CIS 452 - Operating Systems Concepts

Nathan Bowman

Images taken from Silberschatz book

Scheduling Algorithms -- Multilevel Queue

Sometimes there are different categories of processes that benefit from different scheduling algorithms

One example may be foreground vs background processes

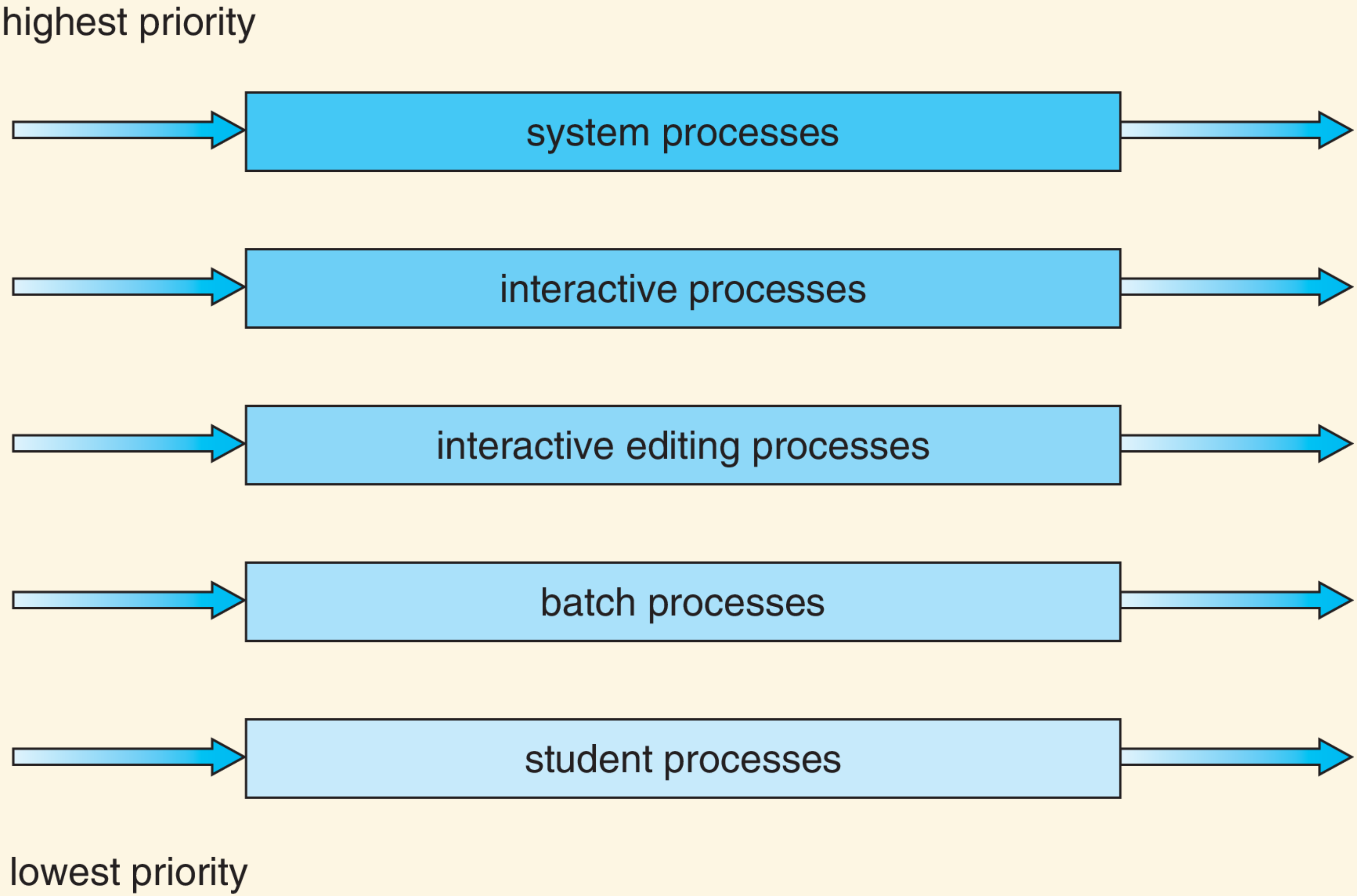
Processes that interact directly with users may need to be scheduled in a way that improves response time, whereas background processes do not need to be run immediately

In a **mutlilevel queue** scheduling algorithm, the ready queue is split into more than one queue

Processes are permanently assigned to a specific queue

The assignment can be based on similar properties to those we discussed with priority scheduling: memory requirements, foreground/background, externally specified priority, ...

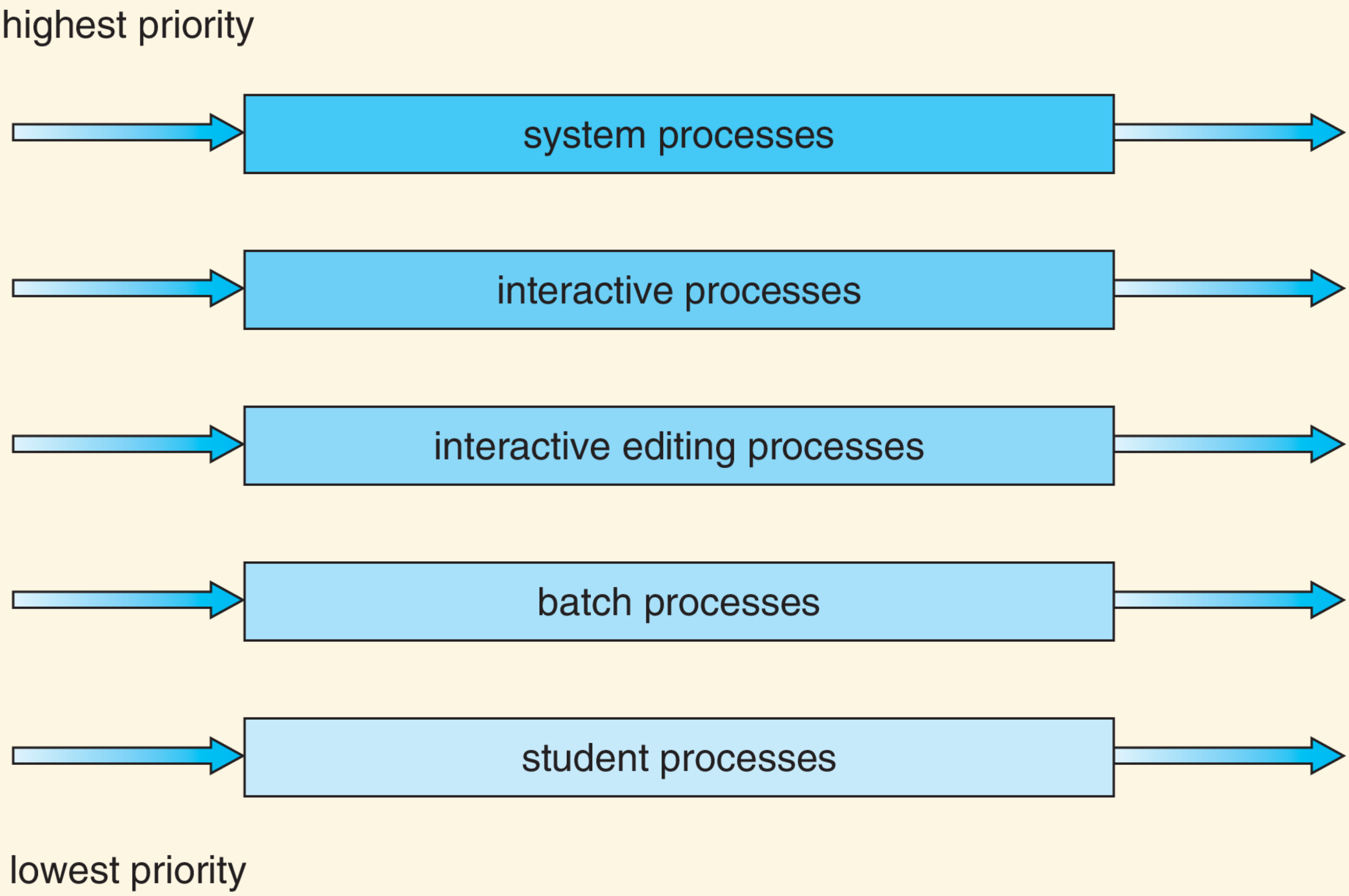
Each of these queues *has its own scheduling algorithm*


highest priority
system processes
interactive processes
interactive editing processes
batch processes
student processes
lowest priority

For example, RR scheduling may make more sense for user-facing processes, whereas background processes can simply be scheduled with FCFS

This still leaves the question of scheduling among queues (as opposed to within a specific queue)

Typically done by fixed-priority preemptive algorithm


highest priority
system processes
interactive processes
interactive editing processes
batch processes
student processes
lowest priority

Priority scheduling may be ineffective if a high-priority queue is regularly filled with processes (Recall the issue of *starvation*)

Could instead assign a fixed proportion of time to each queue -- for example, foreground processes get 80% of CPU time, and remaining 20% goes to background processes

Using a mutlilevel queue essentially turns our scheduling problem into several scheduling problems

- deciding which queue gets to run
- deciding within each queue which process will run -- each queue may have a different algorithm for this