Processes: Scheduling

Every process on a computer is, at any given time, in one of three given states:

1. WAITING: The process is waiting for CPU (Central Processing Unit) time to become available.
2. RUNNING: The process is executing on the CPU.
3. BLOCKED: The process is waiting for some other event, e.g. access to a file, input from a user, etc.

The possible transitions are between these states are:

- WAITING $\longrightarrow$ RUNNING
- RUNNING $\longrightarrow$ WAITING
- BLOCKED $\longrightarrow$ WAITING
- RUNNING $\longrightarrow$ BLOCKED

Processes: Scheduling

Given a number of different processes in the WAITING state, the OS (operating system) much decide which to run, in what sequence: This is *scheduling*. A scheduling policy may be preemptive (where the execution of the process can be interrupted without the process having finished) or non-preemptive.

Scheduling criteria

- **Maximize CPU usage:** Ideally use 100% of CPU cycles. In practice, somewhere from 40 % to 90 %.
- **Maximize throughput:** The throughput is the number of processes completed per unit time.
- **Minimize turnaround time:** The turnaround time is the total time for a process to complete.
- **Minimize waiting time:** This is the time a process spends in the WAITING state.

Processes: Scheduling

Note that

- In general we want to optimize the *average* value of the above parameters, taken over many processes.
- Sometimes we are more interested in minimizing the *variance* rather than maximizing/minimizing the parameter: We may prefer a system that is more predictable but not "the best" over one which has the best average value, but is more unpredictable.

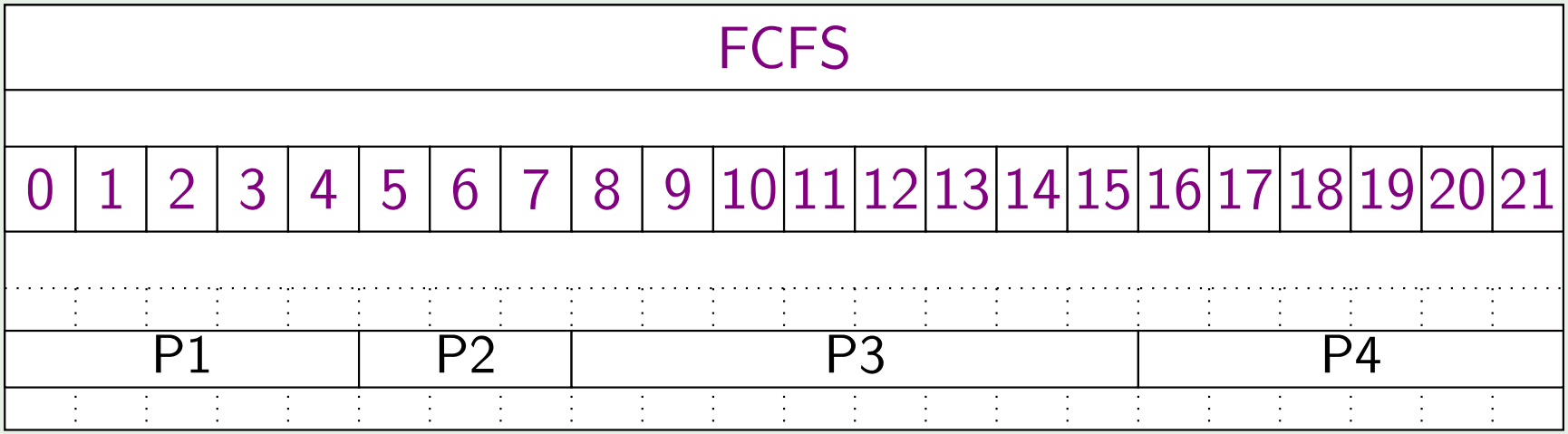
Processes: Scheduling

We consider in what follows the statistics and distributions produced by a number of scheduling policies, and comment on how well they optimize the criteria above. Our analysis looks at four processes:

Process	Arrival time	Process duration
P1	0	5
P2	1	3
P3	2	8
P4	3	6

First Come First Served (FCFS)

This is non-preemptive: The CPU takes processes in the order in which they arrive, and each process executes until termination. (Operates as in the FIFO (First In, First Out) process in a queue.)

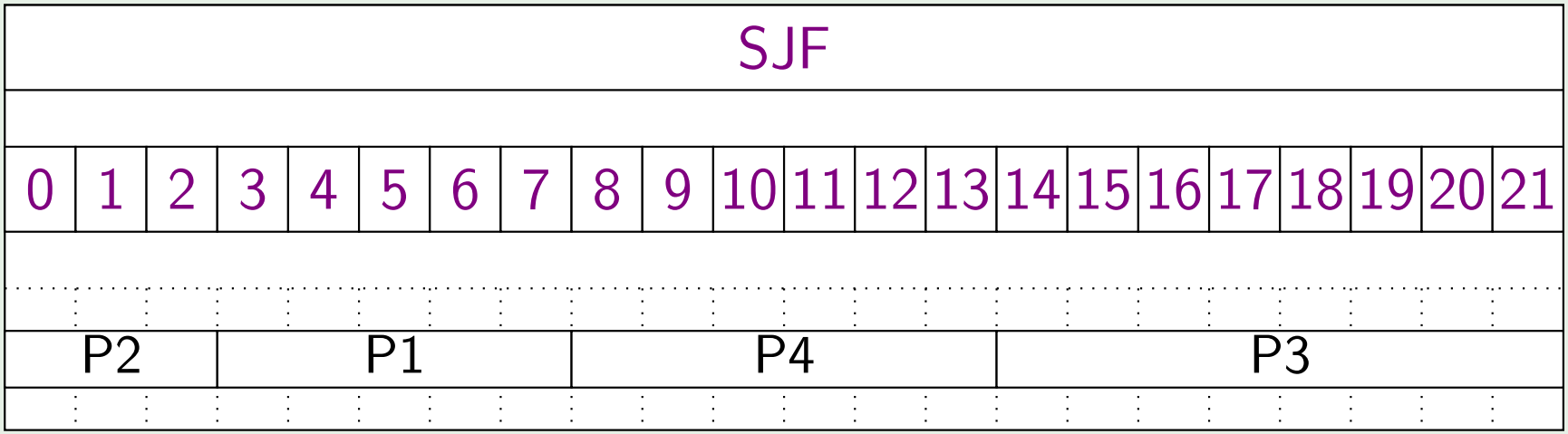


with performance metrics

METRIC:	CPU Usage	Turnaround time	Throughput	Waiting time
	$22/(22+3s)$	$(5 + 7 + 14 + 19)/4 = 11.25$	$4/(22+3s)$	$(0 + 4 + 6 + 13)/4 = 5.75$

Shortest Job First (SJF)

This is non-preemptive: The CPU takes the shortest process first, then the next shortest, etc.



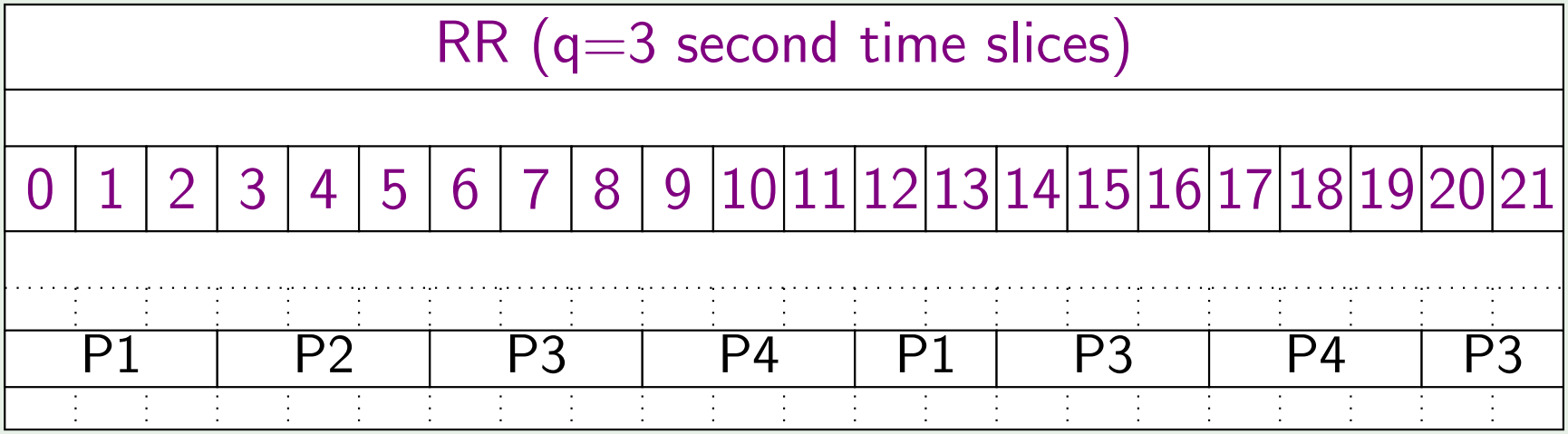
with performance metrics

METRIC:	CPU Usage	Turnaround time	Throughput	Waiting time
	$22/(22+3s)$	$(8 + 2 + 20 + 11)/4 = 10.25$	$4/(22+3s)$	$(3 + (-1) + 12 + 5)/4 = 4.75$

So SJF beats FCFS, can be shown that it has optimum minimum average Waiting times.

Round Robin (RR)

This is preemptive: The CPU allows a fixed time interval (quantum) to each process in turn. When that quantum is used, the current process goes to the back of the queue.

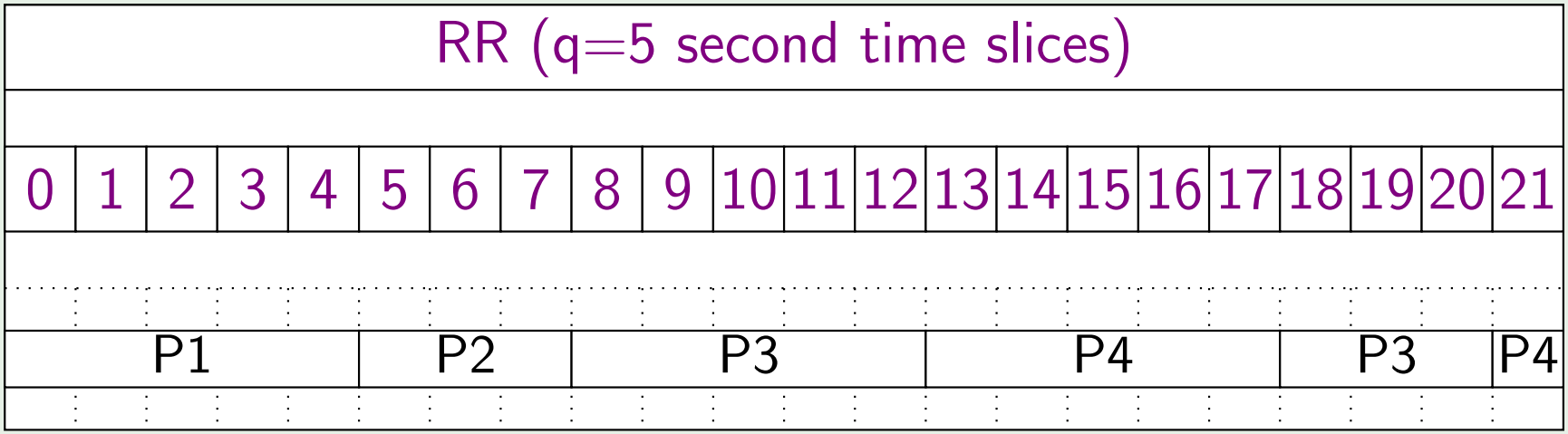


with performance metrics

METRIC:	CPU Usage	Turnaround time	Throughput	Waiting time
	$22/(22+7s)$	(14 + 5 + 20 + 17)/4= 14	4/(22+7s)	$(0+2+4+6+9+5+5+3)/4 = 8.5$

Note that for large time slices, RR degenerates into FCFS. For small time slices, too much time is lost swapping processes in and out.....

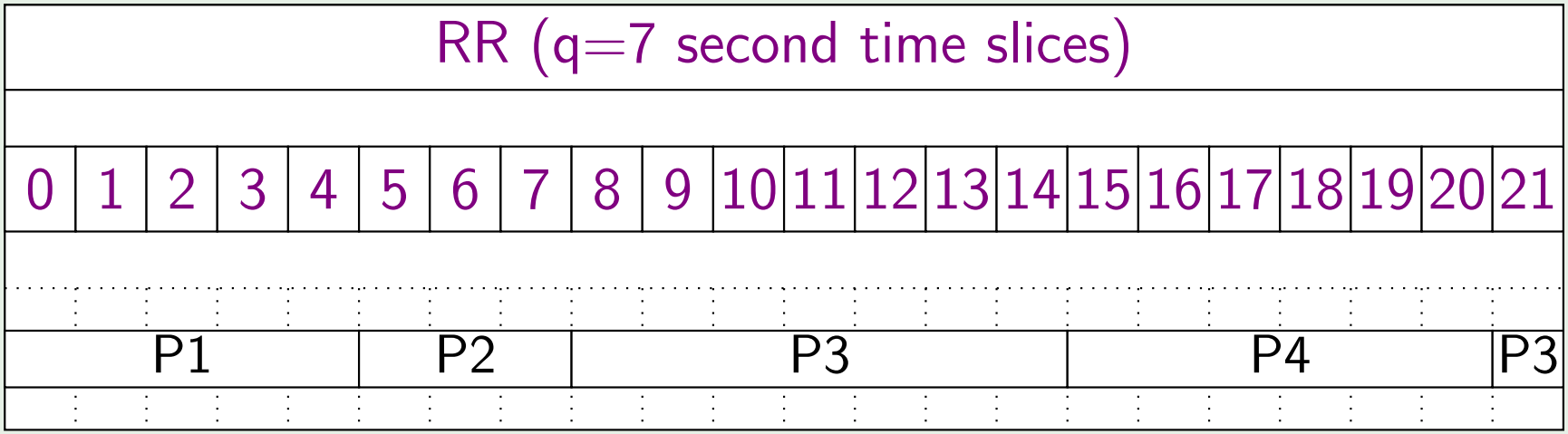
Round Robin (RR)



with performance metrics

METRIC:	CPU Usage	Turnaround time	Throughput	Waiting time
	$22/(22+5s)$	$(5 + 7 + 19 + 19)/4 = 12.25$	4/(22+5s)	$(0 + 4 + 6 + 10 + 5 + 3)/4 = 7$

Round Robin (RR)

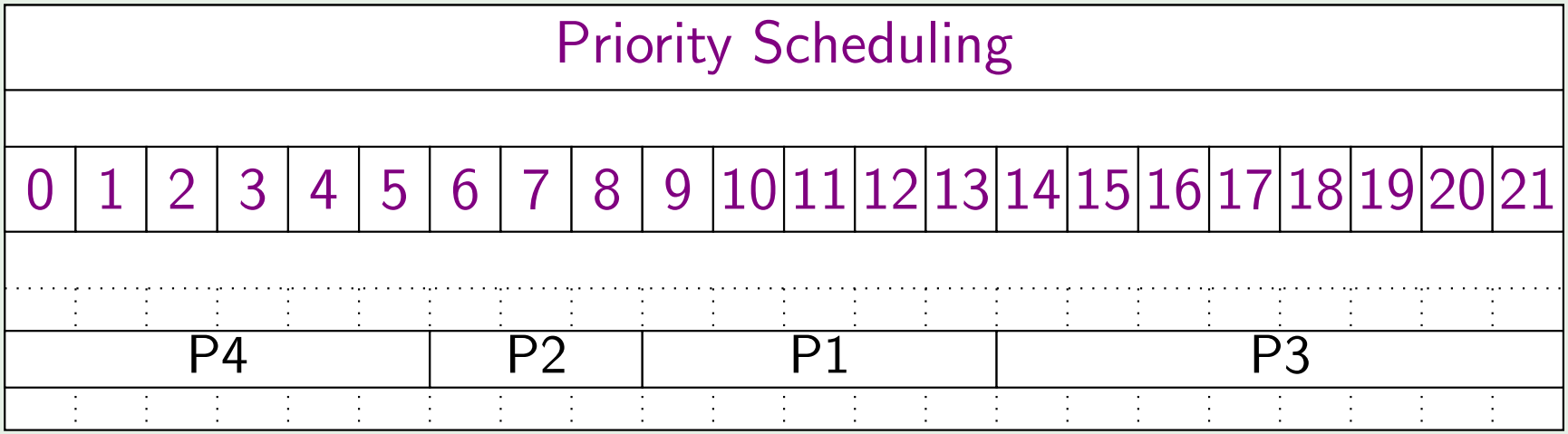


with performance metrics

METRIC:	CPU Usage	Turnaround time	Throughput	Waiting time
	$22/(22+4s)$	(5 + 7 + 20 + 18)/4= 12.25	4/(22+4s)	$(0+4+6+12+6)/4 = 7$

Priority Scheduling

Each process comes with an assigned priority, and higher priority processes run first. For processes with equal priority, use FCFS. Let us suppose in our example our four processes have priorities $(P1, P2, P3, P4)$ of $1, 2, 1, 3$ respectively.

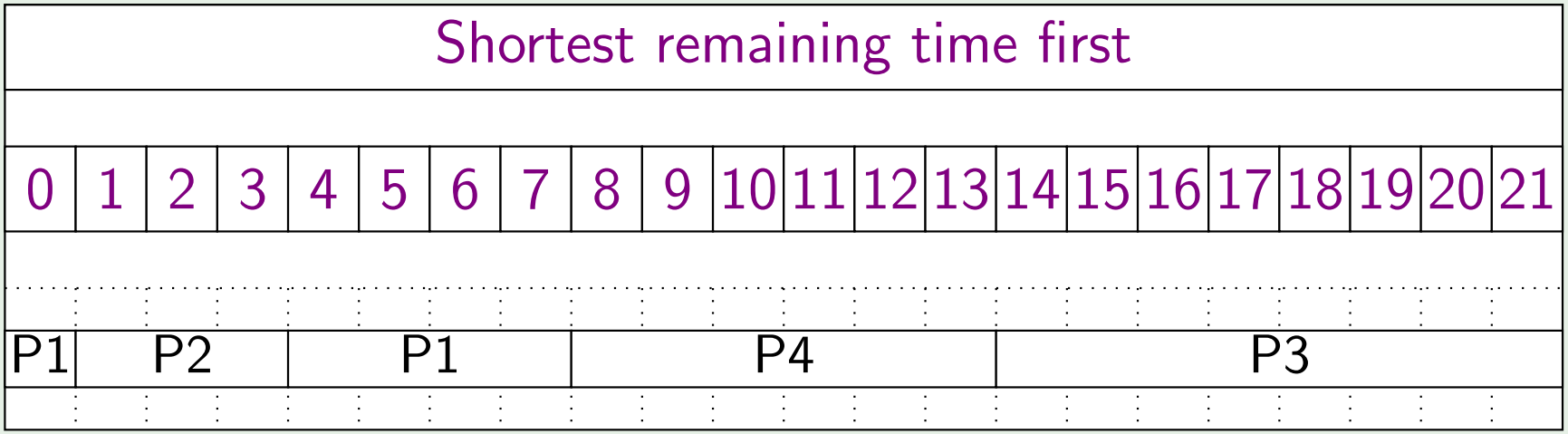


with performance metrics

METRIC:	CPU Usage	Turnaround time	Throughput	Waiting time
	$22/(22+3s)$	$(14 + 8 + 20 + 3)/4 = 11.25$	4/(22+3s)	$(8+5+12+0)/4 = 6.25$

Shortest remaining time first (SRTF)

Preemptive scheduling where the process with shortest time-to-completion is assigned to the CPU



with performance metrics

METRIC:	CPU Usage	Turnaround time	Throughput	Waiting time
	$22/(22+4s)$	(8 + 3 + 20 + 11)/4= 10.5	4/(22+4s)	$(3+0+12+5)/4 = 5$