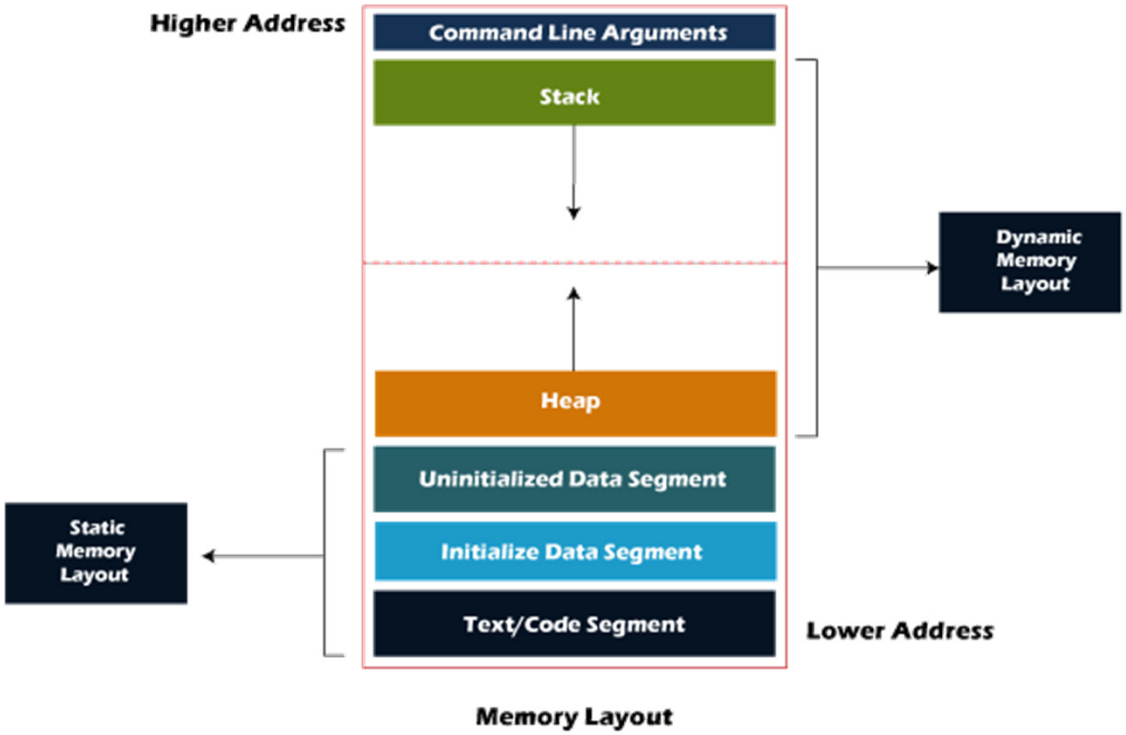
Processes

Theoretical Aspects

Stefano Quer, Pietro Laface, and Stefano Scanzio
Dipartimento di Automatica e Informatica
Politecnico di Torino
skenz.it/os stefano.scanzio@polito.it

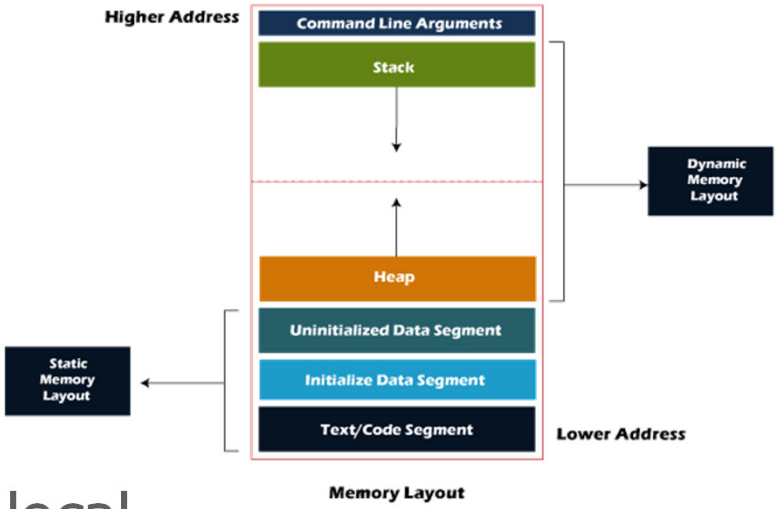
Program

- ***Algorithm:*** a logical procedure that in a finite number of steps solves a problem
- ***Program:*** formal expression of an algorithm by means of a programming language
  - Sequence of code lines
  - Static entity



Process

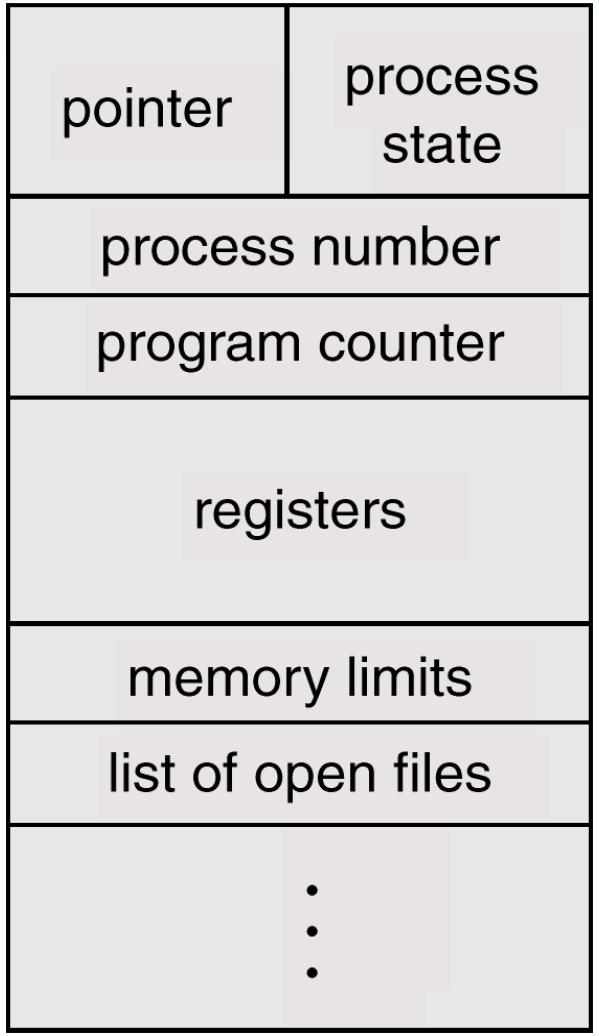
- ***Process:*** a **sequence** of operations performed by a program in execution on a given set of input data.
  - Dynamic entity
  - Program in execution (running)
    - Text area (executable code)
    - Data area (global variables)
    - Stack (function parameters and local variables)
    - Heap (dynamic variables allocated during the process execution)
    - Registers (Program counter, stack pointer, etc.)



Process Control Block (PCB)

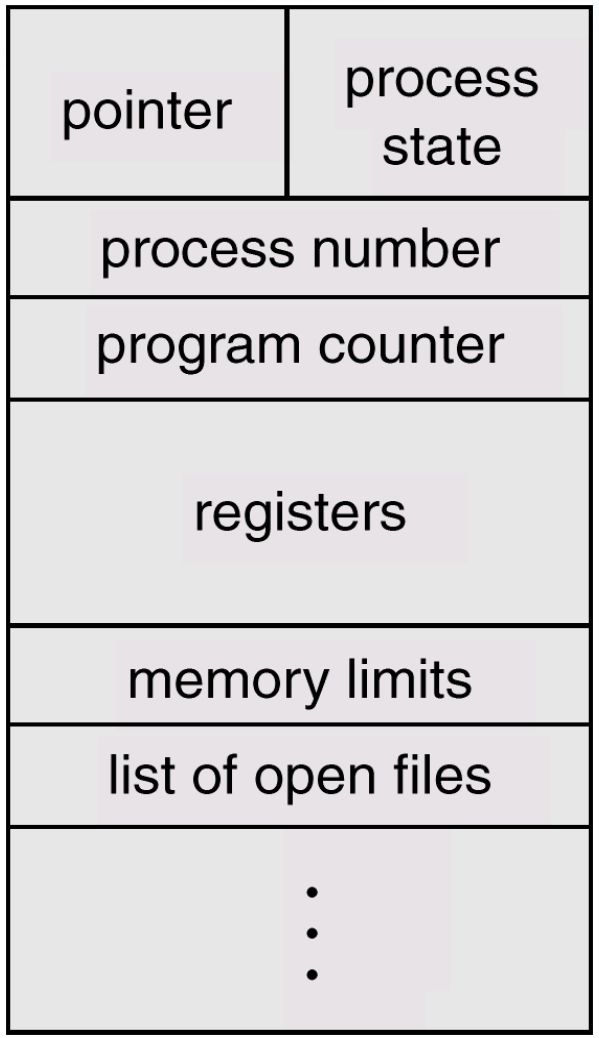
Process Control Block (PCB)

- The kernel stores for each process a set of data, e.g.,
  - The process state
    - New, Ready, Running, Waiting, Terminated
  - Copy of the CPU registers
    - Their number and type is hardware-dependent
  - The program counter
    - Address of the next instruction to be executed



Process Control Block (PCB)

- Data useful for CPU scheduling
  - Priority, pointers to queues, etc.
- Data useful for memory management
  - Base register, Limit register, Segment and paging registers, etc.
- Signal table
  - signal handlers
- Various administration data
  - CPU usage, limits, etc.
- I/O status information
  - I/O device list, open files, etc.

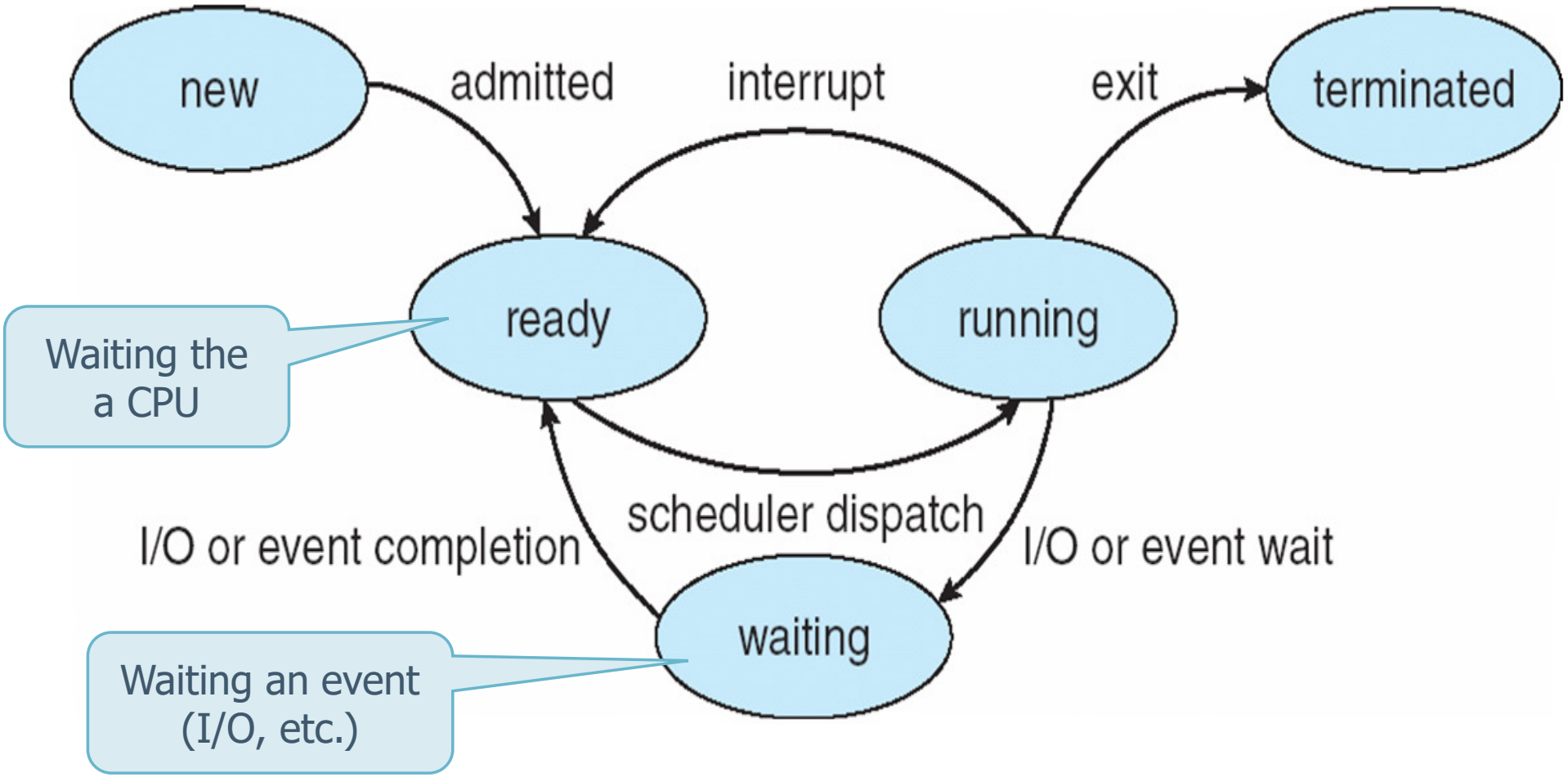


Process state

- During its execution a process change its state
  - **New**: process is created and submitted to the OS
  - **Running**: a CPU is allocated to the process (in execution)
  - **Ready**: logically ready to run, waiting that a CPU is available
  - **Waiting**: for an event or for resources
  - **Terminated**: releases the resource it is using

State diagram

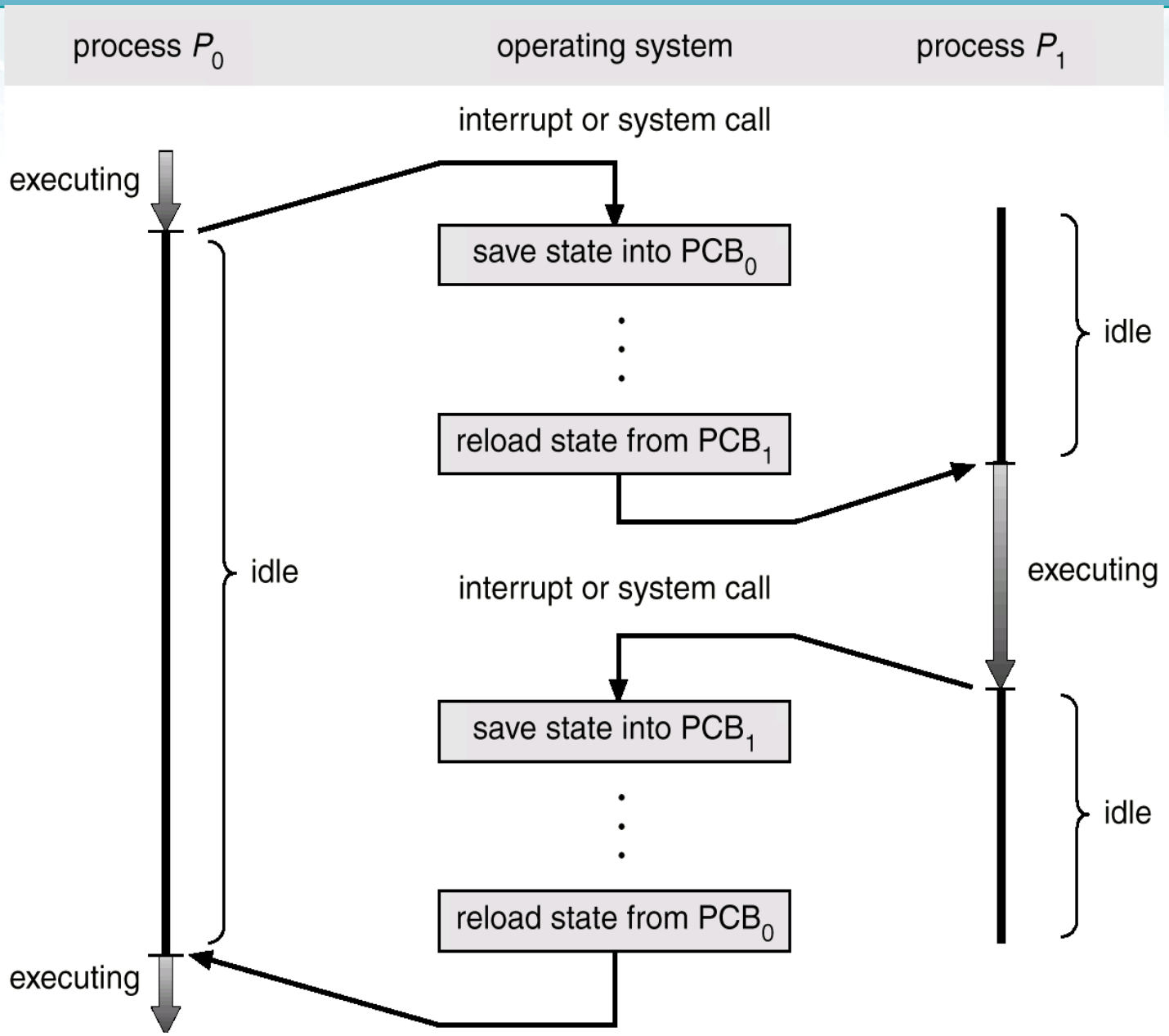
- The possible state evolution of a process is described by a state diagram



Context switching

- When the CPU is assigned to another process, the kernel
  - Save the state of the running process
  - Load the state previously saved for the new process
- The time devoted to the **context switching** is overhead, i.e., time not directly useful for any process
- The amount of time for context switching is hardware-dependent

Context switching



Process scheduling

- Multiprogramming aims at maximizing the CPU usage by processes
- Processes can be classified as
  - I/O-bound
    - Spend more time for I/0 than for computation
    - Require short CPU service times
  - CPU-bound
    - Spend more time for computation than for I/0
    - Require long CPU service times

CPU scheduler

- The context switching operations are controlled by scheduler of the CPU
  - The goal of the scheduler is to
    - Maximize the CPU usage (by means of the processes)
    - Attempting to satisfy various system-level requests timely (hardware and software interrupts)
  - Ultimately, the scheduler is in charge of
    - Determine when the current process should finish its execution
    - Select the next process to run from the available processes

Continues in section u09s01