# 3. Assignment

**Process scheduling**

*Issue: May 11—Due: May 18*

## Exercise 5: *UNIX process scheduling.* 10 Points

The process scheduler used in BSD UNIX utilises a priority scheduling algorithm (see lecture):

Initially, each process starts with a *base priority* in the range $[-20, 20]$, which on average is 0[1]. Based on system load, the scheduler may change the priority of a process. The process with the highest priority is scheduled next.

If a process has used few CPU time recently, its priority is increased. The scheduling algorithm quickly "forgets" CPU consumption in order to favour processes with alternating CPU workload: The algorithm forgets 90% of the most recently used CPU time within $5 \times$ *load* seconds, where *load* is the average number of processes in *ready* state during the last few minutes[2].

Note the distinction between CPU-consumption, and other things a process might do while being executed, e.g., performing I/O operations.

**Discuss the following issues:**

1. Are CPU-intensive processes preferred if the system is under high load, or if the system is under low load?
2. How is an I/O-intensive process scheduled after it is done with its I/O operation?
3. Is it possible that a very CPU-intensive process repeatedly becomes postponed for ever?
4. When the system load increases, what effect do CPU-intensive processes have on the interactive response times of the system?

**Send your answers** in a plain text email to Alexander and Stefan[3]. If you insist on attachments, have an eye on UTF-8 encoding and UNIX end-of-line conventions of the attached plain text file.

[1] See the `nice`(1) command which allows a user to *be nice*, i.e., to start a process with reduced priority.
[2] The `top`(1) command displays load averages, priority, and nice values of all processes.
[3] alexander.holupirek@uni-konstanz.de, stefan.klinger@uni-konstanz.de