The O(1) Linux Scheduler

Alexandre David
B2-206
Introduction

- **Scheduler (OS):** Components that decides which process to run next.
  - How to choose?
  - Processes run for *time slice* units of time = granularity.
  - Scheduler *policy* determines what runs when.

- **Different kinds of processes:**
  - IO-bound – must be responsive, wait most of the time, little computation.
  - Processor-bound – run most of the time, lots of computations.
Introduction

- Priority based scheduling: Runnable processes with time slice left and the highest priority always run.
  - Sort them?
  - Which data structure?
  - Which priority? Priority is dynamically modified to fulfill scheduling policies:
    - fast process response or
    - high process throughput.
The Scheduler

- List of runnable processes = “runqueue”.
  - 2 priority arrays: one active and one expired, swapped with pointers.

- Priority array:
  - bitmap for priorities,
  - with lists of processes per priority level.

- Finding the highest runnable process = finding the first bit set to 1.
  - Independent on the number of processes.
  - Dependent on the (fixed) number of priority levels.
Priority Bitmap

The `priority_bitmap` is a 140-bit array, where each bit represents a task's priority. Bit 0 represents the highest priority (priority 0), and Bit 139 represents the lowest priority (priority 139).

The `schedule()` function is called to schedule tasks. It starts with the `sched_find_first_set()` function to find the first set of tasks with the highest priority.

1. **Bit 0 (priority 0):** The first bit is set to indicate the highest priority task.
2. **List of all runnable tasks, by priority:** Each priority level has a list of tasks that can be run.
3. **Bit 6:** Bit 6 indicates the list of runnable tasks for priority 6.

Run first process in the list.