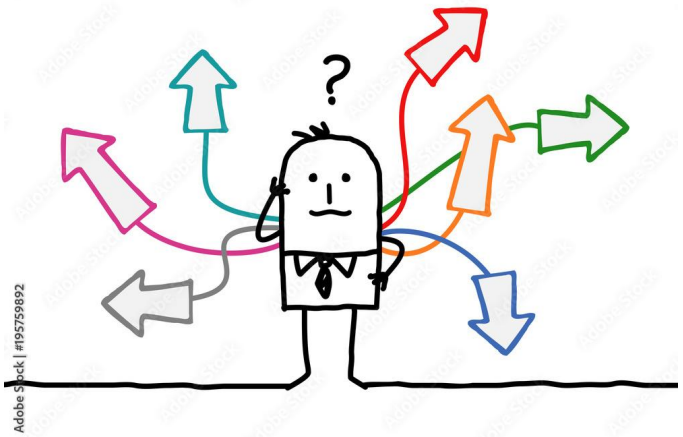
Operational Model vs. Waitfree Model

Achour Mostefaoui

ANRs Estate / Ducat Le Cap Hornu 16-18 mars 2022

Distributed Computations

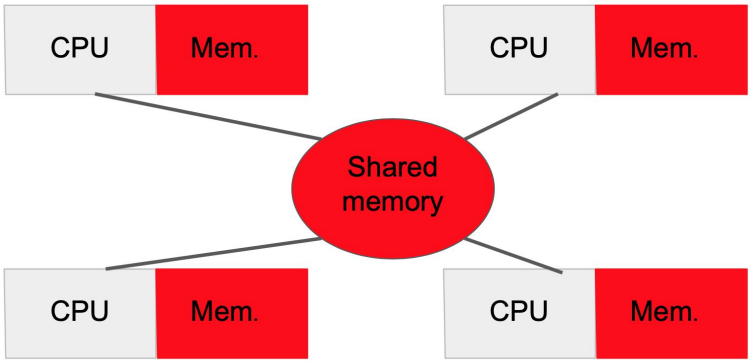
- Processes and concurrency
  - Processes may interact directly through shared data structures …
    - implemented over a shared memory system
    - or, through messages exchanged by the different processes



Shared Memory and Shared Data Structures

The ABD simulation (Attiya, Bar-Noy and Dolev 1995)

- It has been proved in 1995 that a shared memory (shared registers) can be emulated over a distributed system provided that there is a majority of processes that do not crash



Consistency and Progress Conditions

- A data structure is defined by two properties:
  - A safety property
  - A progress condition
- Safety: questions the meaningfulness of the results returned by the operations
- Progress: will there be a returned value, for whom and when?

Strong Consistency (linearizability and sequential consistency)

- Linearizability and sequential consistency cannot be distinguished in an asynchronous system
- Sequential consistency is “cheaper” than linearizability
- However, linearizability is a local property: if all objects are linearizable, then the whole computation is linearizable!
- A distribution computation is a partial order of events.
- A good consistency criterion consists in totally ordering all events
  - linearizability: total order on all events + causality + real-time order
  - sequential consistency: total order on all events + causality

Weak Consistency

- CAP Theorem (Consistency, Availability and Partition): Impossibility to ensure the three properties at once in purely asynchronous systems prone to process crashes (Gilbert & Lynch 2002)
- Moreover, even in synchronous failure free systems, the operations cannot be local (Attiya & Welch 1994)
- In those situations, one can use weak consistency conditions:
  - Cache coherence
  - Causal consistency
  - Eventual consistency
  - PRAM consistency
  - Serializability ...

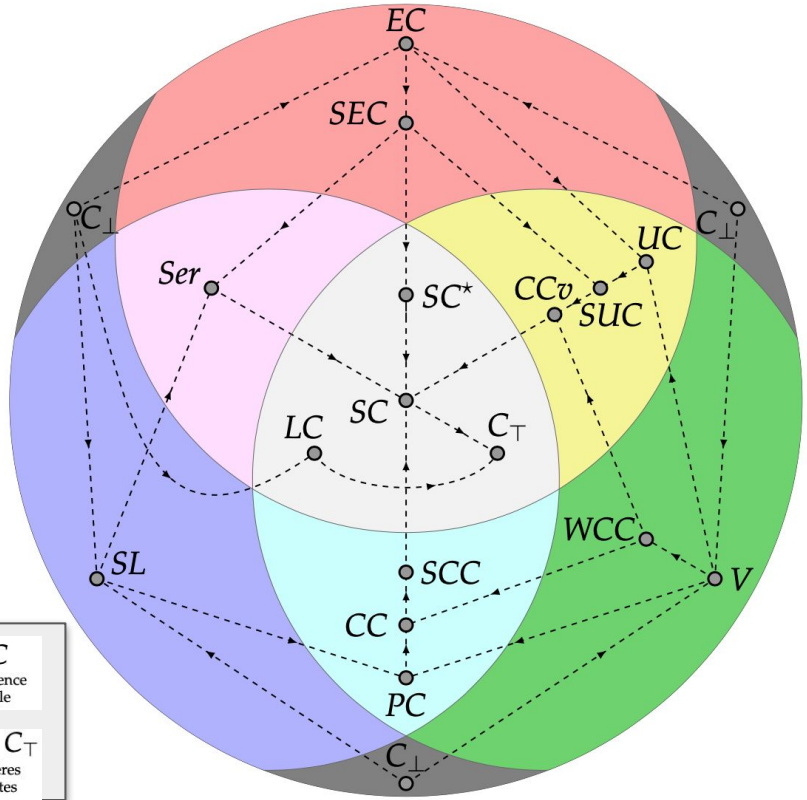
Weak Consistency

The world of consistency conditions (from M. Perrin PhD thesis)

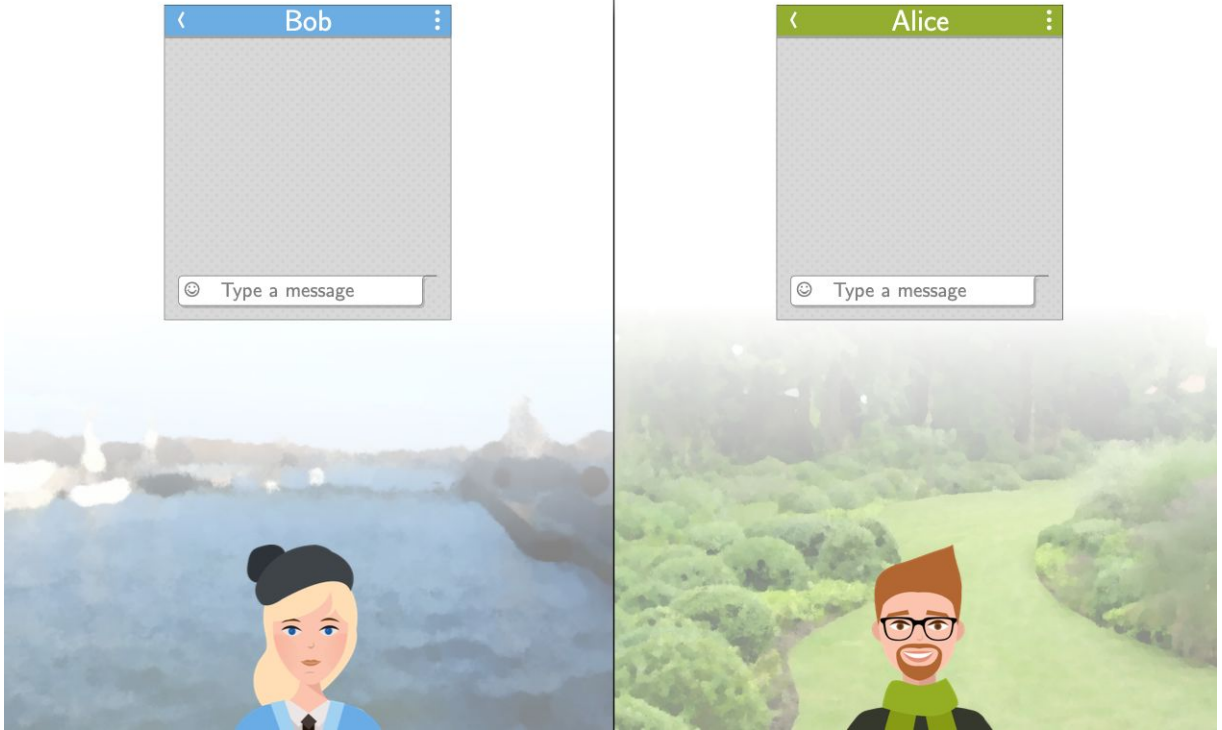
There are 3 basic families of consistency conditions

A consistency condition that merges all of the three families falls into strong consistency

SC Cohérence séquentielle	*PC* Cohérence pipeline	*SEC* Convergence forte	*SUC* Cohérence d'écritures forte	*CCv* Convergence causale	*WCC* Cohérence causale faible	*SCC* Cohérence causale forte	*LC* Cohérence locale
*SC** Cohérence de cache	*SL* Localité d'état	*EC* Convergence	*UC* Cohérence d'écritures	*Ser* Sérialisabilité	*V* Validité	*CC* Cohérence causale	$C_\perp$ $C_\top$ Critères limites

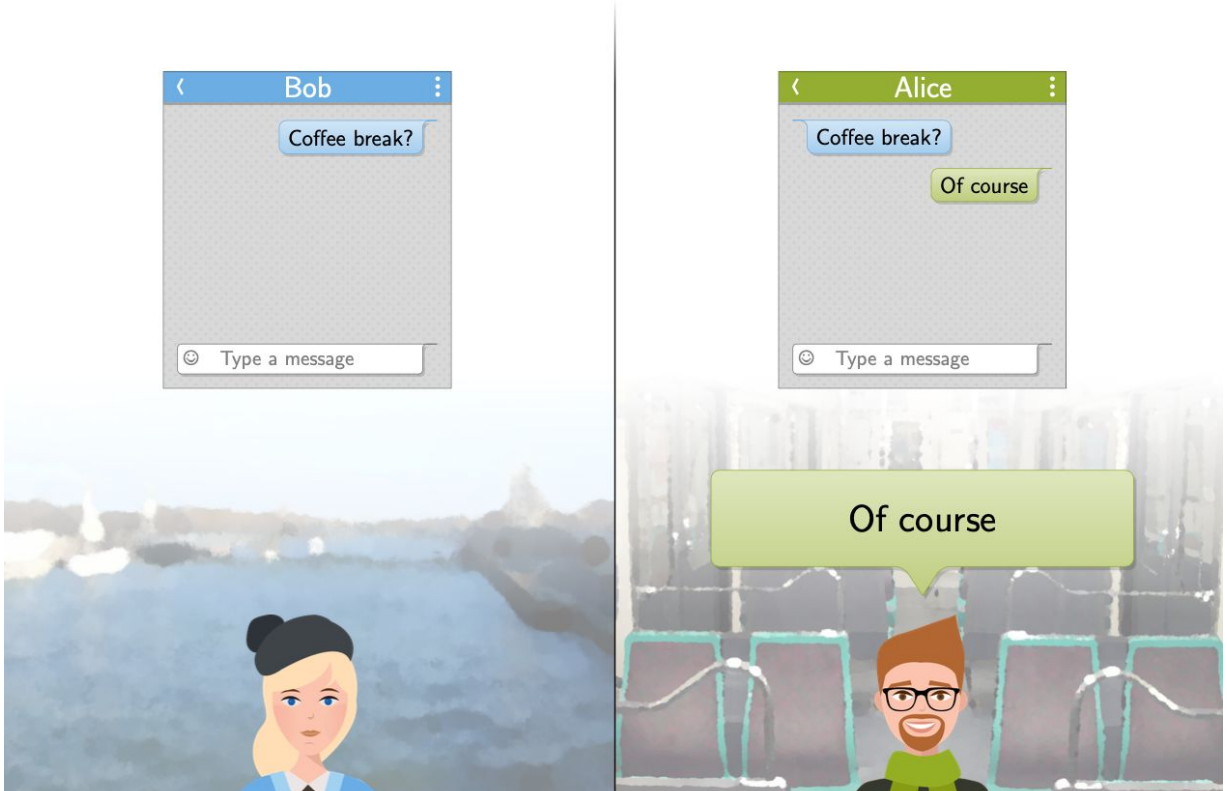


Instant Messaging

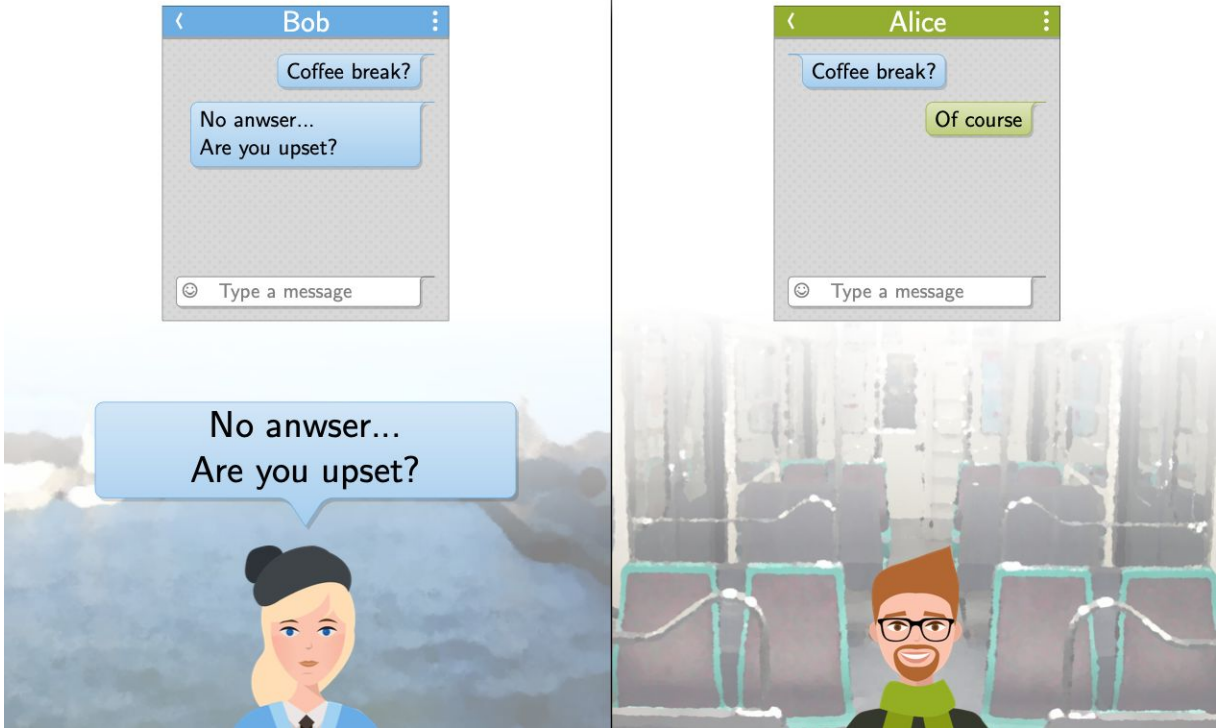


Instant Messaging

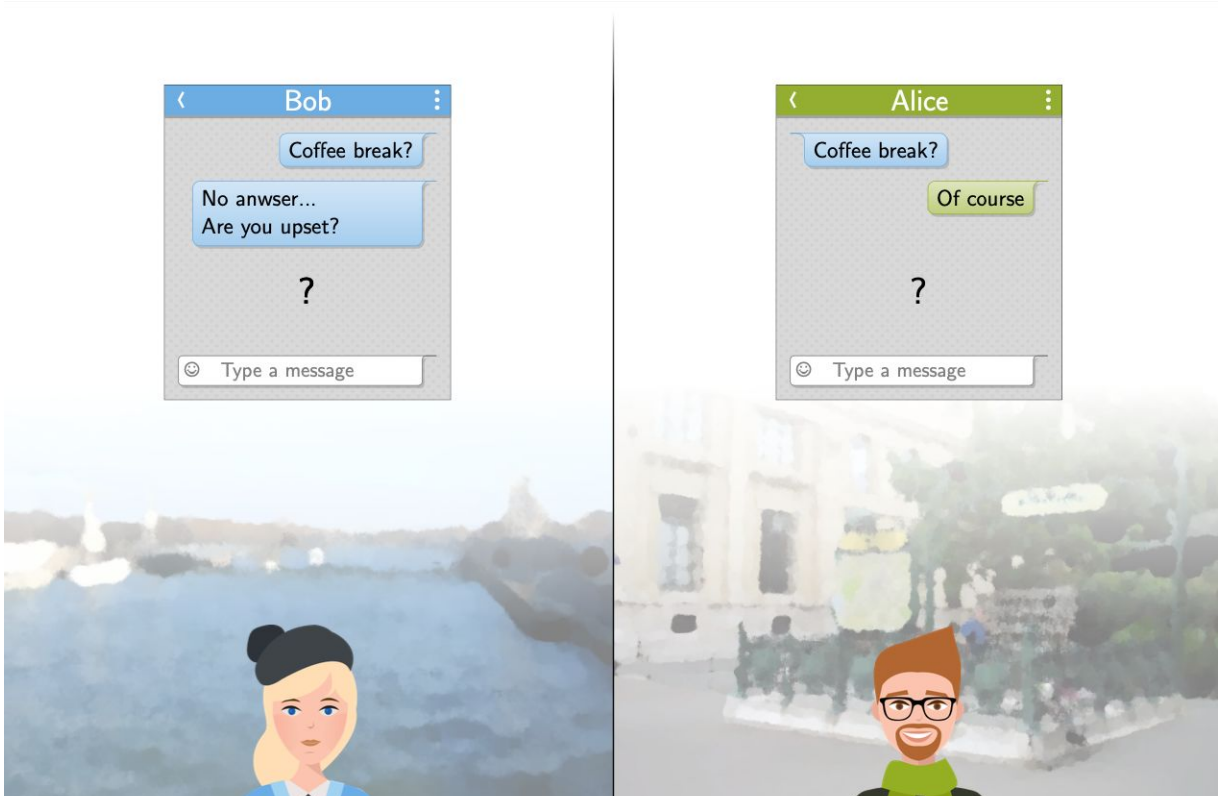
Instant Messaging



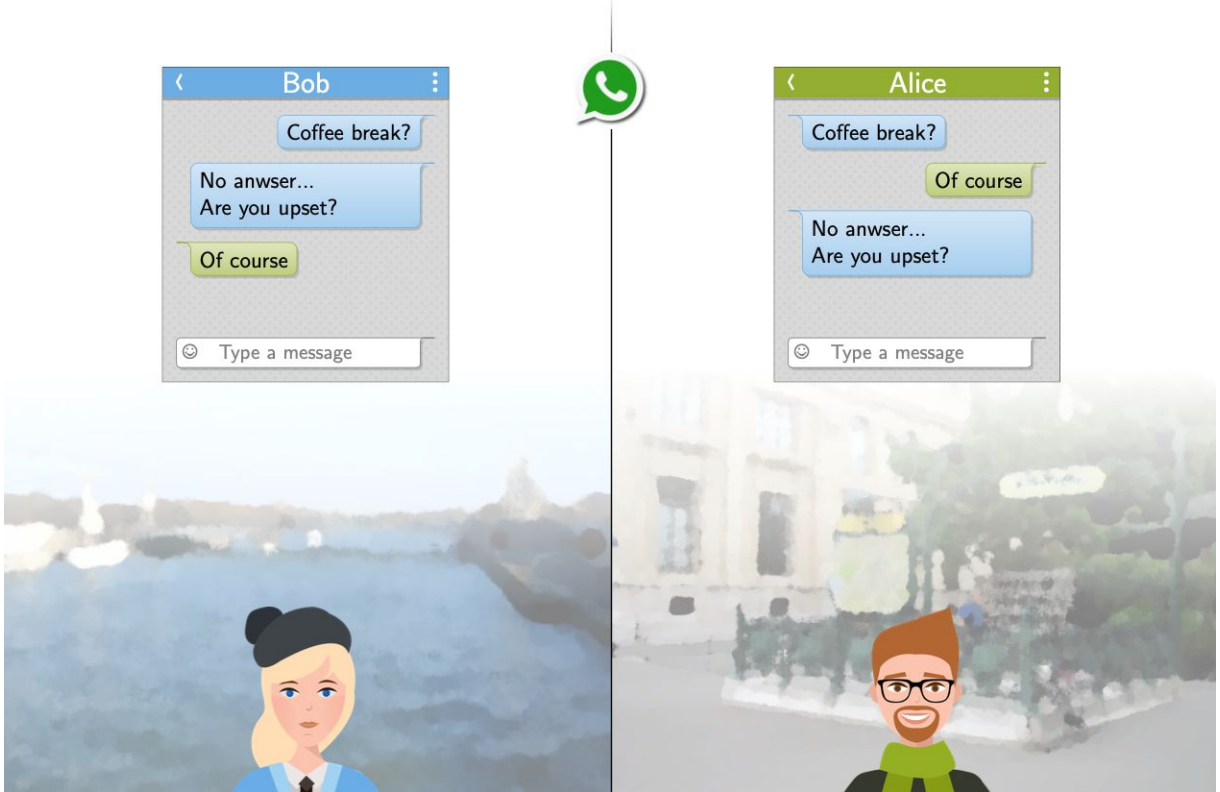
Instant Messaging



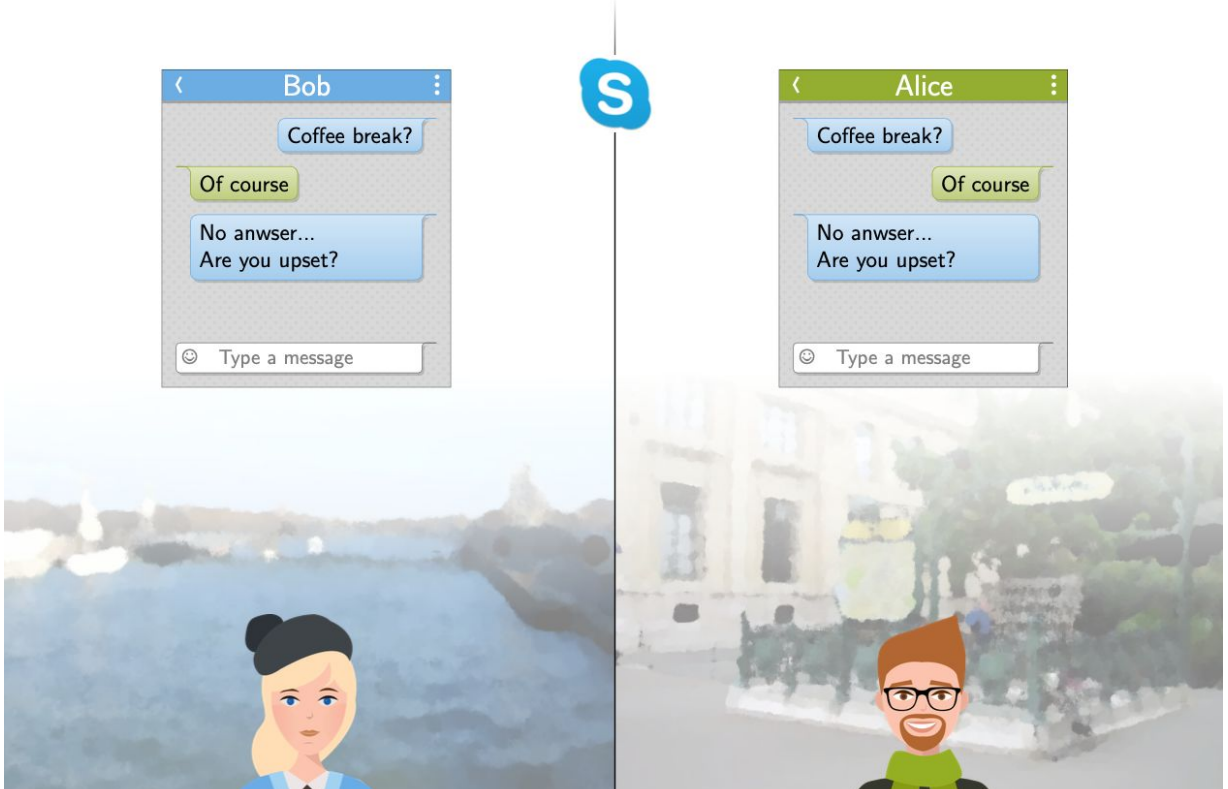
Instant Messaging



Instant Messaging



Instant Messaging



Weak Consistency

Small experience with instant messaging:
Snapshat, Messenger, Whatsapp, Skype, Hangouts, etc.

- Hangouts: serializability
  - message sending can be aborted
- Whatsapp: PRAM consistency
  - local consistency (perhaps the least consistent instant messaging)
- Skype: strong eventual consistency
  - messages can be reordered afterwards (all users eventually see all messages in the same order)

The Classical Waifree Model

Processes

- Asynchronous : no bounds on the execution time
- May crash : no waiting possible

Communication

- Message passing
- Asynchronous : no bounds on message transfer delays

The Operational Model

Hypothesis: Restrictive on the type of algorithms

- Objects are fully replicated
- Read operations are local
- Messages can only be sent during update operations

Remark:

- Optimal in the number of messages

Space Complexity of Some Data Structures

Current results on eventually consistent shared objects:

- Sets ($\mathcal{O}(n \log(m))$), Counters ($\mathcal{O}(n)$), Registers ($\mathcal{O}(\log(m))$), Multi-value Registers ($\mathcal{O}(n \log(m))$) [1].
- Data Stores (Sets ($\mathcal{O}(n \log(m))$), Multi-value Registers ($\mathcal{O}(n \log(m))$)) [2].
- Collaborative Editors ($\mathcal{O}(m)$) [3].

[1] Burckhardt S, Gotsman Al, Yang H, Zawirski M : Replicated data types: specification, verification, optimality
[2] Attiya H, Ellen F, Morrison A : Limitations of Highly-Available Eventually-Consistent Data Stores
[3] Attiya H, Burckhardt S, Gotsman A, Morrison A, Yang H, Zawirski M : Specification and complexity of collaborative text editing

Are the Two Models Equivalent?

Theorem

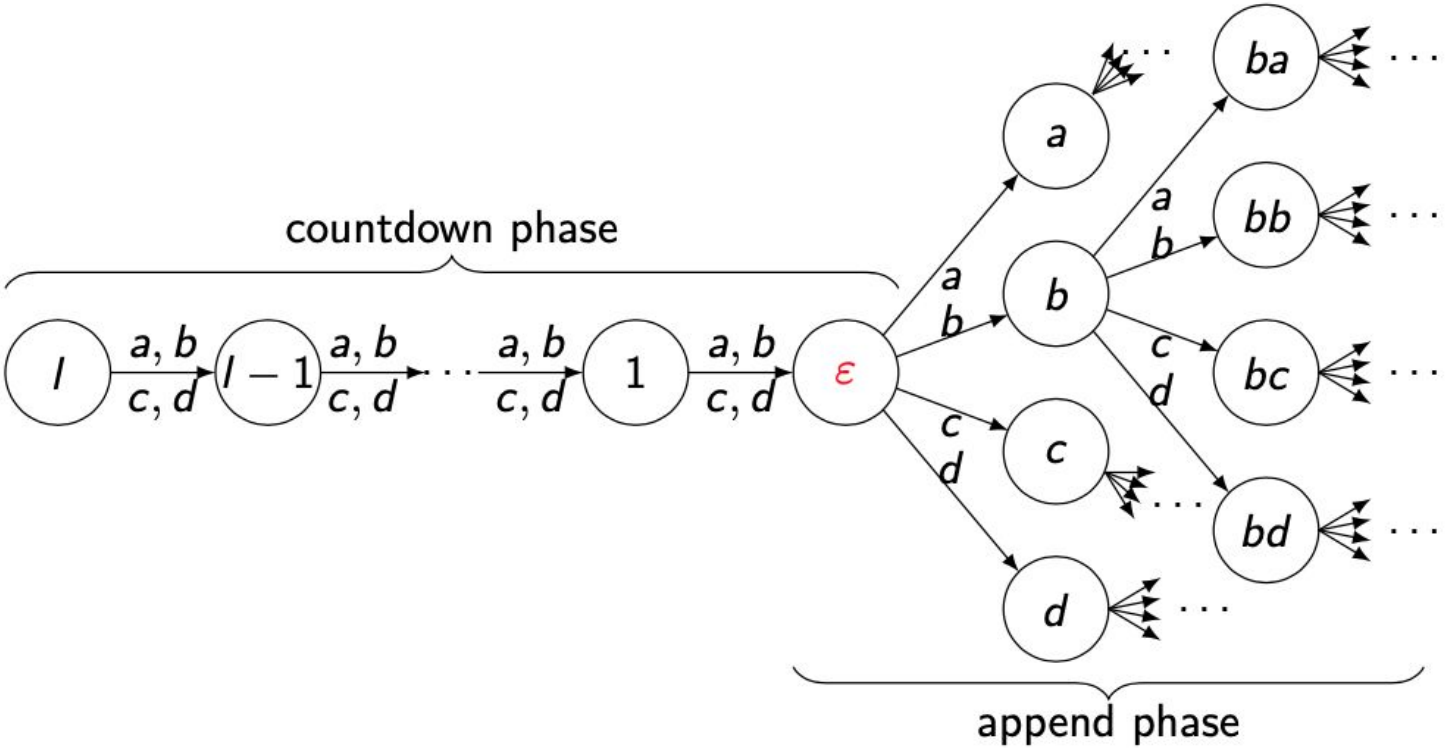
We prove that there is:

- One object O
- One execution E
- One implementation of O in the Wait-Free model I_{WF}

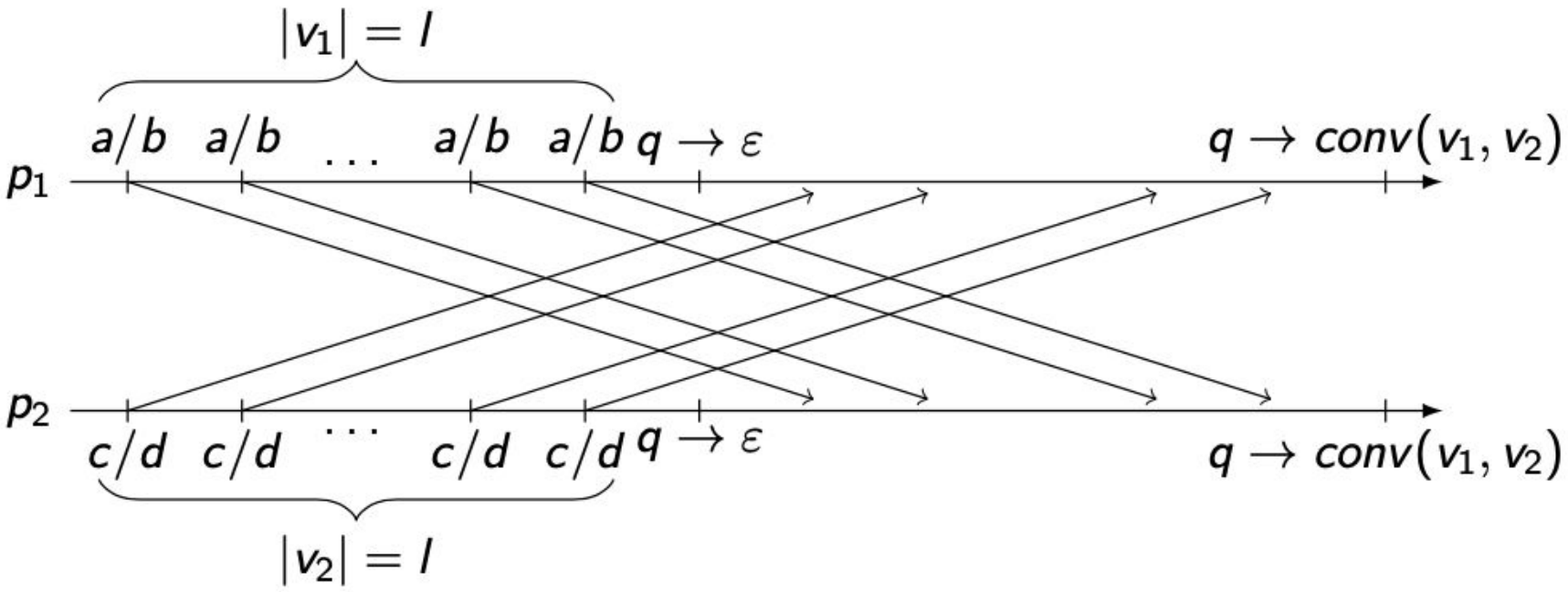
Such that:

Any implementation of O in the operational model I_{OM} takes strictly more bits of local memory than I_{WF} in E.

The *l*-Countdown-Append Object

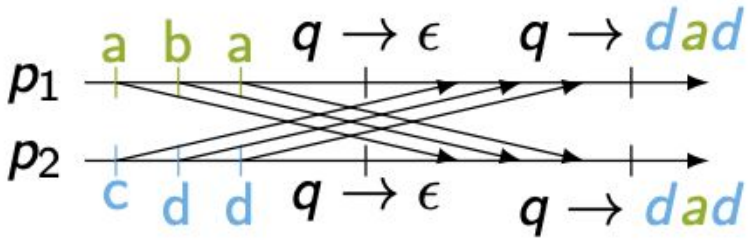


The l-CA Object in the Operational Model

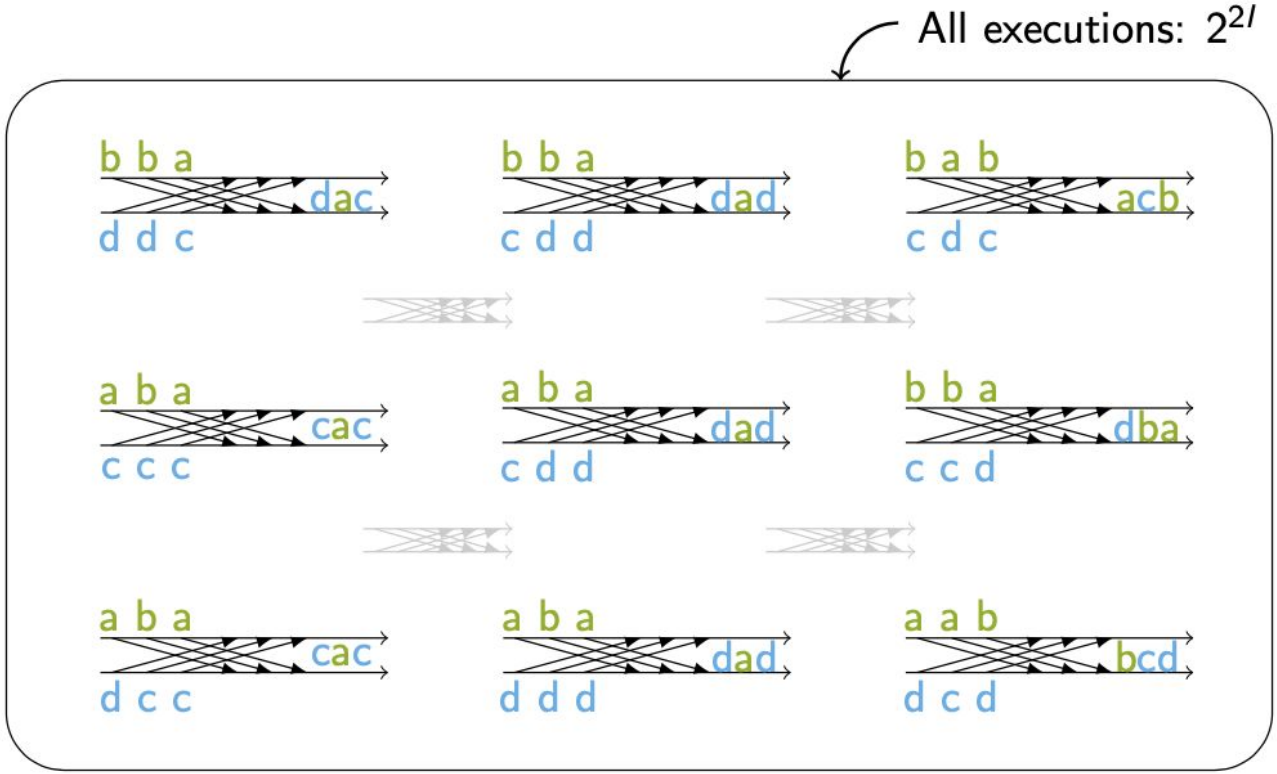


The l-CA Object in the Operational Model

An example with $l = 3$:



The *l*-CA Object in the Operational Model



The l-CA Object in the Operational Model

Lemma

There is an execution X in which any implementation of the l-CA in the operational model requires at least $\left(\frac{l}{2} - 1\right)$ bits of local memory in the ε state.

The l-CA Object in the Classical Model

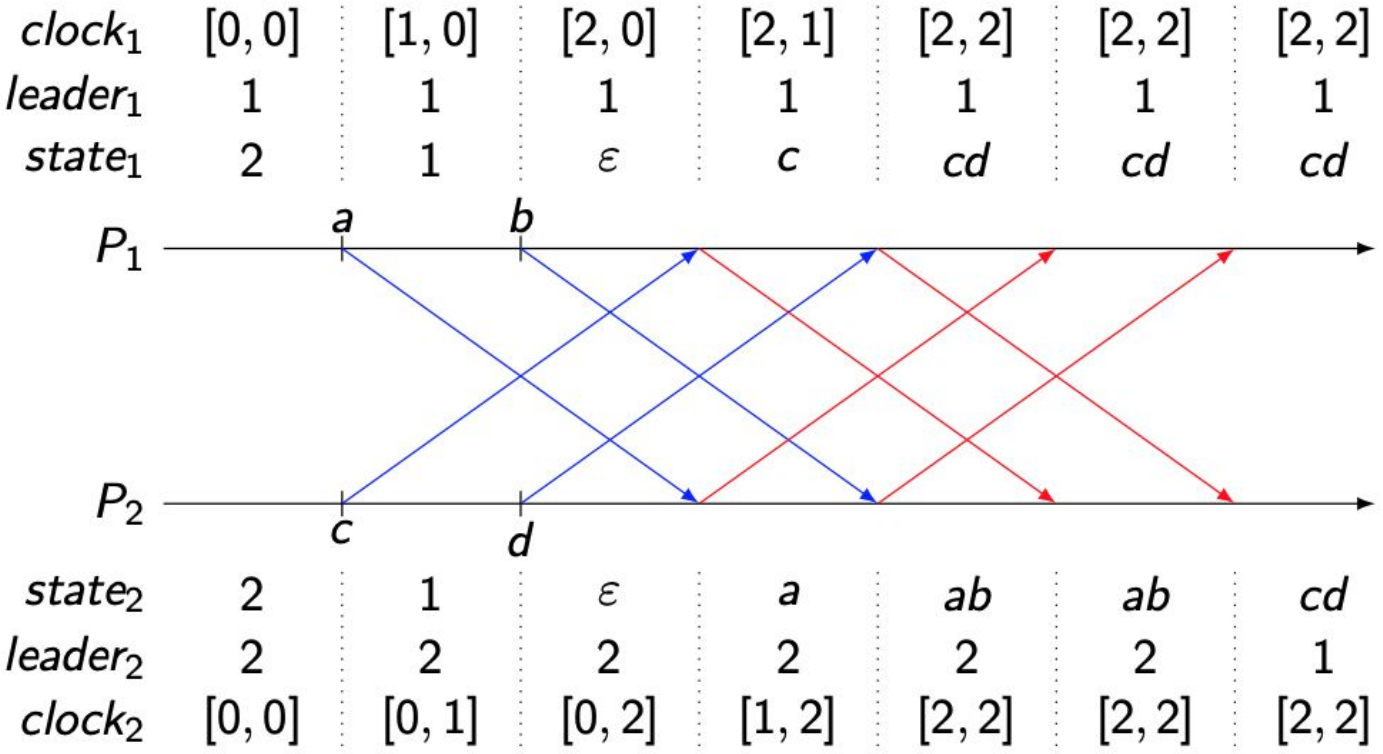
- Each process maintains a vector of version numbers

- Each time the state of a process changes, it broadcasts its new state (state transfer)
- A process keeps the state with the highest associated version vector

Lemma

The algorithm has a local memory complexity of $\mathcal{O}(n \log(l))$.

The *l*-CA Object in the Classical Model



The Best of Both Worlds

We proposed a generic algorithm that combines the two approaches:

- The normal behavior is the one of the operational model.
- A global logical time defines logical phases of size k (a constant)
- Version vectors are reinitialized at the each phase
- State transfer happens only if asynchronism is high