Sistemas Distribuídos de Larga Escala (Large Scale Distributed Systems) Class Information

1º M.EIC

Carlos Baquero (cbm@fe.up.pt)
Pedro F. Souto (pfs@fe.up.pt)

October 18, 2021

Staff

- Carlos Baquero (cbm@fe.up.pt)
- Pedro Ferreira do Souto (pfs@fe.up.pt)

Context

Distributed System (Def.) A system with two or more processes:

1. executing on different computers
2. communicating via messages
   - with a no negligible delay (wrt computation)

Large Scale System A system with:

Thousands processes likely in different data centers

Geographically distributed high latency communication

Heavy load up to millions of request per second

Several administrative domains

Think about:

- Some services on the Web (Gmail, Google Search, Amazon, Netflix, Facebook)
- The software infrastructure of cloud providers (Amazon, Microsoft, Google, IBM)
  - Cloud-based applications
- Peer-to-peer systems (Bitcoin)

Objectives

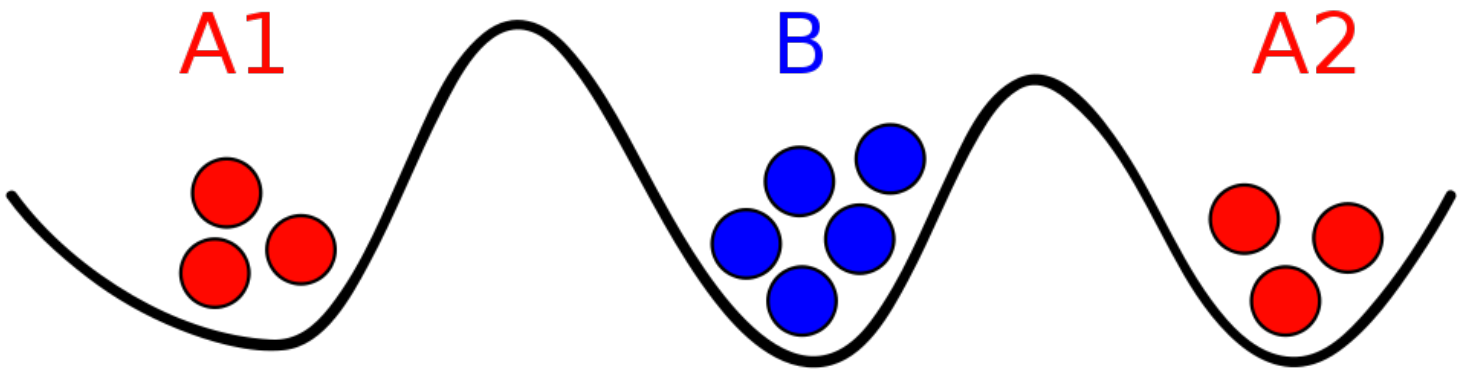
1. Understand the foundations of distributed computing;
2. Be able to:
   - design and implement distributed large-scale applications for partition-prone and high latency networks;
   - analyse distributed solutions and evaluate their fitness to the problem at hand.

Two (gangster) Generals Paradox

A group of gangsters are about to pull off a big job. The plan of action is prepared down to the last detail. Some of the men are holed up in a warehouse across town, awaiting precise instructions. It is absolutely essential that the two groups act with complete reliance on each other in executing the plan.

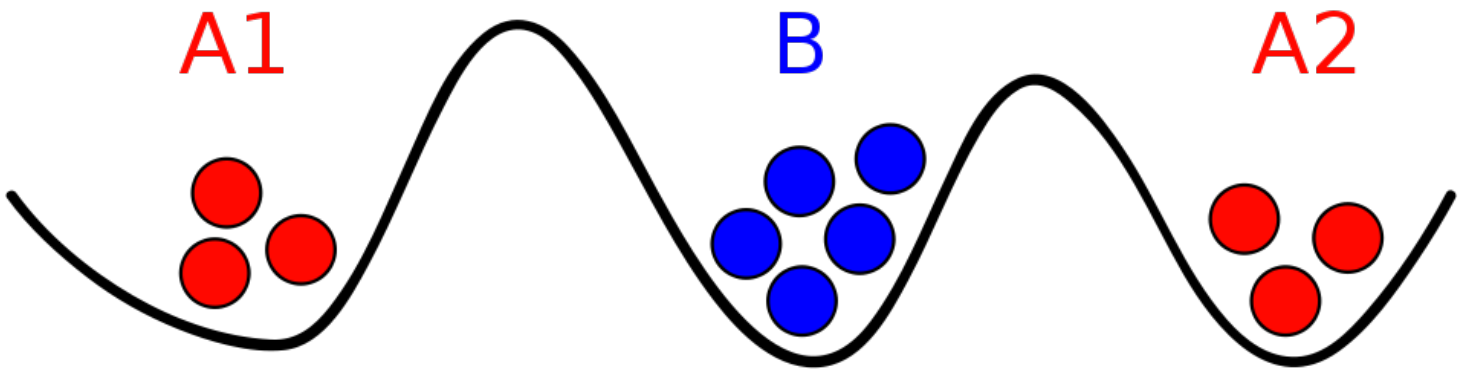
in “Some Constraints and Trade-offs in the Design of Network Communications”. Akkoyunlu, Ekanadham and Huber. 1975.

Two (gangster) Generals Paradox



- Red gangsters have more mobsters if together, but need to attack at the same time
- Messengers are unreliable
- How to coordinate an attack?

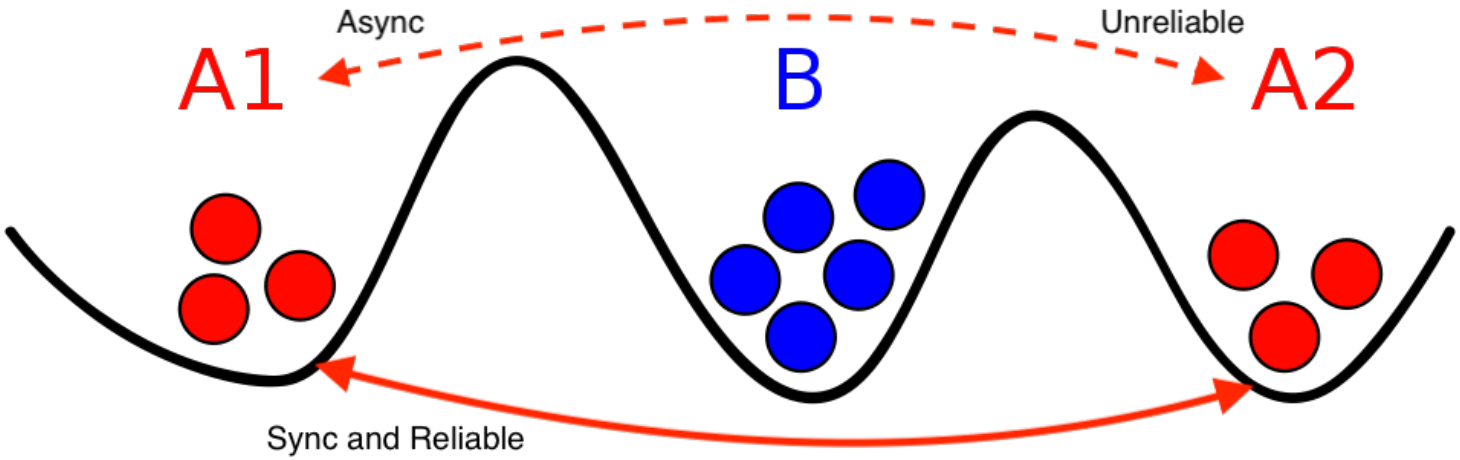
Two (gangster) Generals Paradox



- Red gangsters have more mobsters if together, but need to attack at the same time
- Messengers are unreliable
- How to coordinate an attack?

`http://en.wikipedia.org/wiki/Two_Generals'_Problem`

Two (gangster) Generals Paradox



- The battleground is shared unreliable medium
- Build a private reliable tunnel
- Problem (mostly) solved

Synchronous model

Expected in small scale and cost dominated. Ex: Cars subsystems

- Processing delays have a known bound
- Message delivery delays have a known bound
- Rate of drift of local clocks has a known bound

Asynchronous model

Adapted to large scale and can spread costs. Ex: Internet

- Processing delays are unbounded or unknown
- Message delivery delays are unbounded or unknown
- Rate of drift of local clocks is unbounded or unknown

Consistency Models

Eventually Consistent. CACM 2009, Werner Vogels

- In an ideal world there would be only one consistency model: when an update is made all observers would see that update.
- Building reliable distributed systems at a worldwide scale demands trade-offs between consistency and availability.

CAP theorem. PODC 2000, Eric Brewer

Of three properties of shared-data systems – data consistency, system availability, and tolerance to network partition – only two can be achieved at any given time.

Both strong and eventual consistency are covered in the course

Prerequisites

- Distributed Systems
  - Operating Systems – concurrency
  - Computer Networks
  - Programming

Syllabus: Part 1/3 - Communication and Processing for Large Scale

Message Oriented Middleware

- Message queues
- Publish-Subscribe

Processing

Quorum systems

- Linearizability (strong consistency)

Syllabus: Part 2/3 - Large Scale Systems

Scalable Distributed Topologies

- Graphs, Gossip, Broadcast

System Design for Large Scale

- P2P, Cloud and planetary systems

Physical and Logical Time

High Availability under Eventual Consistency

- Local First and Convergence

Syllabus: Part 3/3 - Large Scale Byzantine Algorithms

Byzantine Generals

Byzantine Quorums

Distributed Ledgers

- Blockchain

Supporting Material

Textbook

van Steen, M. and Tanenbaum, A. S.
Distributed Systems: Principles and Paradigms, 3rd Ed. (2017)
(available for free upon request)

Small fun "book" focused on the data center

Distributed Systems: for fun and profit.

Topologies and Networks

Duncan J. Watts Small Worlds

Large scale systems

Alex Petrov Database Internals

Martin Kleppmann Designing Data-Intensive Applications

(Mini-)Projects

1. Reliable Publish-Subscribe System
   - Groups of 3/4 students
   - Due date: November 26 @ 20:00 (Friday)
   - Must be built on top of ZeroMQ
   - Report
2. Peer-to-peer Based Timeline Application
   - Groups of 3/4 students
   - Due date: January 21 @ 20:00 (Friday)
   - Presentation: during the last week of the semester

Note 0 You can use your favourite programming language.

Note 1 Both projects and their report/presentation must be submitted via GitLab@FEUP

- We'll create the Git projects and will assign them to you.

Note 2 Both projects have the same weight.

(Mini-)Projects: Grading

- Grading is individual
- We grade each project assuming the expected number of group members (4)
- To that grade we apply a **contribution factor** computed from the contribution using a piecewise linear function:
  - "Breaking point": 25%
  - Factor: 0 at 0%, 1 at 25%, 1.15 at 100%
- You are expected to perform peer-evaluation

Exam

Exam

- In Moodle
  - True/False questions
  - Multiple-choice questions
  - Open questions
- Closed books **with cheat sheet**
  - A4 (both sides)
  - **Handwritten by yourself**

  Failure to comply means your exam will be **nullified**.

Final Grade

All students

$$G = min(0.45P + 0.45F + 0.1C, F + 3, P + 3)$$

where:

- G course final grade
- P average of the grades in both projects ($P \geq 10$)
- C class participation
- F final exam grade ($F \geq 10$)

The final grade cannot exceed in more than 3 points (in 20) both the projects' average grade and the final exam grade

Academic Integrity

- You are allowed to discuss the projects
  - For each project, there will be a discussion forum on Moodle
- **But** all code (and other artifacts) submitted for evaluation should be either:
  - Authored by the group members
  - Or authorized by us, **and** due credit should be given both in the report and in the source file (in the case of code).
- We will use tools to automatically detect plagarism
- Check out the Declaração de Princípios sobre a Integridade Académica na UP

Thank you!
Questions?

Announcements

Lectures start 10 minutes after the hour

- 16:10 on Mondays and Thursdays

Labs start next week (10 minutes after the hour)

Course material available on the course's team on MS Teams

Important dates:

	Due date
1st project	November 26 @ 20:00 (Friday)
2nd project	January 21 @ 20:00 (Friday)

Presentation of 2nd project in the last week.