

# Parallel and Distributed Systems

Federico Matteoni

A.A. 2021/22

# Index

|     |                        |   |
|-----|------------------------|---|
| 0.1 | Introduction . . . . . | 2 |
|-----|------------------------|---|

## 0.1 Introduction

Prof.: Marco Danelutto

**Program** Techniques for both parallel (single system, many core) and distributed (clusters of systems) systems. Principles of parallel programming, structured parallel programming, parallel programming lab with standard and advanced (general purpose) **parallel programming frameworks**.

**Technical Introduction** Each machine has more cores, perhaps multithreaded cores, but also GPUs (maybe with AVX support, which support operations floating point operations, **flops**, in a single instruction).

Between 1950 and 2000 the VLSI technology arised, integrated circuits which nowadays are in the order of 7nm (moving towards 2nm): printed circuits!

In origin, everything happened in a single clock cycle: fetch, decode, execute, write results in registers, with perhaps some memory accesses. Then we had more complex control where in a single clock cycle we do just one of the phases (fetch *or* decode *or*...), like a **pipeline**. More components are used the higher the frequency but the more power we need to dissipate, and we're coming to a point where the power we need to dissipate is too much and risks to melt the circuit, so we're reaching a **physical limit** in chip miniaturization. But temperature and computing power do not go in tandem: computing power is proportional to the chip dimensions, while temperature is proportional to the area. So it's better to put more processors (**cores**) and let them work together rather than make a bigger single processor. An approach is to have few powerful cores and more less powerful cores (for example, in the Xeon Phi processors). Now, the processors follow this architecture, with the performance of a single core decreasing a bit with every generation but it's leveled by adding more cores.

Up to the 2000, during the single core era, code written years before will run faster on newer machines. Now, code could run slower due to not exploiting more cores and the decreasing in performance of the single core.

With accelerators the situation is even more different: for example GPUs, accelerator for graphics libraries, with their own memory and specialized in certain kinds of operations.