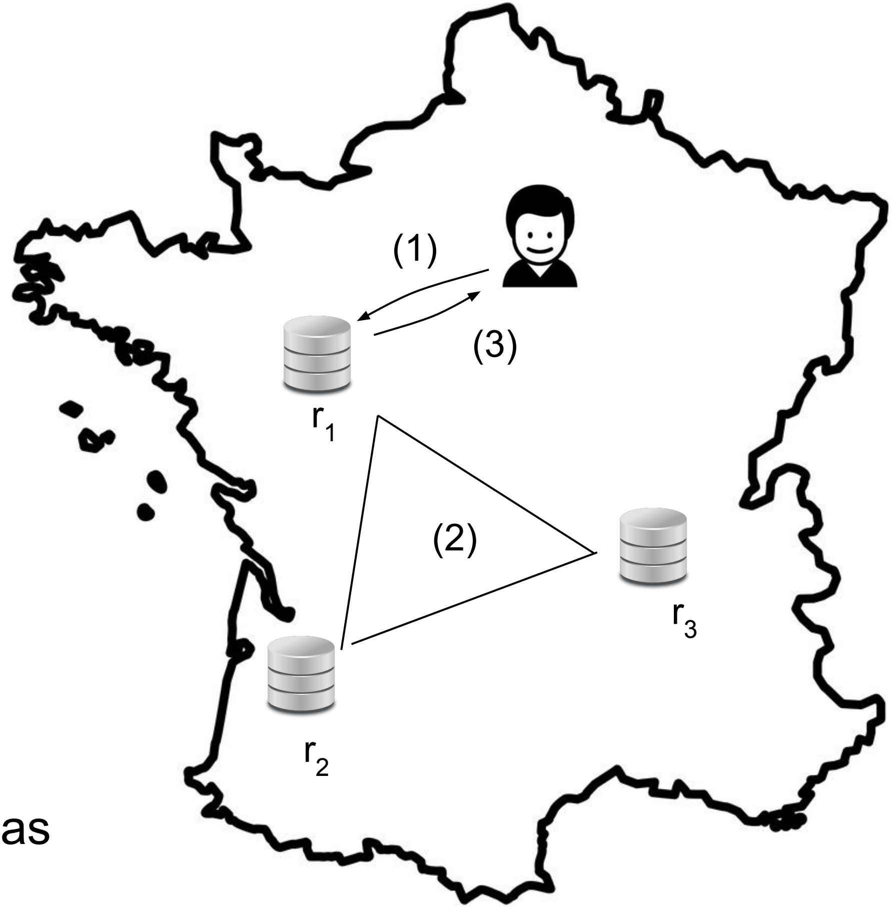
Leaderless State-Machine Replication: An Overview

Pierre Sutra

Télécom SudParis
Institut Polytechnique de Paris

Per3S Workshop, 13.06.2022

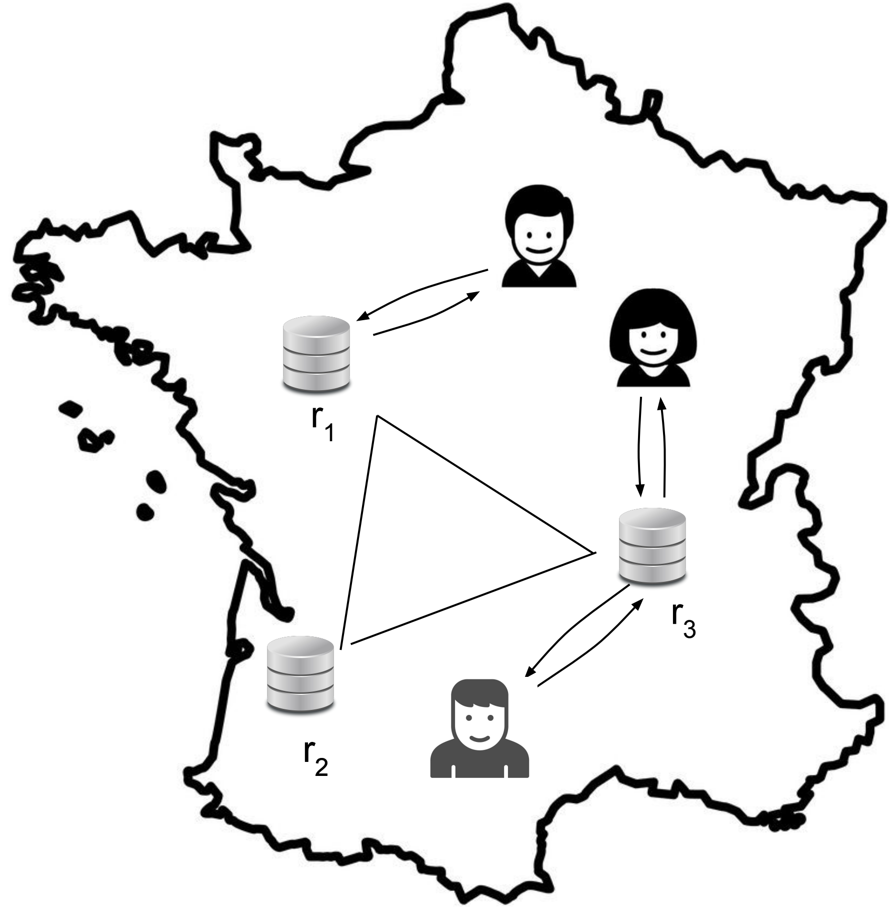
Context: geo-replication



(1) command
(2) some protocol
(3) response

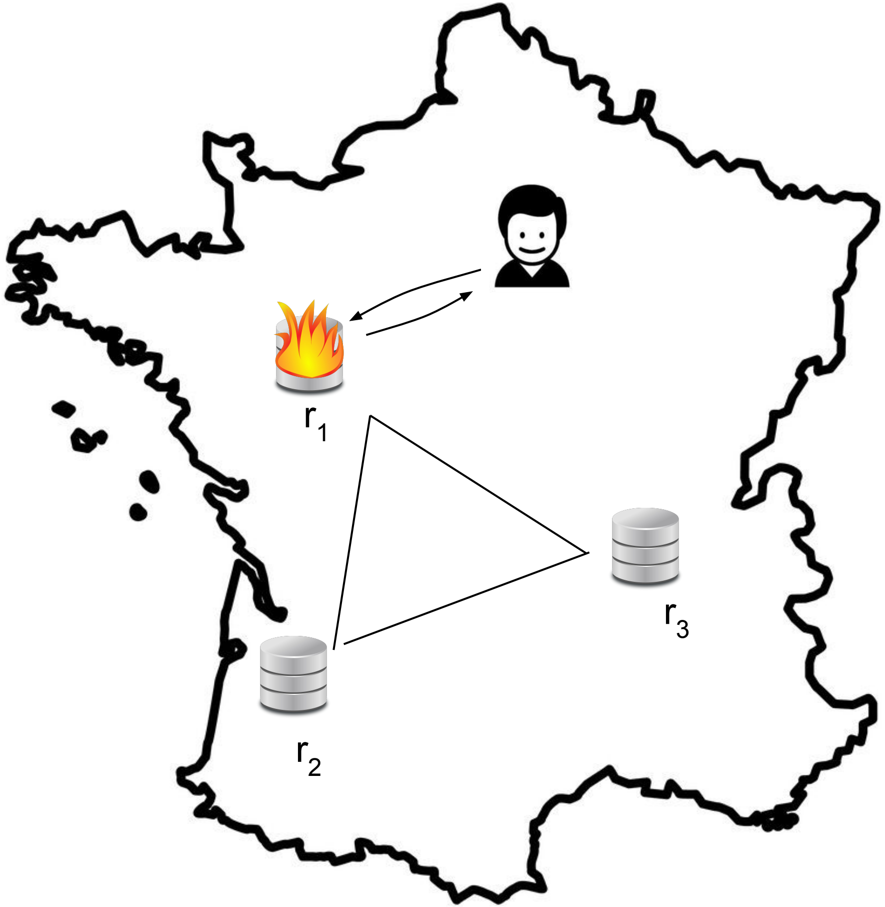
r_1, r_2, .. = data replicas

Context: geo-replication



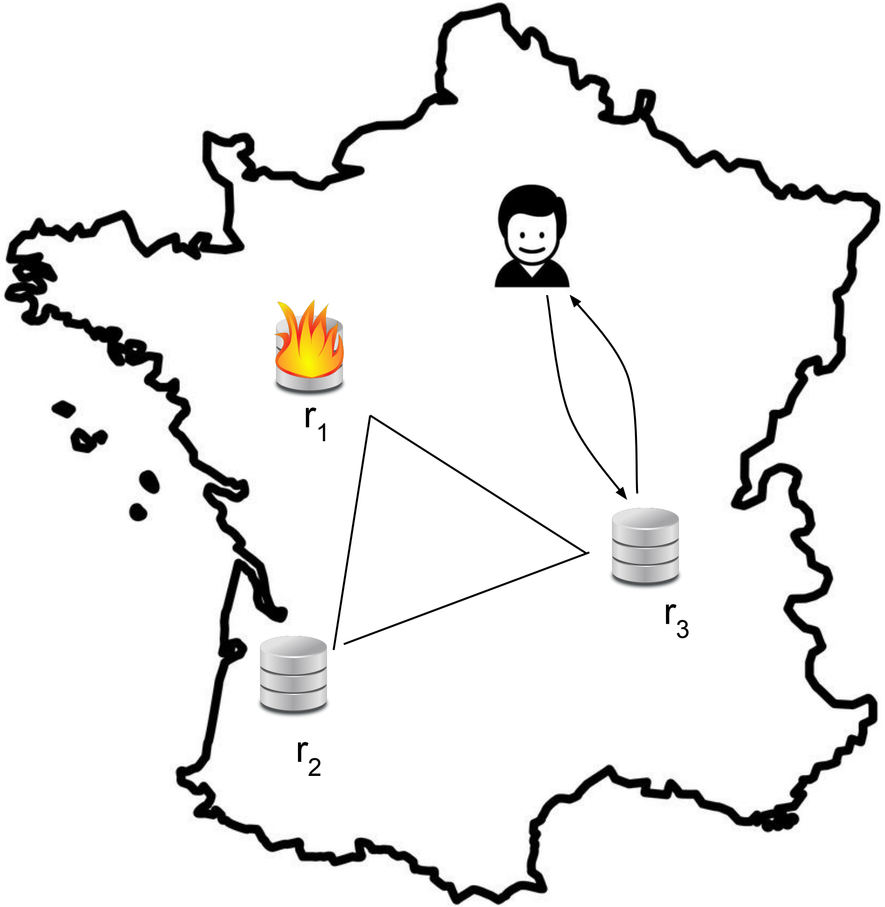
Problematic:
transparent efficient
geo-replication

Context: geo-replication



Problematic:
transparent efficient
geo-replication

Context: geo-replication

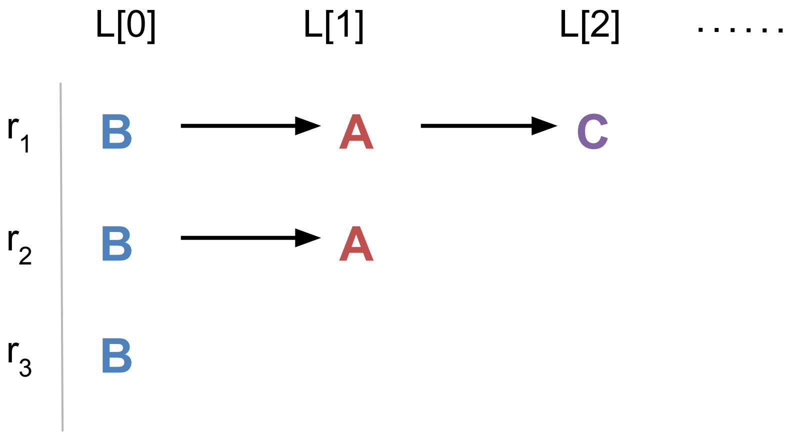


Problematic: *transparent efficient* and *robust* geo-replication

Classic State-Machine Replication [*Paxos, Raft*]

Each replica holds a log L

For each i, *agree* on command L[i]



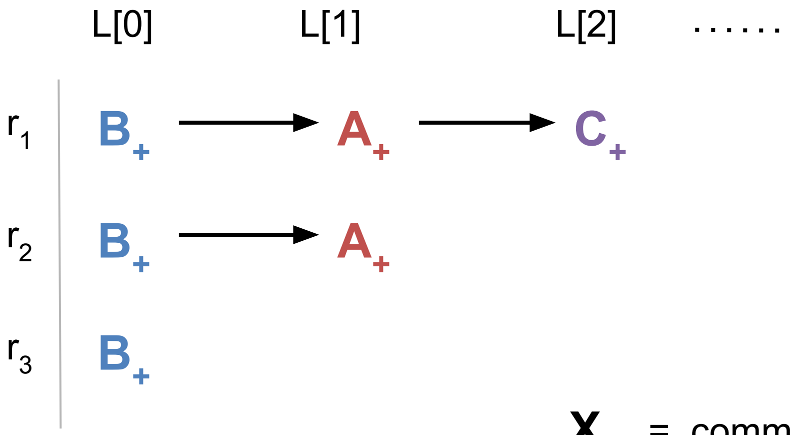
	L[0]	L[1]	L[2]	……
r_1	B →	A →	C	
r_2	B →	A		
r_3	B			

Classic SMR

Each replica holds a log L

For each i, *agree* on command L[i]

Execute commands in log order



X_+ = command is executed

APACHE ZooKeeper™

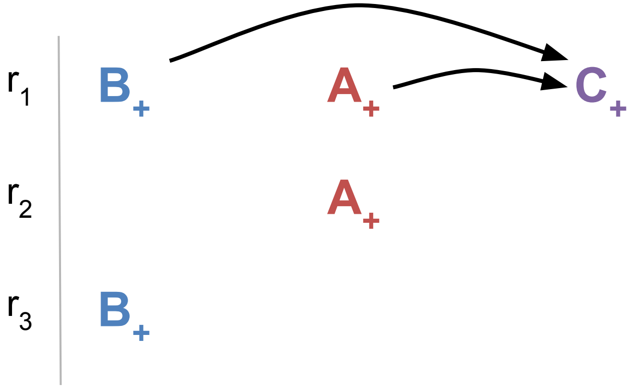
Cloud Spanner

etcd

Apache CASSANDRA

Generic SMR [*GPaxos*, *GBcast*]

Execute *non-commuting* commands in the same order in the log



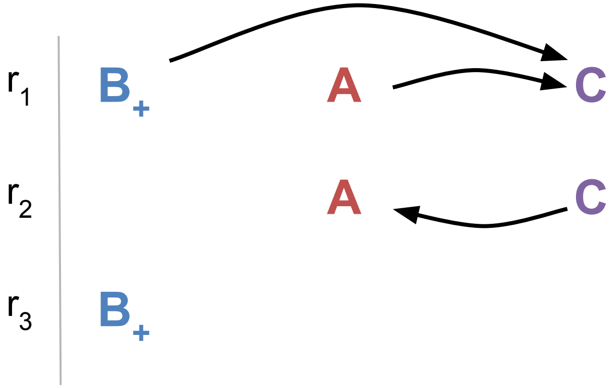
A = x ← 42

B = y ← 7

C = z ← x + y

Leaderless SMR [*DISC'05, SOSP'13*]

Execute *non-commuting* commands according to the same graph



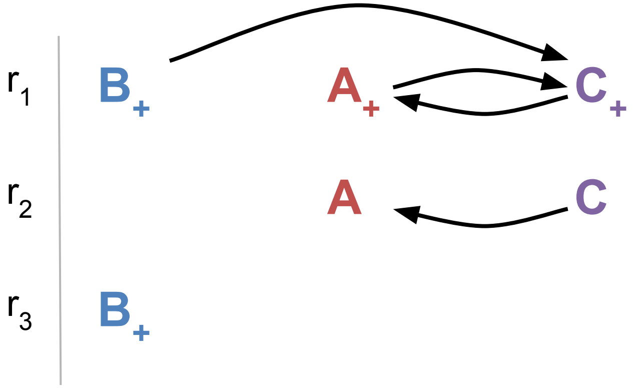
dep(A) = {C}

dep(C) = {B, A}

dep(B) = ∅

Leaderless SMR

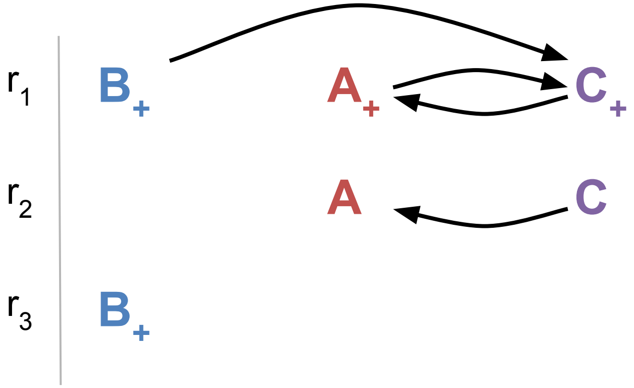
Execute *non-commuting* commands according to the same graph



- operation **X** executed once dep(**X**) transitively closed
- cycles are broken deterministically

Leaderless SMR

Execute *non-commuting* commands according to the same graph

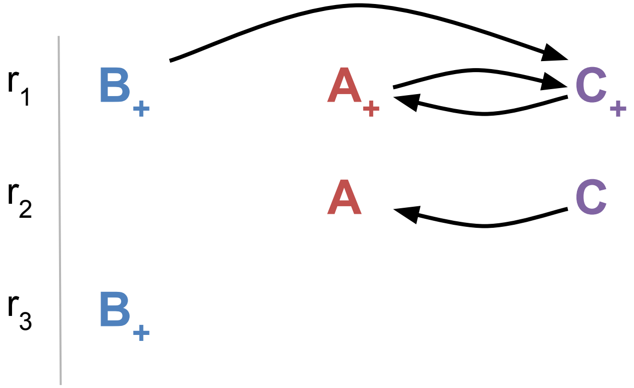


Properties

- replicas agree *on* dep(**X**)
- (**X**, **Y**) non-commuting then **X** ∈ dep(**Y**) or **Y** ∈ dep(**X**)

Leaderless SMR

Execute *non-commuting* commands according to the same graph



Properties

- replicas *agree on* dep(**X**)
- (**X**, **Y**) non-commuting then **X** ∈ dep(**Y**) or **Y** ∈ dep(**X**)

Egalitarian Paxos [*SOSP'13*]

EPaxos uses *2f+1* replicas.

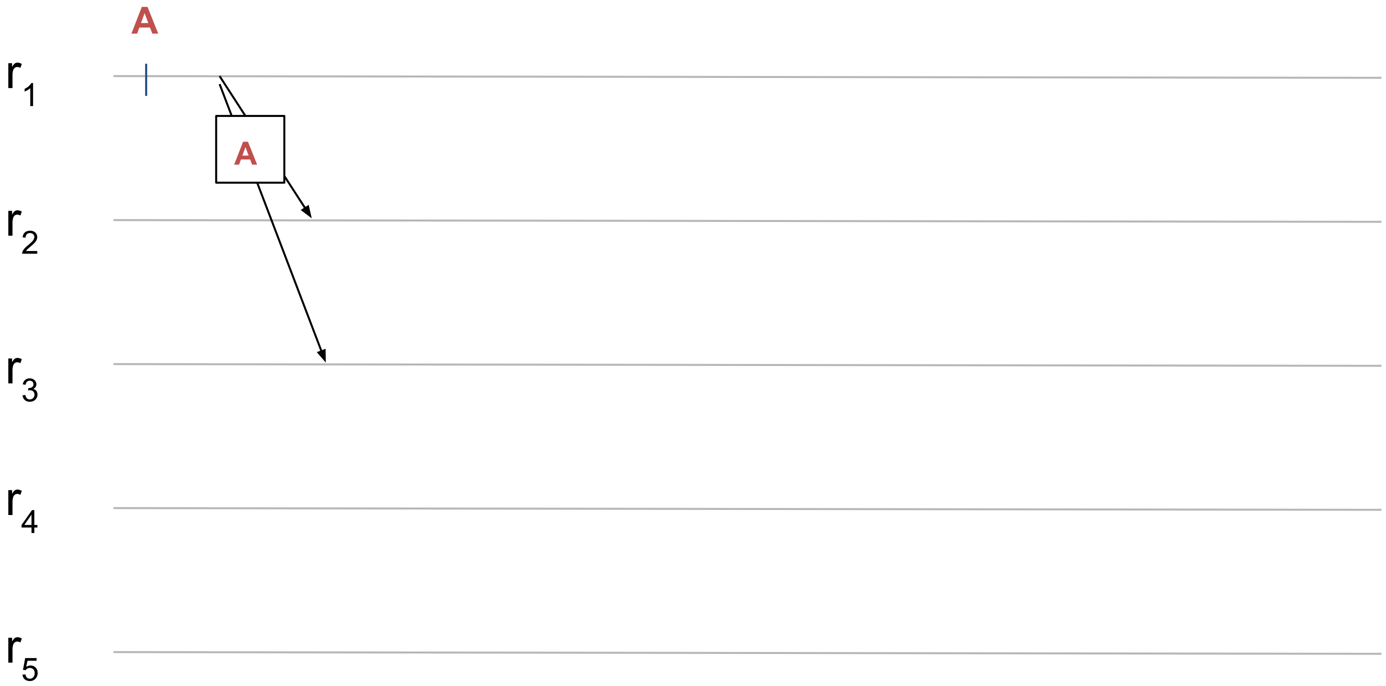
When a client executes command **X**

- pick a replica
- this replica is the *coordinator* for **X**, coord(**X**)
- coord(**X**) runs consensus over dep(**X**)

To do consensus on dep(**X**)

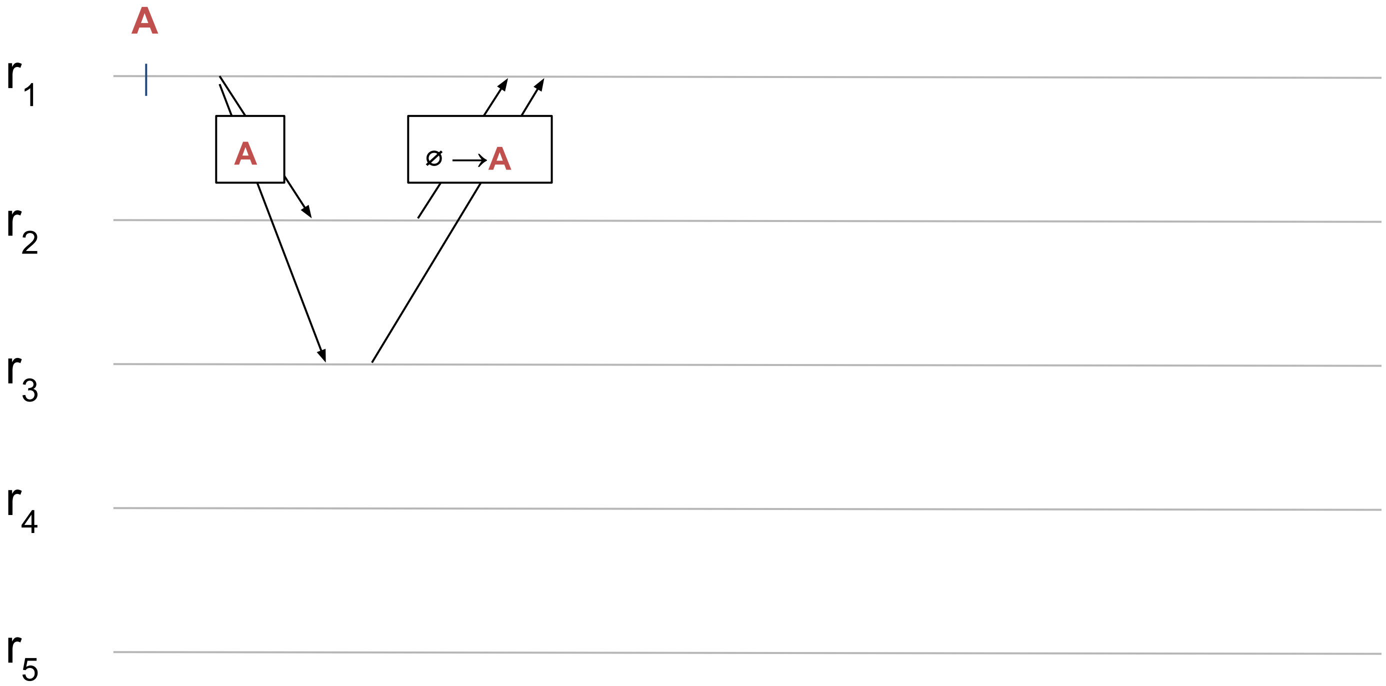
- try to agree spontaneously by contacting a fast quorum *(f+f/2 replicas)*
- if this fails, ask a slow quorum *(f+1 replicas)*

EPaxos



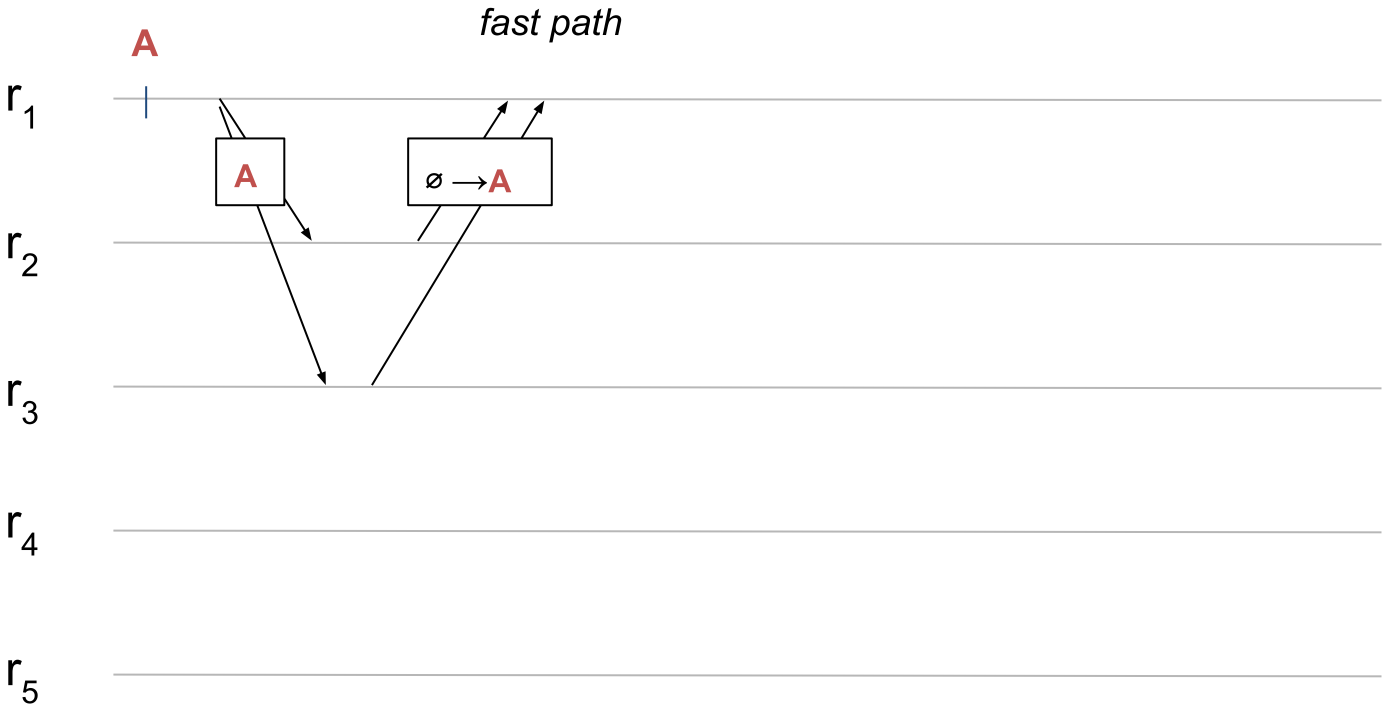
(n=5, f=2)

EPaxos



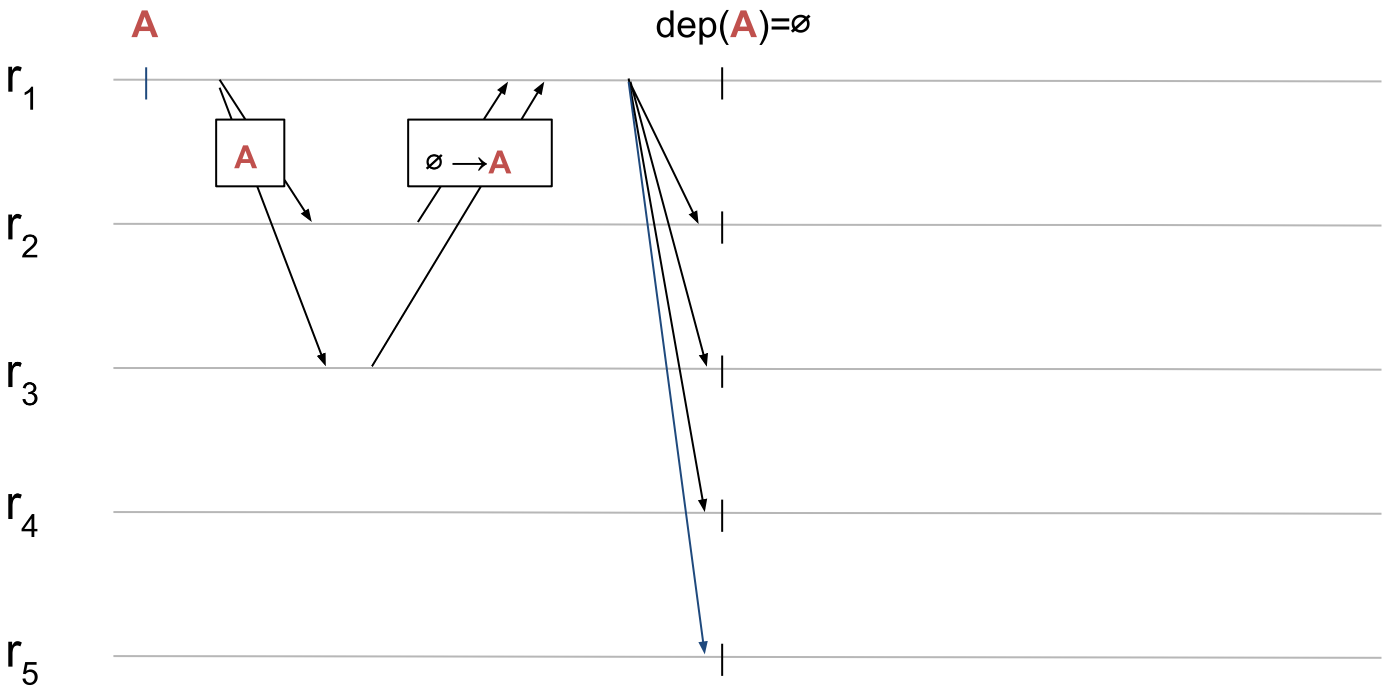
(n=5, f=2)

EPaxos



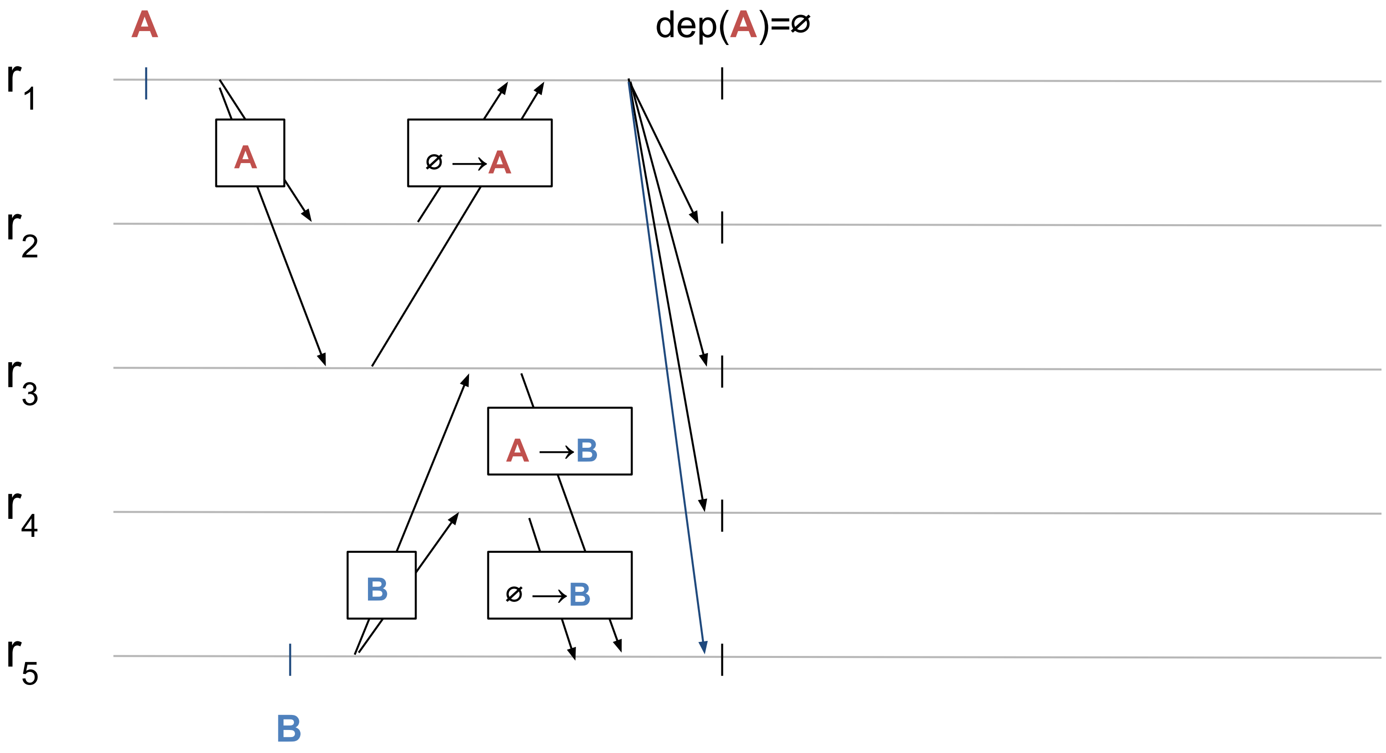
(n=5, f=2)

EPaxos



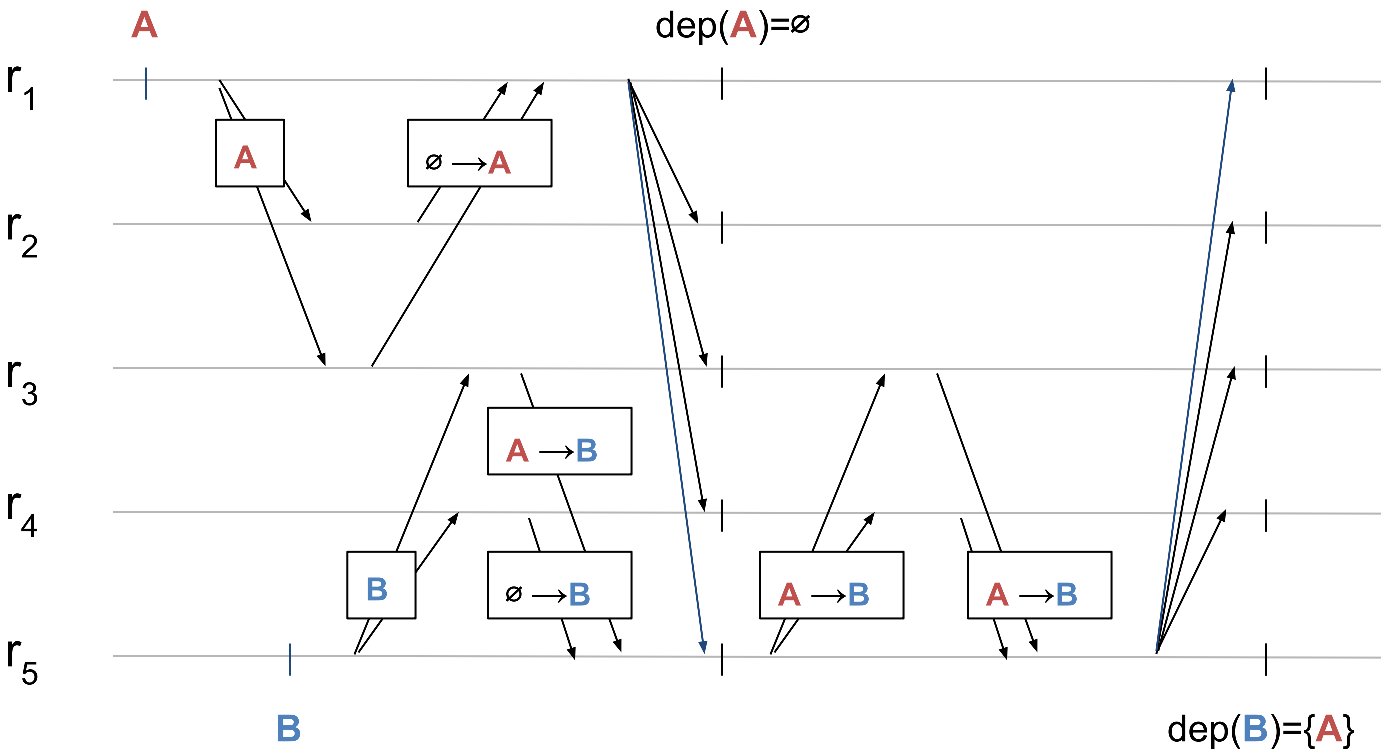
(n=5, f=2)

EPaxos



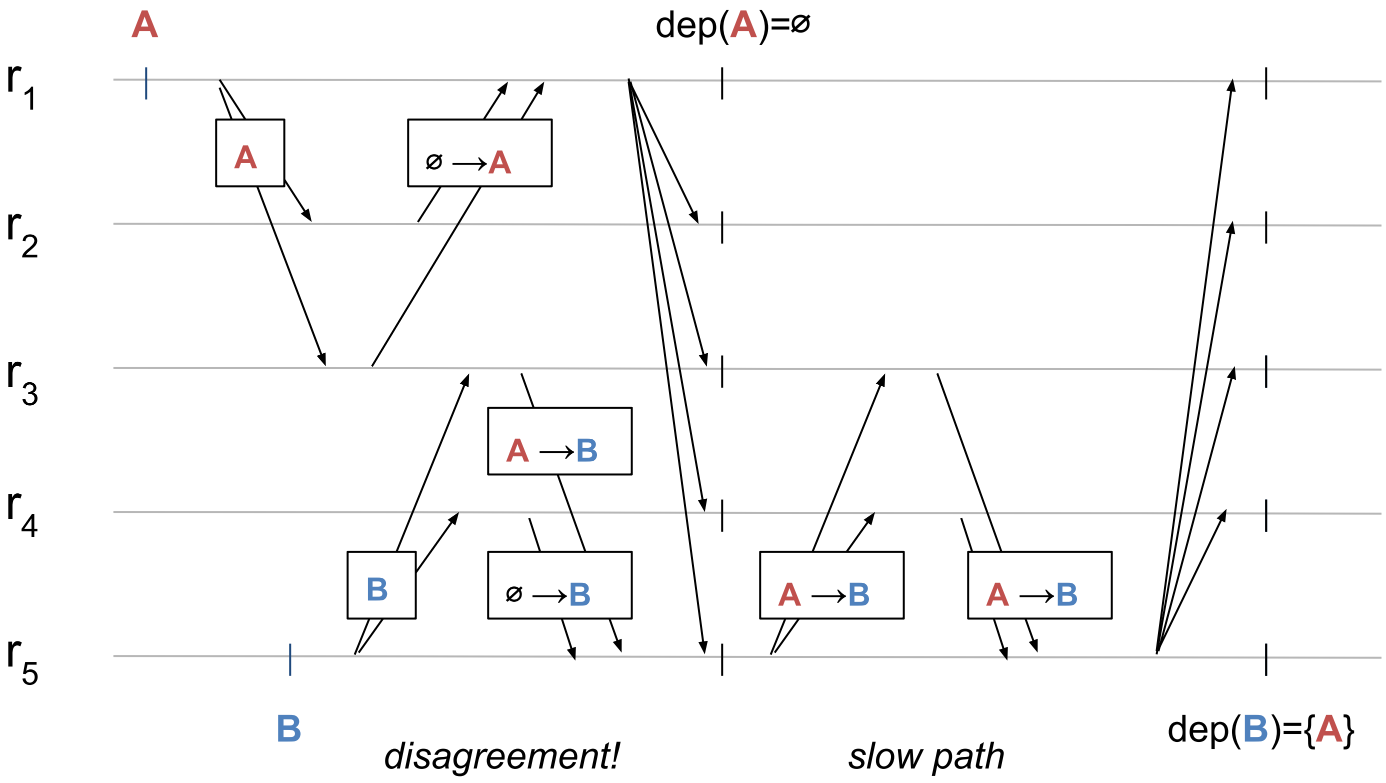
(n=5, f=2)

EPaxos



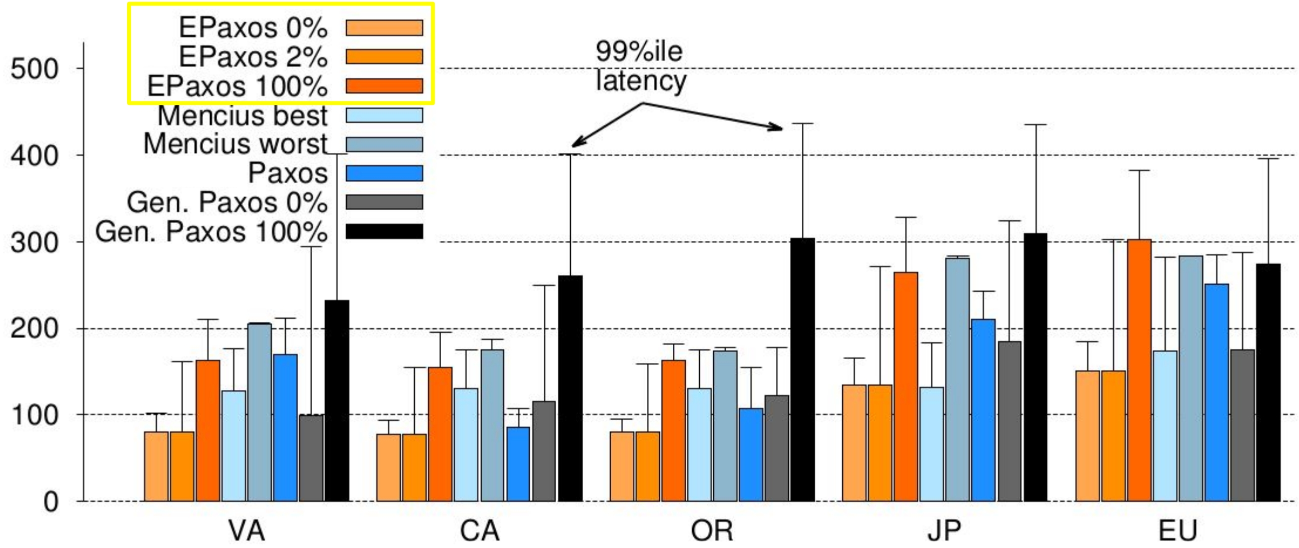
(n=5, f=2)

EPaxos


A
dep(A)=∅
r1
r2
r3
r4
r5
A
∅ →A
A →B
B
∅ →B
A →B
A →B
B
dep(B)={A}
disagreement!
slow path

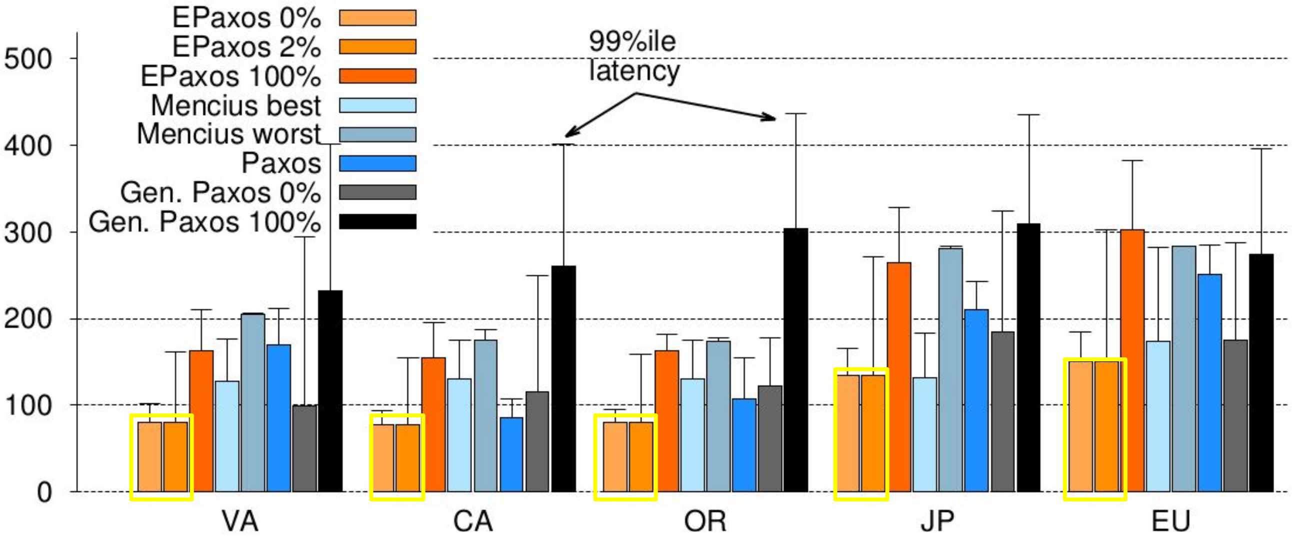
(n=5, f=2)

EPaxos - AWS experiments



VA, CA, .. = datacenters
x% = ratio of commuting commands
(≤ 2% is typical)

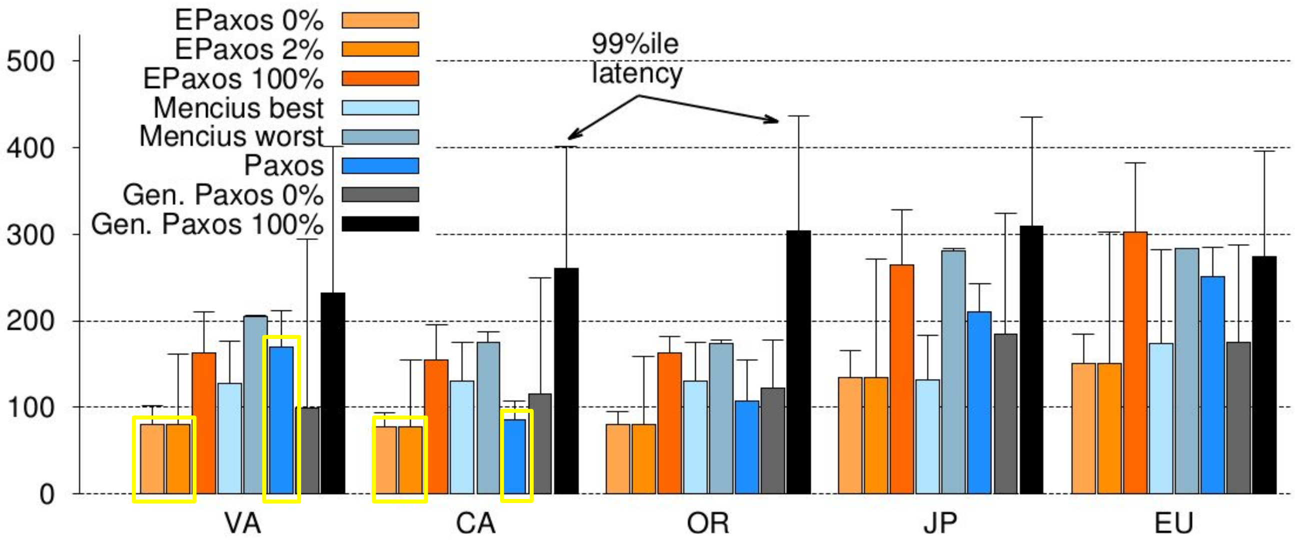
EPaxos - AWS experiments



Takeaways:

- leaderless SMR is <u>faster</u>

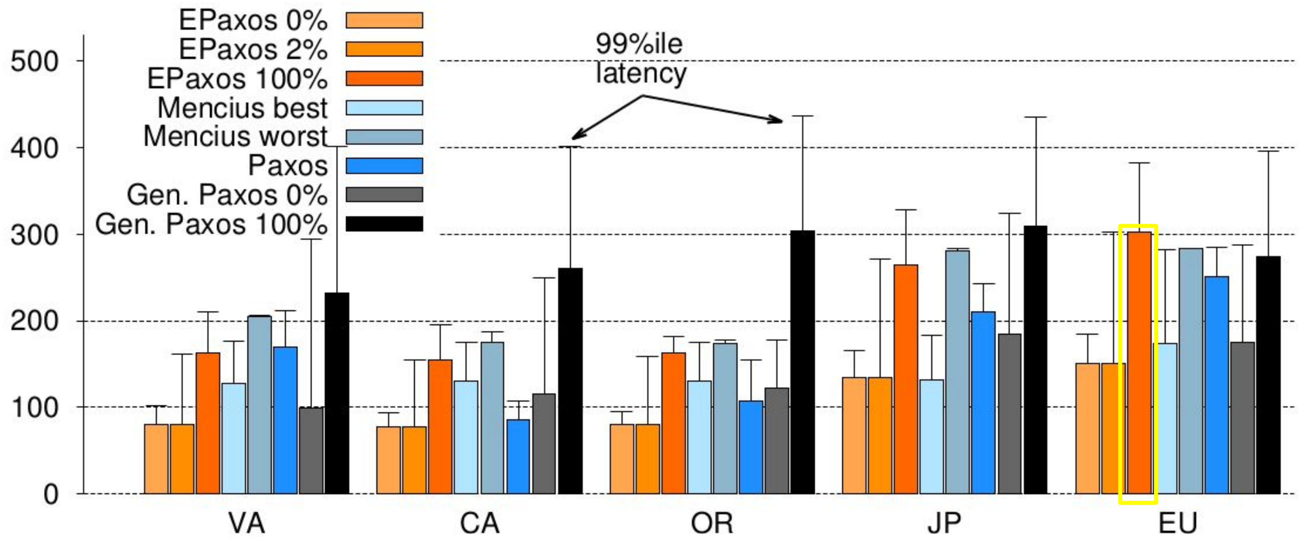
EPaxos - AWS experiments



Takeaways:

- leaderless SMR is faster and <u>more fair</u>

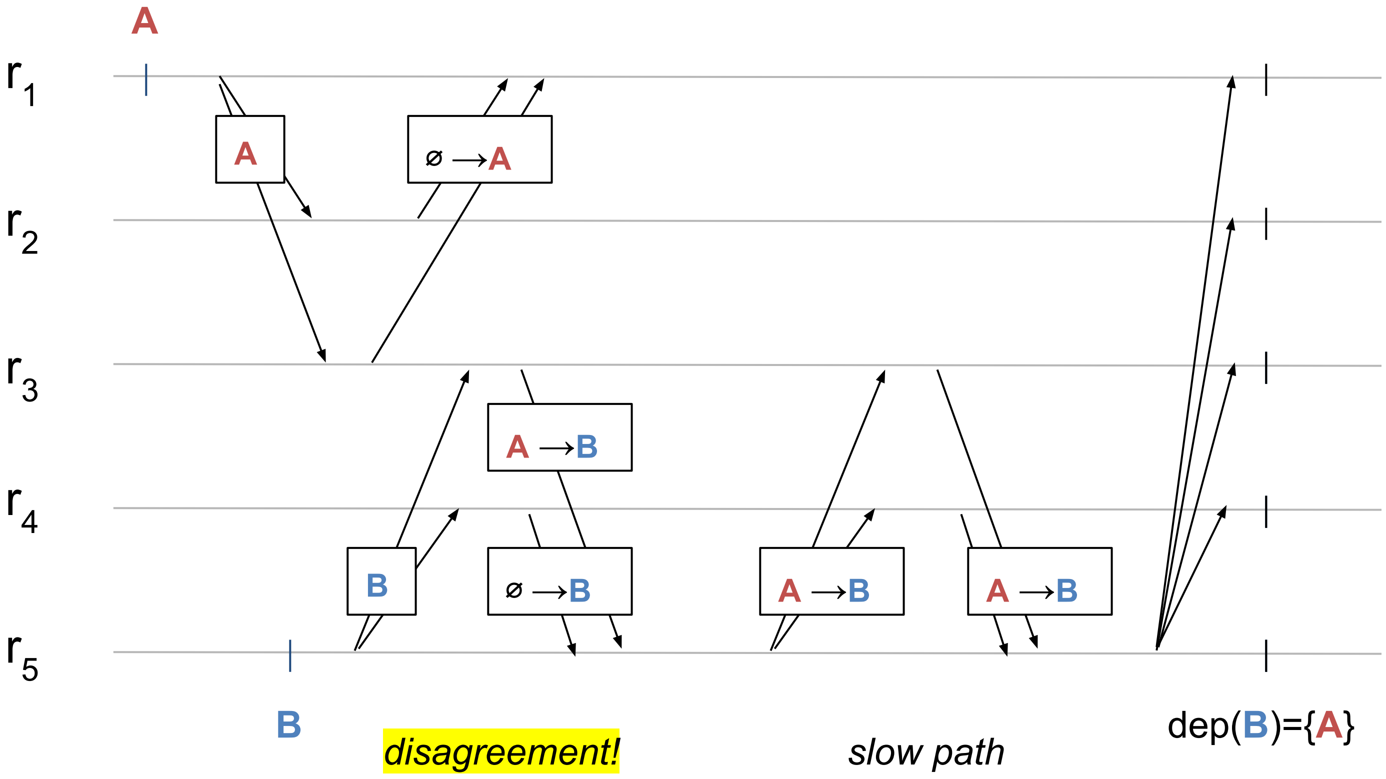
EPaxos - AWS experiments



Takeaways:

- leaderless SMR is faster and more fair
- but commands *should commute*

EPaxos



(n=5, f=2)

Atlas [*Eurosys'20*]

goal: avoid disagreement

Consider a bag of items E, the *k-threshold union* of E, written $\dot{\cup}_k$ E, are the items reported at least k+1 times in the sets of E
formally,

$$\dot{\cup}_k E \overset{\text{def}}{=} \{ Y : \mathrm{count}(Y) \geq k+1 \}$$

Example:

let $E = \{E_1,E_2,E_3\}$ with $E_1= \{$**A**,**B**,**C**$\}$, $E_2= \{$**A**,**C**$\}$ and $E_3= \{$**A**$\}$
then

- $\dot{\cup}_1 E = \{$**A**,**C**$\}$,
- $\dot{\cup}_2 E = \{$**A**$\}$,

Atlas

goal: to avoid disagreement

EPaxos fast path condition:

let Q be a fast path quorum (*f*+*f/2* replicas)

given $q \in Q$, let dep_q be the dep. reported by q

then

fast-path **iff** $\forall$ q,p $\in$ Q. $dep_q = dep_p$

Atlas

goal: to avoid disagreement

Atlas fast path condition:

given $q \in Q$, let dep_q be the dep. reported by q

then

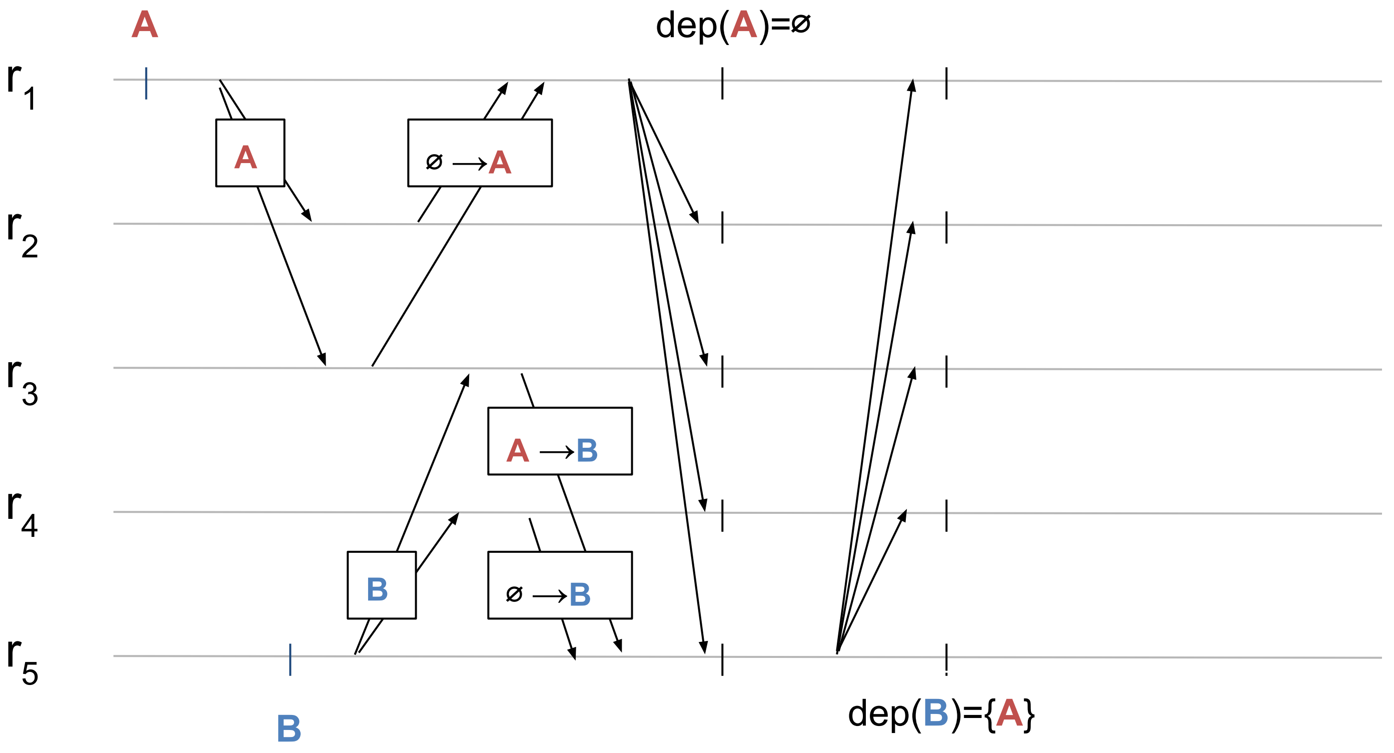
fast-path **iff** $\cupdot_f Q = \cup_q dep_q$

(i.e., every dep. is reported at least f+1 times)

why this works?

- if a failure occurs, the dep. reported by any majority quorum in Q is exactly $\cupdot_f Q$

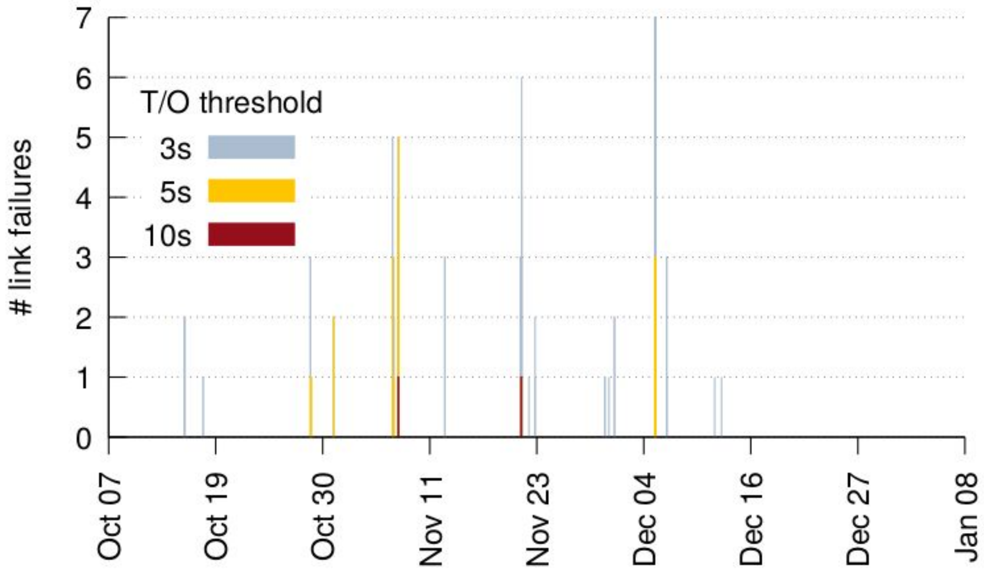
Atlas



** the coordinator counts as the union of the reported deps.*

(n=5, f=1)

Atlas - asynchrony in practice

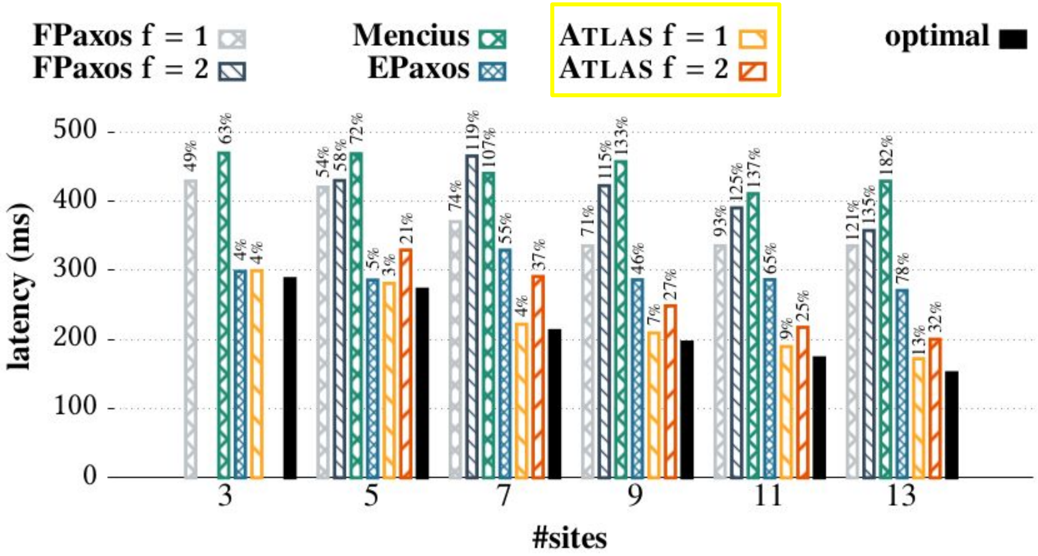


13 GCP sites
all-to-all ping
over 3 months

Takeaways:

- concurrent link failures is a rare event at scale
- at most one slow site during the exp. (f=1)

Atlas - GCP experiments

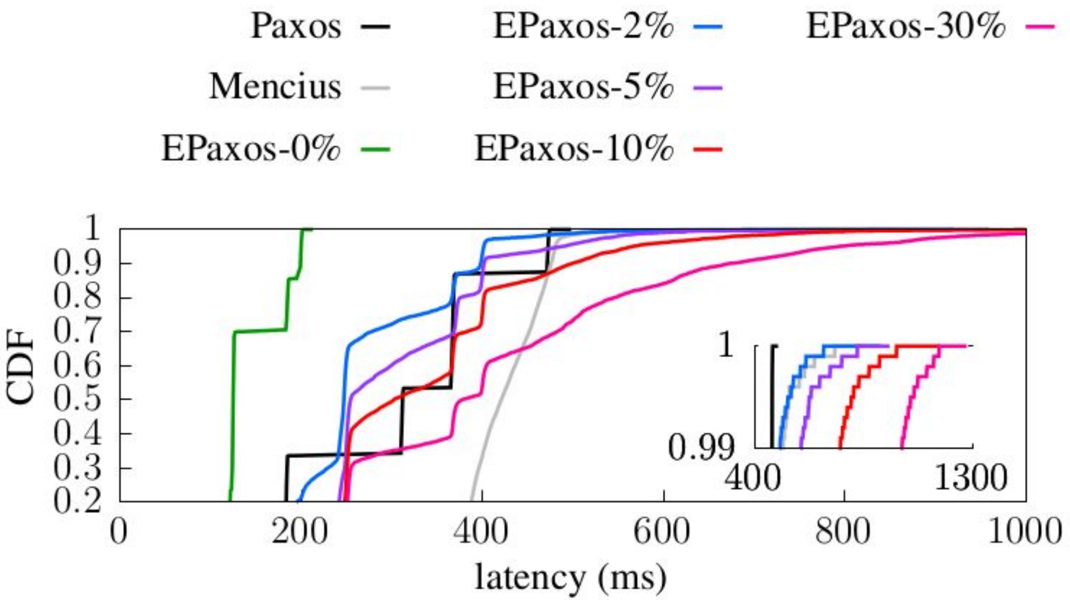


x% = how far from optimal

Takeaways:

- Atlas better than EPaxos for large-scale deployment (n ≥ 5)

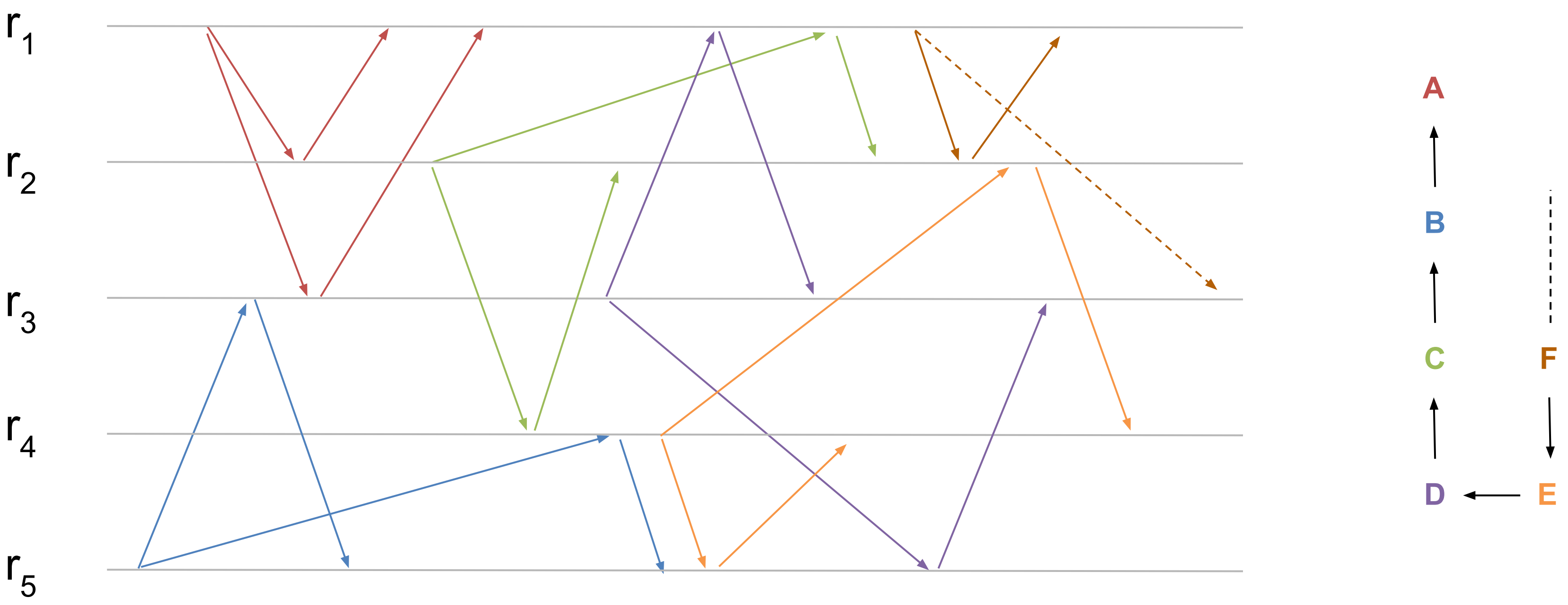
Tail latency [*DISC'20, NSDI'21*]



Takeaways:

- Tail latency in leaderless SMR protocols is a problem

Tail latency



(n=5, f=1)

Tempo [*Eurosys’21*]

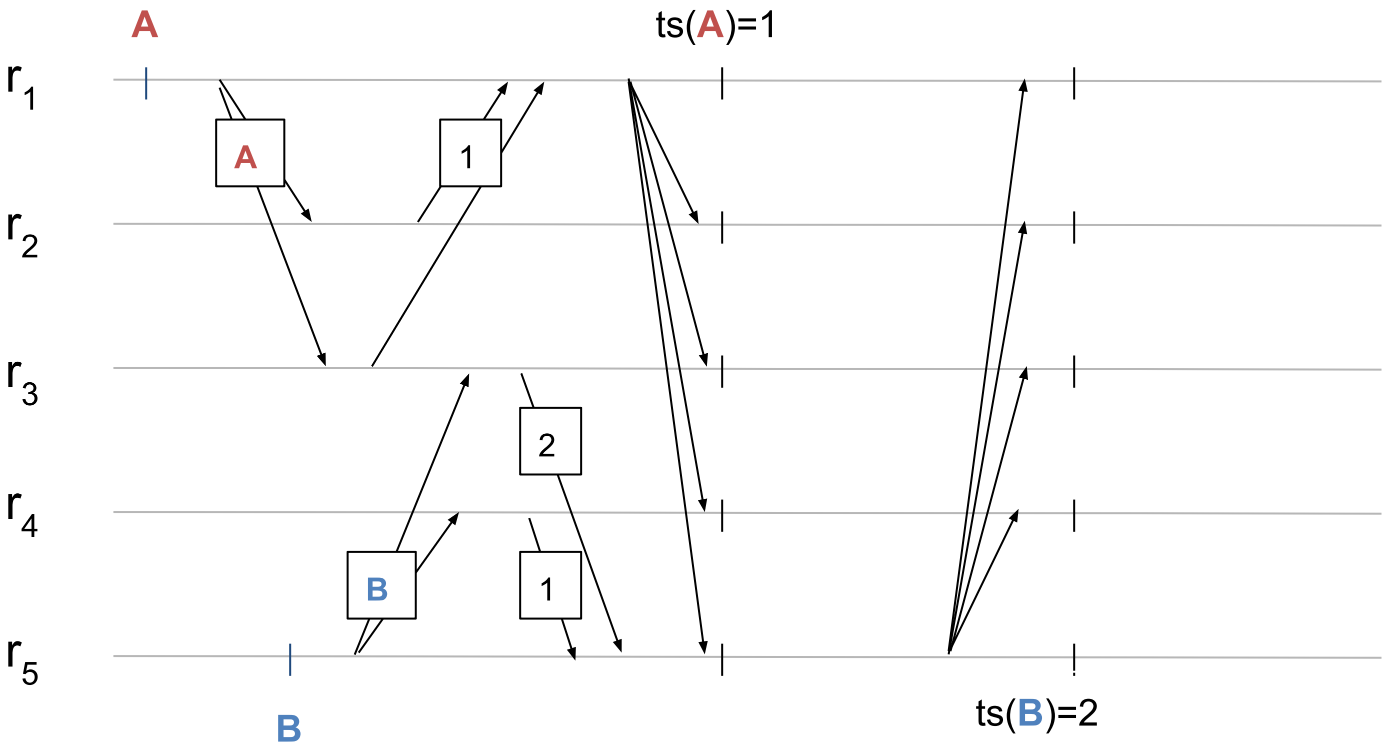
goal: tame tail latency

key idea: agree on a timestamp per command
+ make the timestamp *stable*

Tempo fast path condition:

given $q \in Q$, let ts_q be the timestamp reported, or *promised,* by q
then
fast-path **iff** let $t = \max\{ ts_q : q \in Q\})$
then $\mathrm{count}(t) \geq f+1$

Tempo



(n=5, f=1)

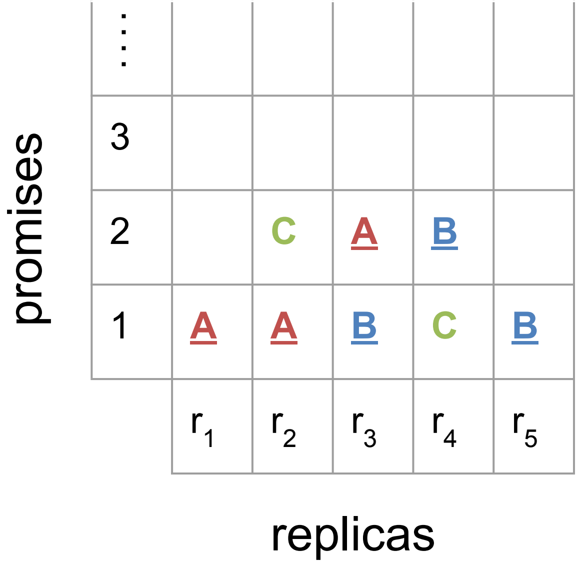
Tempo - background stability mechanism

A command is stable once

- its timestamp, say t, is agreed;
- every command with a timestamp lower (or equal) to t is stable; and
- a quorum reports promises higher (or equal) to t.

Stable commands are executed in the order of timestamps (ties are broken arbitrarily)

Here, A;B
as ts(A) = ts(B) and A < B

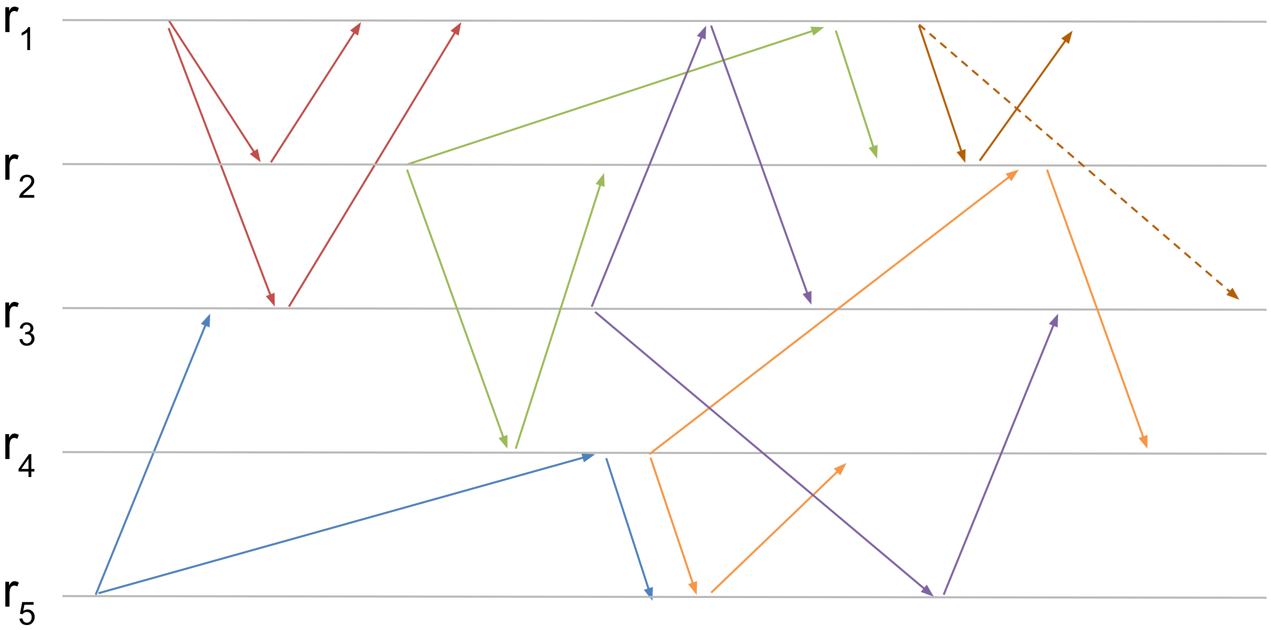


promises					
⋮					
3					
2		C	A	B	
1	A	A	B	C	B
	r_1	r_2	r_3	r_4	r_5

replicas

X = command is stable

Tempo - background stability mechanism

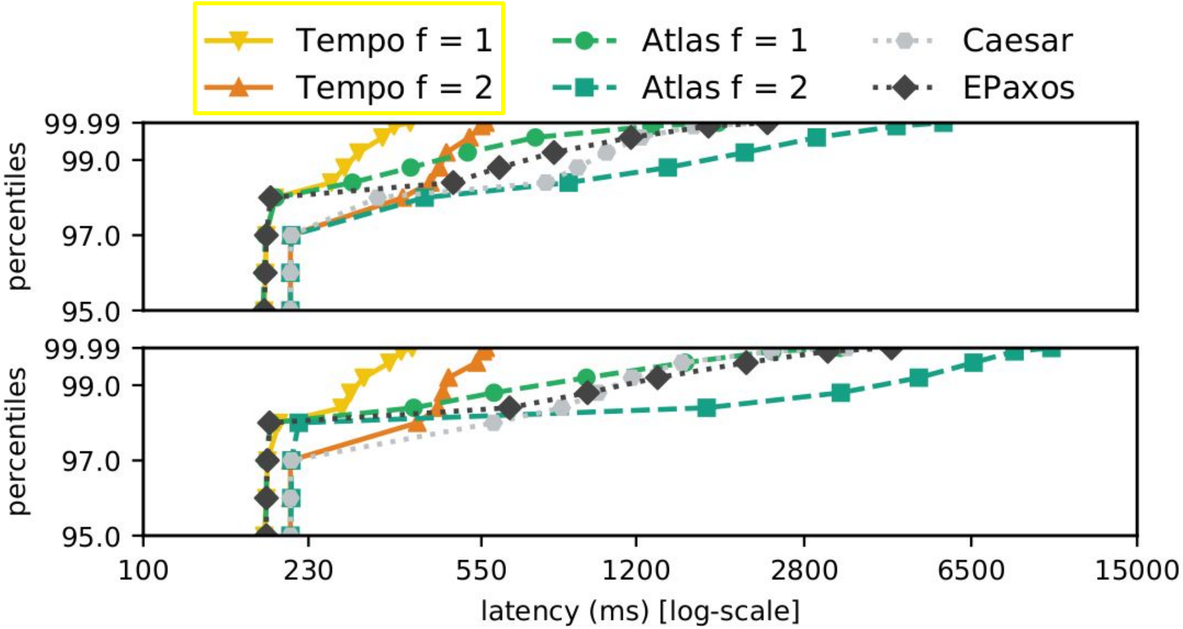


⋮					
3	C		D		D
2	D	C	A	B	E
1	A	A	B	C	B
	r_1	r_2	r_3	r_4	r_5

A;B;C

(n=5, f=1)

Tempo



5 GCP sites
512/256 (top/bottom)
clients per site
conflict rate is 2%

Takeaways:

- Tempo improves tail latency in leaderless SMR

Conclusion

Leaderless SMR

- graph-based ordering of commands
- a coordinator per command **X**
    - runs consensus on dep(**X**)
- better than Paxos/Raft

Future directions

- higher scalability
- byzantine failures

References

[Paxos] The Part-Time Parliament, *L. Lamport, ACM TOCS, 1998.*

[BGcast] Generic Broadcast, *F. Pedone, A. Schiper, DISC, 1999.*

[GPaxos] Generalized Consensus and Paxos, *L. Lamport, Tech Report, 2005.*

[DISC'05] Generic Optimistic Broadcast, *F. Pedone, P. Zieliński, DISC, 2005.*

[SOSP'13] There is more consensus in Egalitarian parliaments, *I. Moraru, D. G. Andersen, M. Kaminsky, SOSP, 2013.*

[Raft] In search of an understandable consensus algorithm, *D. Ongaro, J. Ousterhout, USENIX ATC, 2014.*

[DISC'20] Leaderless State-Machine Replication: Specification, Properties, Limits, *T. F. Rezende, P. Sutra, DISC, 2020.*

[Eurosys'20] State-Machine Replication for Planet-Scale Systems, *V. Enes, M. Perrin, C. Baquero, A. Gotsman, P. Sutra, Eurosys, 2020.*

[NSDI'21] EPaxos Revisited, *S. Tollman, J. Ousterhout, NSDI, 2021.*

[Eurosys'21] Efficient Replication via Timestamp Stability, *V. Enes, C. Baquero, A. Gotsman, P. Sutra, Eurosys, 2021.*