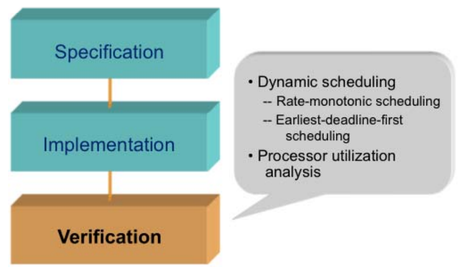
CHALMERS

Real-Time Systems


Specification
Implementation
Verification
• Dynamic scheduling
-- Rate-monotonic scheduling
-- Earliest-deadline-first scheduling
• Processor utilization analysis

CHALMERS

Dynamic scheduling

General properties:

- On-line schedule generation
  - Schedule determined by run-time behavior controlled by priorities or time quanta
  - Feasibility must be tested off-line by predicting run-time behavior
  - Configuration phase encompasses generation of priorities or time quanta for each task
- Mutual exclusion must be handled on-line
  - Support for mutual exclusion needed in real-time kernel (e.g., semaphores, disabling interrupts)
- Precedence constraints must be handled on-line
  - Dependent tasks must synchronize using semaphores or time offsets

CHALMERS

Dynamic scheduling

Advantages:

- High flexibility
  - Schedule can easily adapt to changes in the system, e.g., new tasks can be added dynamically
- External events are handled efficiently
  - I/O units handled via interrupt which activates a task
- Efficient for different types of tasks
  - Sporadic tasks can be easily supported (via suitable priority assignment)
  - Scheduling algorithms are often optimal

CHALMERS

Dynamic scheduling

Disadvantages:

- Complicates communication between tasks
  - Exact time of data availability is not known in advance, which requires extra synchronization between tasks
  - Task execution is difficult to adapt to existing time-triggered (TDMA) network protocols (but does work very well with many priority-based network protocols, e.g., CAN and Token Ring)
- Task execution becomes indeterministic
  - Temporary deviations (”jitter”) in task periodicity may occur
  - Exact feasibility tests often have high time complexity
  - Low observability (difficult to debug)

CHALMERS

Dynamic scheduling

How is task scheduling done?

- Using static or dynamic priorities:
  - Ready tasks are stored in a queue, sorted by priority
  - At scheduling decisions, the task with highest priority is selected
- Using time quanta: (”round-robin”)
  - Ready tasks are stored in a circular FIFO queue
  - Each task gets access to the processor for a certain time interval (quantum); real-time clock is used for interrupting the execution
  - New scheduling decisions can be taken sooner if the executing task terminates or gets blocked

In this course, we only study dynamic scheduling using priorities.

CHALMERS

Dynamic scheduling

How are task priorities assigned?

- Static assignment:
  - Rate-monotonic scheduling
  - Deadline-monotonic scheduling
  - Weight-monotonic scheduling
- Dynamic assignment:
  - Earliest-deadline-first scheduling
  - Least-laxity-first scheduling

In this course, we only study rate-monotonic, deadline-monotonic and earliest-deadline-first scheduling.

CHALMERS

Dynamic scheduling

How is the scheduler implemented?

- Create a queue for the ready tasks
  - Each element in the queue refers to a PCB
  - The elements in the queue are sorted according to task priorities; if multiple tasks have equal priority, the sorting is arbitrary (e.g., FIFO)
- The queue is updated at external or internal events
  - An external event is one that occurs in the environment (the controlled system); for example: an I/O unit generates an interrupt because data has become available at a sensor
  - An internal event is one that occurs within the computer system; for example: a timer generates an interrupt because a certain point in time has been reached

CHALMERS

Rate-monotonic scheduling

Properties:

- Uses static priorities
  - Priority is determined by task frequency (rate): the task with the highest rate (= shortest period) receives highest priority
- Theoretically well-established
  - Sufficient feasibility test can be performed in linear time (under certain simplifying assumptions)
  - Exact feasibility test is an NP-complete problem
  - RM is optimal among all scheduling algorithms that use static task priorities
    (shown by C. L. Liu and J. W. Layland in 1973)

CHALMERS

Earliest-deadline-first scheduling

Properties:

- Uses dynamic priorities
  - Priority is determined by how critical the task is at a given point in time: the task whose absolute deadline is closest in time receives highest priority
  - Can be used for periodic, sporadic and aperiodic tasks
- Theoretically well-established
  - Exact feasibility test can be performed in linear time (under certain simplifying assumptions)
  - EDF is optimal among all scheduling algorithms that use dynamic task priorities (shown by C. L. Liu and J. W. Layland in 1973)

CHALMERS

Dynamic scheduling

What techniques for feasibility testing exist?

- Processor utilization analysis (for static/dynamic priorities)
  - The fraction of processor time that is used for executing the task set may not exceed a given bound
  - Used for traditional RM and EDF
- Response time analysis (for static priorities)
  - Worst-case response time for each task is calculated and compared against the deadline of the task
  - Used for generalized DM
- Processor demand analysis (for dynamic priorities)
  - The accumulated computation demand for the task set under a given time interval must not exceed the length of the interval
  - Used for generalized EDF

CHALMERS

Processor utilization analysis

The utilization U for a set of periodic tasks is the fraction of the processor's capacity that is used for executing the tasks.

Since C_i / T_i is the fraction of processor time that is used for executing task τ_i the utilization for n tasks is

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

CHALMERS

Simple feasibility test for RM

(Sufficient condition)

A sufficient condition for rate-monotonic scheduling based on the utilization U is

$$U = \sum_{i=1}^{n} \frac{C_i}{T_i} \leq n\left(2^{1/n} - 1\right)$$

where n is the number of tasks.

This is a classic feasibility test presented by C. L. Liu and J. W. Layland in 1973.

CHALMERS

Simple feasibility test for RM

(Sufficient condition)

Observe that it is possible to derive a conservative lower bound on utilization by letting $n \rightarrow \infty$.

$$\lim_{n\rightarrow\infty} n\left(2^{1/n} - 1\right) = \ln 2 \approx 0.693$$

This means that a set of tasks (regardless of number of tasks) whose total utilization does not exceed 0.693 is always schedulable with RM!

CHALMERS

Simple feasibility test for RM

(Sufficient condition)

The test is valid under the following assumptions:

1. All tasks are independent.
   - There must not exist dependencies due to precedence or mutual exclusion
2. All tasks are periodic or sporadic.
3. Task deadline equals the period ($D_i = T_i$).
4. Task preemptions are allowed.

CHALMERS

Simple feasibility test for RM

(Sufficient condition)

The proof of the condition uses the following theorem:

The worst-case response time for a task occurs at a critical instant (where all tasks arrive at the same time.)

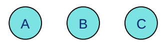
The feasibility test is derived using an analysis of this special case. The proof also shows that if the task set is schedulable for the critical instant case, it is also schedulable for any other case. We refrain from analyzing the proof …

CHALMERS

Example: scheduling using RM

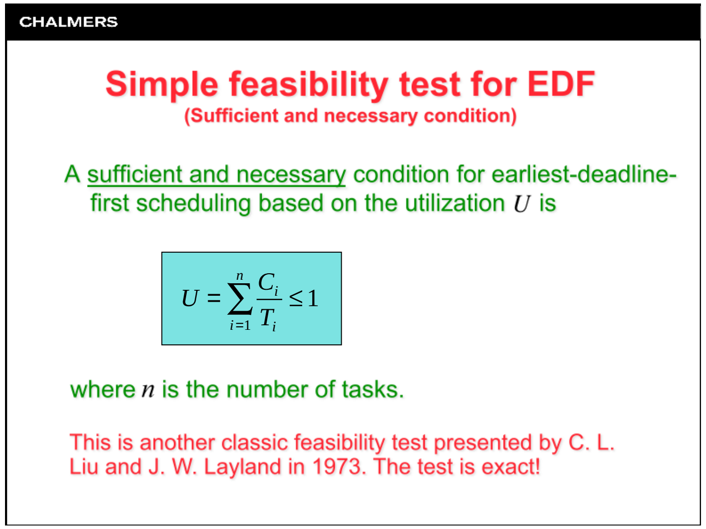
Problem: Assume a system with tasks according to the figure below. The timing properties of the tasks are given in the table. Schedule the tasks using rate-monotonic scheduling (RM).

a) What is the utilization of the task set?
b) What is the outcome of Liu & Layland's feasibility test for RM?
c) Show that the tasks are schedulable using RM.



Task	C_i	O_i	T_i
A	1	0	3
B	1	0	4
C	1	0	5

We solve this on the blackboard!



CHALMERS

Simple feasibility test for EDF

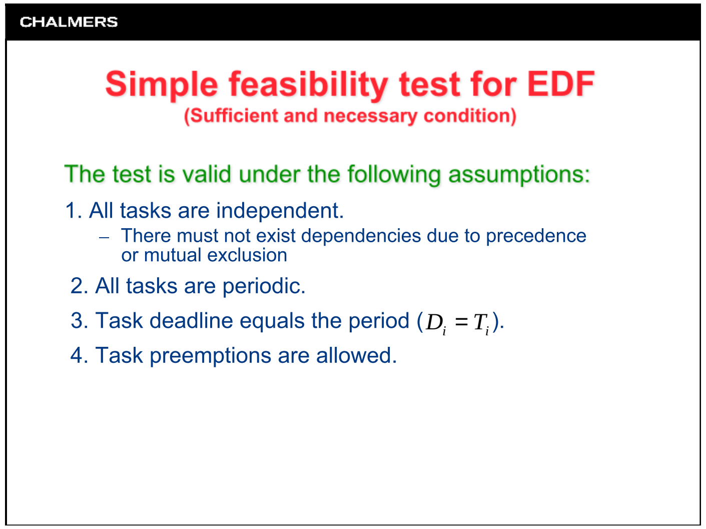
(Sufficient and necessary condition)

A sufficient and necessary condition for earliest-deadline-first scheduling based on the utilization U is

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

where n is the number of tasks.

This is another classic feasibility test presented by C. L. Liu and J. W. Layland in 1973. The test is exact!



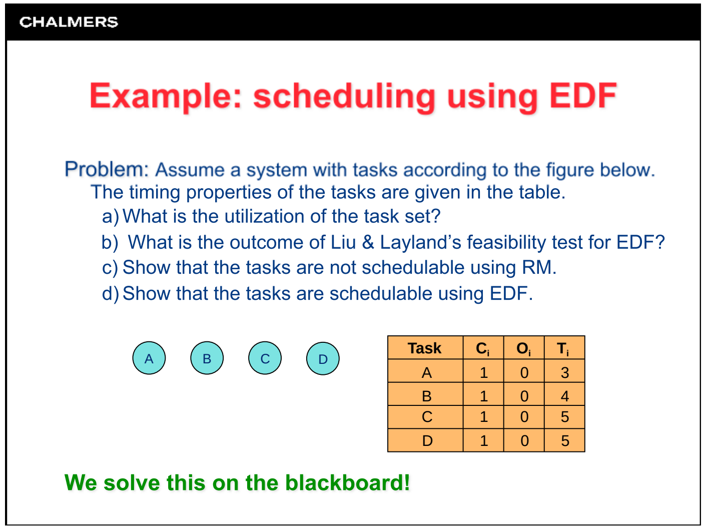
CHALMERS

Simple feasibility test for EDF

(Sufficient and necessary condition)

The test is valid under the following assumptions:

1. All tasks are independent.
   – There must not exist dependencies due to precedence or mutual exclusion
2. All tasks are periodic.
3. Task deadline equals the period ($D_i = T_i$).
4. Task preemptions are allowed.



CHALMERS

Example: scheduling using EDF

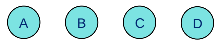
Problem: Assume a system with tasks according to the figure below. The timing properties of the tasks are given in the table.

a) What is the utilization of the task set?

b) What is the outcome of Liu & Layland's feasibility test for EDF?

c) Show that the tasks are not schedulable using RM.

d) Show that the tasks are schedulable using EDF.



Task	C_i	O_i	T_i
A	1	0	3
B	1	0	4
C	1	0	5
D	1	0	5

We solve this on the blackboard!