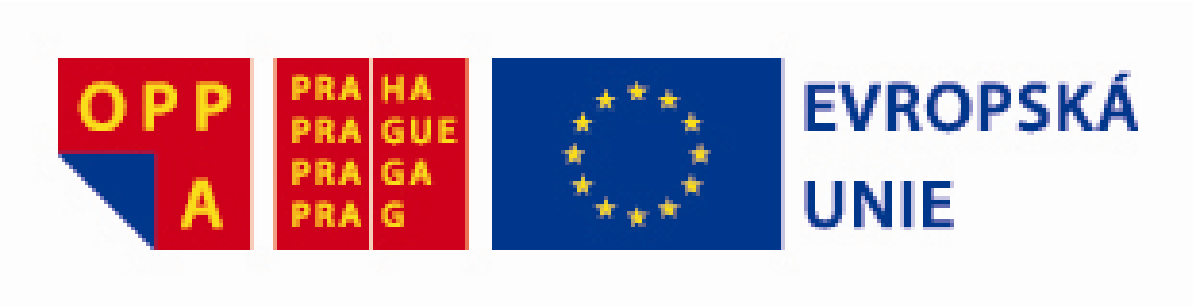
Inovace tohoto kurzu byla v roce 2011/12 podpořena projektem CZ.2.17/3.1.00/33274 financovaným Evropským sociálním fondem a Magistrátem hl. m. Prahy.



Evropský sociální fond
Praha & EU: Investujeme do vaší budoucnosti

Department of Distributed and Dependable Systems D3S

Embedded and Real-time Systems

More on offline scheduling

http://d3s.mff.cuni.cz

Tomáš Bureš

<bures@d3s.mff.cuni.cz>

CHARLES UNIVERSITY IN PRAGUE

Faculty of Mathematics and Physics

Offline scheduling

- Also known as static or pre-run-time scheduling
  - Static schedule (time table) created before we start the system
  - Run-time dispatching: just follows the generated time table
- Analysis
  - “proof by construction”
- Application area
  - Safety-critical systems
  - Used e.g. for control system of Boeing 777

Structure of an Offline Schedule

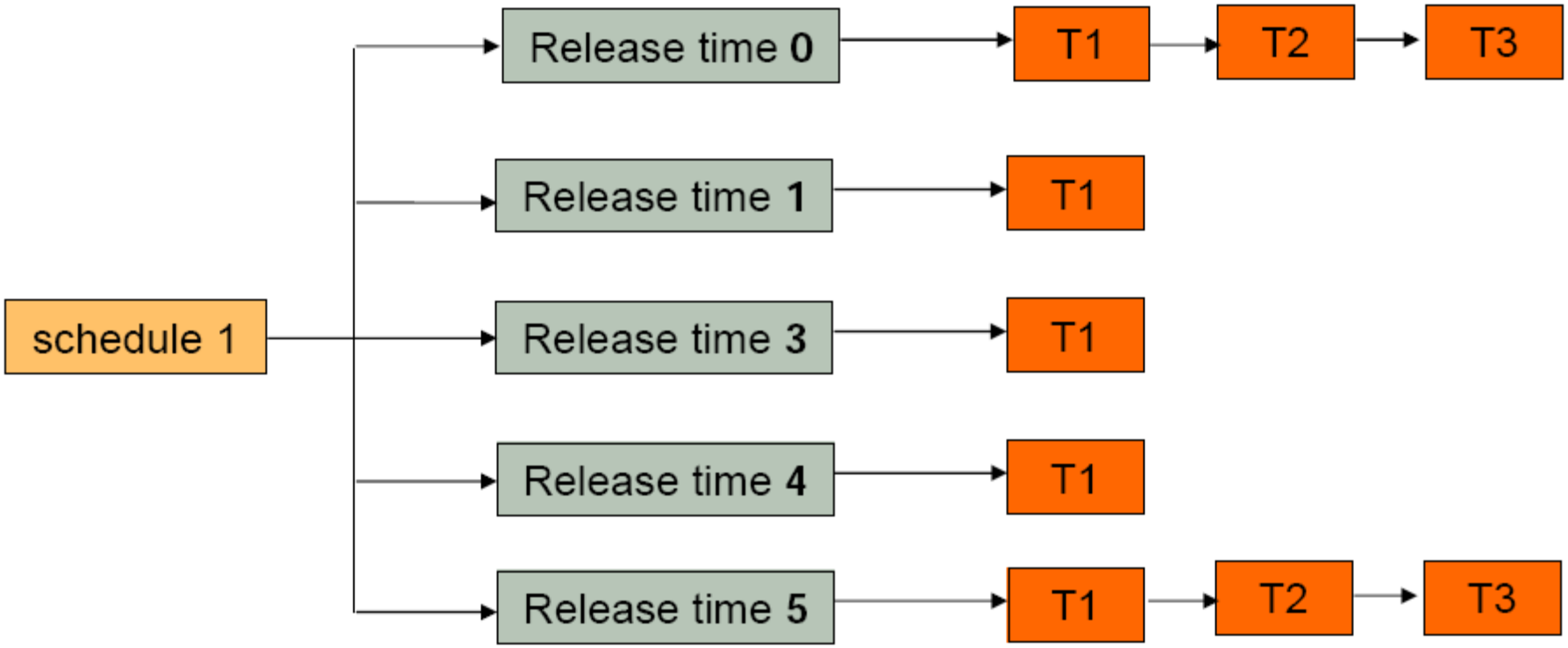
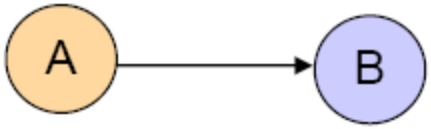


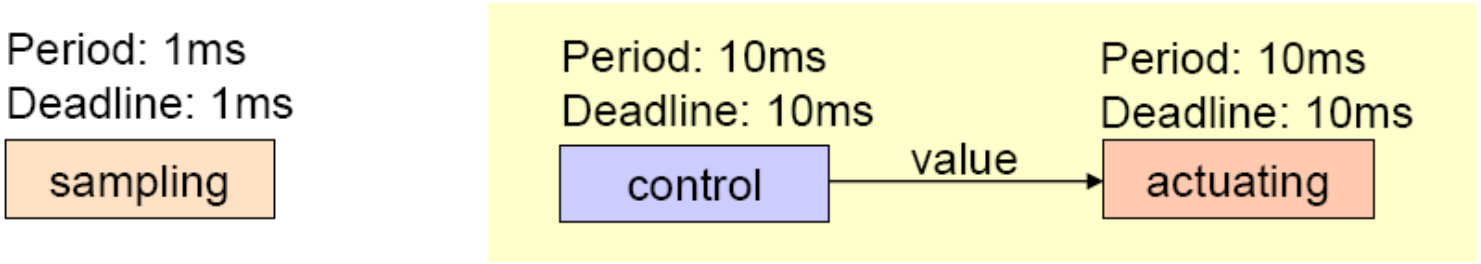
Figure from Issovic, D.:Real-time systems, basic course

Precedence Relation

- If A precedes B then both tasks run with the same period, but A must complete before B starts to execute.



- Example:
  - Assume a small control system in which we need to sample the environment with the rate of 1000 Hz and control and actuate at 100 Hz



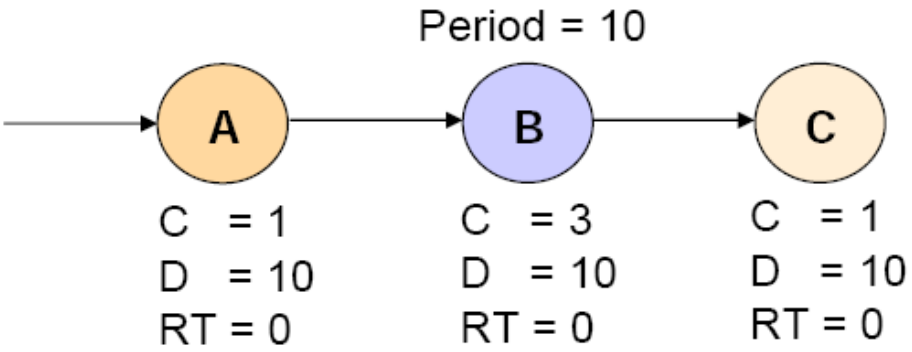
- We must specify a precedence relation between the control task and the actuator task so that the control task always sends the latest calculated value to the actuator task

Figure from Issovic, D.:Real-time systems, basic course

Simple Offline Scheduler

- Each task is described with three temporal parameters:
  - C_i – execution time
  - D_i – deadline
  - RT_i – release time
- Some additional relations:
  - Precedence order between task
  - Period for precedence graphs

- Example precedence graph:



Goal: to find a schedule that satisfies the requirements

Figure from Issovic, D.:Real-time systems, basic course

Steps of Offline Scheduling

- Three steps in offline scheduling:
  - Construct a joint graph with period equal to the least common multiple of all precedence graph's periods
  - Generate a search tree
  - Traverse the search tree to find a solution

Steps of Offline Scheduling

- Example:

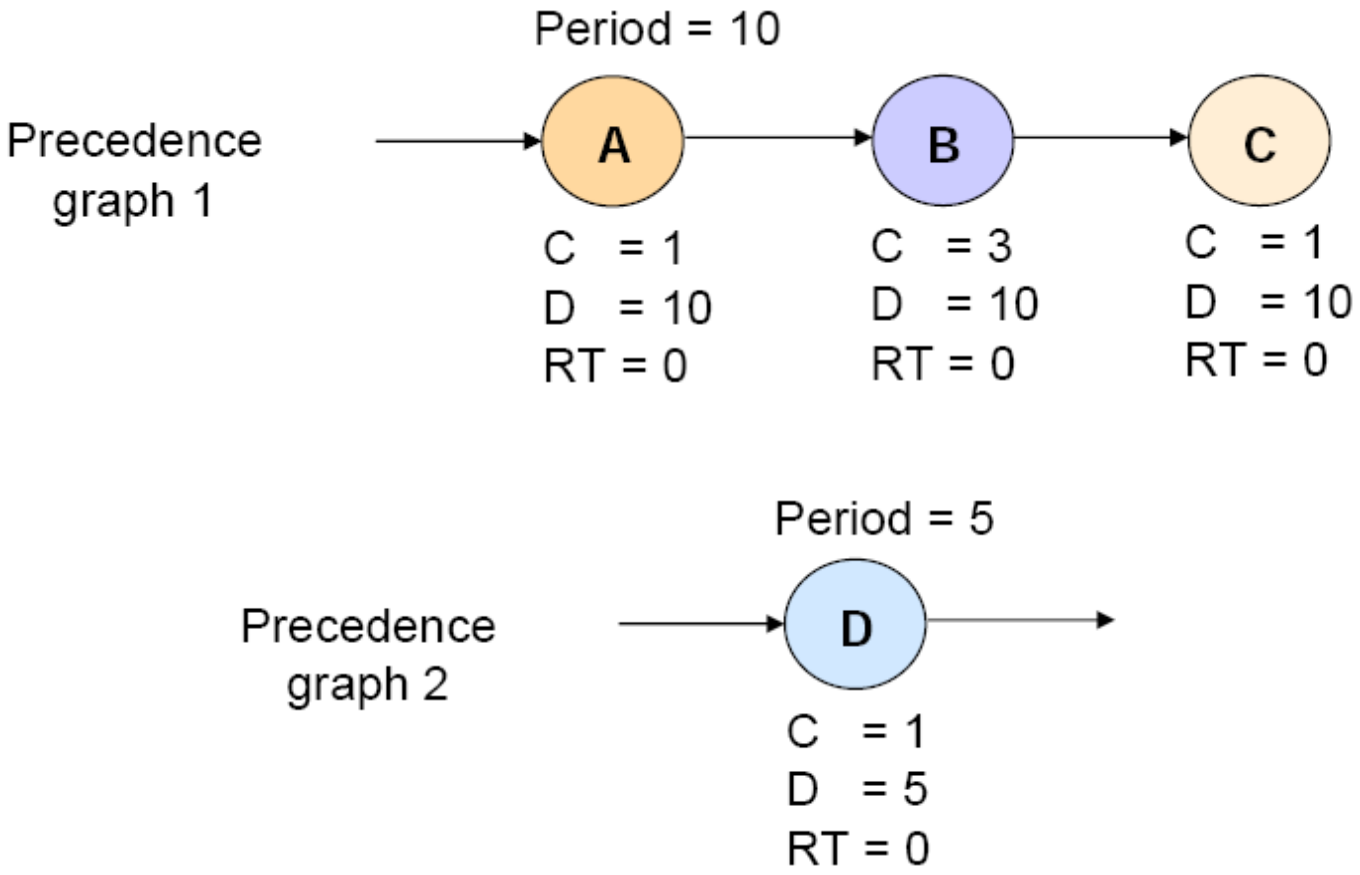
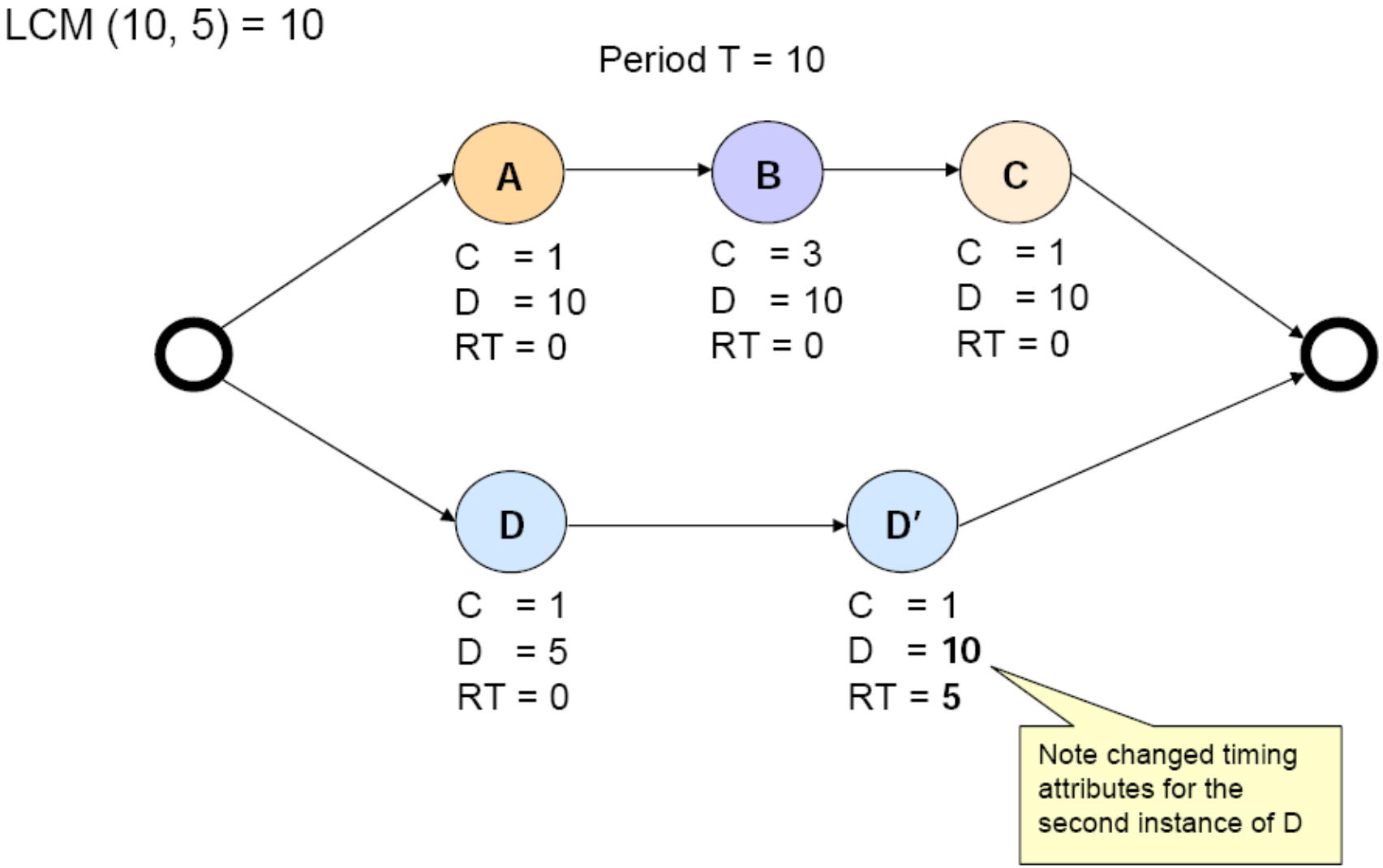
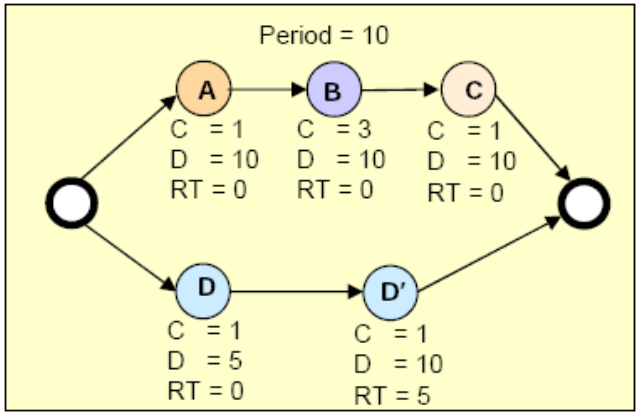


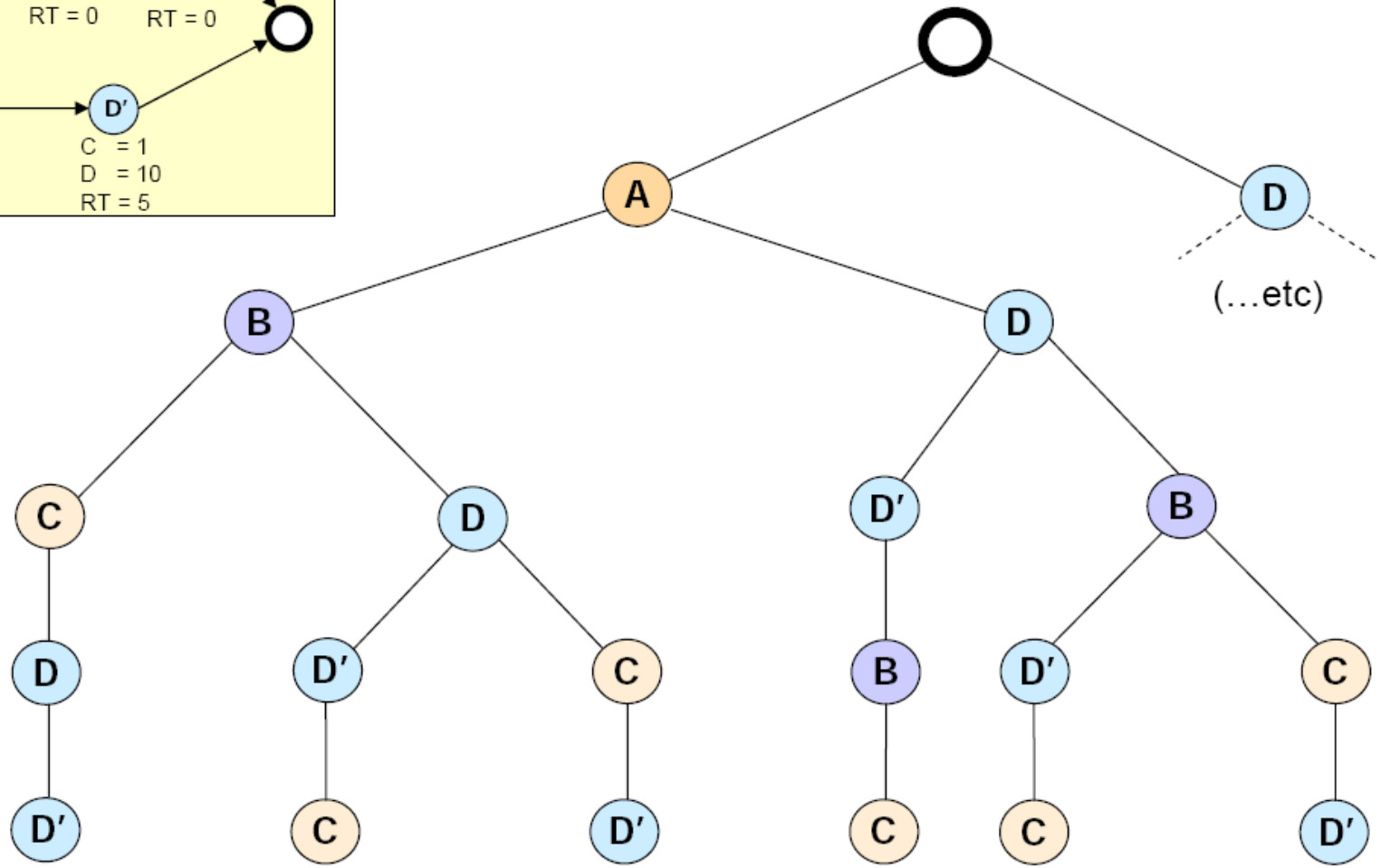
Figure from Issovic, D.:Real-time systems, basic course

Step 1: Construct Joint Graph

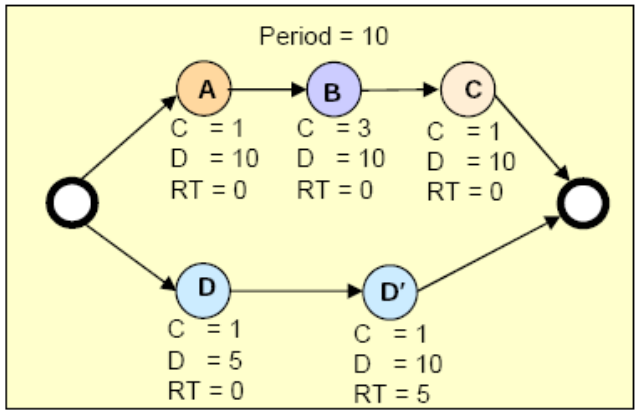


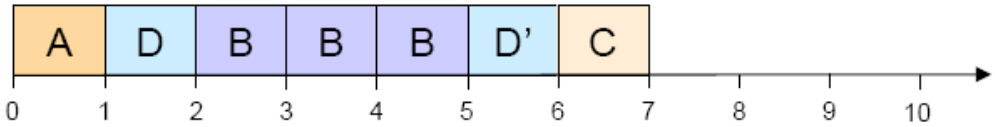
Step 2: Generate Search Tree

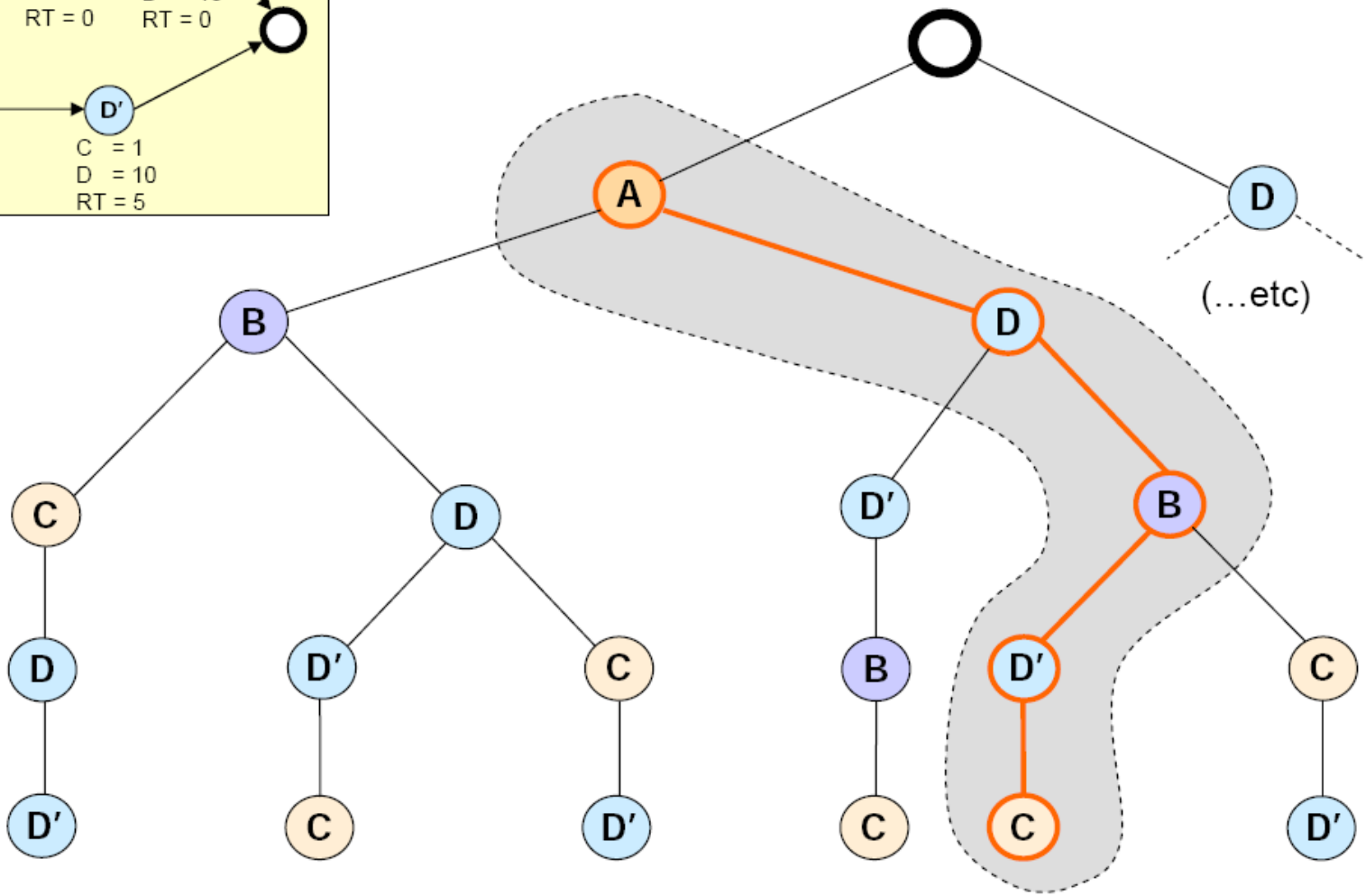




Step 3: Traverse the Search Tree







Complexity

- The size of the search tree increases exponentially for each new task we add to the system $\Longrightarrow$ it can take very long time to find a solution

- If we allow preemption
  - (+) higher probability that we'll find a solution
  - (-) bigger search tree

- We must use some strategy – heuristic
  - Ex 1: Chose the task with the shortest deadline
  - Ex 2: Chose the task with the longest execution time

Mutual Exclusion and Communication

- Mutual Exclusion
  - Solved on the level of the schedule by not allowing two tasks to run concurrently
- Communication
  - Solved by a system task that copies the data between the run of communicating tasks

Mode Changes

- Processes controlled by hard real-time systems may exhibit mutually exclusive phases of operation and control. This means that the functionality is different at different system states. In this case, we can brake the system into different modes of operation.

- Example 1: Modes and mode transitions in a vehicle

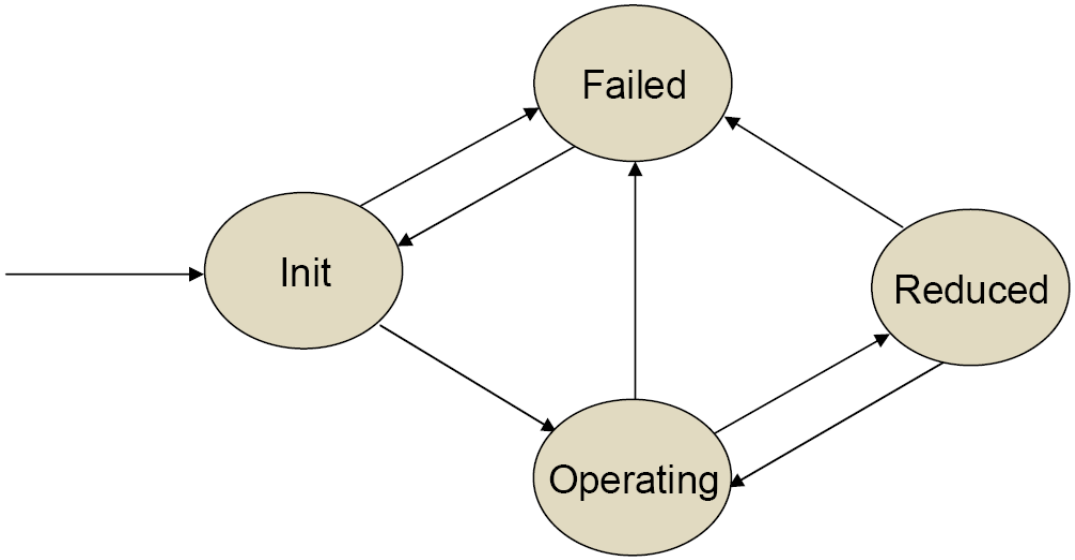


Figure from Issovic, D.:Real-time systems, basic course

Mode Changes

- Each mode has its own offline schedule ⟹ when we switch a mode we switch a schedule

- Example 2: Modes in an airplane

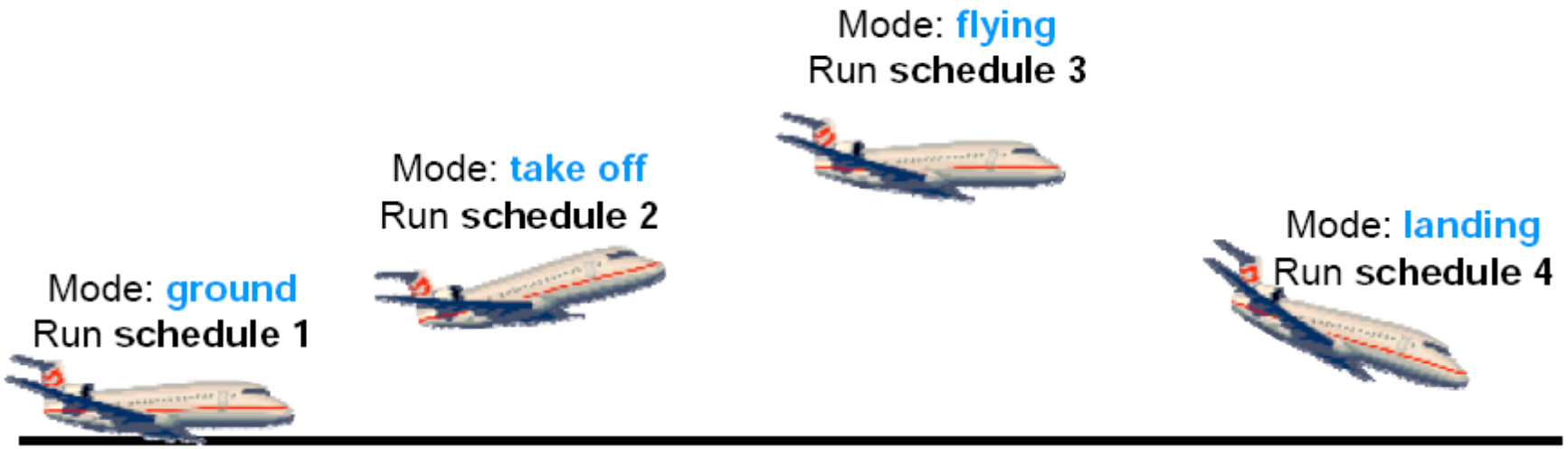


Figure from Issovic, D.:Real-time systems, basic course

Schedule Switch upon Mode Change