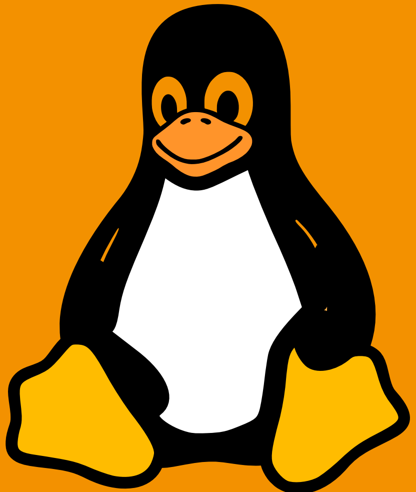
Start: 8:01

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Prof. Dr. Florian Künzner
Technical University of Applied Sciences Rosenheim, Computer Science

OS 11 – Scheduling



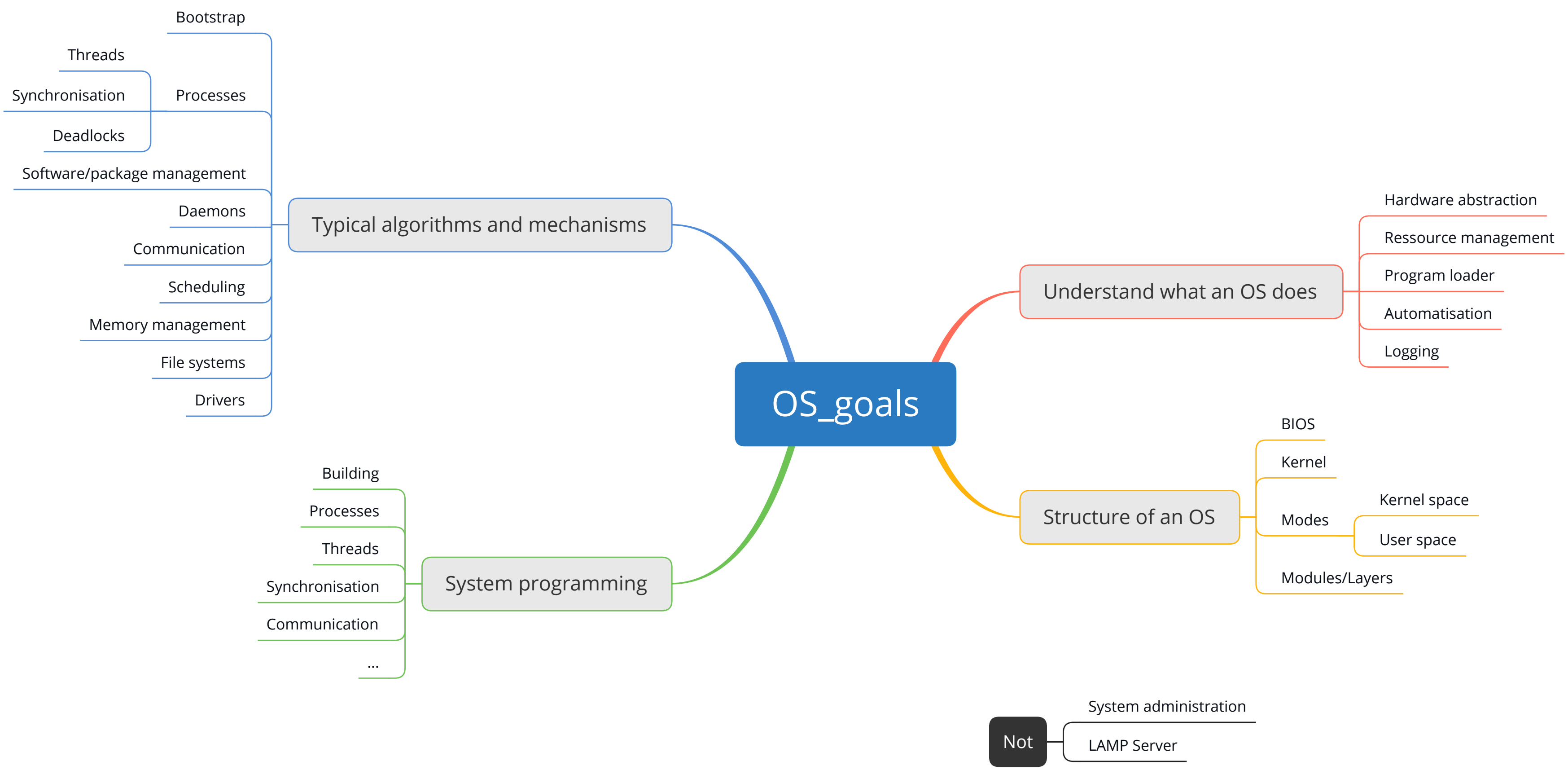
source: iconspng.com

The lecture is based on the work and the documents of Prof. Dr. Ludwig Frank

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Goal



CAMPUS Rosenheim
Computer Science

Goal

OS::Scheduling

- Scheduling theory and terms
- Scheduling strategies
- Scheduling on Linux

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Scheduling

Scheduling is a technique to **distribute computing resources** like processor time, bandwidth, memory, or device I/O **to the processes** on a computer system.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Scheduler

The **scheduler** is such a **program** that **distributes** the **resources** to the processes.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Scheduling time frame

Long term scheduling

Determines which processes (jobs) are admitted to the system for processing.

Mid term scheduling

Allocation of bandwidth, memory, or device I/O to a process/thread.

Short term scheduling

Allocation of a process/thread to a CPU core.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Scheduling time frame

Long term scheduling

Determines which processes (jobs) are admitted to the system for processing.

Mid term scheduling

Allocation of bandwidth, memory, or device I/O to a process/thread.

Short term scheduling

Allocation of a process/thread to a CPU core.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Scheduling time frame

Long term scheduling

Determines which processes (jobs) are admitted to the system for processing.

Mid term scheduling

Allocation of bandwidth, memory, or device I/O to a process/thread.

Short term scheduling

Allocation of a process/thread to a CPU core.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Scheduling time frame

Long term scheduling

Determines which processes (jobs) are admitted to the system for processing.

Mid term scheduling

Allocation of bandwidth, memory, or device I/O to a process/thread.

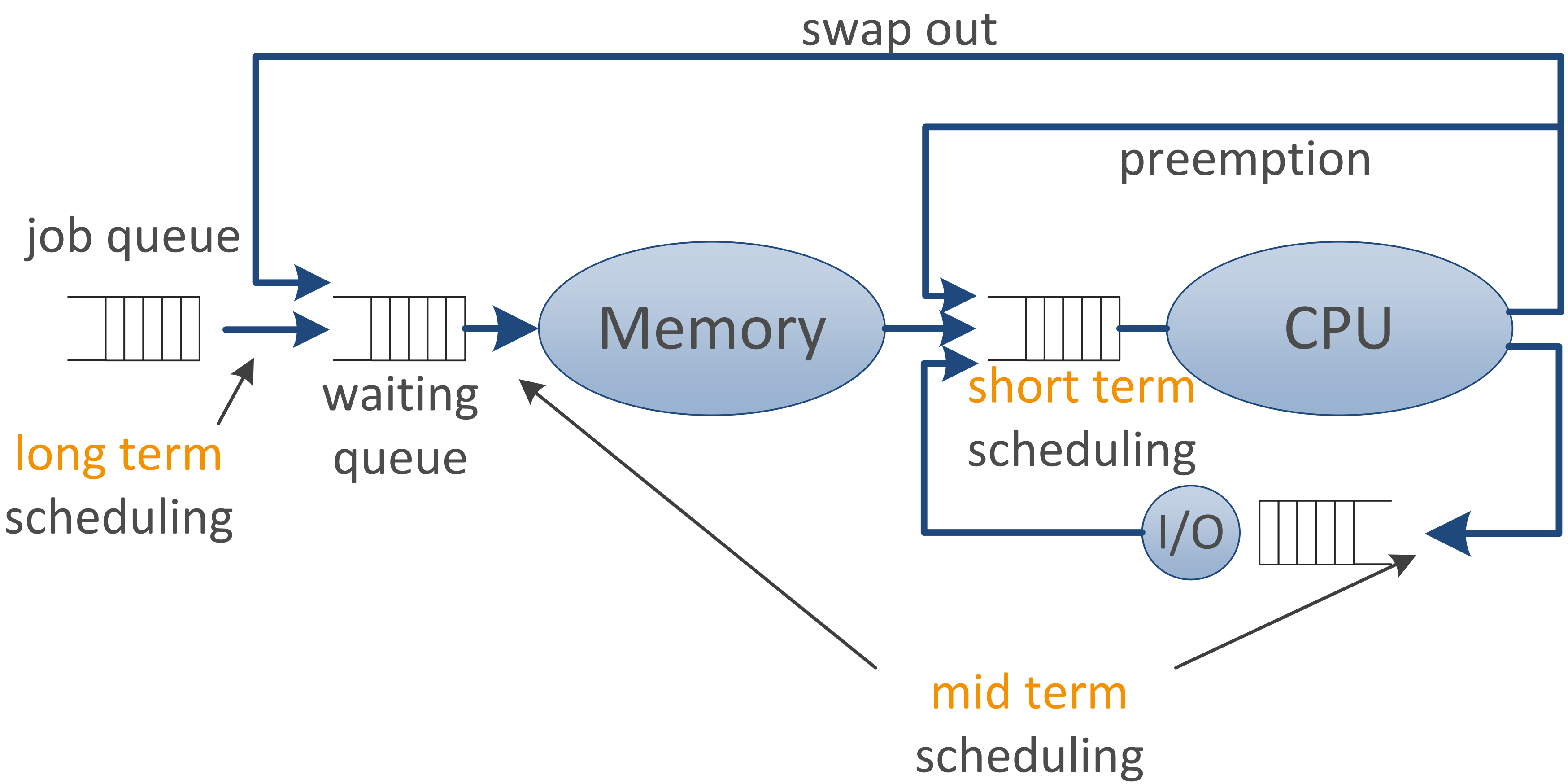
Short term scheduling

Allocation of a process/thread to a CPU core.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Scheduling time frame overview

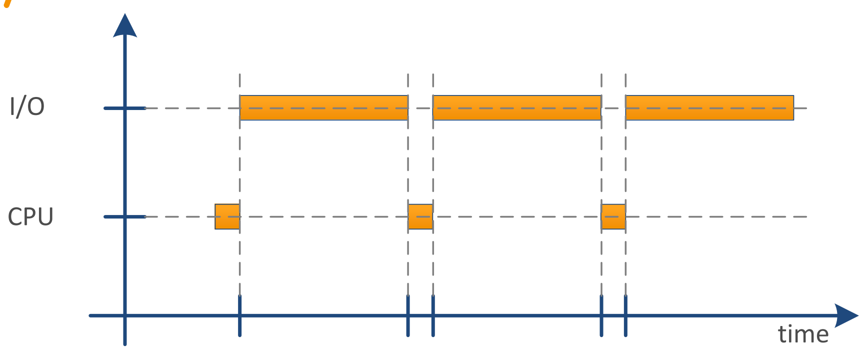


CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

I/O-bound vs CPU-bound

I/O-bound

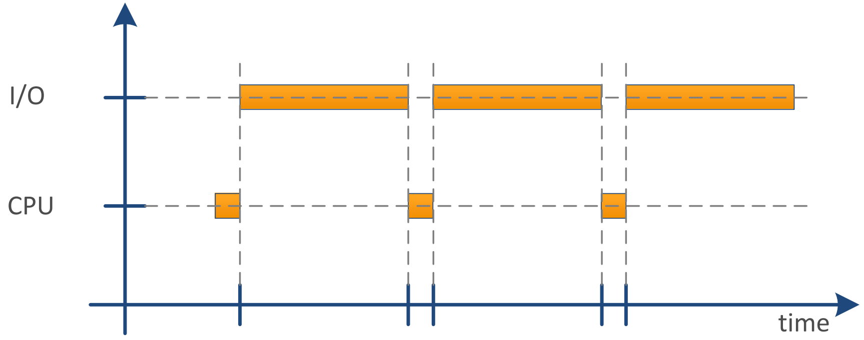


CPU-bound

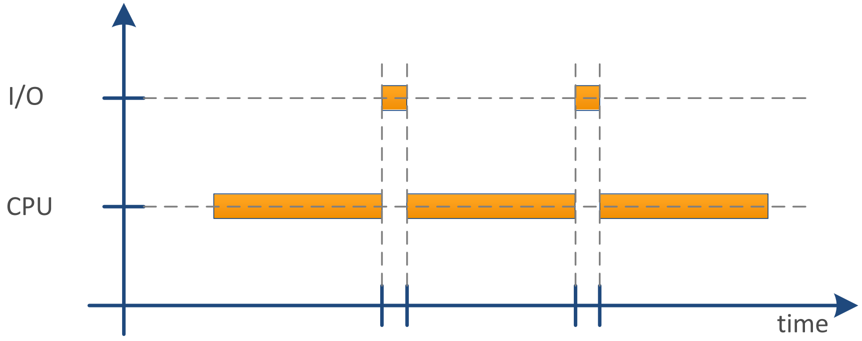
CAMPUS Rosenheim
Computer Science

I/O-bound vs CPU-bound

I/O-bound



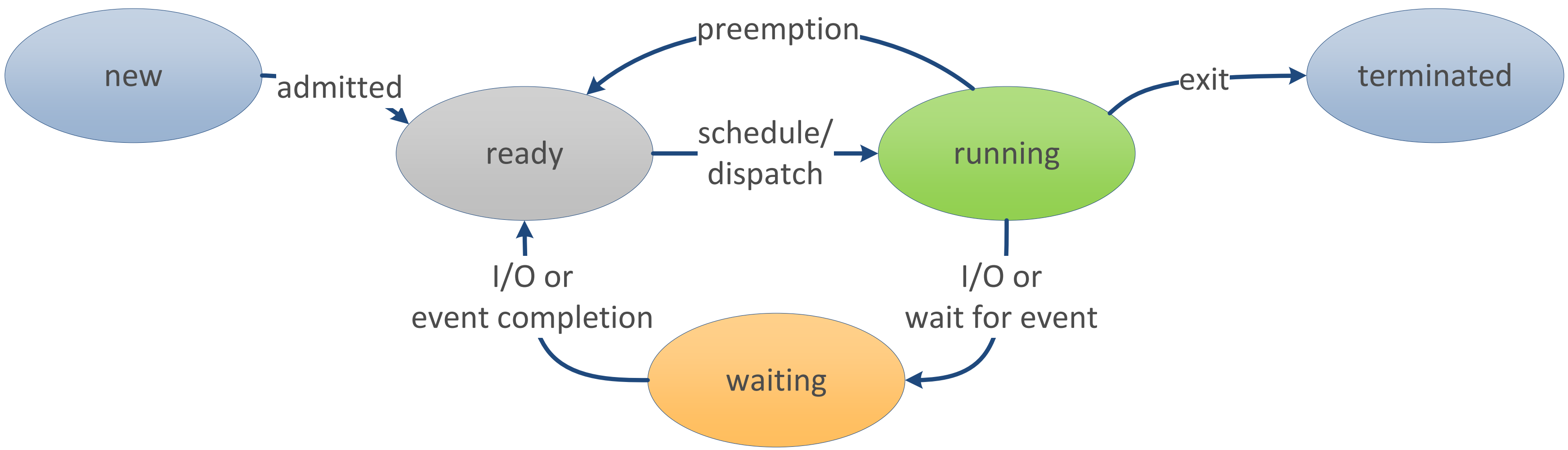
CPU-bound



CAMPUS Rosenheim
Computer Science

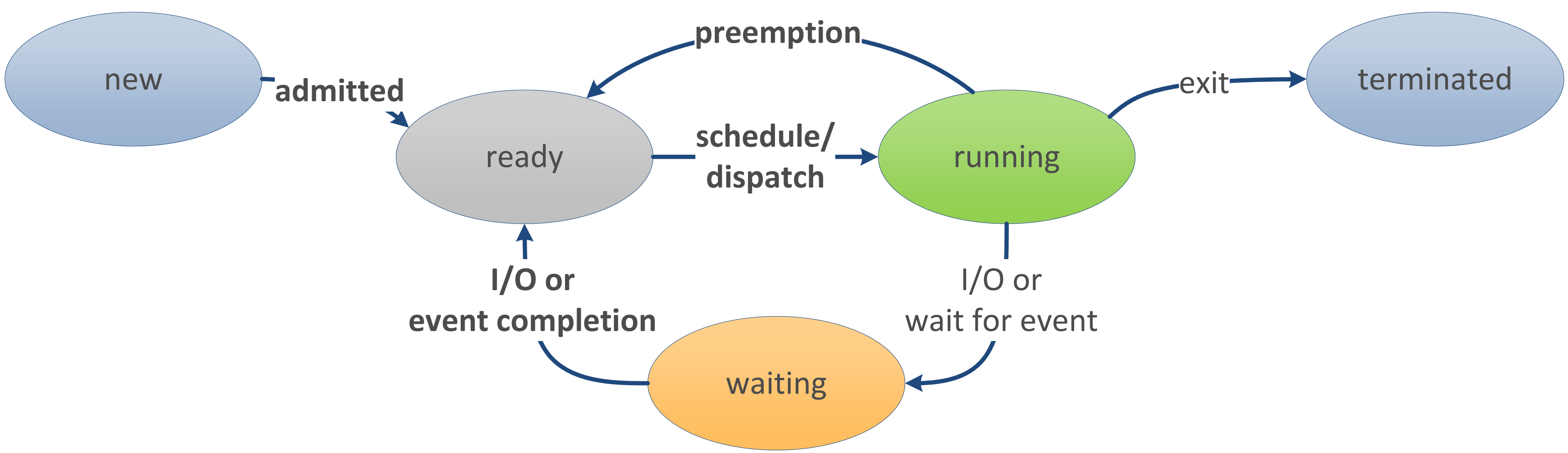
Technische Hochschule Rosenheim
Technical University of Applied Sciences

Process states



CAMPUS Rosenheim
Computer Science

Process states



CAMPUS Rosenheim
Computer Science

Scheduling terms

Term	Description
CPU usage	CPU usage up to 100% if possible.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Scheduling terms

Term	Description
CPU usage	CPU usage up to 100% if possible.
throughput	The number of completed processes in a time frame. Should be as high as possible.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Scheduling terms

Term	Description
CPU usage	CPU usage up to 100% if possible.
throughput	The number of completed processes in a time frame. Should be as high as possible.
arrival time	The point in time at which a process arrives for execution in the system.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Scheduling terms

Term	Description
CPU usage	CPU usage up to 100% if possible.
throughput	The number of completed processes in a time frame. Should be as high as possible.
arrival time	The point in time at which a process arrives for execution in the system.
processing time	The time a process takes to run on the CPU.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Scheduling terms

Term	Description
CPU usage	CPU usage up to 100% if possible.
throughput	The number of completed processes in a time frame. Should be as high as possible.
arrival time	The point in time at which a process arrives for execution in the system.
processing time	The time a process takes to run on the CPU.
waiting time	The time a process has to wait until it can run.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

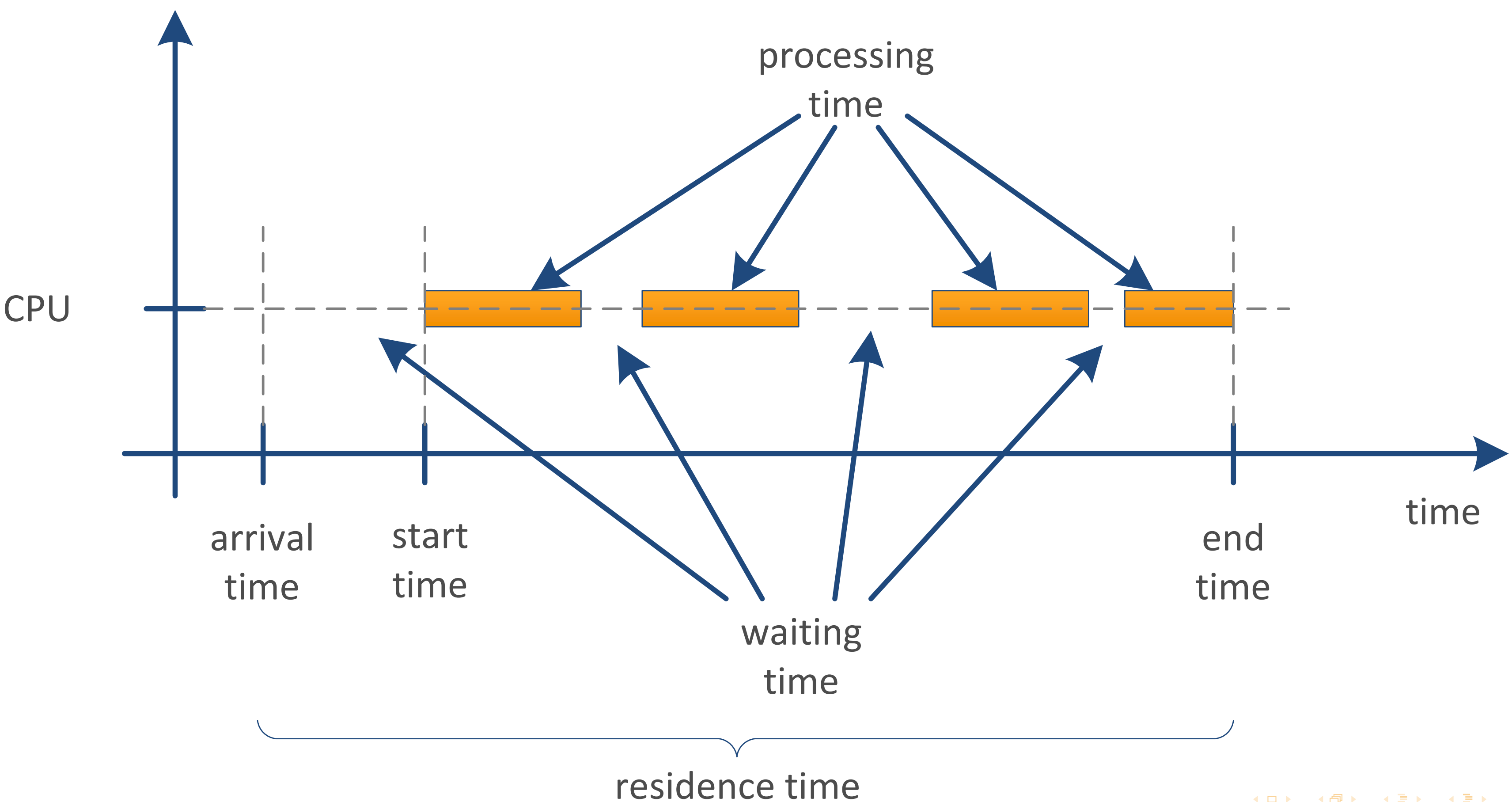
Scheduling terms

Term	Description
CPU usage	CPU usage up to 100% if possible.
throughput	The number of completed processes in a time frame. Should be as high as possible.
arrival time	The point in time at which a process arrives for execution in the system.
processing time	The time a process takes to run on the CPU.
waiting time	The time a process has to wait until it can run.
residence time	The total time a process takes to finish (= processing time + waiting time).

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Time aspects



CAMPUS Rosenheim
Computer Science

System categories

System type	Description

System categories

System type	Description
Job processing system	Typically users submit jobs (programs with its parameters) to a system. The job scheduler decides when the job starts.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

System categories

System type	Description
Job processing system	Typically users submit jobs (programs with its parameters) to a system. The job scheduler decides when the job starts.
Interactive system	With an interactive system, the users work directly: a PC (terminal, desktop), smartphone, ...

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

System categories

System type	Description
Job processing system	Typically users submit jobs (programs with its parameters) to a system. The job scheduler decides when the job starts.
Interactive system	With an interactive system, the users work directly: a PC (terminal, desktop), smartphone, ...
Real-time system	A real-time system usually observes and controls a physical process in the real world. It must guarantee that it reacts fast enough.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Scheduling Goals

All systems

fairness	-	Every process can run on the CPU.
policy enforcement	-	The system's policy is enforced.
balance	-	All parts of the system are busy.

Job processing system

throughput	-	Maximize the number of jobs in a time frame.
residence time	-	Minimize the residence time for each job.
CPU usage	-	The CPU is constantly used as long as there are jobs in the queue.

Interactive system

response time	-	Respond quickly to requests.
proportionality	-	Consider the requirements for all users.

Real-time system

meet deadline	-	Meet always the deadline of all processes.
predictability	-	Always guarantee the same periodic execution (small jitter).

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Scheduling Goals

All systems

fairness	-	Every process can run on the CPU.
policy enforcement	-	The system's policy is enforced.
balance	-	All parts of the system are busy.

Job processing system

throughput	-	Maximize the number of jobs in a time frame.
residence time	-	Minimize the residence time for each job.
CPU usage	-	The CPU is constantly used as long as there are jobs in the queue.

Interactive system

response time	-	Respond quickly to requests.
proportionality	-	Consider the requirements for all users.

Real-time system

meet deadline	-	Meet always the deadline of all processes.
predictability	-	Always guarantee the same periodic execution (small jitter).

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Scheduling Goals

All systems

fairness	-	Every process can run on the CPU.
policy enforcement	-	The system's policy is enforced.
balance	-	All parts of the system are busy.

Job processing system

throughput	-	Maximize the number of jobs in a time frame.
residence time	-	Minimize the residence time for each job.
CPU usage	-	The CPU is constantly used as long as there are jobs in the queue.

Interactive system

response time	-	Respond quickly to requests.
proportionality	-	Consider the requirements for all users.

Real-time system

meet deadline	-	Meet always the deadline of all processes.
predictability	-	Always guarantee the same periodic execution (small jitter).

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Scheduling Goals

All systems

fairness	-	Every process can run on the CPU.
policy enforcement	-	The system's policy is enforced.
balance	-	All parts of the system are busy.

Job processing system

throughput	-	Maximize the number of jobs in a time frame.
residence time	-	Minimize the residence time for each job.
CPU usage	-	The CPU is constantly used as long as there are jobs in the queue.

Interactive system

response time	-	Respond quickly to requests.
proportionality	-	Consider the requirements for all users.

Real-time system

meet deadline	-	Meet always the deadline of all processes.
predictability	-	Always guarantee the same periodic execution (small jitter).

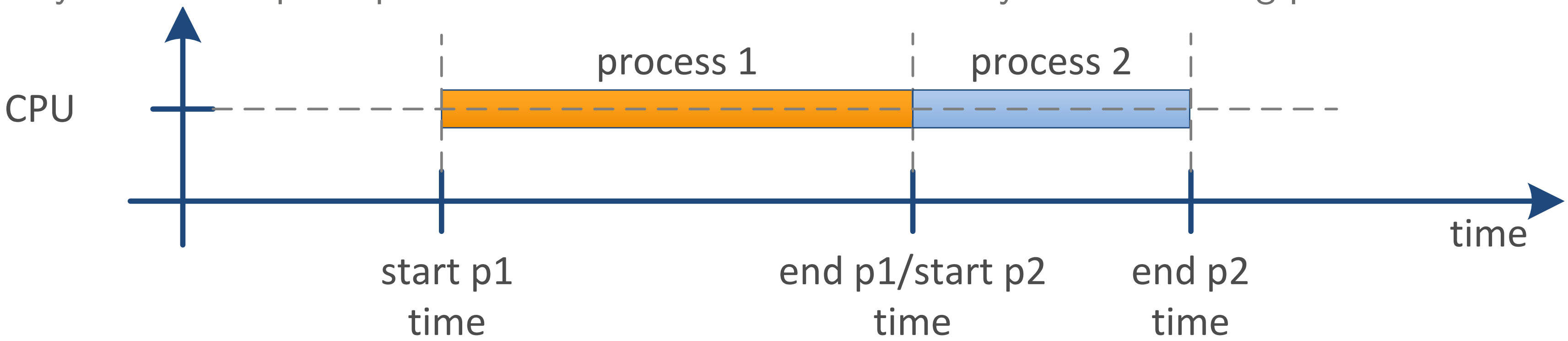
CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Preemption

Non-preemptive system

A system is non-preemptive if the CPU cannot be taken away from a running process.



Preemptive system

A system is preemptive if the CPU can be taken away from a running process.

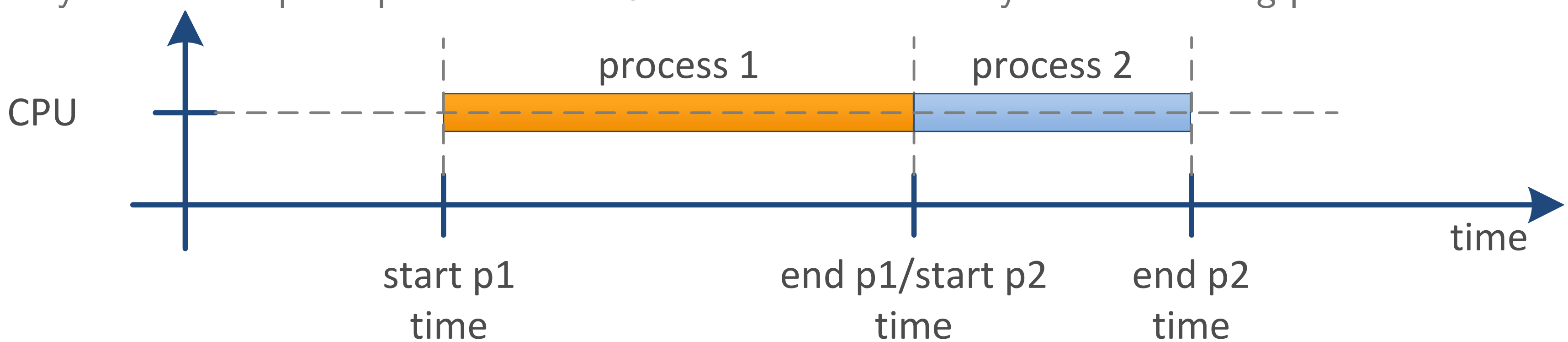
CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Preemption

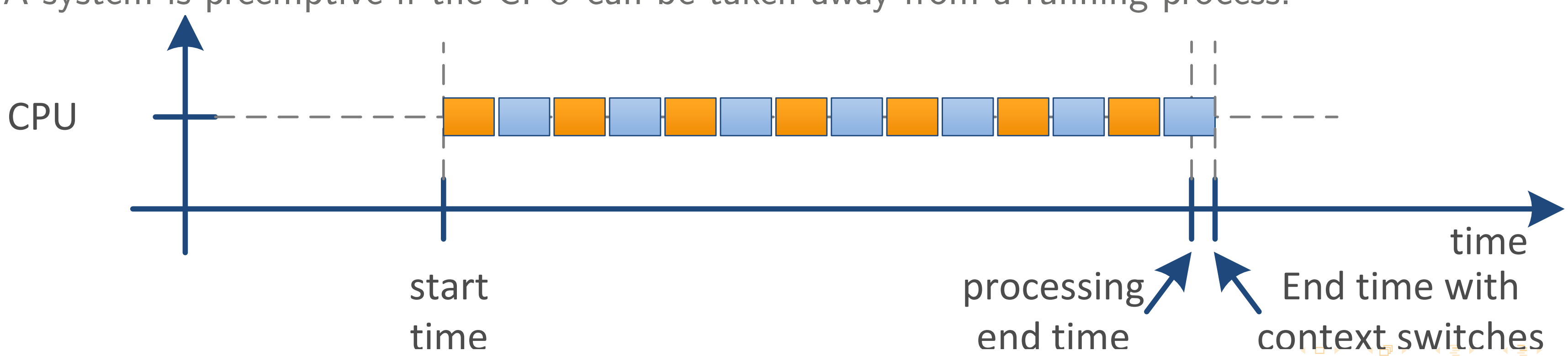
Non-preemptive system

A system is non-preemptive if the CPU cannot be taken away from a running process.



Preemptive system

A system is preemptive if the CPU can be taken away from a running process.



CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Context switch

A context switch **changes the active process or thread** on the CPU. This may be expensive (takes some time).

Procedure:

Time for context switches:

Source for measured times: https://blog.tsunanet.net/2010/11/how-long-does-it-take-to-make-context.html

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Context switch

A context switch **changes the active process or thread** on the CPU. This may be expensive (takes some time).

Procedure:

- Save the register content of the currently running process into its PCB.
- Select a new process to run.
- Load/restore memory information into the CPU (MMU) from its PCB.
- Load/restore the register content of the new process from its PCB.

Time for context switches:

Source for measured times: https://blog.tsunanet.net/2010/11/how-long-does-it-take-to-make-context.html

CAMPUS Rosenheim
Computer Science

Context switch

A context switch **changes the active process or thread** on the CPU. This may be expensive (takes some time).

Procedure:

- Save the register content of the currently running process into its PCB.
- Select a new process to run.
- Load/restore memory information into the CPU (MMU) from its PCB.
- Load/restore the register content of the new process from its PCB.

Time for context switches:

Source for measured times: https://blog.tsunanet.net/2010/11/how-long-does-it-take-to-make-context.html

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Context switch

A context switch **changes the active process or thread** on the CPU. This may be expensive (takes some time).

Procedure:

- Save the register content of the currently running process into its PCB.
- Select a new process to run.
- Load/restore memory information into the CPU (MMU) from its PCB.
- Load/restore the register content of the new process from its PCB.

Time for context switches:

Source for measured times: https://blog.tsunanet.net/2010/11/how-long-does-it-take-to-make-context.html

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Context switch

A context switch **changes the active process or thread** on the CPU. This may be expensive (takes some time).

Procedure:

- Save the register content of the currently running process into its PCB.
- Select a new process to run.
- Load/restore memory information into the CPU (MMU) from its PCB.
- Load/restore the register content of the new process from its PCB.

Time for context switches:

Source for measured times: https://blog.tsunanet.net/2010/11/how-long-does-it-take-to-make-context.html

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Context switch

A context switch **changes the active process or thread** on the CPU. This may be expensive (takes some time).

Procedure:

- Save the register content of the currently running process into its PCB.
- Select a new process to run.
- Load/restore memory information into the CPU (MMU) from its PCB.
- Load/restore the register content of the new process from its PCB.

Time for context switches:

Source for measured times: https://blog.tsunanet.net/2010/11/how-long-does-it-take-to-make-context.html

Context switch

A context switch **changes the active process or thread** on the CPU. This may be expensive (takes some time).

Procedure:

- Save the register content of the currently running process into its PCB.
- Select a new process to run.
- Load/restore memory information into the CPU (MMU) from its PCB.
- Load/restore the register content of the new process from its PCB.

Time for context switches:

- Time to switch processes: $\approx 3600ns$ per context switch (Intel E5440 CPU)
- Time to switch threads: $\approx 1300ns$ per context switch (Intel E5440 CPU)

Source for measured times: `https://blog.tsunanet.net/2010/11/how-long-does-it-take-to-make-context.html`

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Context switch

A context switch **changes the active process or thread** on the CPU. This may be expensive (takes some time).

Procedure:

- Save the register content of the currently running process into its PCB.
- Select a new process to run.
- Load/restore memory information into the CPU (MMU) from its PCB.
- Load/restore the register content of the new process from its PCB.

Time for context switches:

- Time to switch processes: $\approx 3600ns$ per context switch (Intel E5440 CPU)
- Time to switch threads: $\approx 1300ns$ per context switch (Intel E5440 CPU)

Source for measured times: `https://blog.tsunanet.net/2010/11/how-long-does-it-take-to-make-context.html`

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Context switch

A context switch **changes the active process or thread** on the CPU. This may be expensive (takes some time).

Procedure:

- Save the register content of the currently running process into its PCB.
- Select a new process to run.
- Load/restore memory information into the CPU (MMU) from its PCB.
- Load/restore the register content of the new process from its PCB.

Time for context switches:

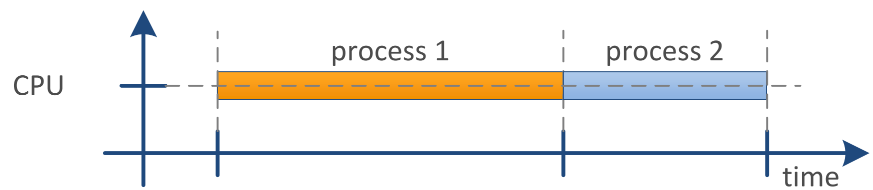
- Time to switch processes: $\approx 3600ns$ per context switch (Intel E5440 CPU)
- Time to switch threads: $\approx 1300ns$ per context switch (Intel E5440 CPU)

Source for measured times: `https://blog.tsunanet.net/2010/11/how-long-does-it-take-to-make-context.html`

CAMPUS Rosenheim
Computer Science

Scheduling: single vs multi core CPU

Single core CPU

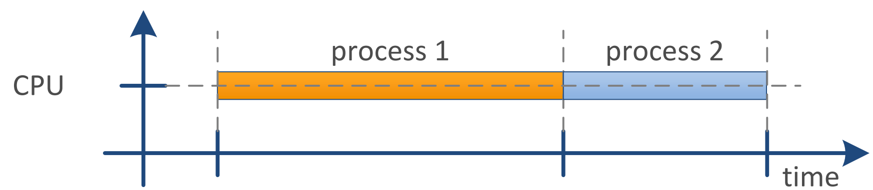


CAMPUS Rosenheim
Computer Science

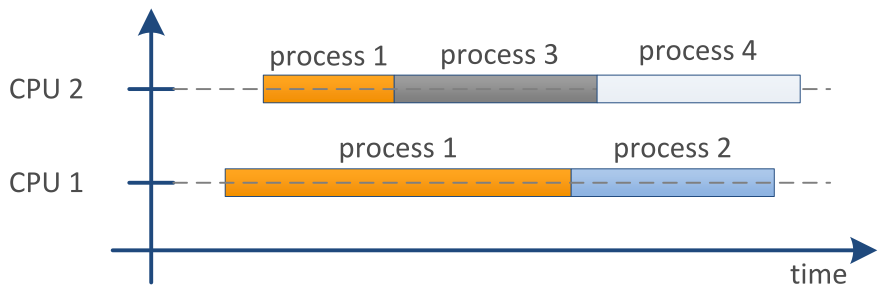
Technische Hochschule Rosenheim
Technical University of Applied Sciences

Scheduling: single vs multi core CPU

Single core CPU



Multi core CPU



CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Questions?

All right? $\Rightarrow$

Question? $\Rightarrow$ and use **chat**

or

speak *after* I ask you to

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Scheduling strategies

How would you schedule processes on your computer?

CAMPUS Rosenheim
Computer Science

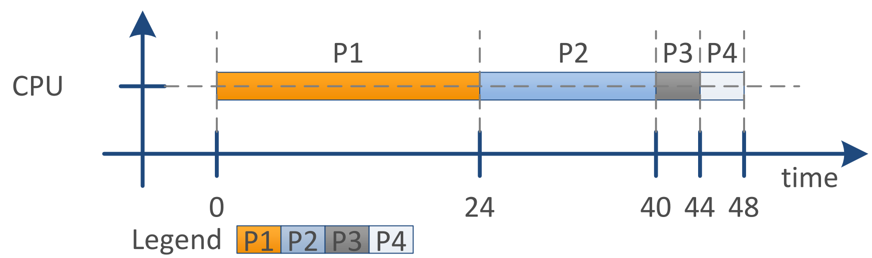
Technische Hochschule Rosenheim
Technical University of Applied Sciences

FCFS - first come first served

The arrival order (time) in the waiting queue is the scheduling order.

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	24
2	0	P2	16	40
3	0	P3	4	44
4	0	P4	4	48

$(24+40+44+48)/4 = 39$



Properties

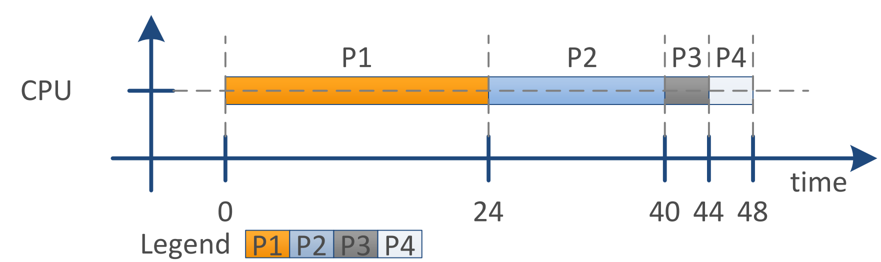
CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

FCFS - first come first served

The arrival order (time) in the waiting queue is the scheduling order.

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	24
2	0	P2	16	
3	0	P3	4	
4	0	P4	4	

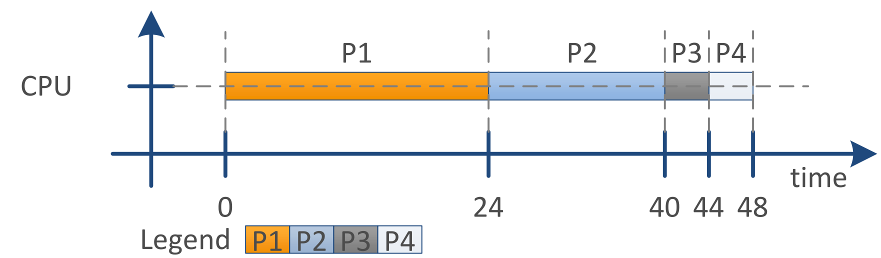


CAMPUS Rosenheim
Computer Science

FCFS - first come first served

The arrival order (time) in the waiting queue is the scheduling order.

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	24
2	0	P2	16	40
3	0	P3	4	
4	0	P4	4	



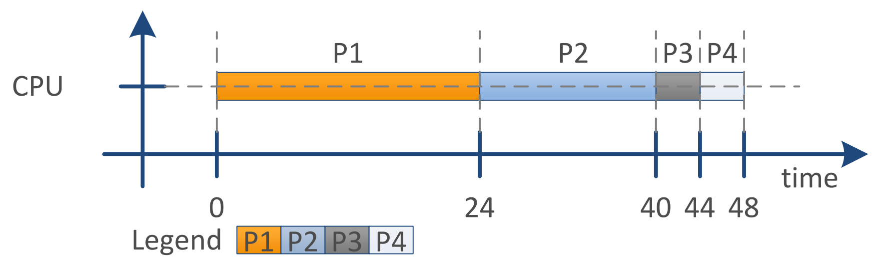
Properties

CAMPUS Rosenheim
Computer Science

FCFS - first come first served

The arrival order (time) in the waiting queue is the scheduling order.

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	24
2	0	P2	16	40
3	0	P3	4	44
4	0	P4	4	



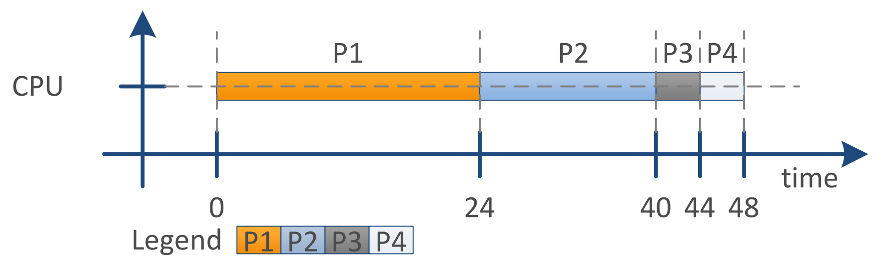
CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

FCFS - first come first served

The arrival order (time) in the waiting queue is the scheduling order.

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	24
2	0	P2	16	40
3	0	P3	4	44
4	0	P4	4	48



CAMPUS Rosenheim
Computer Science

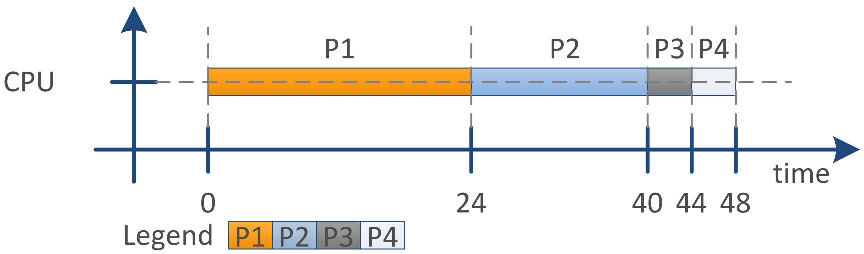
Technische Hochschule Rosenheim
Technical University of Applied Sciences

FCFS - first come first served

The arrival order (time) in the waiting queue is the scheduling order.

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	24
2	0	P2	16	40
3	0	P3	4	44
4	0	P4	4	48

Mean res. time $= (24 + 40 + 44 + 48)/4 = 39$



CAMPUS Rosenheim
Computer Science

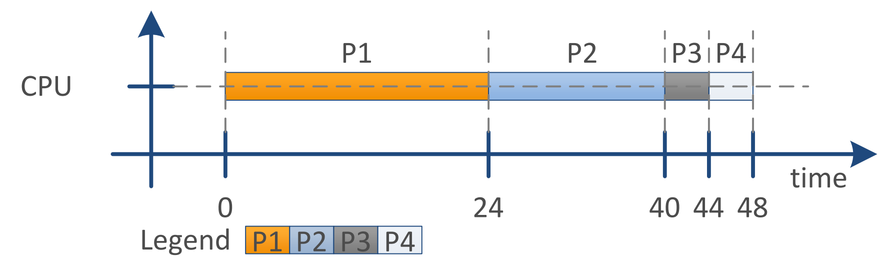
Technische Hochschule Rosenheim
Technical University of Applied Sciences

FCFS - first come first served

The arrival order (time) in the waiting queue is the scheduling order.

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	24
2	0	P2	16	40
3	0	P3	4	44
4	0	P4	4	48

Mean res. time $= (24 + 40 + 44 + 48)/4 = 39$



Properties

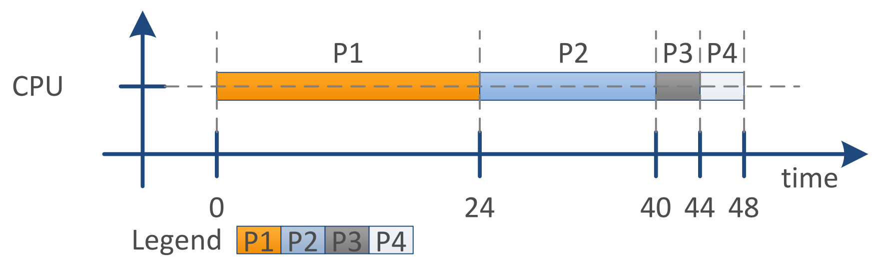
- A faire order (arrival time).
- Small jobs may wait long.
- Not good for interactive system: does not guarantee a good response time.

FCFS - first come first served

The arrival order (time) in the waiting queue is the scheduling order.

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	24
2	0	P2	16	40
3	0	P3	4	44
4	0	P4	4	48

Mean res. time $= (24 + 40 + 44 + 48)/4 = 39$



Properties

- A faire order (arrival time).
- Small jobs may wait long.
- Not good for interactive system: does not guarantee a good response time.

CAMPUS Rosenheim
Computer Science

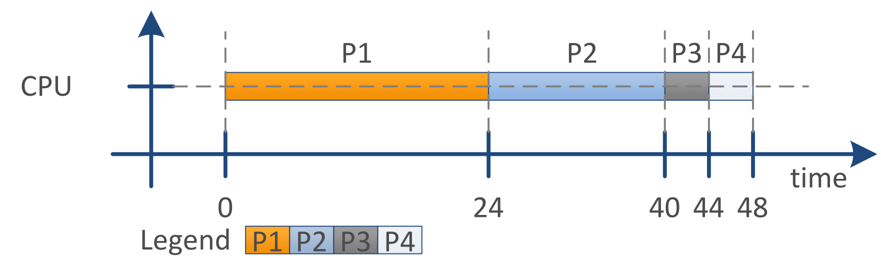
Technische Hochschule Rosenheim
Technical University of Applied Sciences

FCFS - first come first served

The arrival order (time) in the waiting queue is the scheduling order.

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	24
2	0	P2	16	40
3	0	P3	4	44
4	0	P4	4	48

Mean res. time $= (24 + 40 + 44 + 48)/4 = 39$



Properties

- A faire order (arrival time).
- Small jobs may wait long.
- Not good for interactive system: does not guarantee a good response time.

CAMPUS Rosenheim
Computer Science

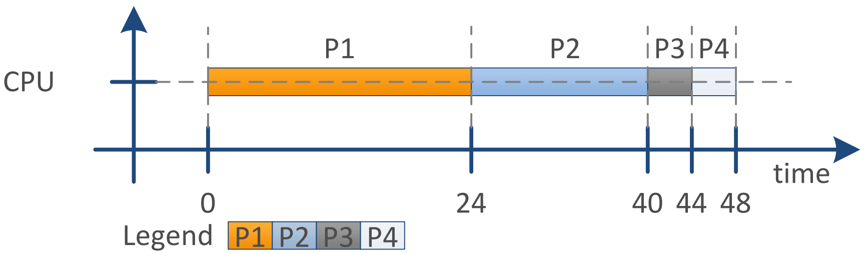
Technische Hochschule Rosenheim
Technical University of Applied Sciences

FCFS - first come first served

The arrival order (time) in the waiting queue is the scheduling order.

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	24
2	0	P2	16	40
3	0	P3	4	44
4	0	P4	4	48

Mean res. time $= (24 + 40 + 44 + 48)/4 = 39$



Properties

- A faire order (arrival time).
- Small jobs may wait long.
- Not good for interactive system: does not guarantee a good response time.

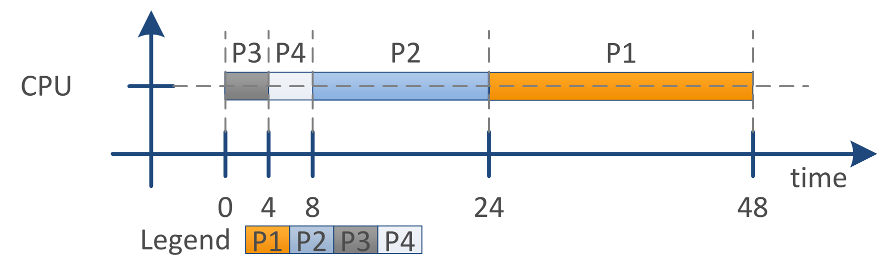
CAMPUS Rosenheim
Computer Science

SJF - shortest jobs first

The job with the smallest processing time is scheduled first.

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	48
2	0	P2	16	24
3	0	P3	4	4
4	0	P4	4	8

Mean res. time $= (48 + 24 + 4 + 8)/4 = 21$



CAMPUS Rosenheim
Computer Science

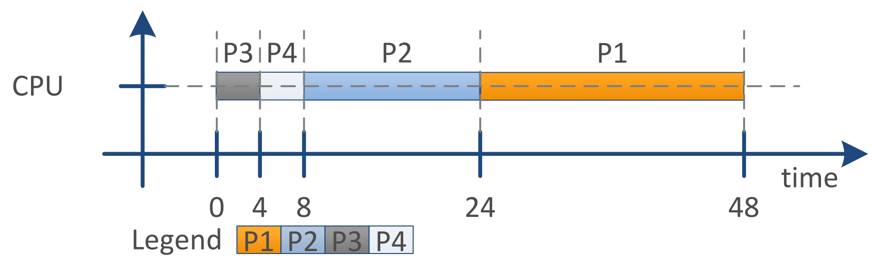
Technische Hochschule Rosenheim
Technical University of Applied Sciences

SJF - shortest jobs first

The job with the smallest processing time is scheduled first.

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	48
2	0	P2	16	24
3	0	P3	4	4
4	0	P4	4	8

Mean res. time $= (48 + 24 + 4 + 8)/4 = 21$



Properties

- SJF optimises the throughput.
- The processing time is often not available (prediction also hard).
- Processing time is predicted: by user, automatically?
- Jobs with a long processing time may not get scheduled (starvation).

CAMPUS Rosenheim
Computer Science

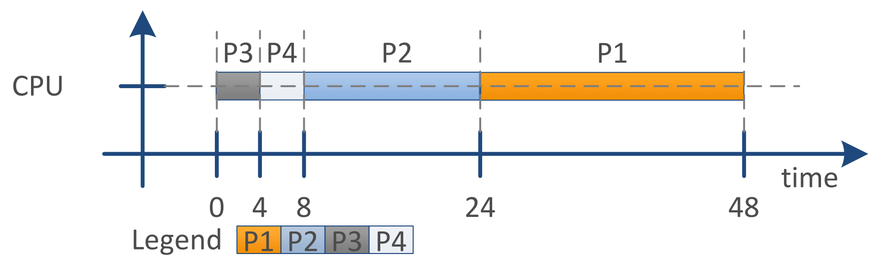
Technische Hochschule Rosenheim
Technical University of Applied Sciences

SJF - shortest jobs first

The job with the smallest processing time is scheduled first.

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	48
2	0	P2	16	24
3	0	P3	4	4
4	0	P4	4	8

Mean res. time $= (48+24+4+8)/4 = 21$



Properties

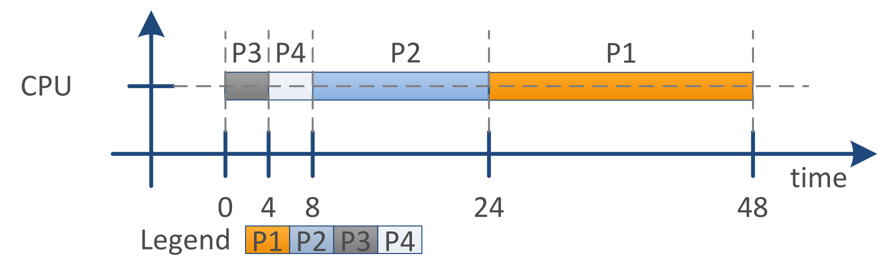
- SJF optimises the throughput.
- The processing time is often not available (prediction also hard).
- Processing time is predicted: by user, automatically?
- Jobs with a long processing time may not get scheduled (starvation).

SJF - shortest jobs first

The job with the smallest processing time is scheduled first.

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	48
2	0	P2	16	24
3	0	P3	4	4
4	0	P4	4	8

Mean res. time $= (48 + 24 + 4 + 8)/4 = 21$



Properties

- SJF optimises the throughput.
- The processing time is often not available (prediction also hard).
- Processing time is predicted: by user, automatically?
- Jobs with a long processing time may not get scheduled (starvation).

CAMPUS Rosenheim
Computer Science

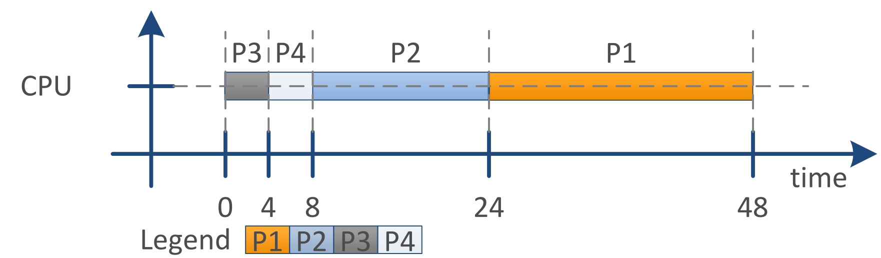
Technische Hochschule Rosenheim
Technical University of Applied Sciences

SJF - shortest jobs first

The job with the smallest processing time is scheduled first.

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	48
2	0	P2	16	24
3	0	P3	4	4
4	0	P4	4	8

Mean res. time $= (48 + 24 + 4 + 8)/4 = 21$



Properties

- SJF optimises the throughput.
- The processing time is often not available (prediction also hard).
- Processing time is predicted: by user, automatically?
- Jobs with a long processing time may not get scheduled (starvation).

CAMPUS Rosenheim
Computer Science

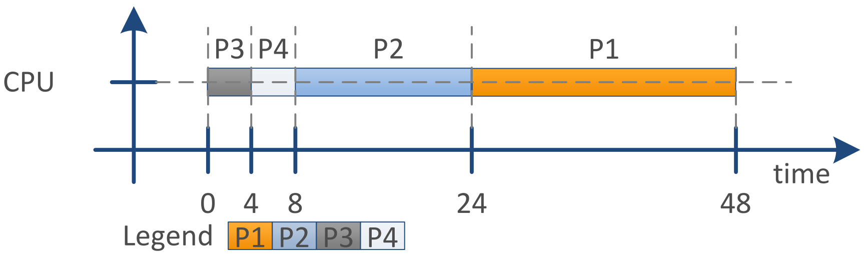
Technische Hochschule Rosenheim
Technical University of Applied Sciences

SJF - shortest jobs first

The job with the smallest processing time is scheduled first.

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	48
2	0	P2	16	24
3	0	P3	4	4
4	0	P4	4	8

Mean res. time $= (48 + 24 + 4 + 8)/4 = 21$



Properties

- SJF optimises the throughput.
- The processing time is often not available (prediction also hard).
- Processing time is predicted: by user, automatically?
- Jobs with a long processing time may not get scheduled (starvation).

CAMPUS Rosenheim
Computer Science

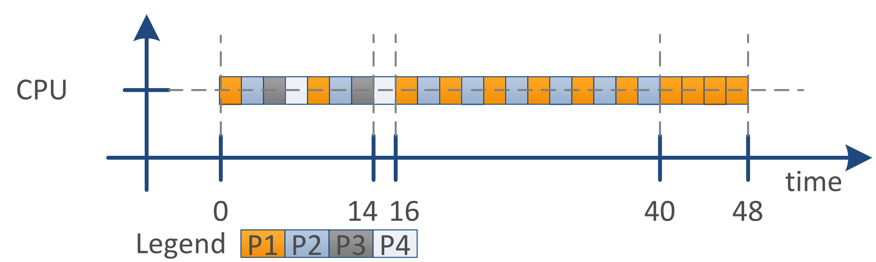
RR - round robin

Every process in the queue get the required resource for a limited amount of time (time slice). Then the preempted process is placed to the last position in the waiting queue.

Here: time slice $= 2$

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	48
2	0	P2	16	40
3	0	P3	4	14
4	0	P4	4	16

Mean res. time $= (48 + 40 + 14 + 16)/4 = 29.5$



CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

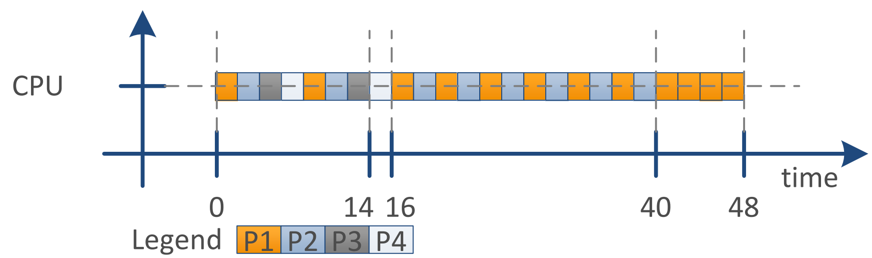
RR - round robin

Every process in the queue get the required resource for a limited amount of time (time slice). Then the preempted process is placed to the last position in the waiting queue.

Here: time slice $= 2$

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	48
2	0	P2	16	40
3	0	P3	4	14
4	0	P4	4	16

Mean res. time $= (48 + 40 + 14 + 16)/4 = 29.5$



Properties

- Good for interactive systems.
- Too many process switches causes context switches–and that are expensive.
- The average waiting time is often longer than with other scheduling strategies.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

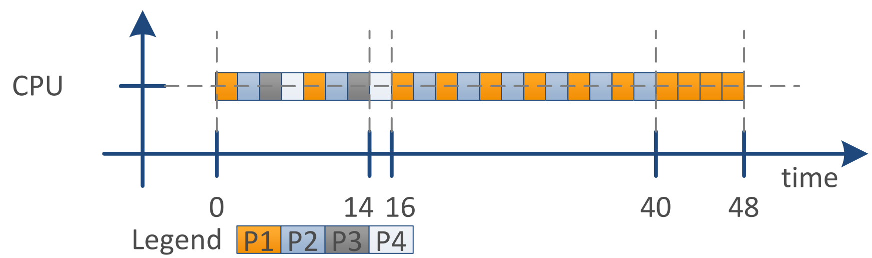
RR - round robin

Every process in the queue get the required resource for a limited amount of time (time slice). Then the preempted process is placed to the last position in the waiting queue.

Here: time slice $= 2$

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	48
2	0	P2	16	40
3	0	P3	4	14
4	0	P4	4	16

Mean res. time $= (48 + 40 + 14 + 16)/4 = 29.5$



Properties

- Good for interactive systems.
- Too many process switches causes context switches–and that are expensive.
- The average waiting time is often longer than with other scheduling strategies.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

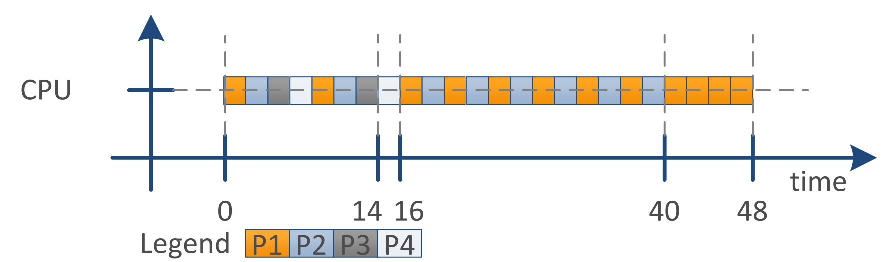
RR - round robin

Every process in the queue get the required resource for a limited amount of time (time slice). Then the preempted process is placed to the last position in the waiting queue.

Here: time slice = 2

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	48
2	0	P2	16	40
3	0	P3	4	14
4	0	P4	4	16

Mean res. time $= (48 + 40 + 14 + 16)/4 = 29.5$



Properties

- Good for interactive systems.
- Too many process switches causes context switches–and that are expensive.
- The average waiting time is often longer than with other scheduling strategies.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

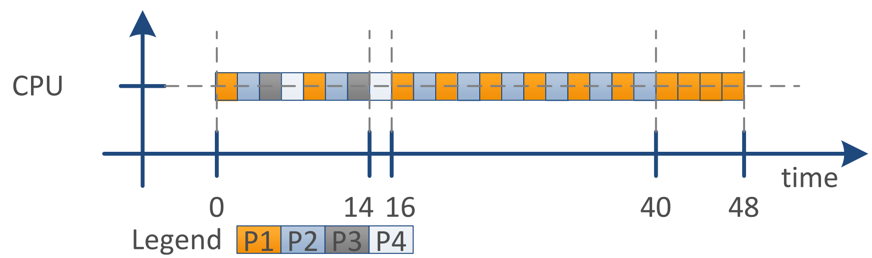
RR - round robin

Every process in the queue get the required resource for a limited amount of time (time slice). Then the preempted process is placed to the last position in the waiting queue.

Here: time slice = 2

Or.	ar. time	Process	proc. time	res. time
1	0	P1	24	48
2	0	P2	16	40
3	0	P3	4	14
4	0	P4	4	16

Mean res. time $= (48 + 40 + 14 + 16)/4 = 29.5$



Properties

- Good for interactive systems.
- Too many process switches causes context switches–and that are expensive.
- The average waiting time is often longer than with other scheduling strategies.

CAMPUS Rosenheim
Computer Science

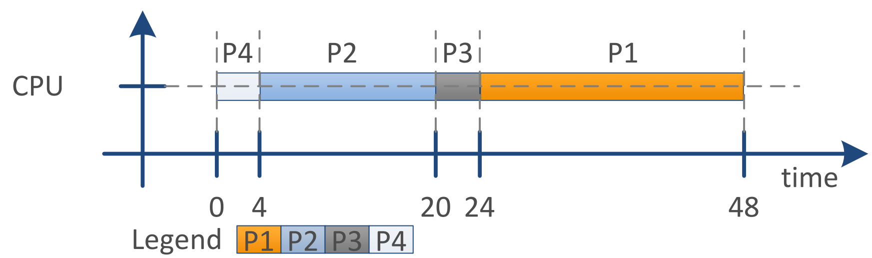
Technische Hochschule Rosenheim
Technical University of Applied Sciences

EDF - earliest deadline first

The process whose deadline ends first is processed first.

Or.	ar. time	Process	proc. time	deadl.	res. time
1	0	P1	24	60	48
2	0	P2	16	20	20
3	0	P3	4	28	24
4	0	P4	4	4	4

Mean res. time $= (48 + 20 + 24 + 4)/4 = 24$



CAMPUS Rosenheim
Computer Science

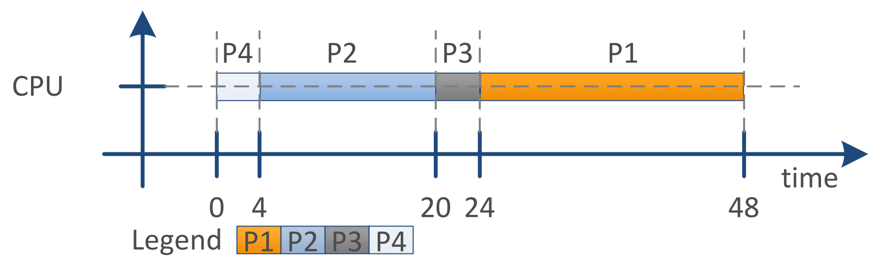
Technische Hochschule Rosenheim
Technical University of Applied Sciences

EDF - earliest deadline first

The process whose deadline ends first is processed first.

Or.	ar. time	Process	proc. time	deadl.	res. time
1	0	P1	24	60	48
2	0	P2	16	20	20
3	0	P3	4	28	24
4	0	P4	4	4	4

Mean res. time $= (48 + 20 + 24 + 4)/4 = 24$



Properties

- Used in real-time systems.
- Is not always optimal on multi-core CPUs.

CAMPUS Rosenheim
Computer Science

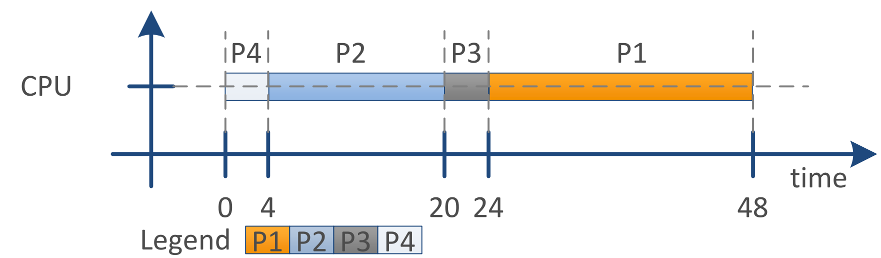
Technische Hochschule Rosenheim
Technical University of Applied Sciences

EDF - earliest deadline first

The process whose deadline ends first is processed first.

Or.	ar. time	Process	proc. time	deadl.	res. time
1	0	P1	24	60	48
2	0	P2	16	20	20
3	0	P3	4	28	24
4	0	P4	4	4	4

Mean res. time $= (48 + 20 + 24 + 4)/4 = 24$



Properties

- Used in real-time systems.
- Is not always optimal on multi-core CPUs.

CAMPUS Rosenheim
Computer Science

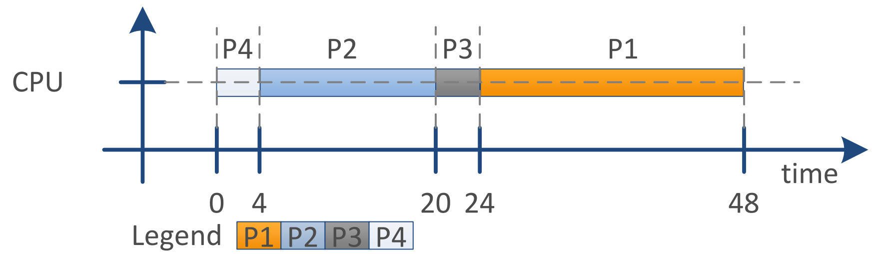
Technische Hochschule Rosenheim
Technical University of Applied Sciences

EDF - earliest deadline first

The process whose deadline ends first is processed first.

Or.	ar. time	Process	proc. time	deadl.	res. time
1	0	P1	24	60	48
2	0	P2	16	20	20
3	0	P3	4	28	24
4	0	P4	4	4	4

Mean res. time $= (48 + 20 + 24 + 4)/4 = 24$



Properties

- Used in real-time systems.
- Is not always optimal on multi-core CPUs.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Priority based

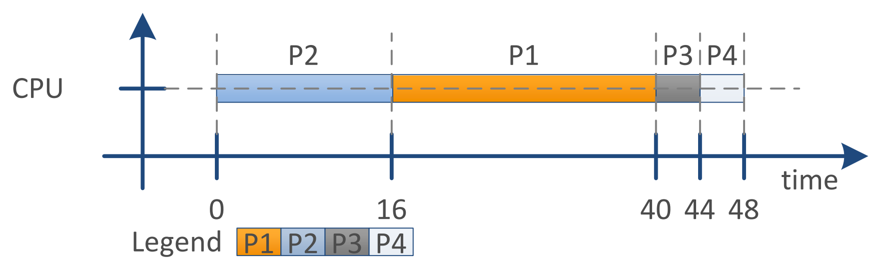
External priority

The process with the highest priority is scheduled first. The user can define the priority on startup or change it during execution.

Here: 1 = highest priority, 4 = lowest priority (depends on OS definition)

Or.	ar. time	Process	proc. time	prio	res. time
1	0	P1	24	2	40
2	0	P2	16	1	16
3	0	P3	4	3	44
4	0	P4	4	4	48

Mean res. time $= (40 + 16 + 44 + 48)/4 = 37$



CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Priority based

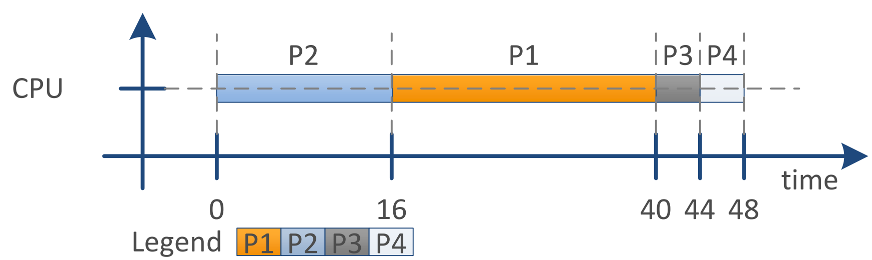
External priority

The process with the highest priority is scheduled first. The user can define the priority on startup or change it during execution.

Here: 1 = highest priority, 4 = lowest priority (depends on OS definition)

Or.	ar. time	Process	proc. time	prio	res. time
1	0	P1	24	2	40
2	0	P2	16	1	16
3	0	P3	4	3	44
4	0	P4	4	4	48

Mean res. time $= (40 + 16 + 44 + 48)/4 = 37$



Properties

- The important processes are scheduled first.
- Long-running processes with a high priority can cause low priority processes to be kept away from the CPU for a long time.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Priority based

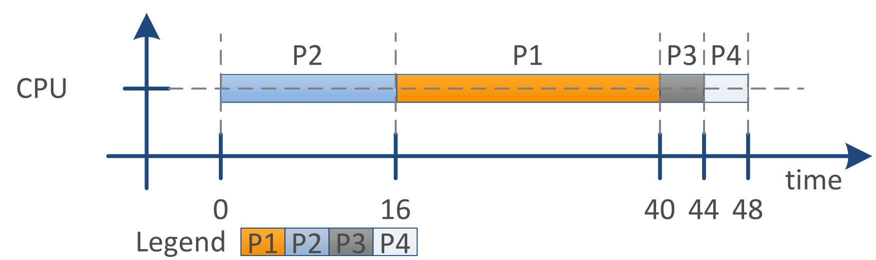
External priority

The process with the highest priority is scheduled first. The user can define the priority on startup or change it during execution.

Here: 1 = highest priority, 4 = lowest priority (depends on OS definition)

Or.	ar. time	Process	proc. time	prio	res. time
1	0	P1	24	2	40
2	0	P2	16	1	16
3	0	P3	4	3	44
4	0	P4	4	4	48

Mean res. time $= (40 + 16 + 44 + 48)/4 = 37$



Properties

- The important processes are scheduled first.
- Long-running processes with a high priority can cause low priority processes to be kept away from the CPU for a long time.

CAMPUS Rosenheim
Computer Science

Priority based

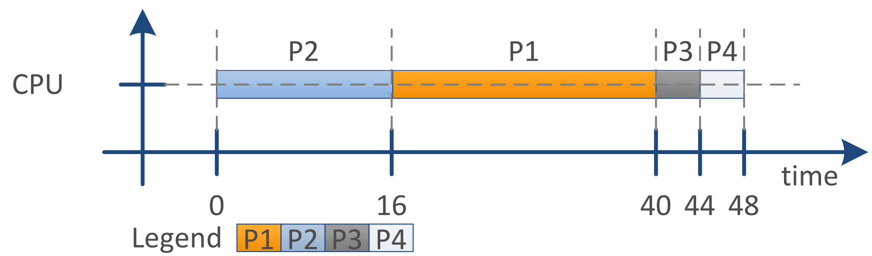
External priority

The process with the highest priority is scheduled first. The user can define the priority on startup or change it during execution.

Here: 1 = highest priority, 4 = lowest priority (depends on OS definition)

Or.	ar. time	Process	proc. time	prio	res. time
1	0	P1	24	2	40
2	0	P2	16	1	16
3	0	P3	4	3	44
4	0	P4	4	4	48

Mean res. time $= (40 + 16 + 44 + 48)/4 = 37$



Properties

- The important processes are scheduled first.
- Long-running processes with a high priority can cause low priority processes to be kept away from the CPU for a long time.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Priority based

Internal priority

After a process runs for a while, its priority is automatically lowered. After some long waiting time, the priority can automatically be increased again.

Properties

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Priority based

Internal priority

After a process runs for a while, its priority is automatically lowered. After some long waiting time, the priority can automatically be increased again.

Properties

- Solves the issue of the external priority.
- Improves the response time.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Priority based

Internal priority

After a process runs for a while, its priority is automatically lowered. After some long waiting time, the priority can automatically be increased again.

Properties

- Solves the issue of the external priority.
- Improves the response time.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Priority based

Internal priority

After a process runs for a while, its priority is automatically lowered. After some long waiting time, the priority can automatically be increased again.

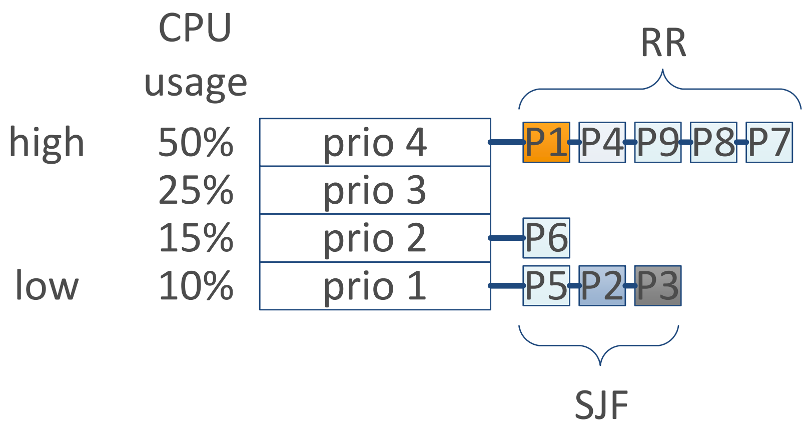
Properties

- Solves the issue of the external priority.
- Improves the response time.

CAMPUS Rosenheim
Computer Science

Multilevel queue scheduling

There exist multiple queues, whereas each can have its own scheduling strategy.

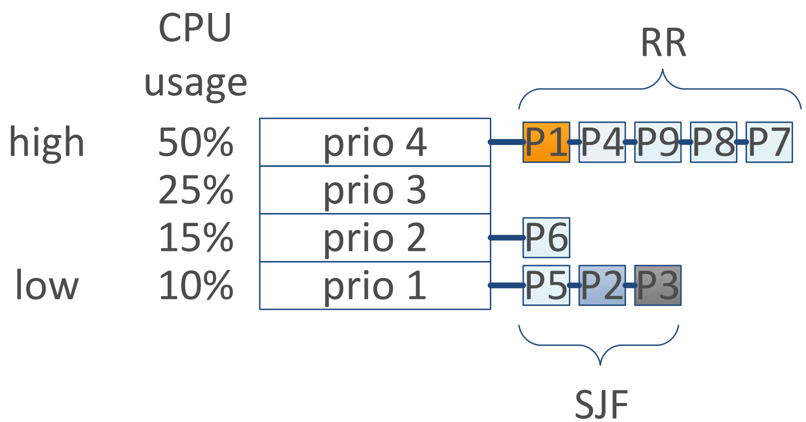


CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Multilevel queue scheduling

There exist multiple queues, whereas each can have its own scheduling strategy.



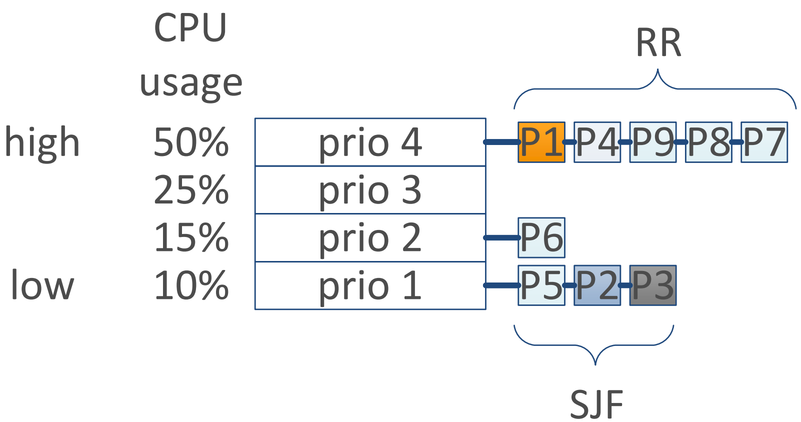
Properties

- Combination of different strategies.
- Each queue can have its own CPU usage.
- Improves response time while taking priorities into account.

CAMPUS Rosenheim
Computer Science

Multilevel queue scheduling

There exist multiple queues, whereas each can have its own scheduling strategy.



Properties

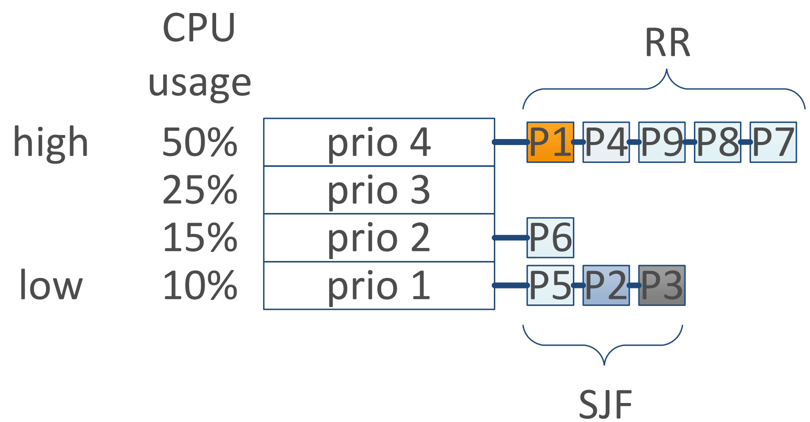
- Combination of different strategies.
- Each queue can have its own CPU usage.
- Improves response time while taking priorities into account.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Multilevel queue scheduling

There exist multiple queues, whereas each can have its own scheduling strategy.



Properties

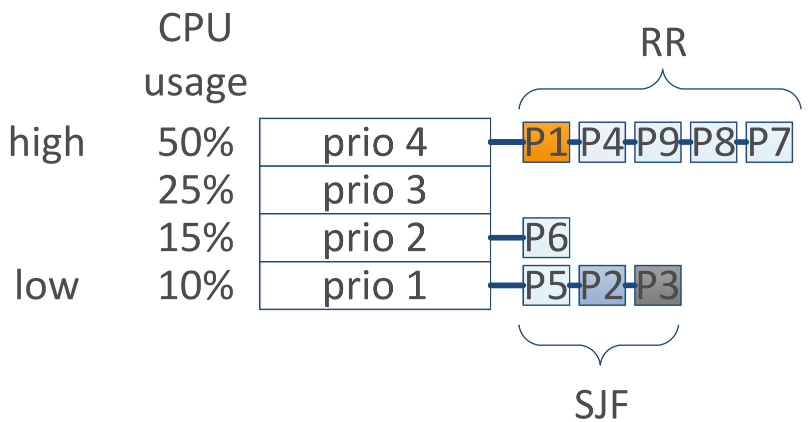
- Combination of different strategies.
- Each queue can have its own CPU usage.
- Improves response time while taking priorities into account.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Multilevel queue scheduling

There exist multiple queues, whereas each can have its own scheduling strategy.



Properties

- Combination of different strategies.
- Each queue can have its own CPU usage.
- Improves response time while taking priorities into account.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Questions?

All right? $\Rightarrow$

Question? $\Rightarrow$ and use **chat**

or

speak *after* I ask you to

CAMPUS Rosenheim
Computer Science

Linux priorities

Supports 140 different priority classes (0...139)
The lower the value, the higher the priority.

Normal processes: 100...139 [0...39]

Parameter	Definition	Description
NI	$NI = -20 \ldots +19$	Nice value ranges from $-20 \ldots +19$. Users can only increase the nice value, but not lower it.
PR	$PR = 20 + NI$	The priority PR ranges from 0...39. Default user processes usually submitted with $NI = 0 => PR = 20$.

Real-time processes: 0...99 [−100...−1]

Parameter	Definition	Description
RT	$RT = 1 \ldots 99$	The real-time priority RT ranges from 1...99
PR	$PR = -1 - RT$	The priority PR ranges from $-100 \ldots -1$.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Linux priorities

Supports 140 different priority classes (0...139)
The lower the value, the higher the priority.

Normal processes: 100...139 [0...39]

Parameter	Definition	Description
NI	$NI = -20 \ldots +19$	Nice value ranges from $-20 \ldots +19$. Users can only increase the nice value, but not lower it.
PR	$PR = 20 + NI$	The priority *PR* ranges from 0...39. Default user processes usually submitted with $NI = 0 => PR = 20$.

Real-time processes: 0...99 [−100...−1]

Parameter	Definition	Description
RT	$RT = 1 \ldots 99$	The real-time priority *RT* ranges from 1...99
PR	$PR = -1 - RT$	The priority *PR* ranges from $-100 \ldots -1$.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Linux priorities

Supports 140 different priority classes (0...139)
The lower the value, the higher the priority.

Normal processes: 100...139 $[0 \ldots 39]$

Parameter	Definition	Description
NI	$NI = -20 \ldots +19$	Nice value ranges from $-20 \ldots +19$. Users can only increase the nice value, but not lower it.
PR	$PR = 20 + NI$	The priority PR ranges from 0...39. Default user processes usually submitted with $NI = 0 => PR = 20$.

Real-time processes: 0...99 $[-100 \ldots -1]$

Parameter	Definition	Description
RT	$RT = 1 \ldots 99$	The real-time priority RT ranges from 1...99
PR	$PR = -1 - RT$	The priority PR ranges from $-100 \ldots -1$.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Linux commands

Command	Description
`top`	Shows processes in live view.
`htop`	Shows processes in live view.
`ps ax -o pid,rtprio,pri,ni,cmd`	Shows processes with pid, rtprio, pri, ni, and its cmd.
`nice -n 15 make -j`	Starts a parallel build with $NI = 15$
`nice -n -5 make -j`	Starts a parallel build with $NI = -5$
`renice -n 15 -p 1000`	Change $NI = 15$ of process with PID 1000.
`renice -n -5 -p 1000`	Change $NI = -5$ of process with PID 1000.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Questions?

All right? $\Rightarrow$

Question? $\Rightarrow$ and use **chat**

or

speak *after* I ask you to

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Summary and outlook

Summary

- Scheduling theory and terms
- Scheduling strategies
- Scheduling on Linux

Outlook

- Memory management

CAMPUS Rosenheim
Computer Science

Summary and outlook

Summary

- Scheduling theory and terms
- Scheduling strategies
- Scheduling on Linux

Outlook

- Memory management