ECE 3140 / CS 3420
EMBEDDED SYSTEMS
LECTURE 15

Prof. José F. Martínez

TR 1:25-2:40pm in 150 Olin

Real Time Systems

Terminology:

- A *job* is a sequence of operations that, in the absence of any other activities, is executed by the processor
- A *task* is a sequence (possibly infinite) of jobs
- Jobs have:
  - A request time r_i (arrival time)
  - A start time s_i
  - A finishing time f_i
  - An absolute deadline d_i

Scheduling Algorithms

- Preemptive or nonpreemptive
- Static or dynamic: are the scheduling decisions based on parameters that change with time?
- Online or offline: are the decisions made apriori with knowledge of task activations, or are they taken at run time based on the set of active tasks?
- Optimal or heuristic: can you prove that the algorithm optimizes a certain criteria or not?

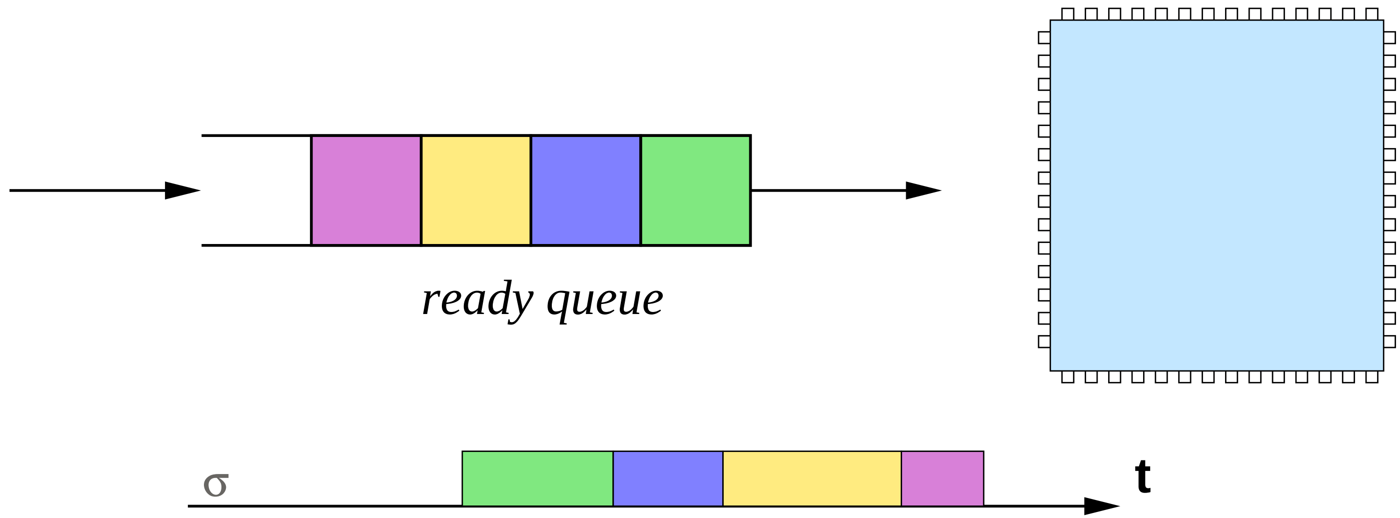
Optimality

Examples:

- Find a feasible schedule if one exists, and:
  - Minimize the maximum lateness
  - Minimize the number of missed deadlines
- Assign a utility value to each task, and maximize the value of the feasible tasks

Classic Scheduling Policies

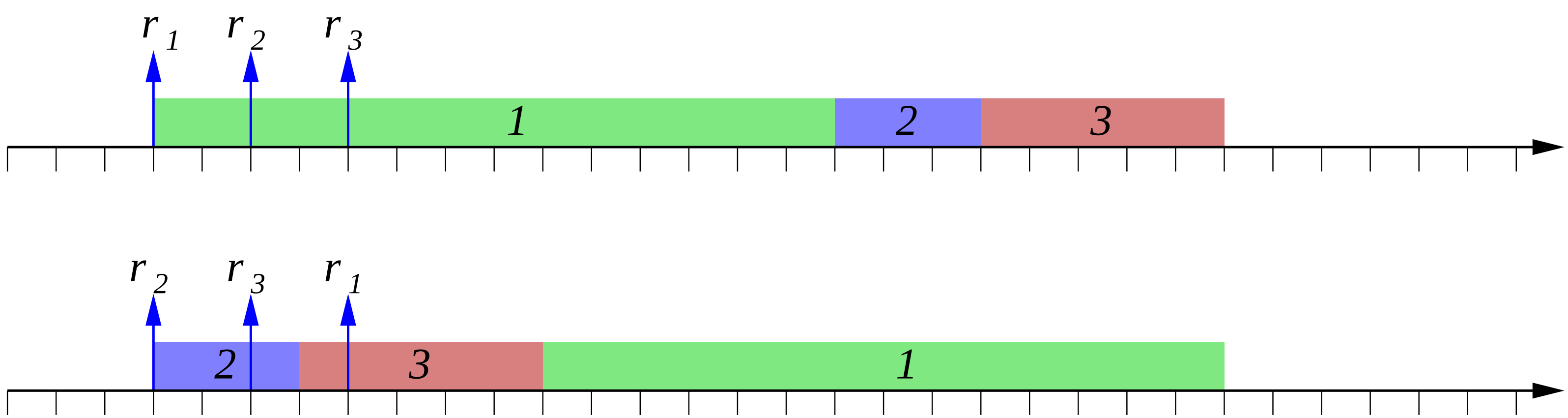
First Come First Served (FCFS):



- Non-preemptive
- Dynamic
- Online
- Heuristic

First Come First Served

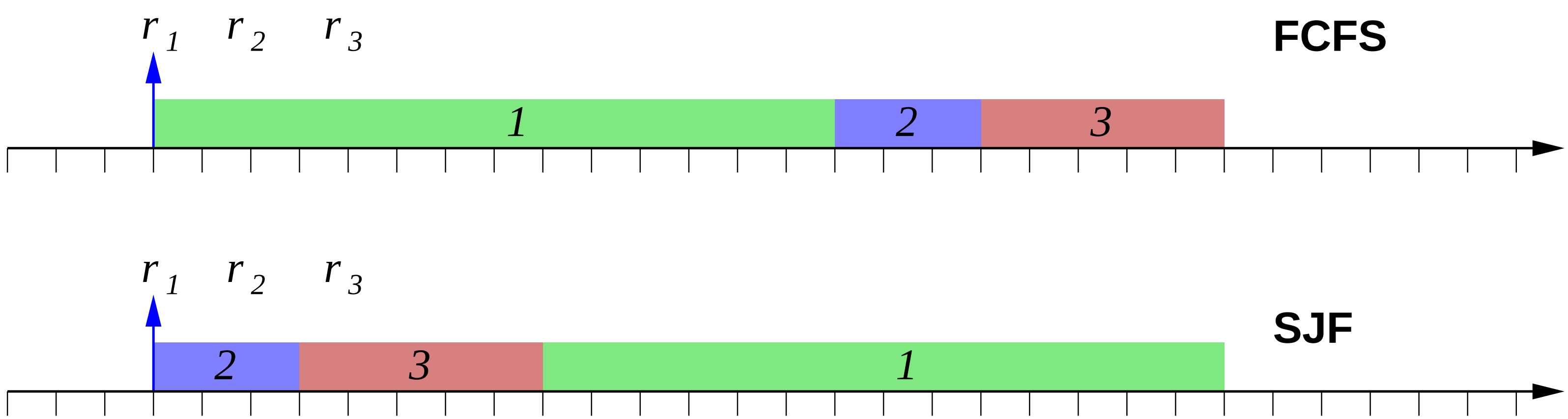
- Very unpredictable: response time depends strongly on task arrivals (response time: $f - r$)



$\Rightarrow$ *not suitable for real-time systems*

SHORTEST JOB FIRST

Shortest Job First (SJF) Policy: pick the task with the shortest computation time



- Non preemptive or preemptive
- Static (c_i is known and fixed)
- Online or offline
- *It minimizes the average response time*

Shortest Job First

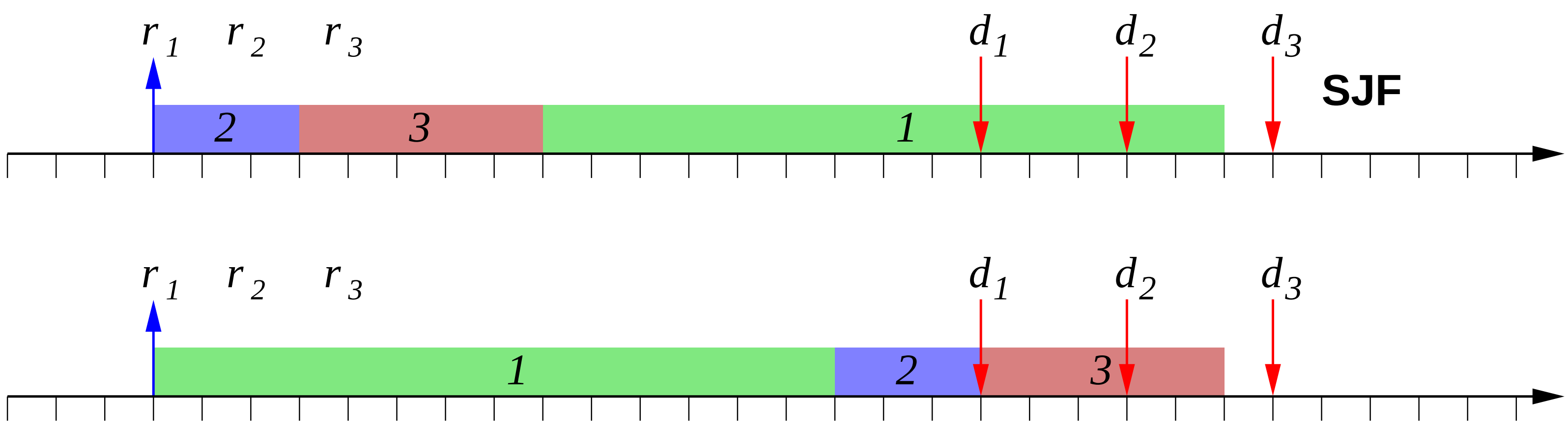
Why does it minimize the average response time? Proof?

- In particular, if $\overline{R}(\sigma)$ is the average response time of a schedule, then:

$$\forall\sigma : \qquad \overline{R}(\sigma_{SJF}) \leq \overline{R}(\sigma)$$

Shortest Job First

What about real-time constraints?



Not suitable for real-time in the sense of feasibility!

Priority Scheduling

- Each task is assigned a priority
  - Example: $p_i \in [0, 255]$ (one byte to store priority)
- Task with the highest priority is selected first
- Tasks with the same priority are scheduled using FCFS

Priority scheduling is:

- Preemptive
- Static or dynamic (if priorities change)
- Online

PRIORITY SCHEDULING

Some issues that have to be considered:

- Starvation:
  Low priority tasks may experience very long delays due to preemption by higher priority tasks

Common approach used:

- Aging:
  Priority increases with waiting time

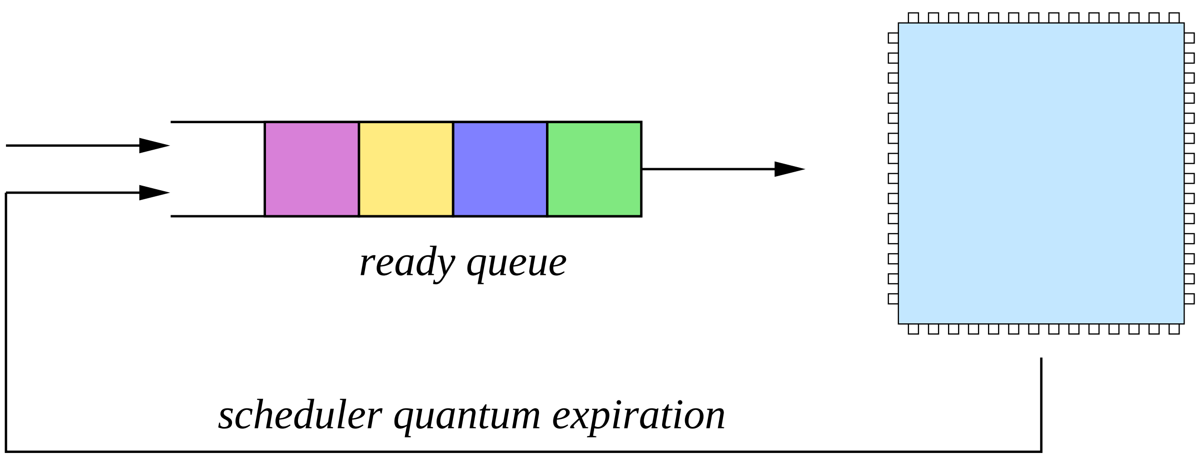
Note that:

- If $p_i \propto 1/c_i$: shortest job first!
- If $p_i = const$: first come first served!

Round Robin Scheduling

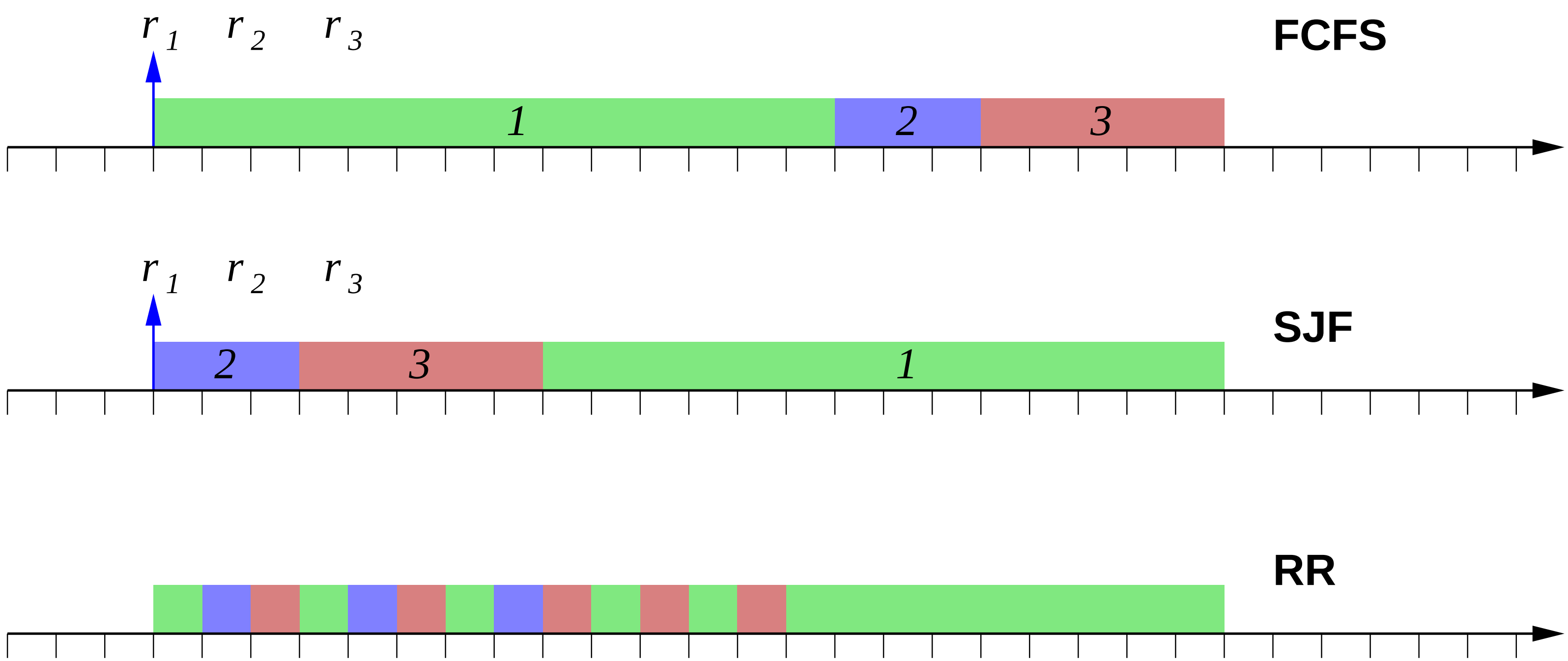
Round Robin (RR):

- The ready queue is FCFS
- However ...
  - Each task cannot execute more than Q time units (the *quantum*)
  - When Q time units have elapsed, the task is put back into the ready queue



Round Robin Scheduling

Example:



Round Robin Scheduling

If there are n tasks in the system:

- Each repeating sequence in the schedule is nQ in length
- In each repeating sequence, a task gets Q units of time
- Suppose context switch time $\approx \delta$

Hence,

$$R_i = f_i - r_i \approx n(Q+\delta)\frac{C_i}{Q} = nC_i\left(1+\frac{\delta}{Q}\right)$$

ROUND ROBIN SCHEDULING

- For very small Q:
  - Each task runs as if it were executing on a *virtual* processor that is n times slower than the real one
- If Q is very large then RR $\equiv$ FCFS

  $\forall i : Q \geq C_i$