Facultat d'Informàtica de Barcelona
Universitat Politècnica de Catalunya

Real-Time Systems

3e non-RT schedulers

Antonio Camacho Santiago
antonio.camacho.santiago@upc.edu

Objective

To understand the difference between RT and non-RT schedulers…

To evaluate the pros and cons of RT and non-RT schedulers …

`a RT scheduler is aimed to ensure timing constraints of the tasks`
`a non-RT scheduler is aimed to satisfy some other criterion`

More details can be found in G.Buttazzo, Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications", 3rd ed., Springer, 2011.

Schedulers basics

Schedulers can be…

Preemptive or non-preemptive:

In preemptive algorithms, the running task can be interrupted at any time to assign the processor to another active task, according to a predefined scheduling policy.

In non-preemptive algorithms, a task, once started, is executed by the processor until completion. In this case, all scheduling decisions are taken as the task terminates its execution

Schedulers basics

Schedulers can be…

Static vs. Dynamic:

Static algorithms are those in which scheduling decisions are based on fixed parameters, assigned to tasks before their activation.

Dynamic algorithms are those in which scheduling decisions are based on dynamic parameters that may change during system evolution.

Schedulers basics

Schedulers can be…

Offline vs. Online:

A scheduling algorithm is used off line if it is executed on the entire task set before tasks activation. The schedule generated in this way is stored in a table and later executed by a dispatcher.

A scheduling algorithm is used online if scheduling decisions are taken at runtime every time a new task enters the system or when a running task terminates.

Schedulers basics

Schedulers can be…

Optimal vs. Heuristic:

An algorithm is said to be optimal if it minimizes some given cost function defined over the task set. When no cost function is defined and the only concern is to achieve a feasible schedule, then an algorithm is said to be optimal if it is able to find a feasible schedule, if one exists.

An algorithm is said to be heuristic if it is guided by a heuristic function in taking its scheduling decisions. A heuristic algorithm tends toward the optimal schedule, but does not guarantee finding it.

Schedulers basics

Schedulers can be…

Guarantee vs Best-effort algorithms:

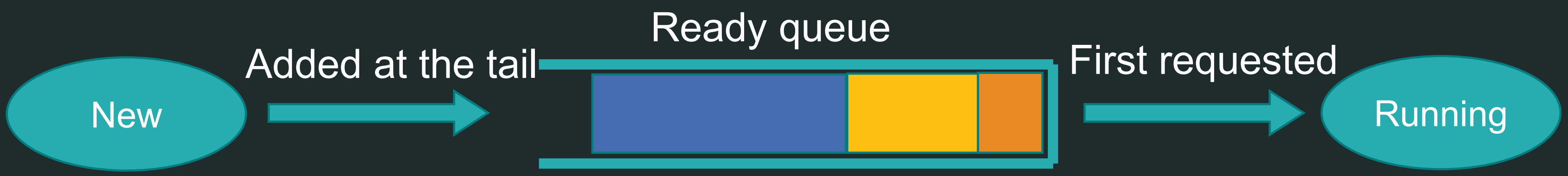
In hard real-time applications that require highly predictable behavior, the feasibility of the schedule should be guaranteed in advance. RT-schedulers are used for this purpose

In certain real-time applications, computational activities have soft timing constraints that should be met whenever possible to satisfy system requirements. In these systems, missing soft deadlines do not cause catastrophic consequences, but only a performance degradation. Non-RT schedulers are used for these systems

First Come First Served (FCFS)

The CPU is assigned to tasks based on their arrival times

It is: non-preemptive, dynamic, online, non-optimal, very unpredictable (response times depend on arrival time)

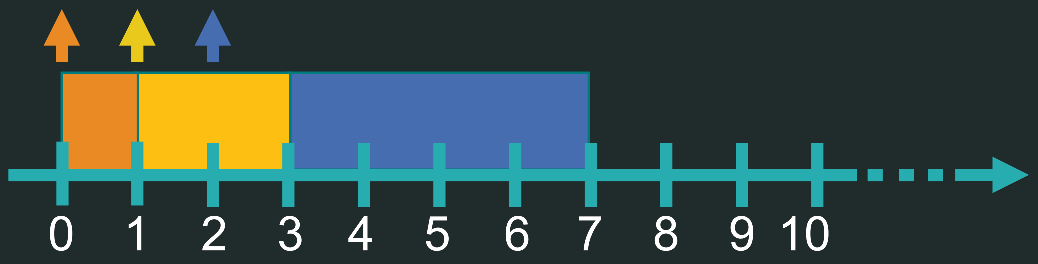


First Come First Served (FCFS)

The response time in FCFS is very unpredictable

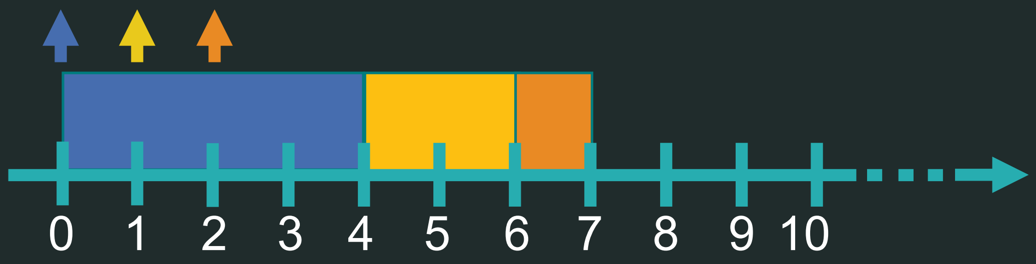
Task τ_i	Computing time c_i (ms)	Arrival time a_i (ms)
τ_1	1	0
τ_2	2	1
τ_3	4	2

Response times:
$R_1 = 1$
$R_2 = 3$
$R_3 = 7$



Task τ_i	Computing time c_i (ms)	Arrival time a_i (ms)
τ_1	1	2
τ_2	2	1
τ_3	4	0

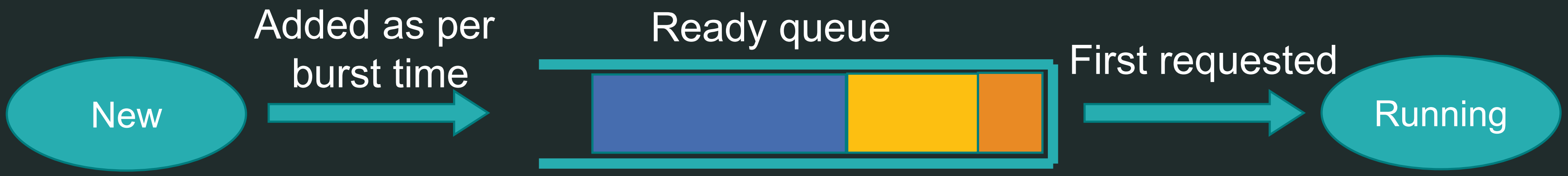
Response times:
$R_1 = 7$
$R_2 = 6$
$R_3 = 4$



Shortest Job First (SJF)

The CPU is assigned to tasks with the shortest computation time

It is: preemptive or not, static (due to c_i is constant), online or offline, minimizes the average response time but it is not optimal in the sense of the feasibility



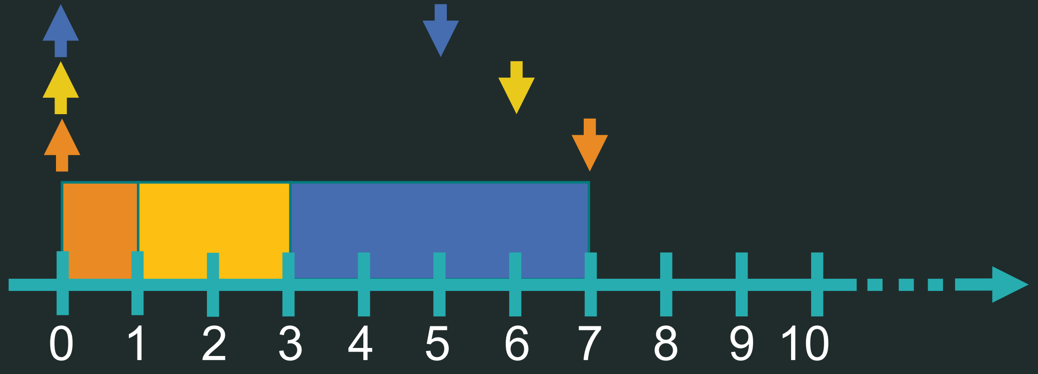
Shortest Job First (SJF)

The CPU is assigned to tasks with the shortest computation time

Assigning priorities based on SJF has some benefits but it is not well suited to ensure timing constraints compared with other schedulers

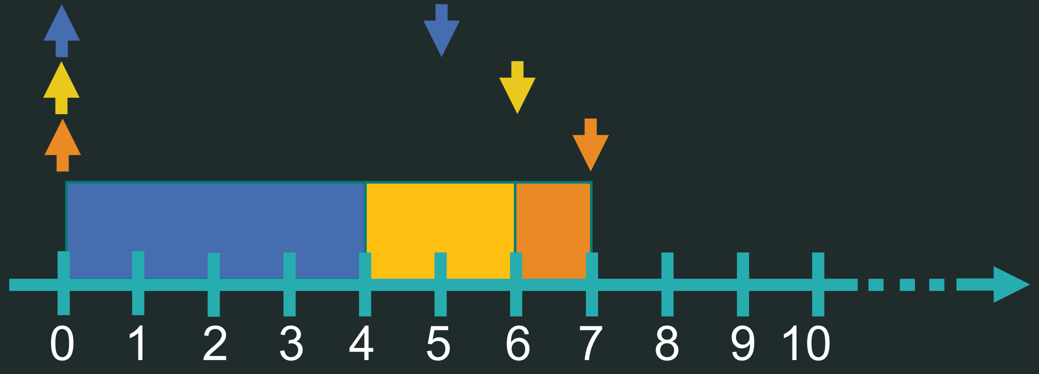
Task τ_i	Computing time c_i (ms)	Deadline D_i (ms)	SJF Priority	EDF Priority
τ_1	1	7	3	1
τ_2	2	5	2	2
τ_3	4	3	1	3

Deadline missed due to SJF priority assignation!!!



SJF scheduling

No deadline missed with EDF priority assignation

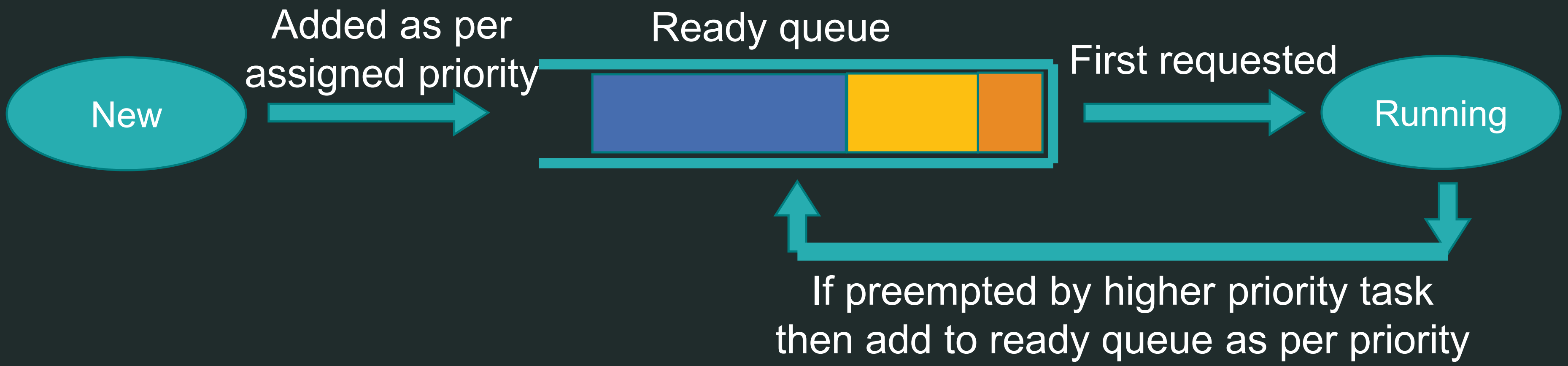


EDF scheduling

Priority scheduling (PS)

The CPU is assigned to tasks based on heuristic priorities

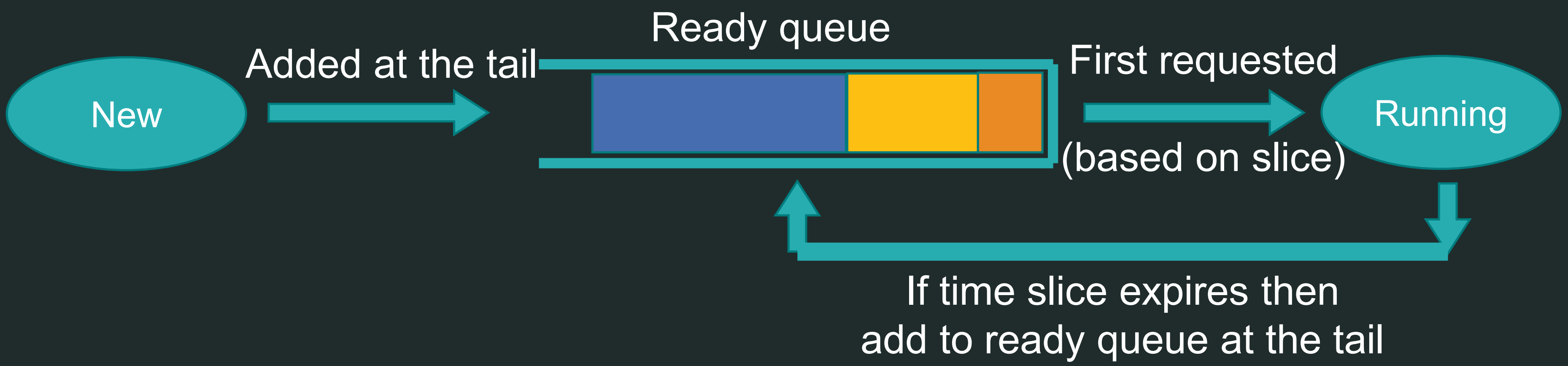
It is: preemptive, static or dynamic, online, may be or not optimal, depends on how priorities are assigned (*aging* increases the priority of a task along waiting time)



Starvation is a typical problem of badly assigned priorities in which low priority tasks may experience long delays due to the preemption of high priority tasks

Round Robin (RR)

The CPU is assigned based on First Come First Served but each task cannot execute more than a time quantum. When time expires, task is put back in queue.
It is: preemptive, static or dynamic, online or offline.



Some schedulers witch require improving the user experience are based on RR.

Schedulers examples

Automotive and avionics: Mostly cyclic schedulers. *Test what you fly, fly what you test!*

Windows, macOs: Multilevel feedback queue (preference to short jobs, IO processes and high processor demand). Time quantum implemented in different queues

Linux: Round Robin + Completely Fair Scheduler (aims to maximize overall CPU utilization while also maximizing interactive performance)

RT-kernels: Fixed priorities for hard RT tasks, non-RT schedulers for soft RT tasks or background tasks

When to use a RT Operating System?

- When application complexity increases. <64KB (do not use RTOS)
- RTOS is typically a third-party software, so lost of control over the code may exists
- Preemptive multitasking design paradigm to deal with complexity
- Pre-tested and pre-integrated communications stacks and drivers provided by RTOS
- Application portability
- System-level debug and analysis tools
- More efficient use of CPU resources

RT vs non-RT schedulers

RT schedulers:

They are focused at guaranteeing the response time of hard real time tasks

Rate Monotonic (RM), Deadline Monotonic (DM) and Earliest Deadline First (EDF) are examples of RT-schedulers

non-RT schedulers:

They are focused at guaranteeing some other criterion, typically not related to hard timing constraints

First Come First Served (FCFS), Shortest Job First (SJF) and Round Robin (RR) are examples of non-RT schedulers

Most RT-kernels combine both types of schedulers to deal with different situations, as for example FreeRTOS which uses RM to select the highest priority running task, and RR for time slicing within tasks of the same priority