

Scheduling

Andre M. Maier, DHBW Ravensburg
dhbw@andre-maier.com

Goals of Process Scheduling

- Maximize CPU utilization
- Fair CPU allocation
- Minimize Turnaround Time
- Minimize Waiting Time
- Maximize the number of processes that complete execution per time unit (“throughput”)

Process Scheduling Times

- **Arrival Time**

The time at which the process arrives in the ready queue.

- **Completion Time**

The time at which a process is completed.

- **Burst Time**

CPU time that a process requires for its execution.

- **Turnaround Time**

= Completion Time – Arrival Time

- **Waiting Time**

The amount of time a process has been waiting in the ready queue.

Two Types of Scheduling Algorithms

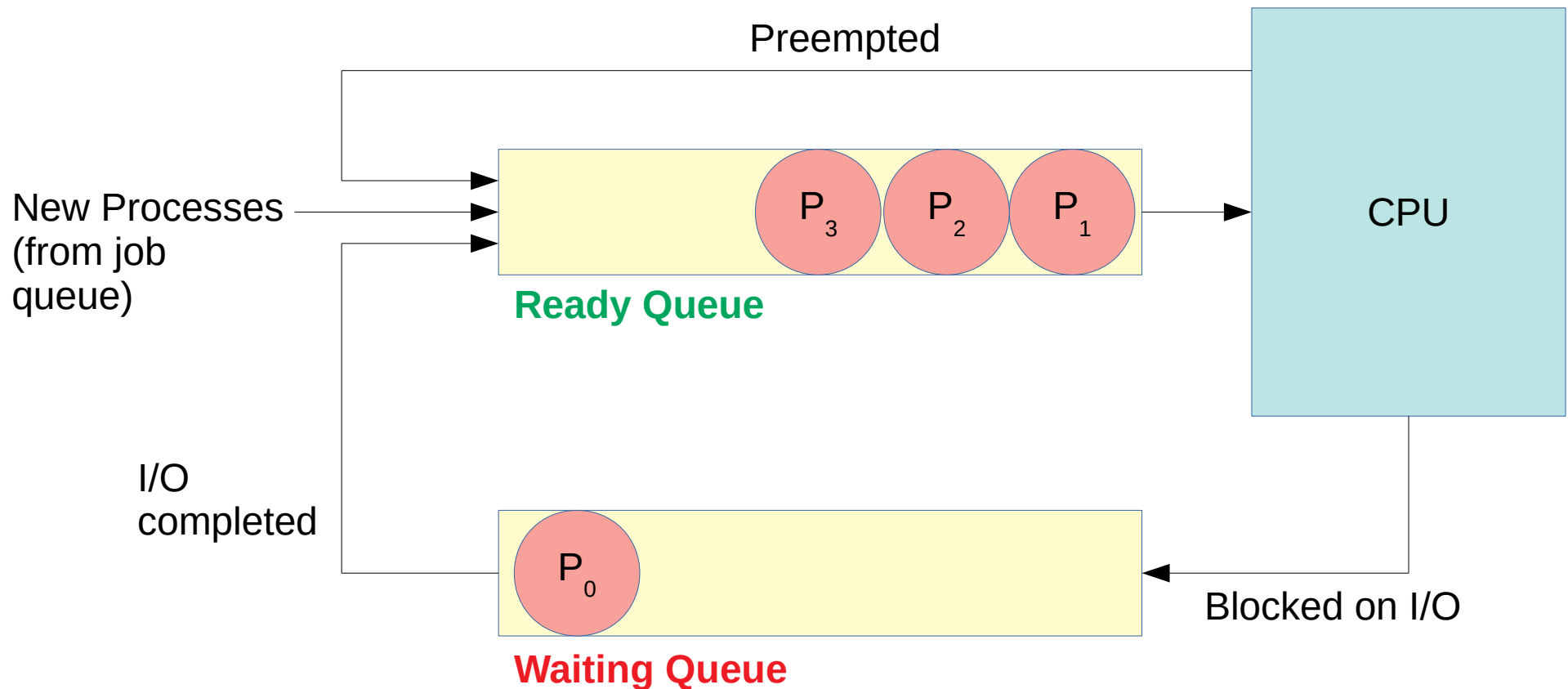
- **Non-preemptive Scheduling**

A process holds the CPU until it terminates or it switches from running to waiting state.

- **Preemptive Scheduling**

A running process can be taken away from the CPU in favor of other processes.

Scheduling Queues (simplified)



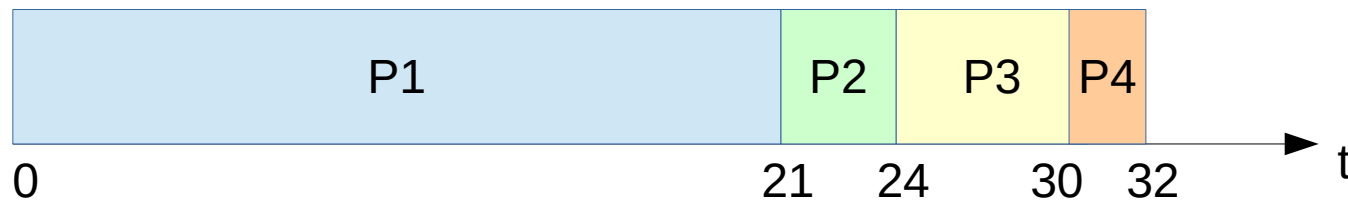
Scheduling Algorithms

- First Come First Serve
- Shortest Job First
- Shortest Remaining Time First
- Round Robin Scheduling
- Priority Based Scheduling
- Highest Response Ratio Next
- Multilevel Queue Scheduling
- Multilevel Feedback Queue Scheduling

First Come First Serve (FCFS) (1)

Process	Burst Time
P1	21
P2	3
P3	6
P4	2

Assumption: Arrival Time of all processes is 0



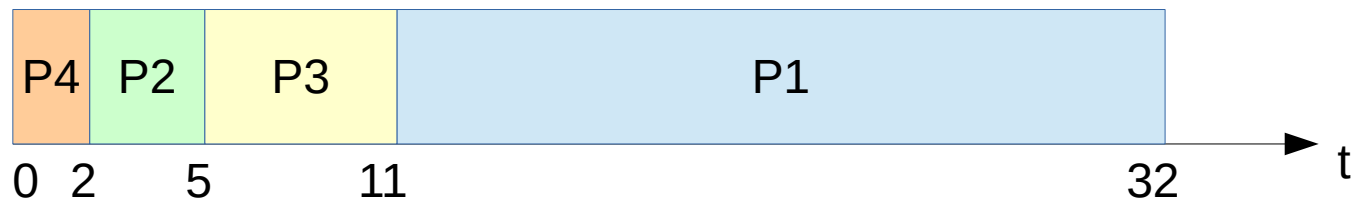
First Come First Serve (FCFS) (2)

- FIFO (First-In-First-Out) queue structure
- Simple to implement
- Non-preemptive
- Does not consider priority
- Low throughput possible due to *convoy effect* (Long processes will delay execution of short processes)
- No starvation (assuming that every process will eventually complete)

Shortest Job First (SJF) (1)

Process	Burst Time
P1	21
P2	3
P3	6
P4	2

Assumption: Arrival Time of all processes is 0

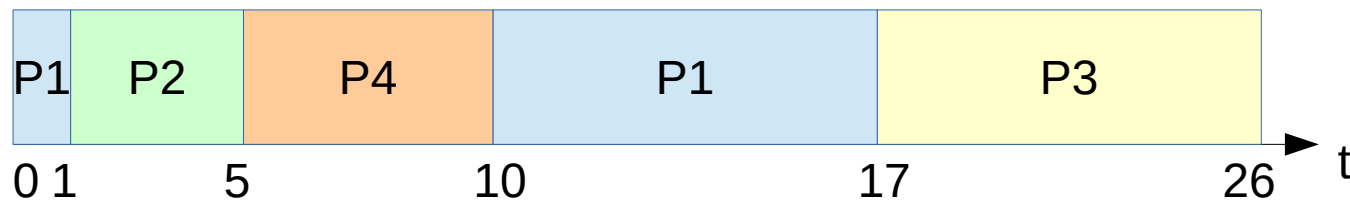


Shortest Job First (SJF) (2)

- Always selects the process with the smallest burst time for execution first.
- A process' priority is the inverse of its predicted CPU burst time.
- The burst time of each process must be known in advance.
- Risk of starvation.

Shortest Remaining Time First (SRTF) (1)

Process	Burst Time	Arrival Time
P1	8	0
P2	4	1
P3	9	2
P4	5	3



Shortest Remaining Time First (SRTF) (2)

- Whenever a process with a shorter burst time arrives, the currently executed process is preempted.
- Overhead due to context switching
- Starvation is possible (if short processes are continually added)

Highest Response Ratio Next (HRRN)

- Similar to SJN with a small modification
- Decision which job is next is based on the highest *response ratio*.

$$\text{response ratio} = 1 + \frac{\text{waiting time}}{\text{estimated run time}}$$

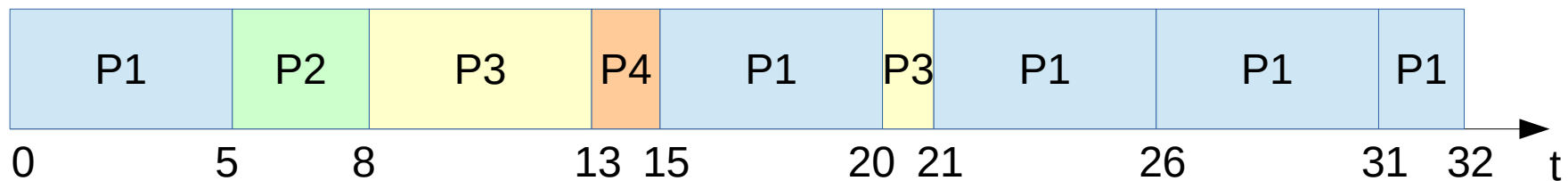
- Jobs that have spent a relatively long waiting time will be preferred
- Mitigates the problem of starvation

Round Robin Scheduling (RR) (1)

Process	Burst Time
P1	21
P2	3
P3	6
P4	2

Quantum: 5

Assumption: Arrival Time of all processes is 0



Round Robbing Scheduling (RR) (2)

- Assigns a fixed time quantum per process and cycles through all processes
- Process is rescheduled if it does not complete within the quantum
- No starvation
- Overhead due to context switching, especially with small time quanta
 - Quantum should be significantly higher than context switch time, e.g. 100 ms when context switch time is $< 10 \mu\text{s}$

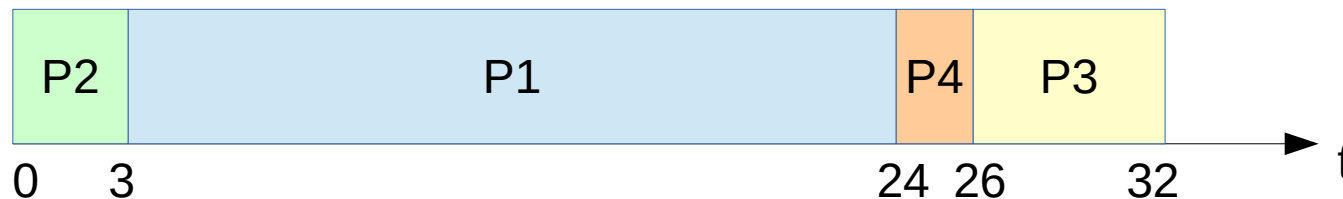
Exercise

- Determine the average waiting time for the previous examples of the following scheduling algorithms:
 - First Come First Serve (FCFS)
 - Shortest Job First (SJF)
 - Shortest Remaining Time First (SRTF)
 - Round Robin (RR)

Priority Based Scheduling (1)

Process	Burst Time	Priority
P1	21	2
P2	3	1
P3	6	4
P4	2	3

Assumption: Arrival Time of all processes is 0



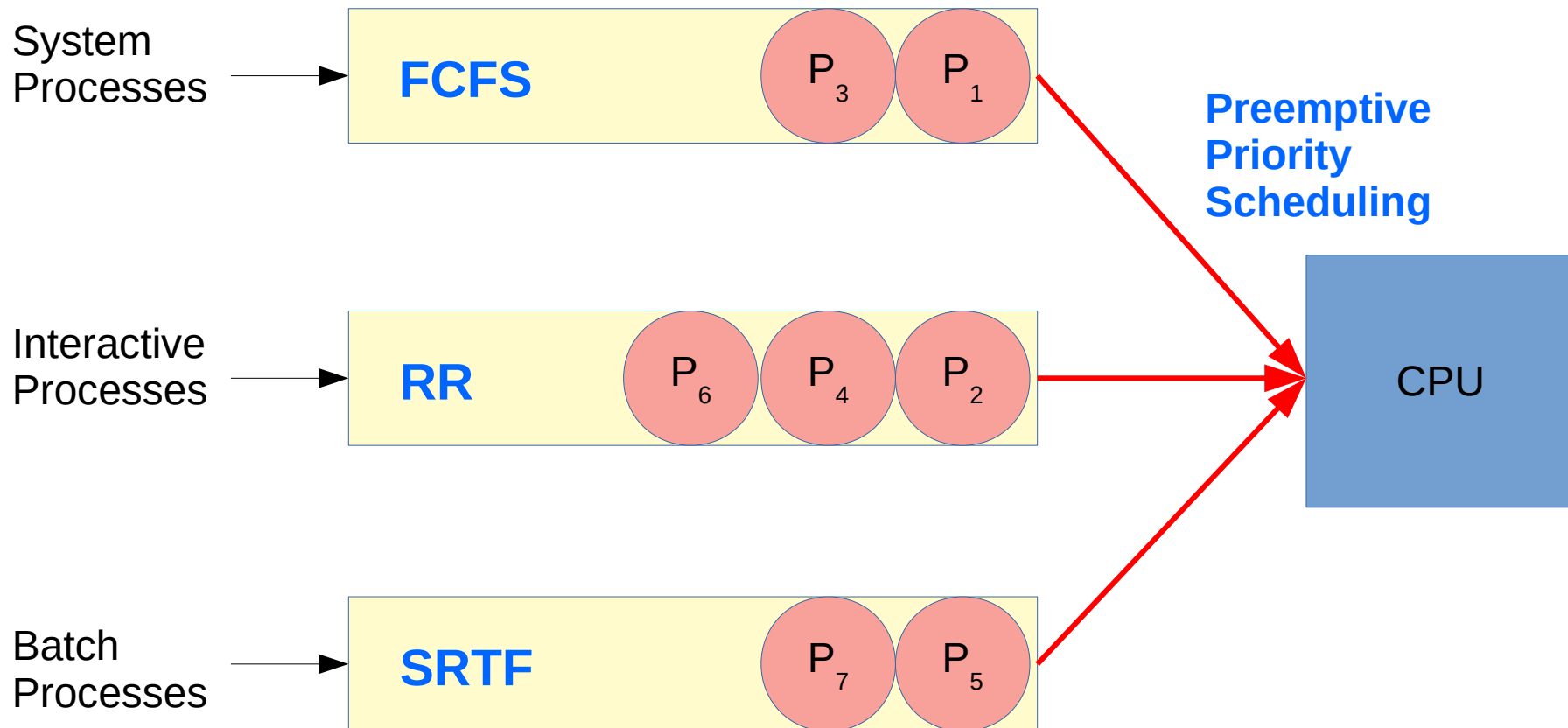
Priority Based Scheduling (2)

- Process with the highest priority is executed first
- Processes with the same priority require a second scheduling algorithm, e.g. FCFS
- Risk of starvation in preemptive priority scheduling

Multilevel Queue Scheduling (1)

- Processes are classified into different groups each of which has its own scheduling requirements.
- Ready queue consists of multiple separate queues each of which has its own scheduling algorithm.
- Additional scheduling is required among the queues (e.g. preemptive priority scheduling)

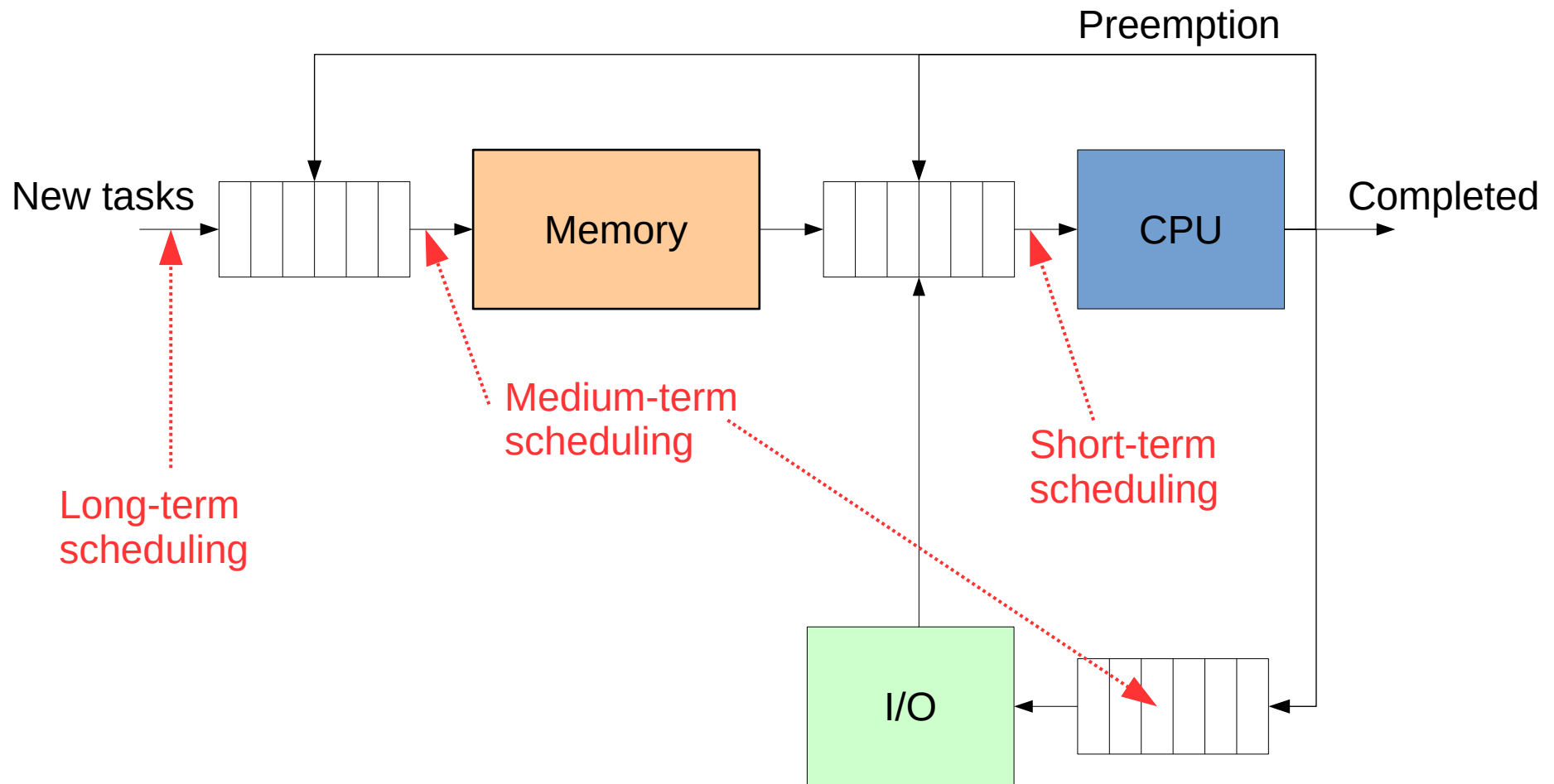
Multilevel Queue Scheduling (Example)



Multilevel Feedback Queue Scheduling

- Enhanced variant of Multilevel Queue Scheduling
- Allows processes to move between the queues based on their spent CPU time
- Processes in high priority queues that spend too much CPU time may be moved to a lower priority queue
- Processes that have been waiting too long in a lower priority queue may be moved to a higher priority queue
- Most general scheduler, but also most complex to implement

Scheduling Strategies in the OS



Lab Exercises (1)

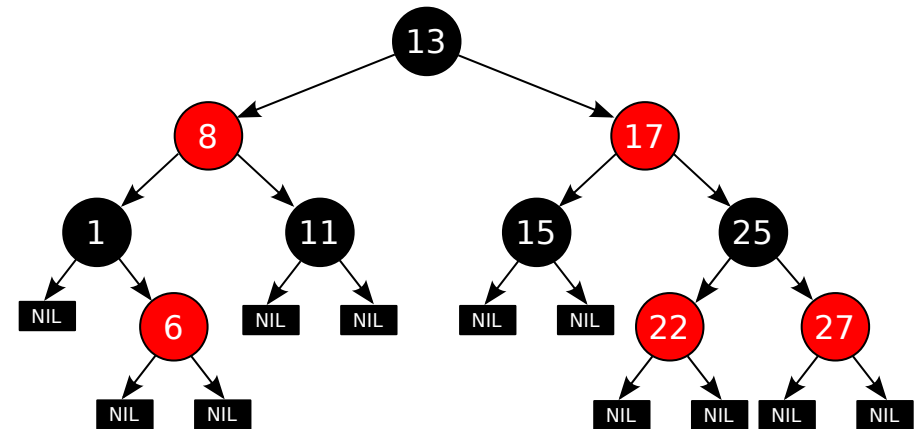
- Implement simulations of the following scheduling algorithms in C
 - First Come First Serve (FCFS)
 - Shortest Remaining Time First (SRTF)
 - Round Robin Scheduling (RR)

Lab Exercises (2)

- What concept does Linux provide from a user's perspective to run processes with different priorities?
- Set up a scenario in which processes with higher priority are granted more CPU time than others.
- What mechanisms does Linux provide to prevent users from depriving other users of CPU time?

Completely Fair Scheduler (CFS) (1)

- Presently used by the Linux Kernel (since 2.6.23, October 2007)
- Developed by Ingo Molnár
- Idea: Ideal Fairness
 - Every Process receives $1/n$ CPU time
 - Sleeping processes “earn” CPU time, i.e. they are given a boost when they wake up



Cburnett, CC BY-SA 3.0 <<http://creativecommons.org/licenses/by-sa/3.0/>>, via Wikimedia Commons

- CFS uses a red-black tree instead of queues as its underlying data structure
- CFS uses nanoseconds granularity accounting

Completely Fair Scheduler (CFS)

(2)

- Each process has a virtual runtime (*vruntime*)

$$vruntime = \frac{runtime}{weight}$$

actual time already spent on the CPU

weight (is dependent on the process' priority/niceness)
See also `/kernel/sched/sched.h` → `prio_to_weight`)

- The higher the weight, the lower the impact of the actual runtime. → Slower increase of virtual runtime for processes with higher priority.
- CFS always chooses the process with the lowest virtual runtime.

Completely Fair Scheduler (CFS)

(3)

- Some related files and tools
 - */proc/stat*
 - */sys/kernel/debug/sched_features*
 - *proc/sys/kernel/sched_**
 - *taskset*
 - *trace-cmd*
 - *kernelshark*
- More technical details on CFS
 - <https://www.kernel.org/doc/html/latest/scheduler/sched-design-CFS.html>
 - <https://doc.opensuse.org/documentation/leap/tuning/html/book-sle-tuning/cha-tuning-taskscheduler.html>

Real-time Operating Systems (RTOS)

- Guarantees a certain behavior within predictable (well-defined) time constraints.
- OS-internal actions must be finished
- Two groups of RTOS
 - Hard
 - Deadlines must always be met
 - Example: Collision avoidance systems in aircrafts
 - Soft
 - Occasionally missed deadlines do not pose critical risks
 - Example: Decoders for media streaming

Earliest Deadline First (EDF)

- Process with the earliest deadline gets dispatched to the CPU
- Algorithm requires a system clock (absolute time)
- Algorithm is preemptive
- All deadlines can be met, as long as sufficient computing resources are available
- Disadvantages
 - In overload scenarios, deadlines will be missed unpredictably
 - Difficult to implement in hardware

Exercises

- Develop the timing diagram for EDF scheduling of two processes P_1 , P_2 , and P_3 with the following parameters:

Process	Arrival Time	Burst Time	Deadline
P1	0	4	20
P2	1	5	15
P3	2	6	16

- Rate-Monotonic Scheduling (RMS) is a fixed-priority real-time scheduling algorithm that is usually preferred over EDF.
 - What are its advantages over EDF?
 - How does it work?

Questions for Review

- What are the goals of process scheduling?
- What is a major disadvantage of SJF and SRTF?
- Why should the time quantum in RR scheduling be significantly higher than the time required to perform a context switch?
- Explain the principle of how a Multilevel Feedback Queue Scheduler works.
- What is the definition of an RTOS?
- What happens in EDF scheduling when the CPU load exceeds 100%?