Impact of the dynamic in distributed systems

Pierre Sens

Delys
LIP6 (Sorbonne Université/CNRS)

Journées non thématiques GDR RSD

Pierre.Sens@lip6.fr

Outline

- Traditional (static) distributed systems
- Modeling dynamic systems
- Causal broadcast and Leader election in a dynamic systems

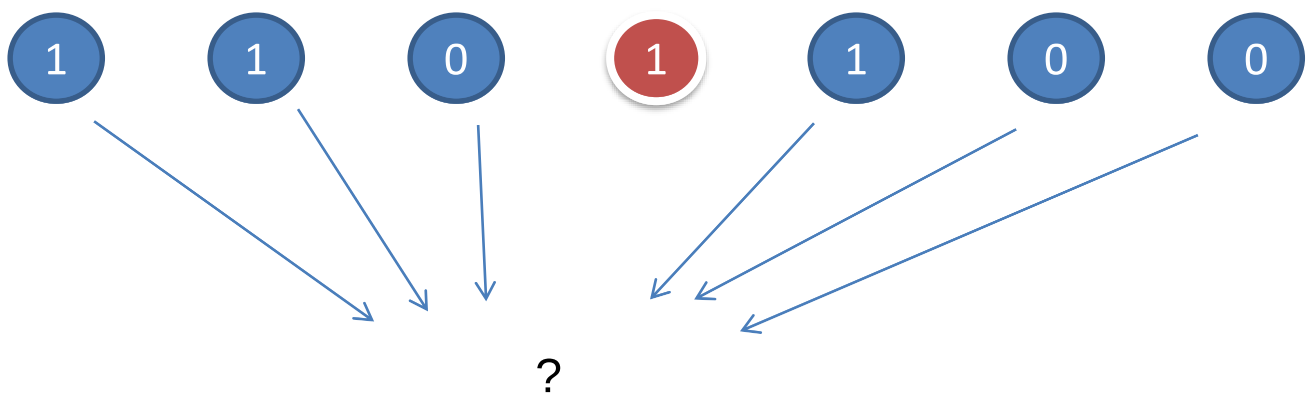
Traditional assumptions

- Connectivity
  - $\pi = \{p1, p2, .., pn\}$ **known processes**
  - n processes strongly connected (**no partition**)
- Time
  - Synchronous (known bound on transmission delays)
  - Asynchronous (no bound)
- Failure
  - processes : crash, omission, byzantine
  - links : reliable, fair lossy, unreliable

A fundamental result

- "Impossibility to solve deterministically the **consensus** in a asynchronous networks with only 1 crash failure" **[Fischer-Lynch-Paterson 85]**

- *The idea*: impossible to distinguish faulty hosts from slow ones



Circumvent FLP impossibility

3 approaches:

- Probabilistic (probabilistic consensus, e.g., Ben-Or)
  - Possibly no termination
- Partial synchrony
  - Add assumptions on the network
  - Eg, There is an unknown bound on the transmission delay
- **Unreliable failure detectors (Chandra, Toueg 91)**
  - an oracle per node provides unreliable information on correct processes

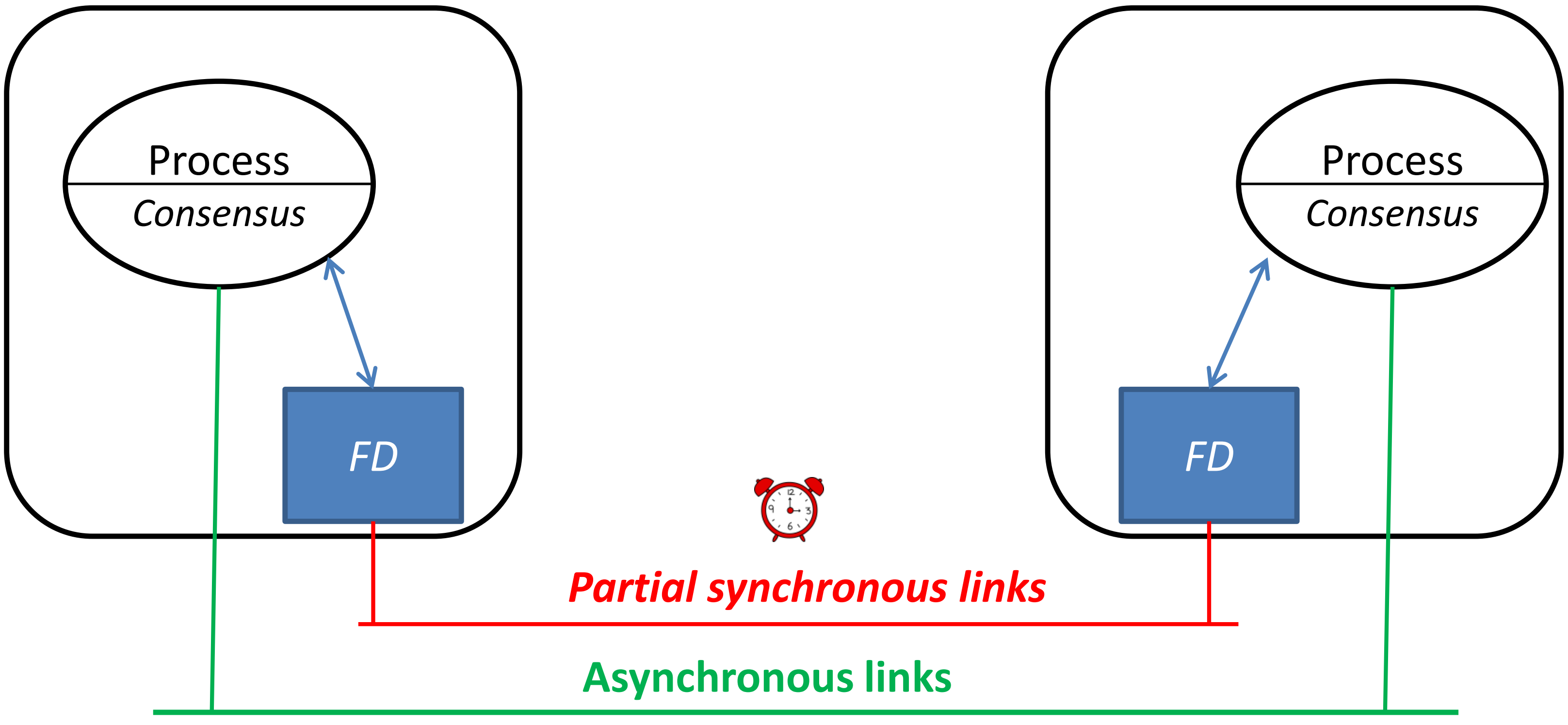
Unreliable FD: Eventual leader

Ω : Output only **one trusted process**, the eventual leader

The leader is eventually the **same correct** process for every correct process

Ω is the weakest FD to solve consensus with a **majority of correct** processes (eg. Paxos)

Implementation of FDs



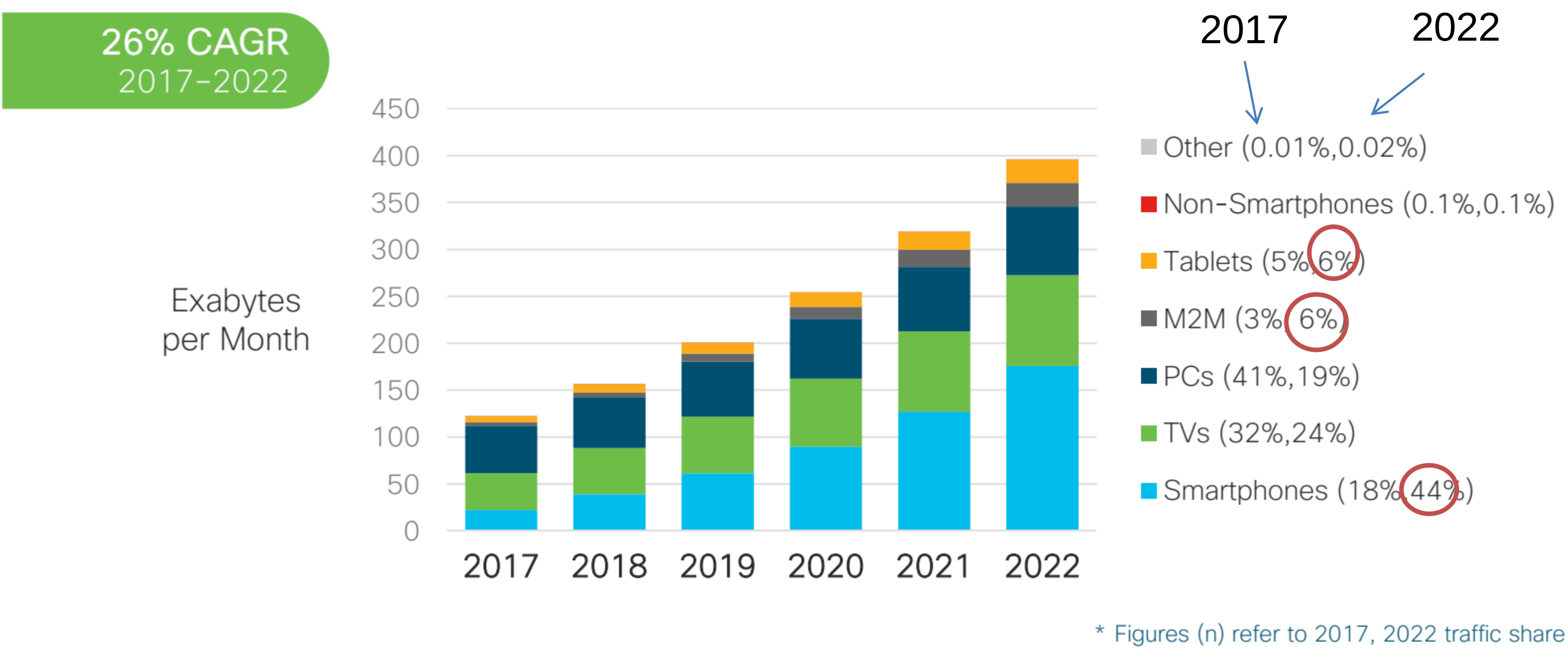
Limits of current implementations

Many implementations of FD target

- **static** systems
  - Membership (set of nodes) is initially set (no arrival)
- **known** topology
  - No change in the topology (no movement)

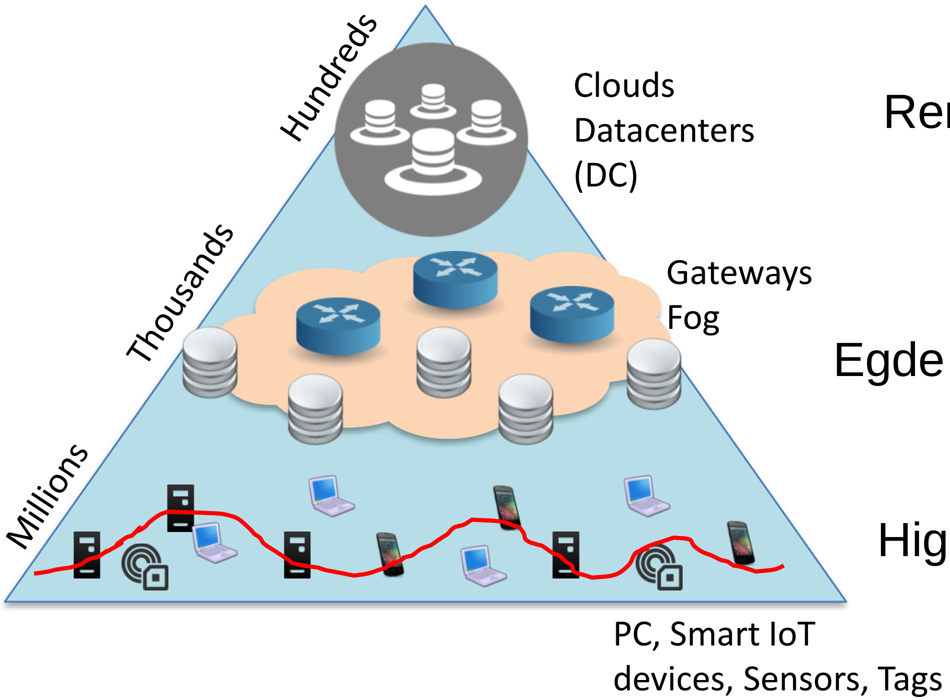
Distributed systems are more and more dynamic

- In 2022, mobile devices will account for a half of global internet traffic



Source: Cisco VNI Global IP Traffic Forecast, 2017–2022

New distributed architectures



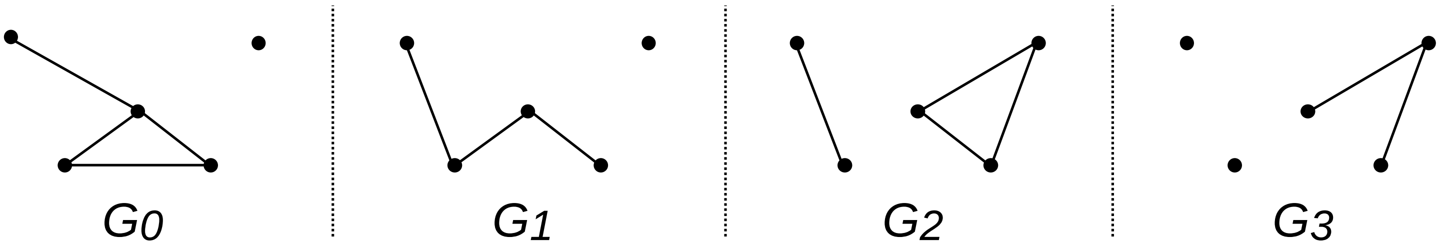
Remote datacenters

Egde and local datacenters

Highly dynamic networks

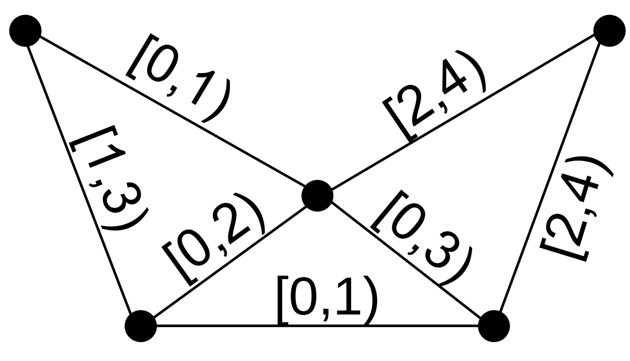
Models for dynamic systems

- Sequence Based [B. Bui-Xuan, A. Ferreira, A. Jarry, JFCS 2003]

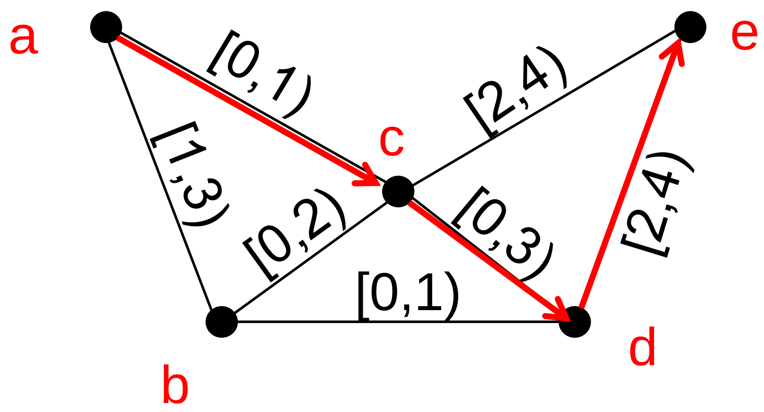


$\mathbb{G} = G_0, G_1, G_2, G_3, \ldots G_i, \ldots, \; i \in \mathbb{N}$

- Time varying graphs (TVG) [A. Casteigts, P. Flocchini, W. Quattrociocchi, N. Santoro, 2012]



TVG: Basic Properties



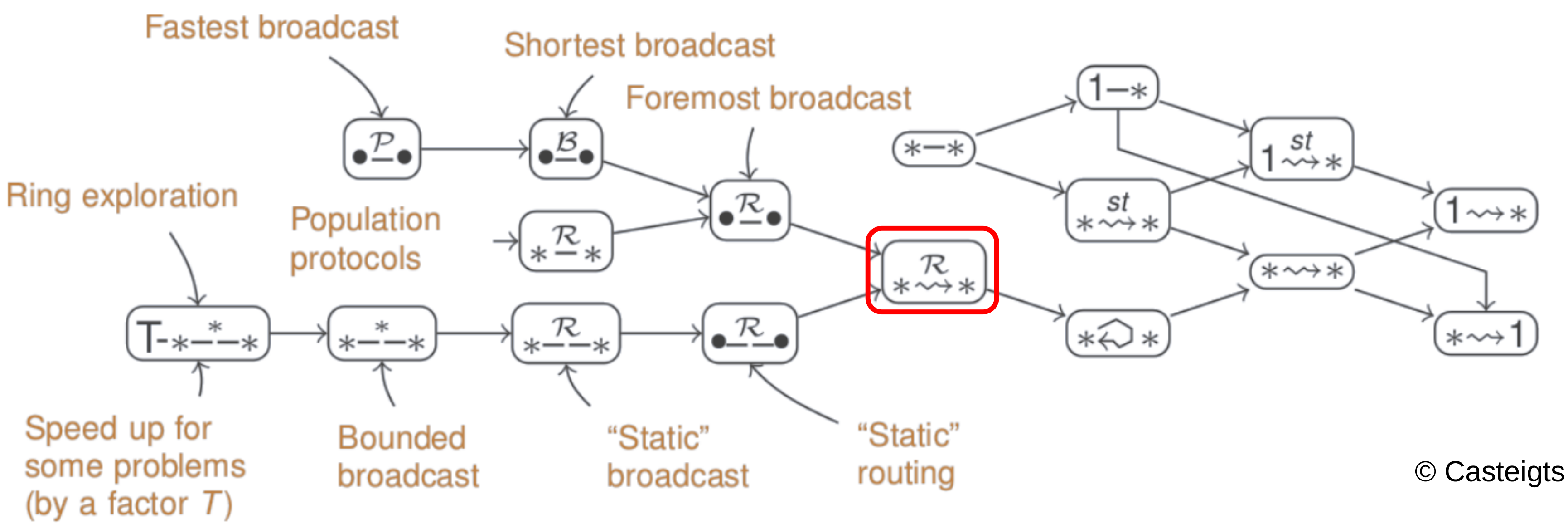
- *Temporal path (a.k.a Journey), e.g., a ↝ e*

  a ↝ *, b ↝ *, c ↝ *, d ↝ *, except e!

- 1 ↝ * $\exists u \in V, \forall v \in V, u \leadsto v$
- * ↝ 1 $\forall u \in V, \exists v \in V, u \leadsto v$
- * ↝ * $\forall u, v \in V, u \leadsto v$

TVG: Classes

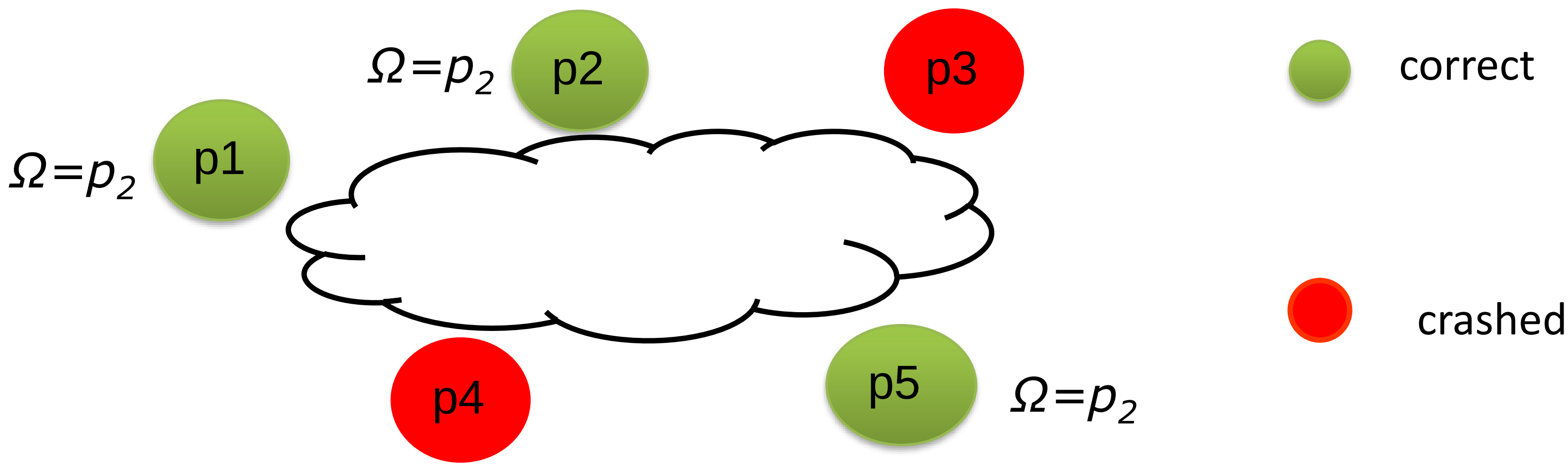
- $u \overset{\mathcal{P}}{\leadsto} v$ - Periodic journey
- $u \overset{\mathcal{B}}{\leadsto} v$ - Bounded journey
- $u \overset{\mathcal{R}}{\leadsto} v$ - Recurrent journey



- What assumption for what problems

Eventual leader election
(Ω : omega failure detector)

- There is a time after which **every correct process** always trusts **the same correct proc**ess



Luciana Arantes[1], Fabiola Greve[2], Véronique Simon[1], and Pierre Sens[1]

LIP6, Inria, France, Federal University of Bahia (UFBA), Brazil [2]

Assumption

- Communication
  - Channels are **fair-lossy**
  - there is no message duplication, modification or creation
- The system is **asynchronous**
  - There are no assumptions on the relative speed of processes nor on message transfer delays.
- Failure model : **crashes**
- The membership is **unknown**
  - A node is not aware about the set of nodes nor the number of them.

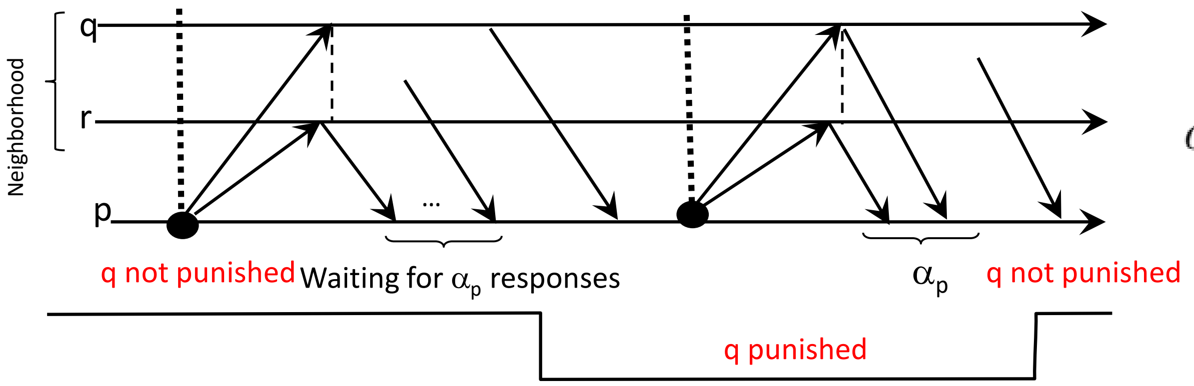
Dynamics of the network

- Dynamic changing topology
  - join/leave of nodes,
  - mobility of nodes, failure of nodes (crash)

- Network connectivity
  - Eventually, the *TVG* is connected over the time
    - There exists a journey between all stable nodes at any time
    - **Network recurrent connectivity (class $* \overset{\mathcal{R}}{\rightsquigarrow} *$)**

An Eventual Leader Election Algorithm

- Principle
  - Election of a leader process based on <u>**punishment**</u>
    - Round counter to control the freshness of the information
  - Periodic local **query-response** exchange
    - Wait for α responses
      - If *q* is <u>**locally known**</u> by *p*, <u>**has not moved**</u>, and <u>**does not respond**</u> to a query of *p* among α_p first responses, *q* is punished by *p*.



$$\alpha_i = |N_i^t| - f_i + 1$$

Ω on dynamic networks

- Each node maintains 3 sets:
  - local_known: the current knowledge about its neighborhood
  - global_known: the current knowledge about the membership of the system

  => set of tuples **<round, node id>**

  - punish: a set of tuples <punish counter, node id>

  leader: the process with the smallest counter in punish set

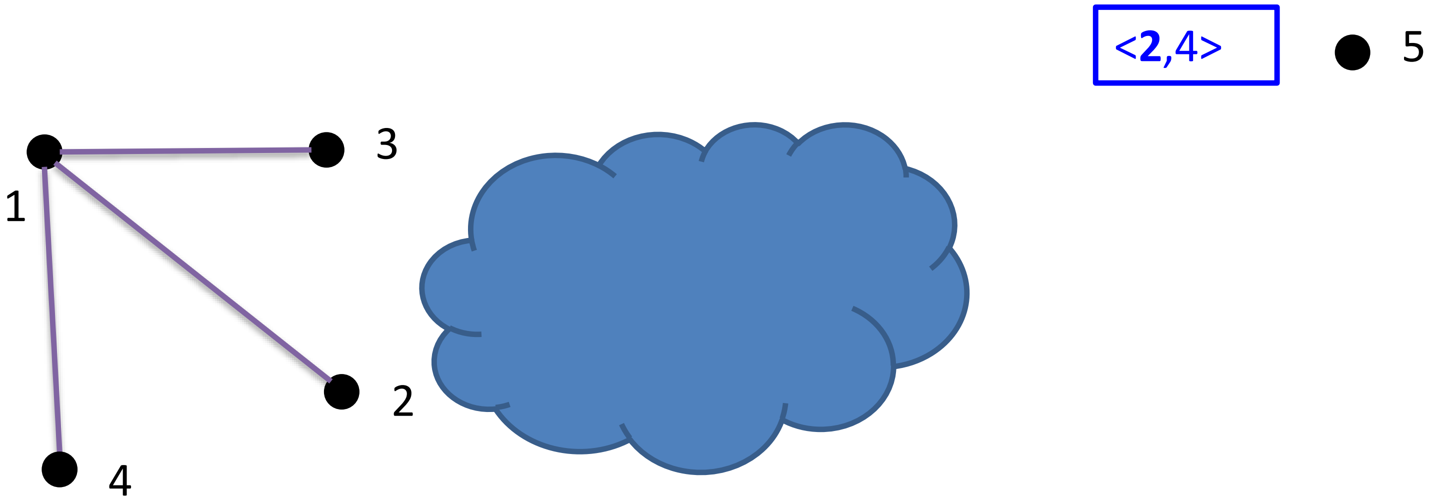
- Diffusion of information over the network by p:
  - *p*'s current round counter
  - set of processes punished by *p*
  - current knowledge of *p* about the membership of the system

Exemple: Mobility of nodes

global_known$_1$ <1,2>,<1,3>,**<2,4>**

punished$_1$ <0,1>,<0,2>,<0,3>,<**3**,4>

local_known$_1$ <1,1>,<1,2>,<1,3>,<1,4>



x:<x,4> in local_known$_1$ < y:<y,4> in global_known$_1$

1 stops punishing 4

Additional properties to ensure eventual election

- *Stable Termination Property (SatP)*:
  - Each *QUERY* must be received by at least one stable and known node

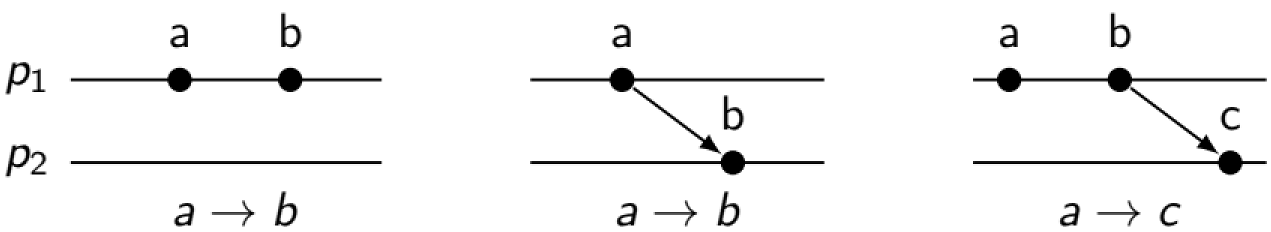
    Necessary for the diffusion of the information

- *Stabilized Responsiveness Property (SRP):*
  - There exists a time *t* after which all nodes of *p* 's neighborhood receive, to every of their queries, a response from *p* which is always among the first responses

    ***SRP* should hold for at least one *stable* known node (the eventual leader)**

Causal broadcast

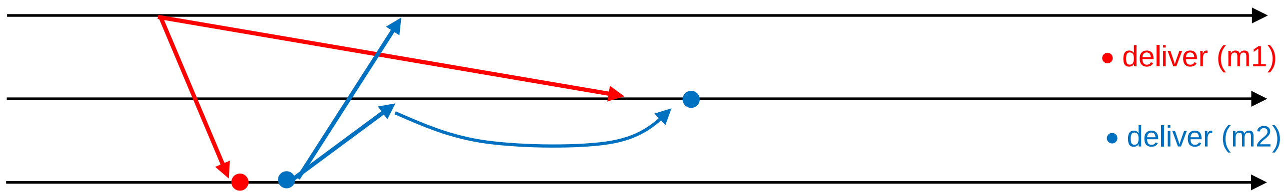
- **Causal order** is defined by the **Happened-Before** relation, which orders events following three rules:



- **Causal broadcast**

Processes **deliver** each message exactly once in causal order:

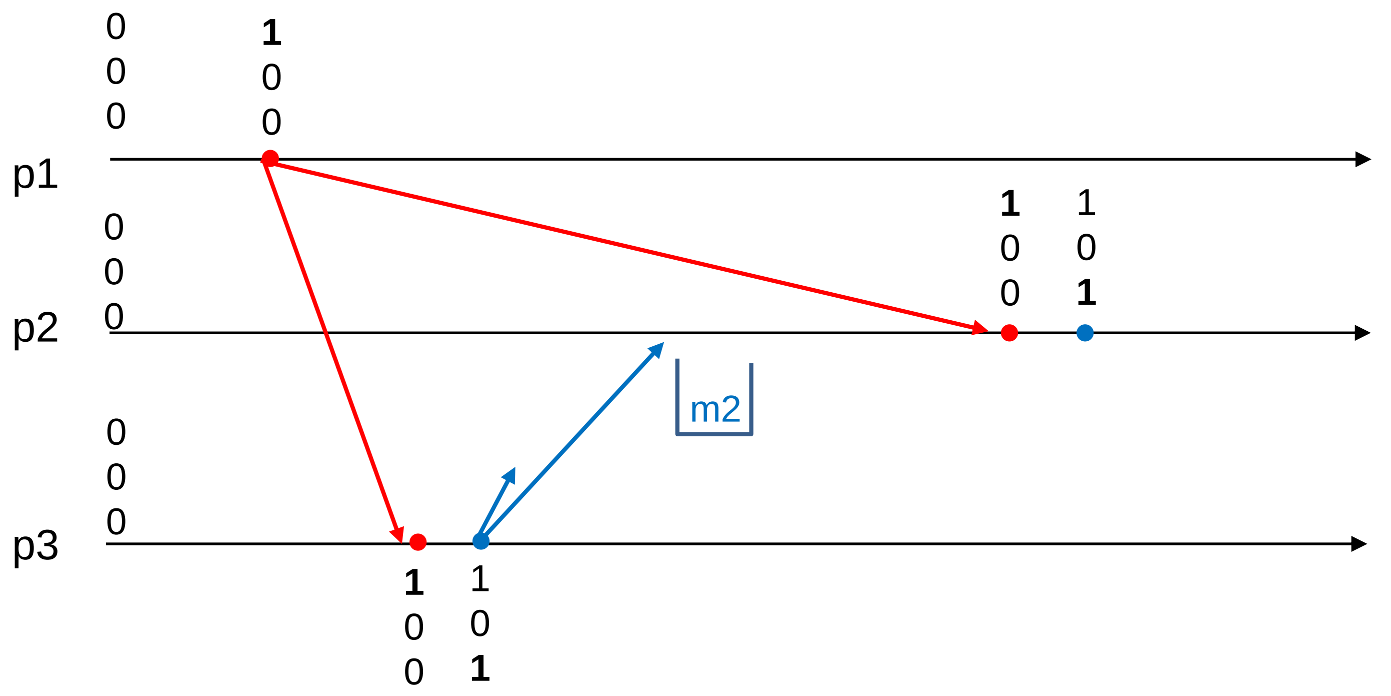
$\forall m1, m2$, broadcast($m1$) $\rightarrow$ broadcast($m2$) $\Rightarrow$ deliver($m2$) $\not\rightarrow$ <u>deliver</u>($m1$)



⇒ Control mechanism + reception of a message it's delivery

Vector clock approach

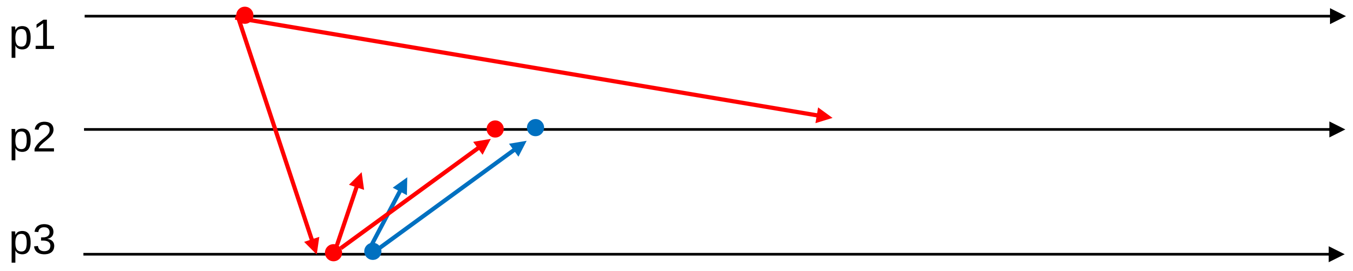
- A vector clock with one entry per node piggybacked on message



⇒ not scalable

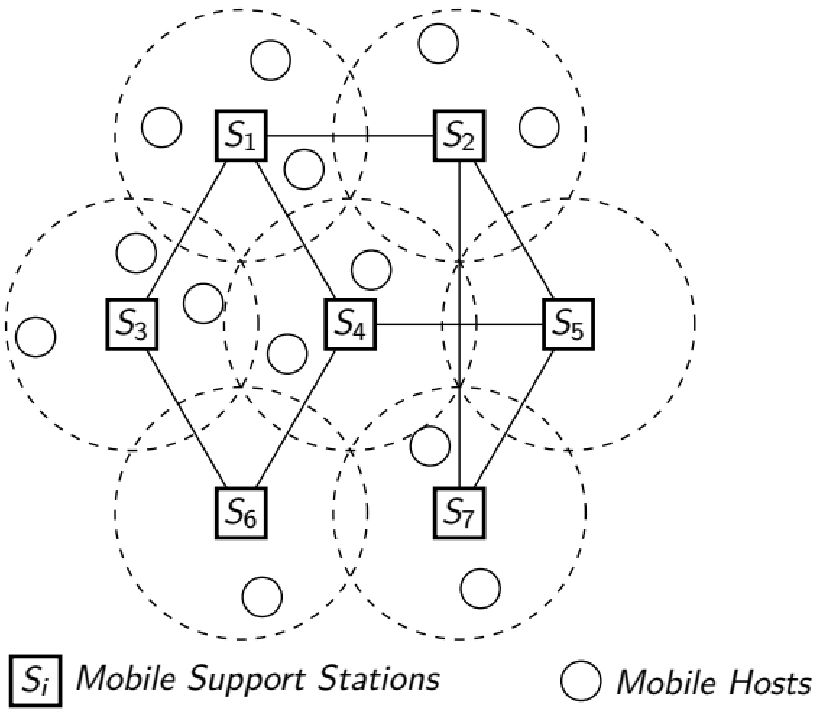
Fifo approach

- Reception(m): deviler(m), retransmission of m



- No control information to order messages
- Hard to add new communication channels

Mobile networks



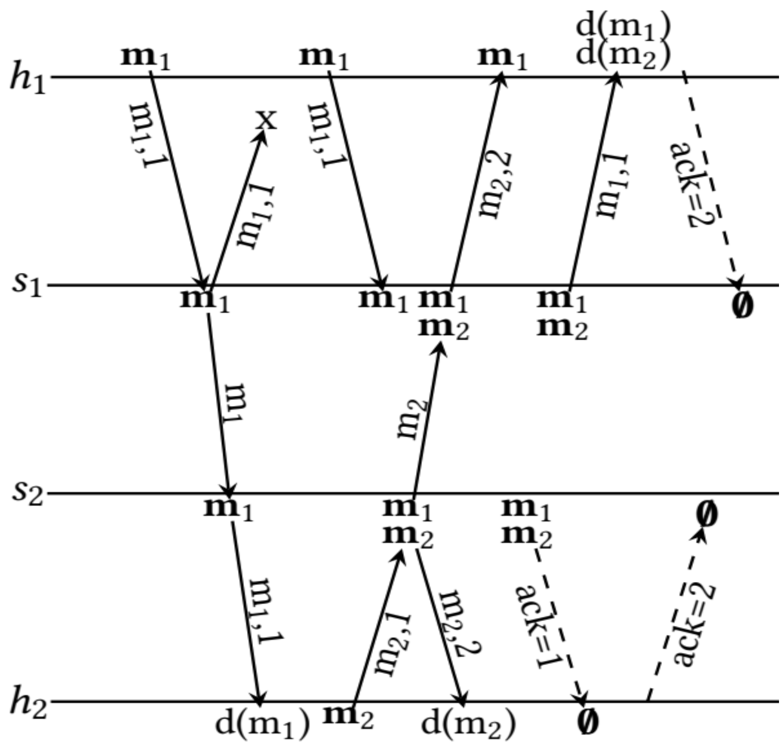
- Hosts capacity limitations: energy, computational, memory
- Stations hold most of the causal information
- Host dynamicity: free movement, leave/join network, failures
- Bandwidth and unreliability of the wireless network

Principles of the algorithm

Hosts are the source of application messages, stations ensure that all hosts deliver them causally

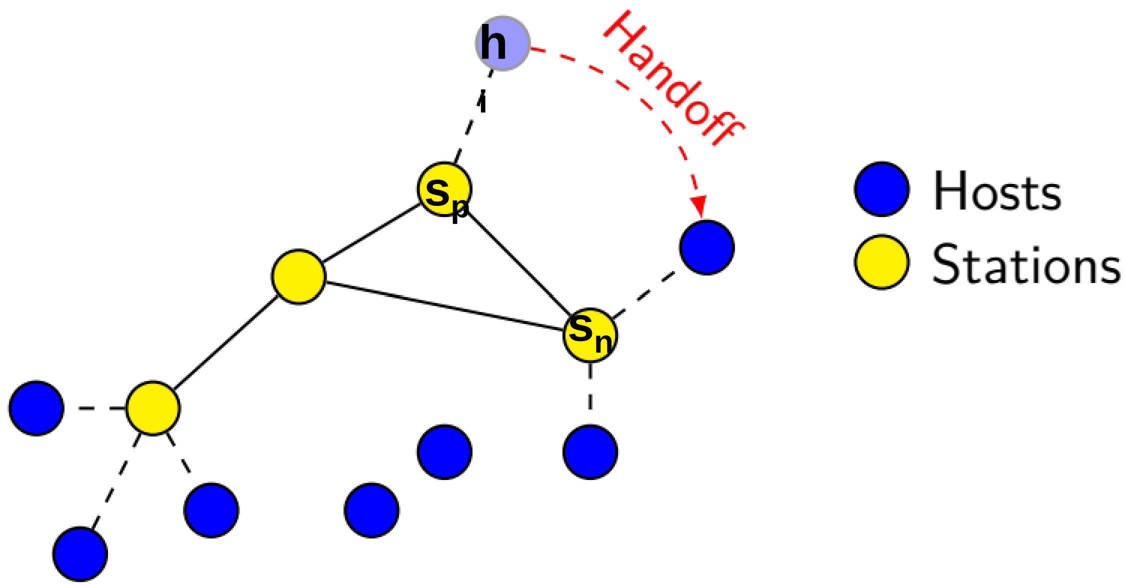
- Each *Host* maintains the **sequence number** of the next expected message.
- Each *Station* assigns **sequence numbers to order** messages inside its cells and **retransmits** messages on wireless and wire (FIFO) channels.
- Inside cells, **ack** included sequence number are periodically sent.
- A *station* discards a message once all its local hosts acknowledge it

Principles: information dissemination



m1 → m2

Mobility: Handoff

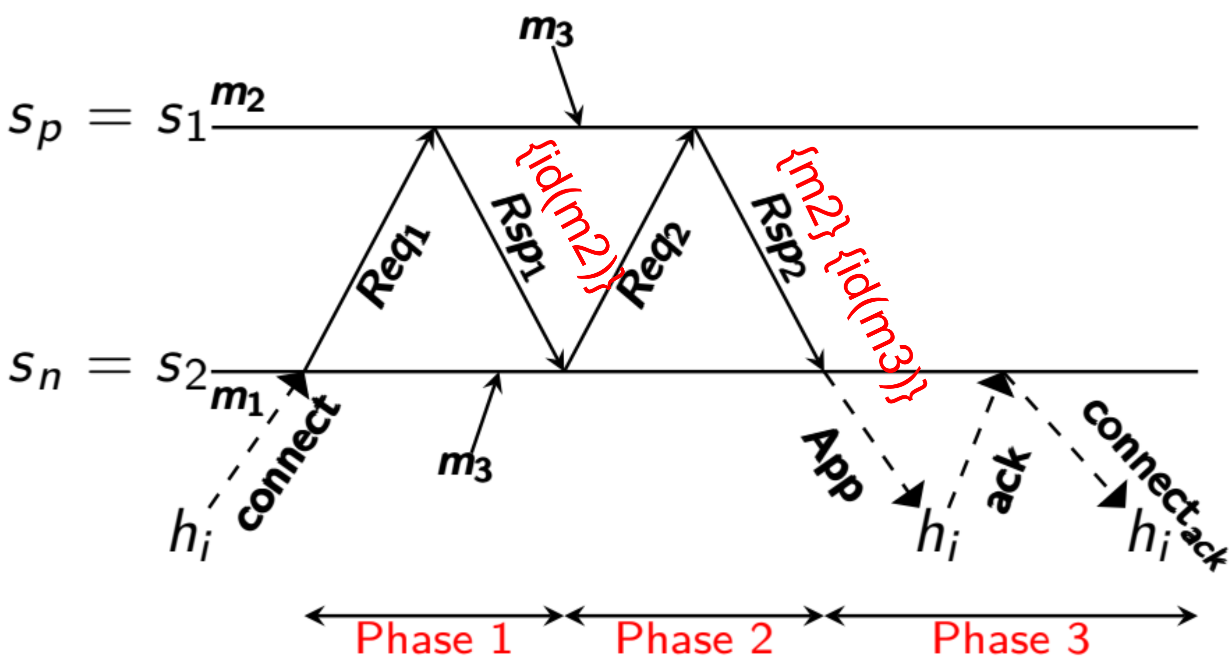


Handoff ($s_p \rightarrow s_n$)

- Phase 1: detection of messages not delivered by h_i
- Phase 2: detection of messages not delivered by h_i among messages that s_n caches.
- Phase 3: initialization of the connection between s_n and h_i.

Handoff exemple

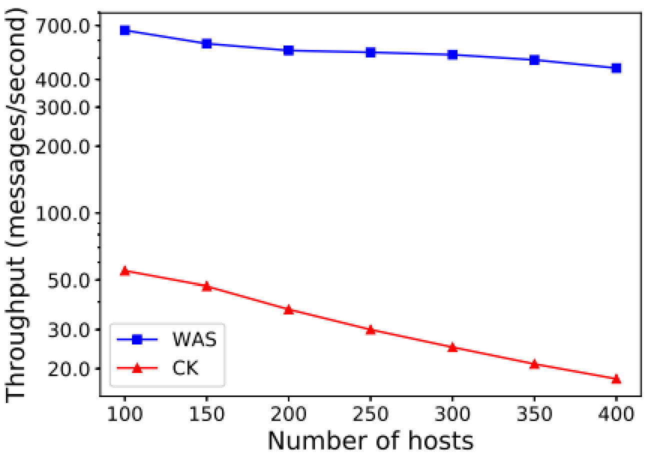
- Initially : h_i delivered m_1, s_p has discarded m_1, s_n discarded m_2
- Both stations receive m_3 during the handoff



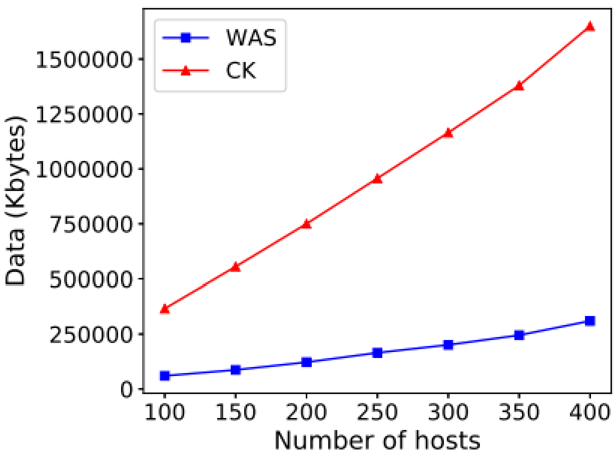
Performance evaluation

- Simulations implemented on **OMNeT++/INET**
  - Host mobility
  - Interferences, simulates network layers
  - Host failures
- Each host broadcasts application messages following a Poisson distribution.
- Hosts move in a straight line with a speed of 5km/h and change direction every 5 seconds
- Comparison with Chandra -Kshemkalyani (CK): a causal multicast algorithm for mobile network with a centralized discard mechanism (end-to-end ack).

Throughput and transmited data



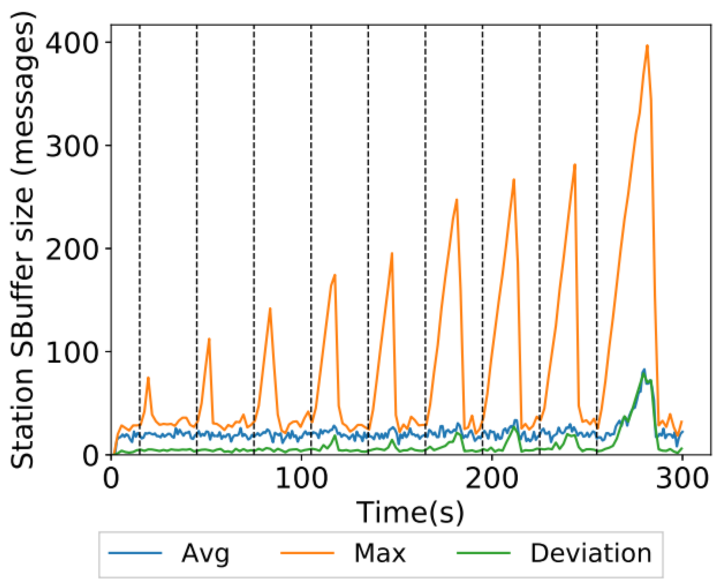
Throughput



Transmited data (wired and wireless network)

Failure injection

- Number of buffered messages at stations
- First host fails at t=10s and lasts 5s, then each 30 seconds another host fails, and the fault duration increases by 2 seconds at each failure



Concluding remarks

- Distributed systems are **dynamic**
- Need to revisit traditional distributed algorithms for and dynamic systems

Open issues

- Theoretical aspects
  - Models : A global model ?
  - Minimal condition in terms of time / connectivity / dynamicity to solve problems (agreement, leader, ordered broadcast, membership …) ?
- Practical aspect
  - Tools to emulated dynamic environments (MSN, Fog, MANET …) in a reproductible way
  - Traces

Prix de thèse GDR RSD – ASF 2023

- Présidents : Xavier Lagrange (IMT Atlantique), Pierre Sens (SU)
- Thèse soutenue entre 1er janvier 2022 et le 31 décembre 2022
- soumettre électroniquement avant le **28 février 2023** :
- un résumé de la thèse en 2 ou 3 pages
- un CV détaillé avec la liste des publications et des brevets,
- les rapports de pré-soutenance des rapporteurs (scannés),
- le rapport de soutenance (scanné),
- une lettre du (ou des) directeur(s) de thèse
- un lien cliquable vers la thèse (pas le document lui-même),
- un lien cliquable vers les transparents de la soutenance et/ou la vidéo de la soutenance,
- des rapports complémentaires que le candidat jugera utile de fournir au jury,
- (si applicable) un lien vers les réalisations techniques comme les logiciels, les études de cas, les brevets

Thank you !