Computer Labs: Processes

2º MIEIC

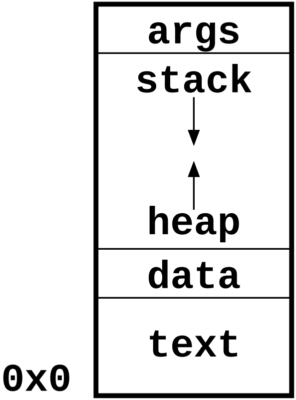
Pedro F. Souto (pfs@fe.up.pt)

October 30, 2012

(Sequential) Process

Abstracts a running program

```
int main(int argc, char *argv[], char* envp[])}
```



`args` Command line args and environment variables de ambiente.

`stack` *Activation frames/records* corresponding to function calls

`heap` Dynamically allocated memory (e.g using `malloc`)

`data` Memory allocated statically (by the compiler) (e.g. the "`Hello, World!`" string)

`text` Machine instructions

Minix is a multitasking OS

```
$ ps ax | more
  PID TTY   TIME CMD
 (-4)    ?  0:46 idle
 (-3)    ?  0:00 clock
 (-2)    ?  0:00 system
 (-1)    ?  0:00 kernel
    5    ?  0:00 pm
    7    ?  0:01 vfs
    4    ?  0:00 rs
    8    ?  0:00 memory
    9    ?  0:00 log
   10    ?  0:00 tty
    3    ?  0:00 ds
   12    ?  0:00 vm
   13    ?  0:00 pfs
    6    ?  0:00 sched
    1    ?  0:00 init
   -- more (43 in all)
```

- And so are Linux and all Windows OSs since XP (at least)

OS support multiple processes (multiprogramming)
for reasons of **efficiency**

Multiprogramming and Efficiency

Problem Processes need to access to I/O devices (monitor, keyboard, mouse, disk, network ...)

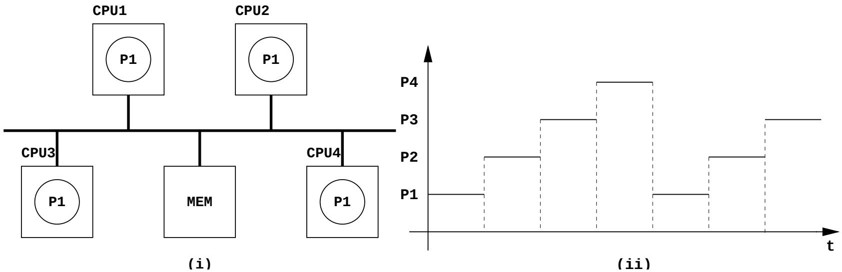
Parameter	Time
CPU cycle	1 ns (1 GHz)
Cache access	$\sim$ 2ns
Memory access	$\sim$ 10 ns
Disk access	$\sim$10 ms

Solution while a process waits for an I/O operation to complete, the OS can allocate the processor to another processor:

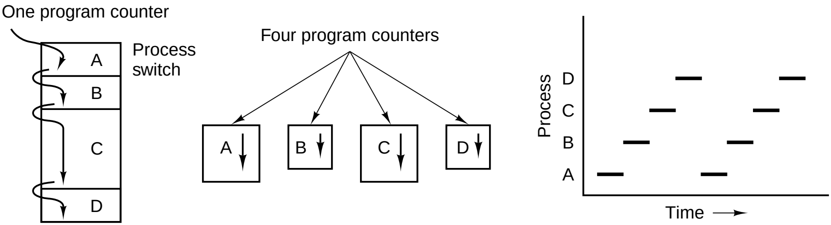
- Upon completion of the I/O operation, the I/O device can generate an interrupt

Multi-process Execution (1/2)

- In a multiprocessor/multicore system (i), each processor/core can execute a different process
- In a uniprocessor system (ii), the OS allocates the processor to the diferent processes (the processor is a resource shared by the different processes): *pseudo-parallelism*.



Multi-process Execution (2/2)

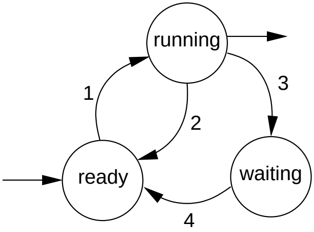


(a) (b) (c)

- The processor is shared by 4 processses;
- The OS creates the illusion that each process executes in its own CPU, i.e. that each process executes in its virtual CPU

States of a Process

- In its lifetime, a process can be in 1 of 3 states:



1. The OS allocates a CPU to the process;
2. The OS allocates the CPU to another process;
3. The process blocks waiting for some event (usually I/O)
4. An event the process was waiting for occurs

Running the CPU executes the process's instructions a executar as instruções do processo;

Waiting the process is waiting for an event (usually the end of an I/O operation)

Ready the process is waiting for the OS to allocate it a CPU, which is executing instructions of another process

Further Reading

- Sections 2, 2.1
  Andrew Tanenbaum, *Modern Operating Systems*, 2nd Ed.