Introduction to Distributed Algorithms

Etienne Renault

October 2, 2020

`https://www.lrde.epita.fr/~renault/teaching/algorep/`

Forewords

A distributed system is a collection of independent computers that appears to its users as a single coherent system

Andrew S. Tanenbaum

A distributed system is one in which the failure of a computer you didn't even know can rend you own computer unusable.

Leslie Lamport

What is a distributed system?

A distributed system is:

- a collection of autonomous computers
- connected through a network
- which enables computers to coordinate their activities
- so that users perceive the system as a single one

Example of Distributed Systems

- Grid/Cluster Computing

Example of Distributed Systems

- Grid/Cluster Computing
- World Wide Web

Example of Distributed Systems

- Grid/Cluster Computing
- World Wide Web
- Network File Server

Example of Distributed Systems

- Grid/Cluster Computing
- World Wide Web
- Network File Server
- Banking Network

Example of Distributed Systems

- Grid/Cluster Computing
- World Wide Web
- Network File Server
- Banking Network
- Peer-to-peer Network

Example of Distributed Systems

- Grid/Cluster Computing
- World Wide Web
- Network File Server
- Banking Network
- Peer-to-peer Network
- Process Control System

Example of Distributed Systems

- Grid/Cluster Computing
- World Wide Web
- Network File Server
- Banking Network
- Peer-to-peer Network
- Process Control System
- Sensors Network

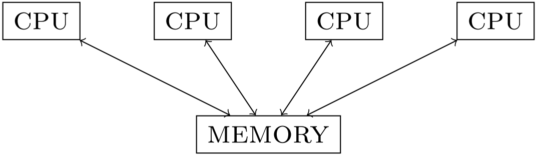
Parallel versus Distributed?

- Parallel architecture

- Distributed architecture

Parallel versus Distributed?

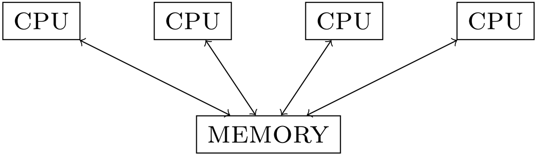
- Parallel architecture



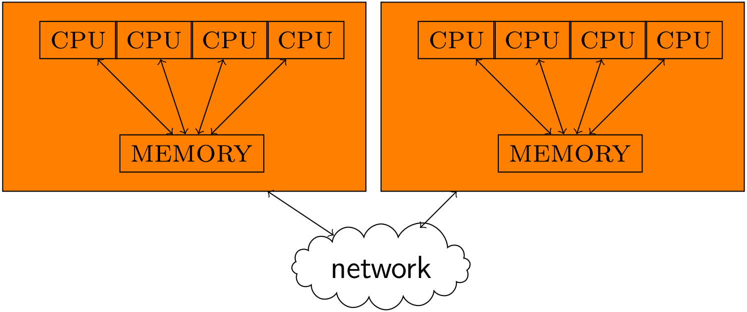
- Distributed architecture

Parallel versus Distributed?

- Parallel architecture



- Distributed architecture



Remark

From a distributed system point of view no real difference between parallelism and distributed system (more details later in this lecture).

Pitfalls in Distributed Systems!

1. The network is reliable

Pitfalls in Distributed Systems!

1. The network is reliable
2. The network is secure

Pitfalls in Distributed Systems!

1. The network is reliable
2. The network is secure
3. The network is homogeneous

Pitfalls in Distributed Systems!

1. The network is reliable
2. The network is secure
3. The network is homogeneous
4. The topology does not change

Pitfalls in Distributed Systems!

1. The network is reliable
2. The network is secure
3. The network is homogeneous
4. The topology does not change
5. Latency is zero

Pitfalls in Distributed Systems!

1. The network is reliable
2. The network is secure
3. The network is homogeneous
4. The topology does not change
5. Latency is zero
6. Brandwidth is infinite

Pitfalls in Distributed Systems!

1. The network is reliable
2. The network is secure
3. The network is homogeneous
4. The topology does not change
5. Latency is zero
6. Brandwidth is infinite
7. Transport cost is zero

Pitfalls in Distributed Systems!

1. The network is reliable
2. The network is secure
3. The network is homogeneous
4. The topology does not change
5. Latency is zero
6. Brandwidth is infinite
7. Transport cost is zero
8. The is one administrator

Challenges distributed systems?

- Scalability

Challenges distributed systems?

- Scalability
  - Avoid Bottlenecks, Good Performances, Physical Ressources

Challenges distributed systems?

- Scalability
  - Avoid Bottlenecks, Good Performances, Physical Ressources
- Heterogeneity

Challenges distributed systems?

- Scalability
  - Avoid Bottlenecks, Good Performances, Physical Ressources
- Heterogeneity
  - Network, Hardware, Operating Systems, Programming Languages, Implementation

Challenges distributed systems?

- Scalability
  - Avoid Bottlenecks, Good Performances, Physical Ressources
- Heterogeneity
  - Network, Hardware, Operating Systems, Programming Languages, Implementation
- Concurrency

Challenges distributed systems?

- Scalability
  - Avoid Bottlenecks, Good Performances, Physical Ressources
- Heterogeneity
  - Network, Hardware, Operating Systems, Programming Languages, Implementation
- Concurrency
  - Managing shared ressources

Challenges distributed systems?

- Scalability
  - Avoid Bottlenecks, Good Performances, Physical Ressources
- Heterogeneity
  - Network, Hardware, Operating Systems, Programming Languages, Implementation
- Concurrency
  - Managing shared ressources
- Failures

Challenges distributed systems?

- Scalability
  - Avoid Bottlenecks, Good Performances, Physical Ressources
- Heterogeneity
  - Network, Hardware, Operating Systems, Programming Languages, Implementation
- Concurrency
  - Managing shared ressources
- Failures
  - Detect, Masking, Tolerating, Recovery, Redundancy

Challenges distributed systems?

- Scalability
  - Avoid Bottlenecks, Good Performances, Physical Ressources
- Heterogeneity
  - Network, Hardware, Operating Systems, Programming Languages, Implementation
- Concurrency
  - Managing shared ressources
- Failures
  - Detect, Masking, Tolerating, Recovery, Redundancy
- Communications

Challenges distributed systems?

- Scalability
  - Avoid Bottlenecks, Good Performances, Physical Ressources
- Heterogeneity
  - Network, Hardware, Operating Systems, Programming Languages, Implementation
- Concurrency
  - Managing shared ressources
- Failures
  - Detect, Masking, Tolerating, Recovery, Redundancy
- Communications
  - Synchronous, Asynchronous

Challenges distributed systems?

- Scalability
  - Avoid Bottlenecks, Good Performances, Physical Ressources
- Heterogeneity
  - Network, Hardware, Operating Systems, Programming Languages, Implementation
- Concurrency
  - Managing shared ressources
- Failures
  - Detect, Masking, Tolerating, Recovery, Redundancy
- Communications
  - Synchronous, Asynchronous
- Topology

Challenges distributed systems?

- Scalability
  - Avoid Bottlenecks, Good Performances, Physical Ressources
- Heterogeneity
  - Network, Hardware, Operating Systems, Programming Languages, Implementation
- Concurrency
  - Managing shared ressources
- Failures
  - Detect, Masking, Tolerating, Recovery, Redundancy
- Communications
  - Synchronous, Asynchronous
- Topology
  - Hierarchical, Decentralized, Ring, Centralized

This class

1. Theoretical, mathematical viewpoint
2. Show some classical solutions and compute complexity
   - Define distributed computing environments
   - Define abstract problems
   - Describe algorithms that solve the problems
   - Analyze complexity
   - Present Impossibility results
3. Practical through a **project** (team of 3)

Objectives

Making you familiar with distributed algorithms since you will use them regardless the compagny you target!

Model Assumptions

- IPC methods: shared memory or message-passing
- Timing
  - Synchronous: rounds
  - Asynchronous: arbitrary speed
  - Partially synchronous models: with timing assumptions with bounds on messages delays, processors speeds and clock rates
- Failures
  - Processors: stoping, byzantine
  - Communication: loss, duplication, failure, recovery

Topic Overview

1. Synchronous model
   - Basic,easy to program
   - Not realistic, but sometimes emulate worse-behaved networks
   - Impossibility for synchronous network carry over worse networks
2. Asynchronous model
   - Realistics but hard to cope with
3. Partially synchronous model
   - Somewhere in between