# Homework 3

Maksim Levental

October 2, 2013

## Problem 1

Yes I think the operating system can make the process runs for exactly its quantum. If the quantum is an integer multiple of the timer interrupt period, which is 1/f. For example if the timer interrupt period is 10ms and the quantum is 100ms. Then every 10ms the scheduler wakes up and checks how much quantum time the process has left. On the 10th time the scheduler sees that the process has no quantum time left and reschedules it.

But if the quantum is not some integer multiple of the timer interrupt period then the process will run for longer than its alotted quantum. For example if the quantum is 101ms and the timer interrupt period is 10ms. Then on the the 10th time the scheduler wakes to see how much time the process has left it does not reschedule the process, because it has 1ms left, but also does not wake again for another 10ms. So the process runs for a total of 110ms, an overage of 9ms. So the upper bound on the runtime of the process is ceilingFunction(q/(1/f))*(1/f)=ceilingFunction(q/p)*p where p = 1/f the period of the timer interrupt.

## Problem 2

For a shorter response time an application with a GUI should be given high priority. Something like a game should get the highest priority because of how critical responsiveness is to the user experience.

For high throughput a web server application should get highest priority so that remote users accessing files on the server can be served quickly and in high volume.

For increased resource utilization priority should be given to applications that are resource intensive applications like MP3 players, video recording software, Internet games.

## Problem 3

A relocation loader is a loader that adjusts memory addresses in programs in operating systems where partitioned memory management is used. This is done to compensate for variations in addresses at which the loading starts when repeat calls of the program are put into different partitions, with different start addresses.

Relocation is an alternative boundary registers because boundary registers are used to make sure that programs don't write and read to/from memory addresses that are beyond those belonging to them, that is to say beyond its own memory partition. They are also used to adjust reference addresses by adding the base address to all addresses referenced by the process in order to be inside the memory partition of the process.

## Problem 4

There are 64K/4K = 16 pages in the virtual address space of process that's running. But there is only room for 32K/4K = 8 page frames in main memory. The virtual address space goes from 0 to 64*1024 - 1= 65535 with pages having width 4096 and the first page going from 0 to 4095, the second 4096 to 8191, etc.

### a

31460/4096 = 7.68 so this virtual address is in the 7th page. This page is not in a page frame because there is no entry for the 7th page in the page table. So this instruction would cause a page fault.

### b

17400/4096 = 4.24 so this virtual address is in the 4th page. This page is in the page table as (4,4) so it's in page frame 4. The offset is 17400

### c

45100/4096 = 11.0107 so this virtual address is in the 11th page. This page is in the page table as (11,7) so its page frame is 7. The offset is 45100

### d

25000/4096 = 6.1035 so this virtual address is in the 6th page. This page is not in a page frame because there is no entry for the 7th page in the page table. So this instruction would cause a page fault.

## Problem 5