Principles of Distributed Computing
Roger Wattenhofer

# Chapter 0

# Introduction

What is Distributed Computing?

- Many processors/nodes active in system at any moment.
- Nodes coordinate or operate without interfering.
- Nodes have certain degree of freedom: own hardware, own code, sometimes own independent task.
- Nevertheless: there will still be sharing of some common resources and information: coordinating is necessary.
- Almost every system today is distributed! Many applications in networking or distributed systems, e.g. P2P, sensor networks, multi-core architectures, etc.
- Foundations of course: algorithms, graph theory, etc.
- We study pearls of distributed computing and network algorithms!

Lots of parameters and models ("coupling level" Parallel Computer ⇔ Internet)

- Synchronous ⇔ Asynchronous
- Homogeneous ⇔ Heterogeneous
- Shared Memory ⇔ Message Passing
- Fast Interconnection Network ⇔ Best-Effort in Graph
- Global Task ⇔ Autonomous Agents
- Correct ⇔ Failures ⇔ Byzantine
- Altruistic ⇔ Selfish ⇔ Malicious
- Concrete models: PRAM, Broadcast or Graph Message Passing

Important issues and paradigms

- Communication: Does not come for free; often dominates local processing and storage (sometimes assumed that local processing is free).
- Incomplete Knowledge: Node does not know what other nodes do. Do nodes know the topology of the network, or just neighbors, and/or just number of nodes in system?
- Failures: Even in the presence of failures of some nodes, the system as a whole may survive. A major reason to build a distributed system.
- Asynchrony: Synchronous (send, receive, compute); asynchronous: algorithms are event driven; there are models in between. Non-determinism through asynchrony.
- Congestion: Storing the information at one node does not scale.
- Locality: Networks are growing. Global information is not always needed; often sufficient when nodes talk to neighbors. Saves costs, and can be simpler (cf., distance-vector routing).
- (Software/Programming)
- (Security)