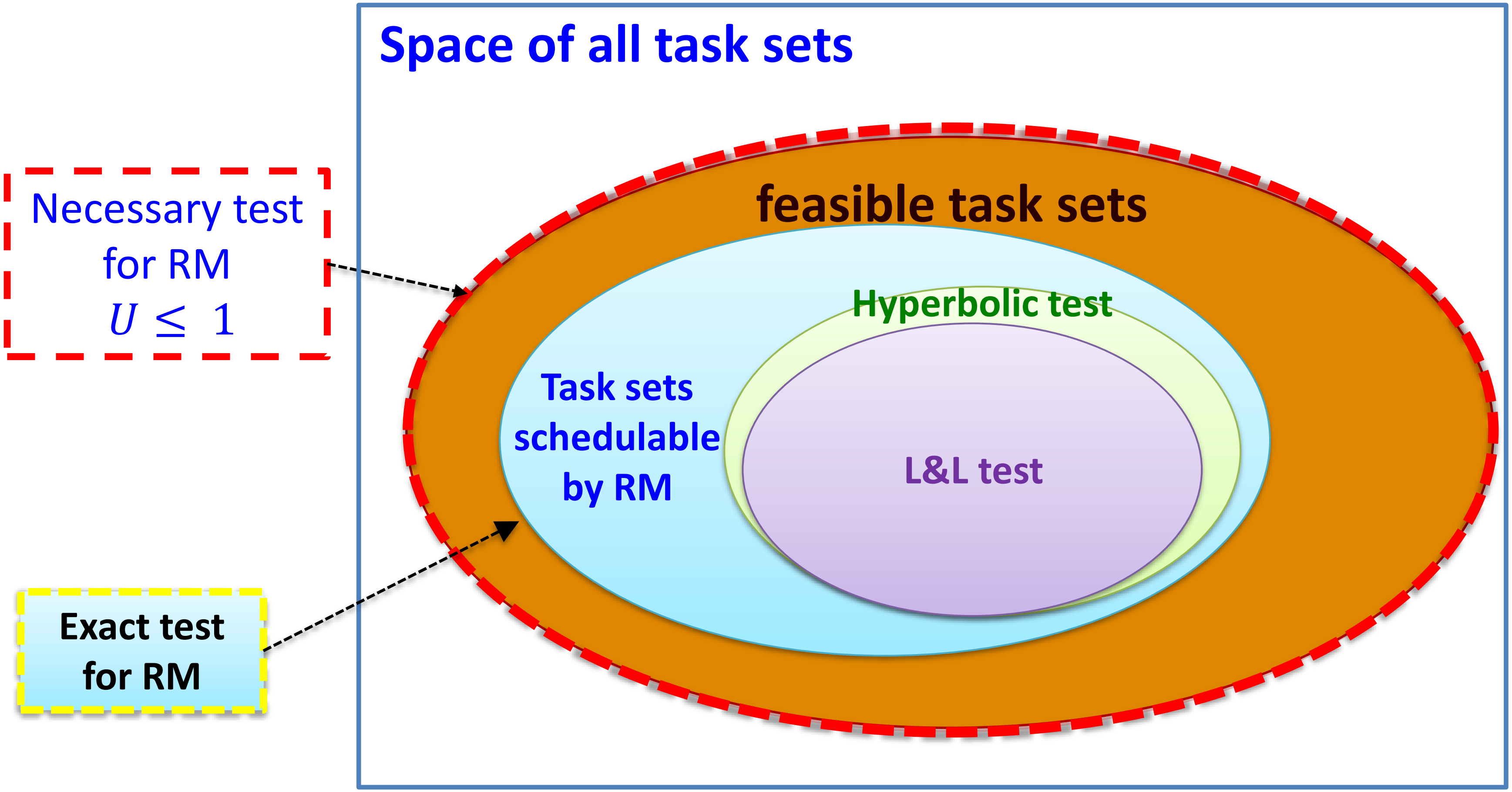
Liu and Layland test

$$\prod_{i=1}^{n} (U_i + 1) \leq 2$$

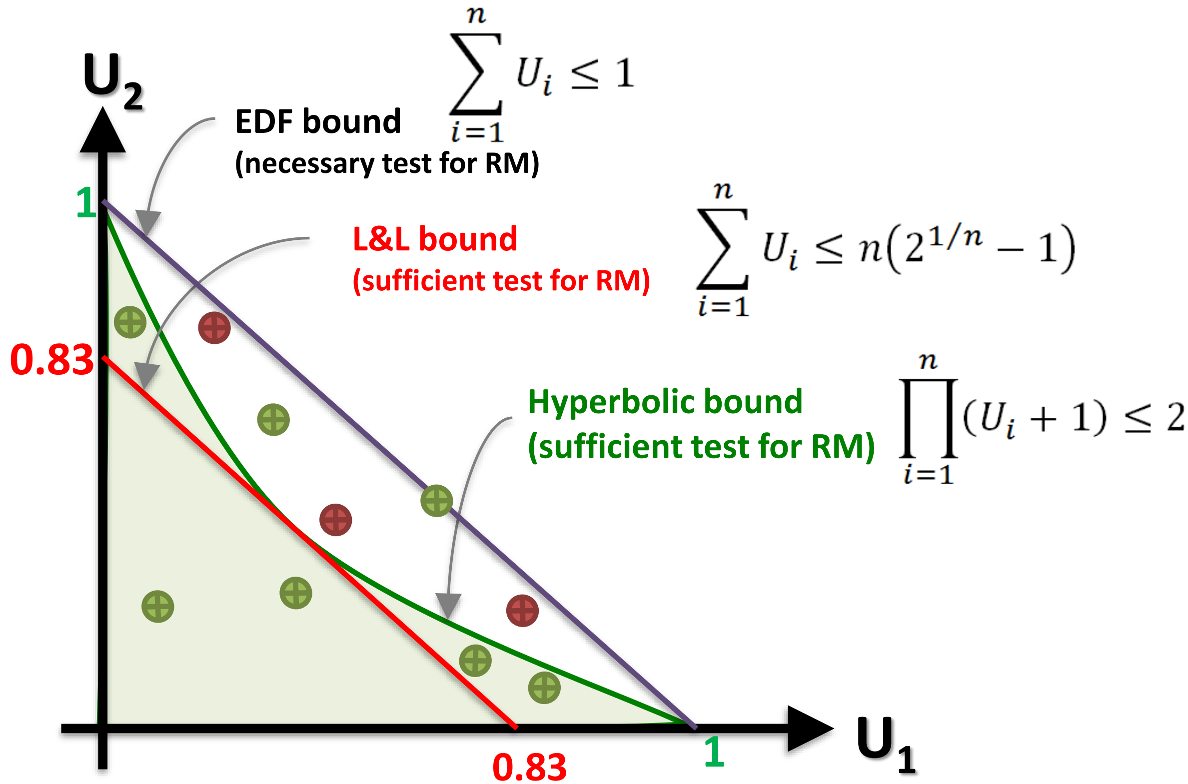
Hyperbolic bound

IRiS

The universe of utilization-based tests for RM



Utilization-based tests and RM scheduling



Bad news:

We <u>cannot</u> have a better <u>utilization-based test</u> than the hyperbolic bound!

Hyperbolic bound is tight

It is impossible to build a new utilization-based test A such that it accepts more task sets than the hyperbolic bound test!

In other words, as long as the only information used in a test is the tasks' utilization, that test will never be better than the hyperbolic bound test!

An improved test for harmonic tasks scheduled under RM

A utilization-based test for harmonic tasks

Han et al. [1997] have proven that **rate monotonic** is **optimal** if $\forall \tau_i, D_i = T_i\ , \sigma_i = 0$ and periods are **harmonic** (task periods are multiples of each others).

Hence, if periods are **harmonic** and $\forall \tau_i, D_i = T_i$, the following schedulability test is both **necessary and sufficient** for RM

$$U \leq 1$$

Try to explain why

C.-C. Han and H.-Y. Tyan. A Better Polynomial-time Schedulability Test for Real-time Fixed-priority Scheduling Algorithms. In IEEE Real-Time Systems Symposium (RTSS), pages 36–45, 1997.


QUIZ TIME

Quiz

- Is this task set feasible?

- Yes
- No
- Maybe (depends on the scheduling algorithm)

τ_i	C_i	T_i	D_i	U_i
1	1	5	5	0.2
2	5	10	10	0.5
3	1	10	10	0.1
4	1	10	10	0.1
5	1	15	15	0.06

U = 0.96

Quiz

- Is this task set feasible?

- Yes
- No
- Maybe (depends on the scheduling algorithm)

τ_i	C_i	T_i	D_i	U_i
1	1	5	5	0.2
2	5	10	10	0.5
3	1	10	10	0.1
4	1	10	10	0.1
5	1	15	15	0.06

U = 0.96

Quiz

- Based on its utilization, is this task set schedulable with RM?

- Yes
- No
- Maybe

τ_i	C_i	T_i	D_i	U_i
1	1	5	5	0.2
2	5	10	10	0.5
3	1	10	10	0.1
4	1	10	10	0.1
5	1	15	15	0.06

U = 0.96

Quiz

- Based on its utilization, is this task set schedulable with RM?

- Yes
- No
- Maybe

τ_i	C_i	T_i	D_i	U_i
1	1	5	5	0.2
2	5	10	10	0.5
3	1	10	10	0.1
4	1	10	10	0.1
5	1	15	15	0.06

U = 0.96

What defines an optimal scheduling policy (in the sense of feasibility)?

- For infeasible task sets, it minimizes the number of deadline misses
- It always generates a feasible schedule
- It generates a feasible schedule for a feasible task set

What defines an optimal scheduling policy (in the sense of feasibility)?

- For infeasible task sets, it minimizes the number of deadline misses
- It always generates a feasible schedule
- It generates a feasible schedule for a feasible task set

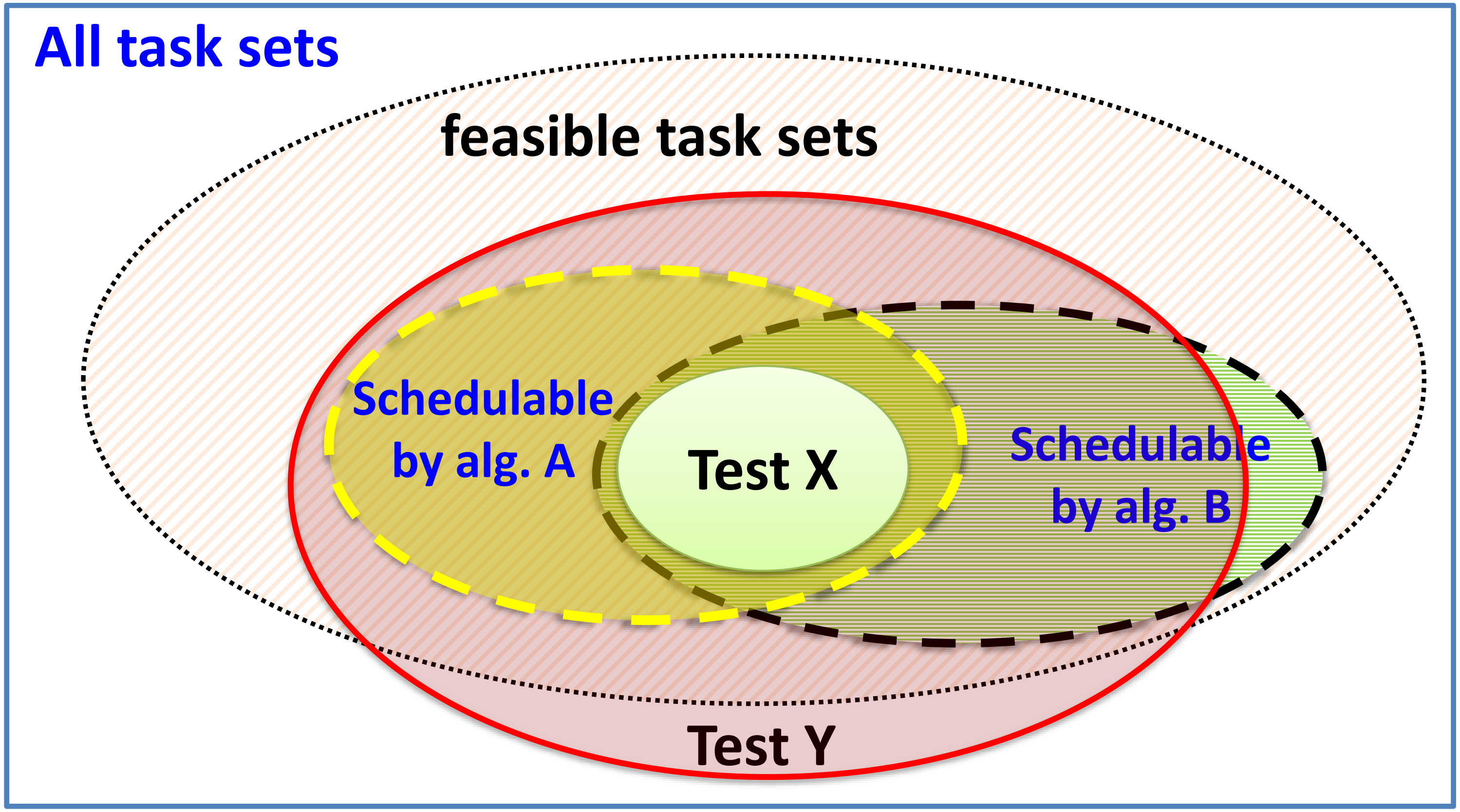
Which policy is optimal for preemptive independent tasks on single core?

- RM
- FIFO
- EDF
- DM

Which policy is optimal for preemptive independent tasks on single core?

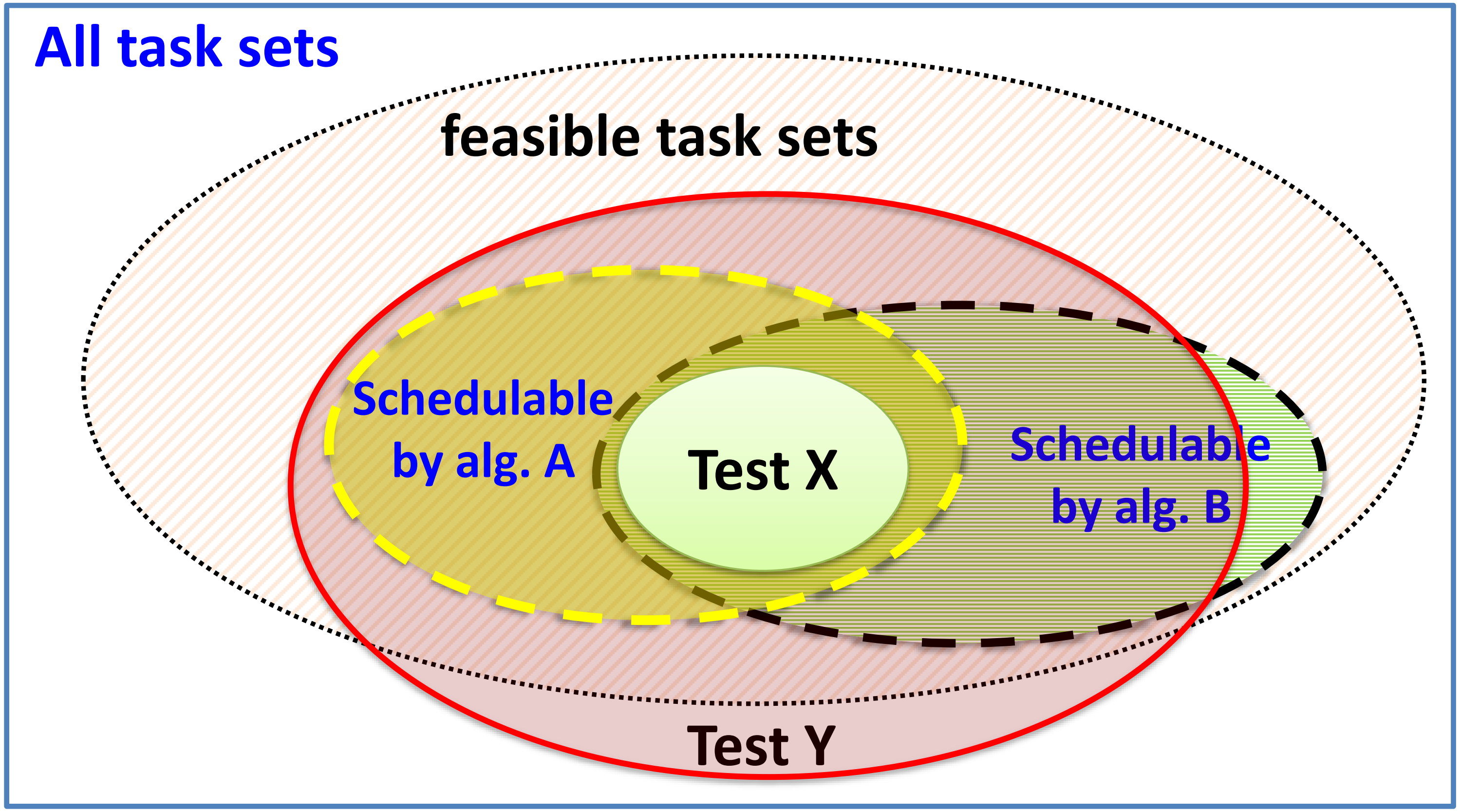
- RM
- FIFO
- EDF
- DM

Pick a correct answer



- Test X is a necessary test for alg. A
- An exact test for B is a sufficient test for A
- Test X is a sufficient test for A and B
- Test Y is not a necessary test for any algorithm

Pick a correct answer



- Test X is a necessary test for alg. A
- An exact test for B is a sufficient test for A
- **Test X is a sufficient test for A and B**
- Test Y is not a necessary test for any algorithm

Summary

- RM, DM, OPA **are all optimal priority assignments** for task-level fixed priority scheduling but under **different sets of assumptions**:
  - **RM**: independent preemptive sporadic or periodic tasks (with $\sigma_i = 0$), single core, $\boldsymbol{D_i = T_i}$ for all tasks
  - **DM**: independent preemptive sporadic or periodic tasks (with $\sigma_i = 0$), single core, $\boldsymbol{D_i \leq T_i}$ for all tasks
  - **OPA**: optimal for any set of preemptive tasks if there exists an **exact schedulability test**
- **EDF** is optimal for independent preemptive tasks on single core
- **Utilization-based** tests for RM
  - **LL-bound**
  - **Hyperbolic bound** (best possible utilization-based test for RM)

IRiS

Is rate-monotonic an optimal scheduling policy (in the sense of feasibility)?