SUPPORTING AND ANALYZING PROBABILISTIC CONSISTENCY IN DISTRIBUTED STORAGE SYSTEMS

Wojciech Golab
wgolab@uwaterloo.ca

UNIVERSITY OF WATERLOO

OUTLINE

Background and Motivation

- quorum replication
- weak consistency using partial quorums

Overview of Prior Work

- probabilistic quorums and random registers
- probabilistically bounded staleness
- consistency benchmarking
- consistency-latency tuning

Ongoing Work at Waterloo

- mathematical model of eventual consistency
- improved consistency-latency tuning

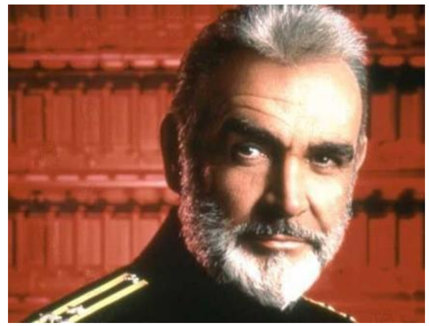
Background and Motivation

ROLES OF RANDOMIZATION

computability

complexity

RANDOMIZED MUTUAL EXCLUSION



adversary
(controls schedule of process steps)

complexity measure:
number of remote memory references (RMRs) required to enter and leave the critical section once

(max. number of processes: *N*) → input → algorithm → output → (order of entry into critical section)

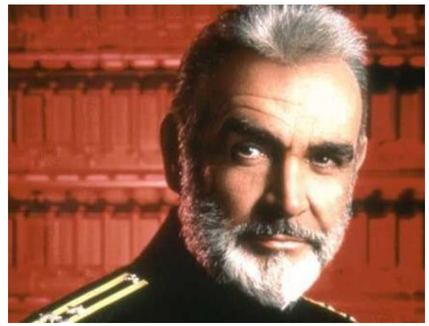
RANDOMIZED MUTUAL EXCLUSION

Time complexity of one passage through a mutual exclusion algorithm in the asynchronous shared memory model with Read and Write operations:

Worst-case: $\Omega(\log N)$
Attiya, Hendler, and Woelfel (2008)

Expected: $O(\log N / \log \log N)$
Hendler and Woelfel (2009/2011)

CONCURRENT DATA STRUCTURE (SHARED MEMORY)

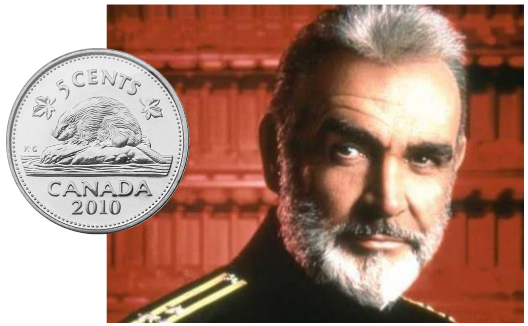


adversary
(controls schedule of process steps and the operations invoked)

complexity measure:
number of steps required to complete one operation

(max. number of processes: *N*) input → algorithm → output (operation responses)

DISTRIBUTED STORAGE SYSTEM
(S.M. ON TOP OF MESSAGE PASSING)

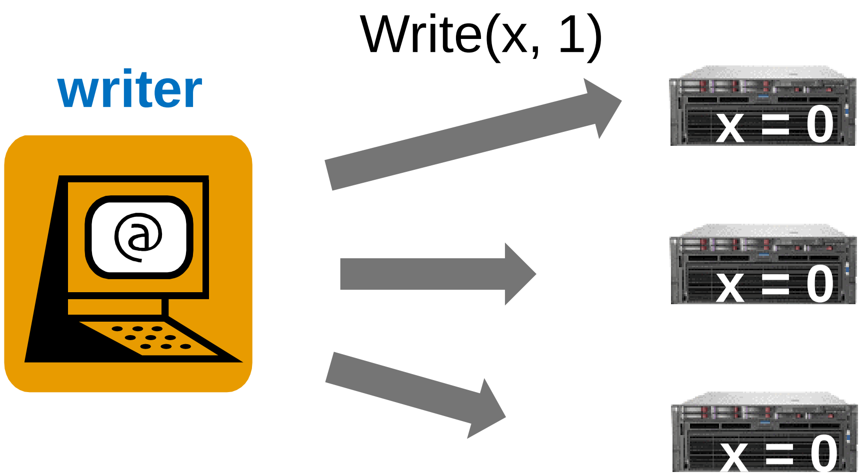


performance metrics:
latency, consistency
(real numbers!)

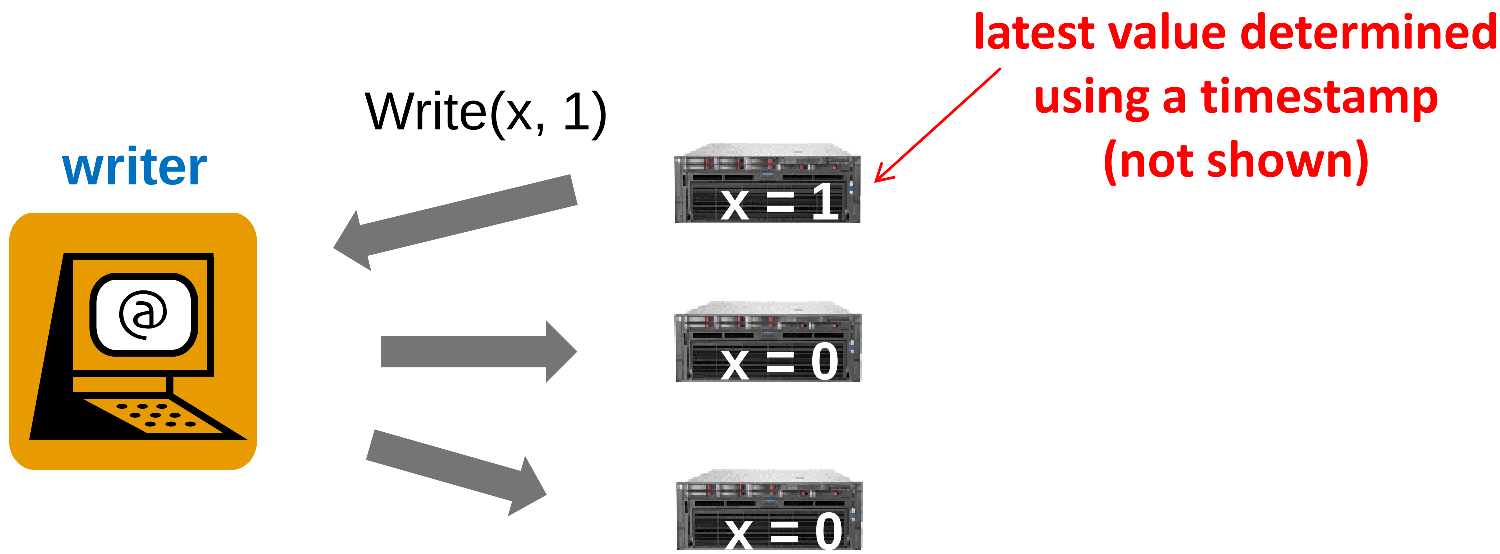
adversary
(controls schedule of process steps, operations invoked, message delays)

(tuning knobs) input → algorithm → output (operation responses)

WEAK CONSISTENCY IN ACTION

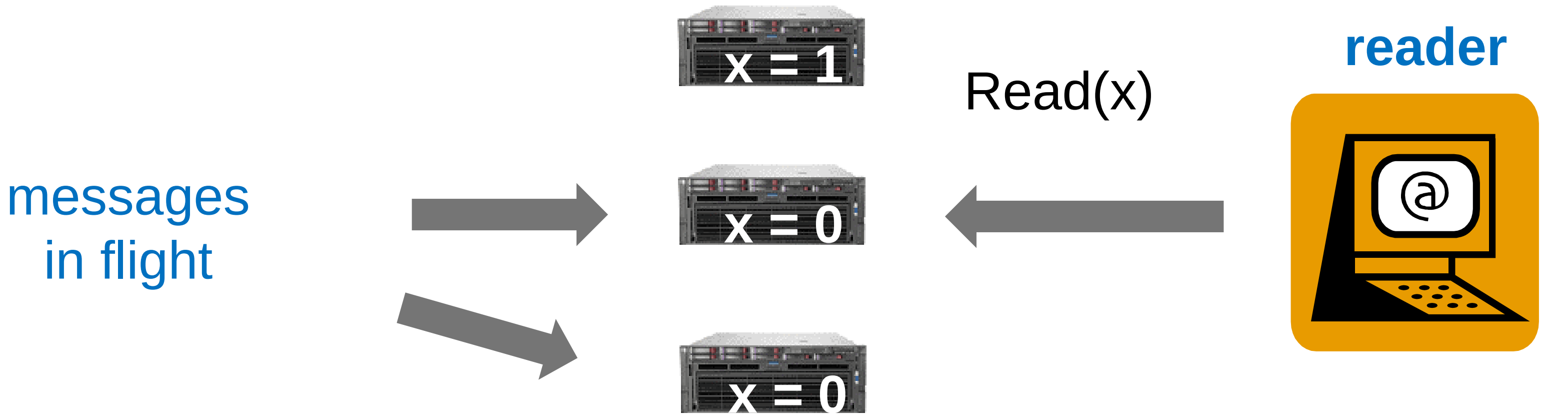


WEAK CONSISTENCY IN ACTION

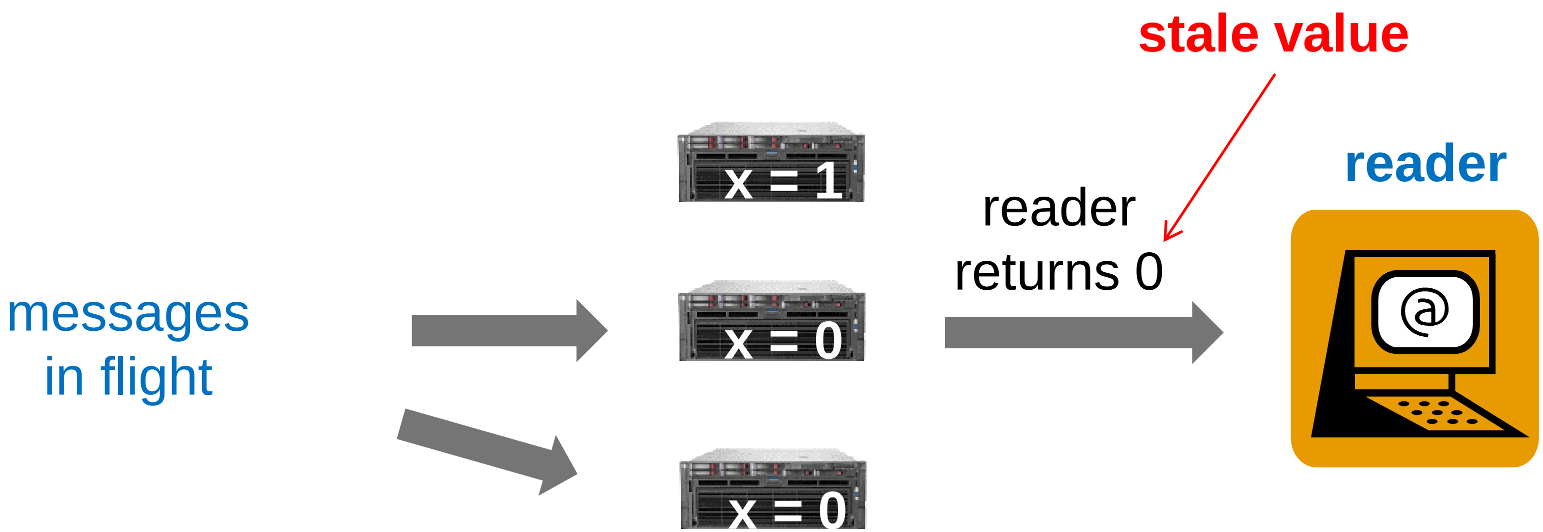


(waiting for one
replica to respond)

WEAK CONSISTENCY IN ACTION

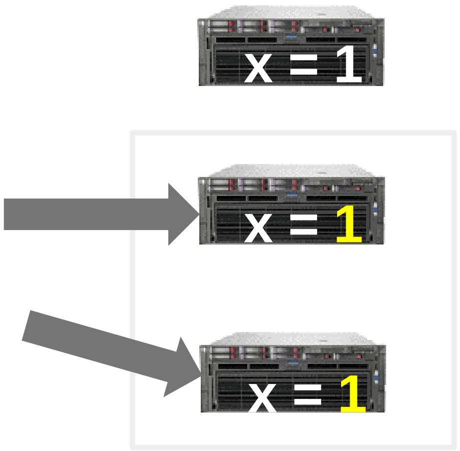


WEAK CONSISTENCY IN ACTION

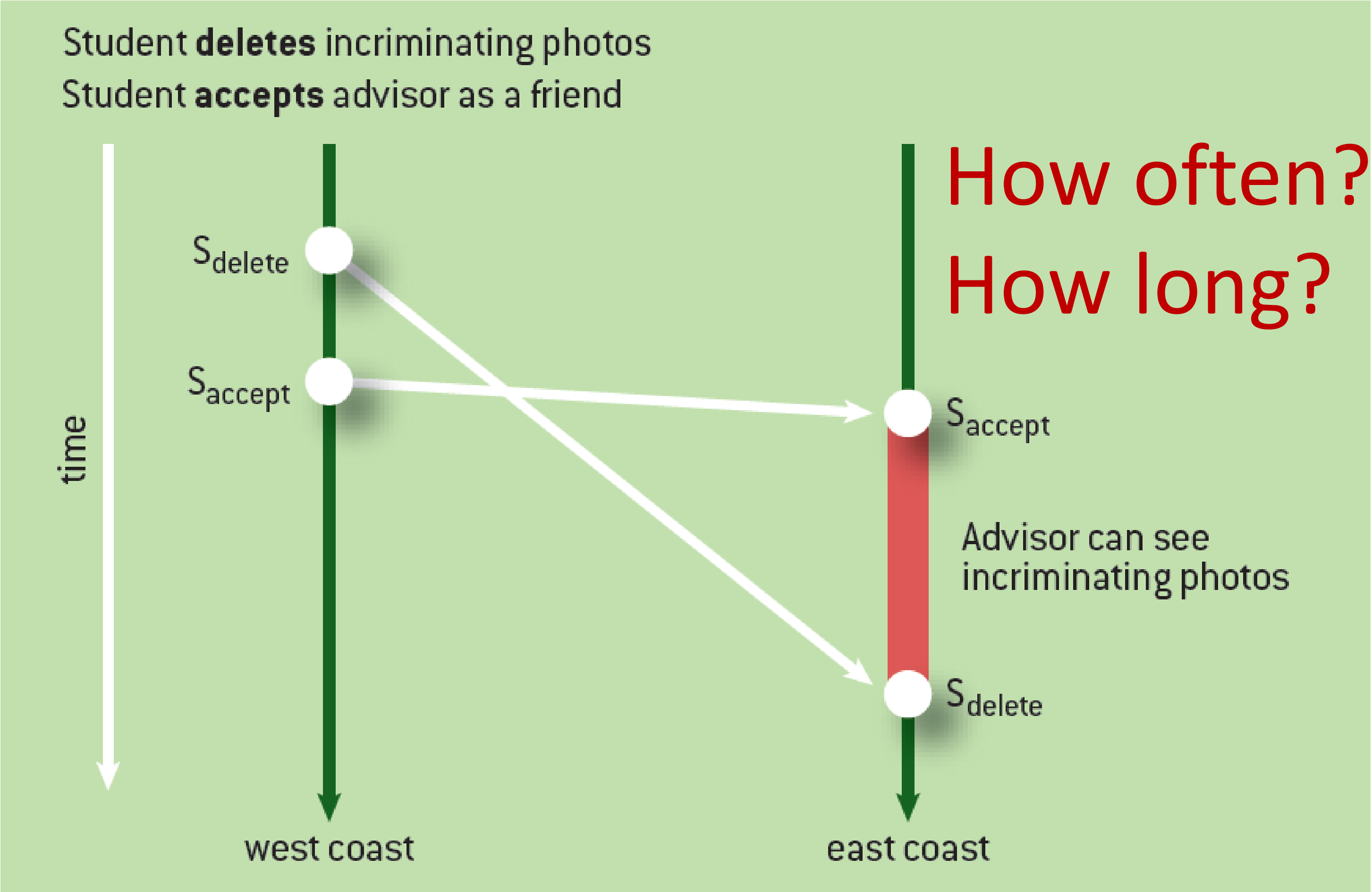


WEAK CONSISTENCY IN ACTION

messages
arrive



STALE READS CONSIDERED DANGEROUS!



"Photo privacy violation" example from
Lloyd, Freedman, Kaminsky, and Andersen (2014)

GOAL

What is the expected proportion of stale reads in the following workload?

- 6 servers
- replication factor 3, partial quorum size 1
- 1000 ops/s/server, Poisson arrivals
- 25% Write, 75% Read operations
- mean network delay 100ms, exponentially distributed
- processing delay 0ms

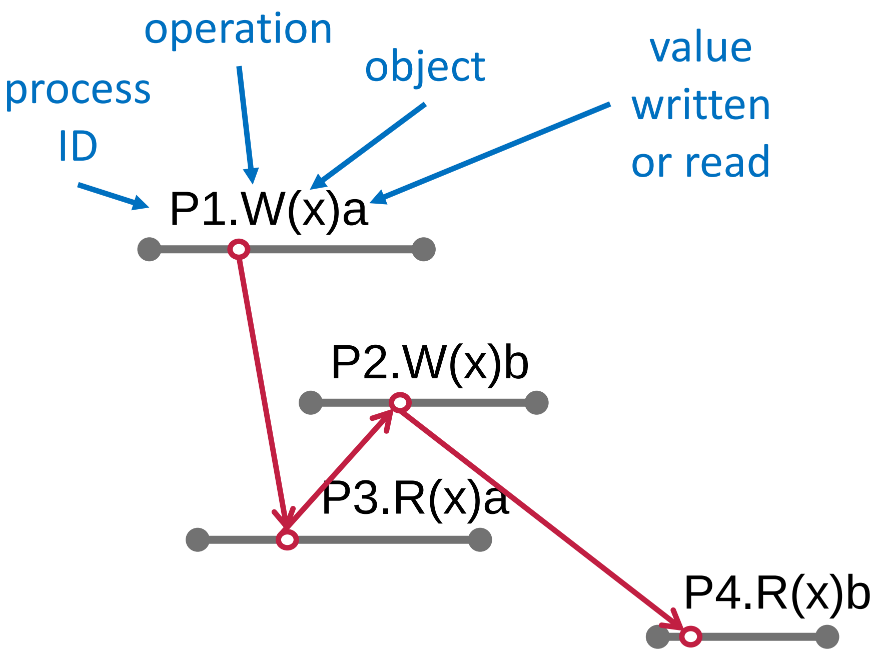
Overview of Prior Work

ASSUMPTIONS

- Read and Write operations
- asynchronous model
- processes may fail by crashing
- network is reliable but delays not bounded

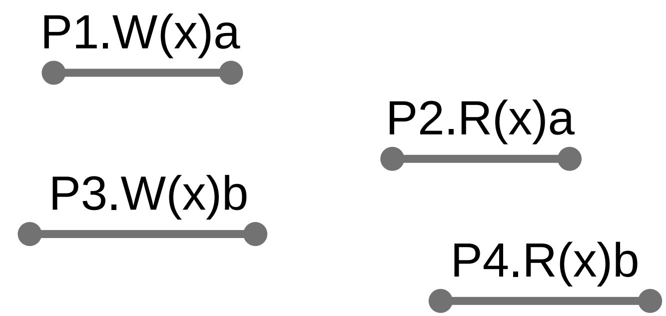
- exceptions: link failures and bounded network delays in some papers

LINEARIZABILITY



P1.W(x)a P3.R(x)a P2.W(x)b P4.R(x)b

linearizable



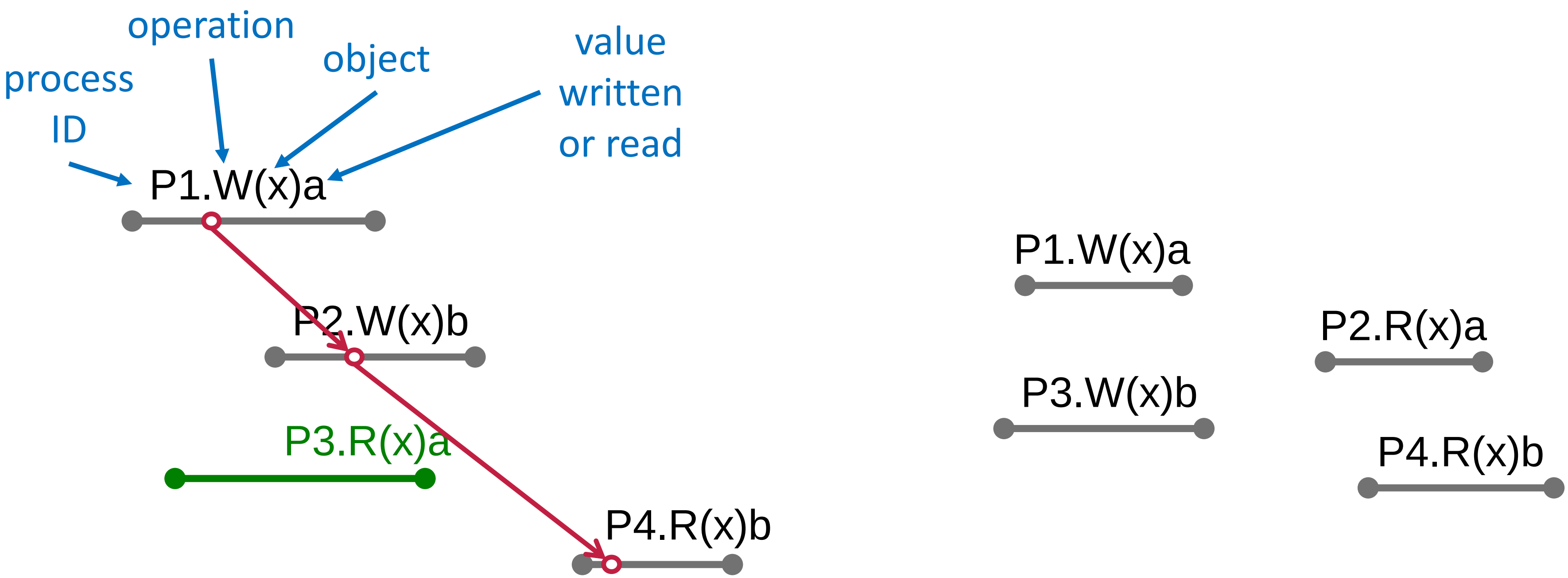
?W(x)? ?W(x)? ?R(x)? ?R(x)?

not linearizable

time →

Herlihy and Wing (1990)

REGULARITY (GENERALIZED)



P1.W(x)a P2.W(x)b P4.R(x)b

?W(x)? ?W(x)? ?R(x)? ?R(x)?

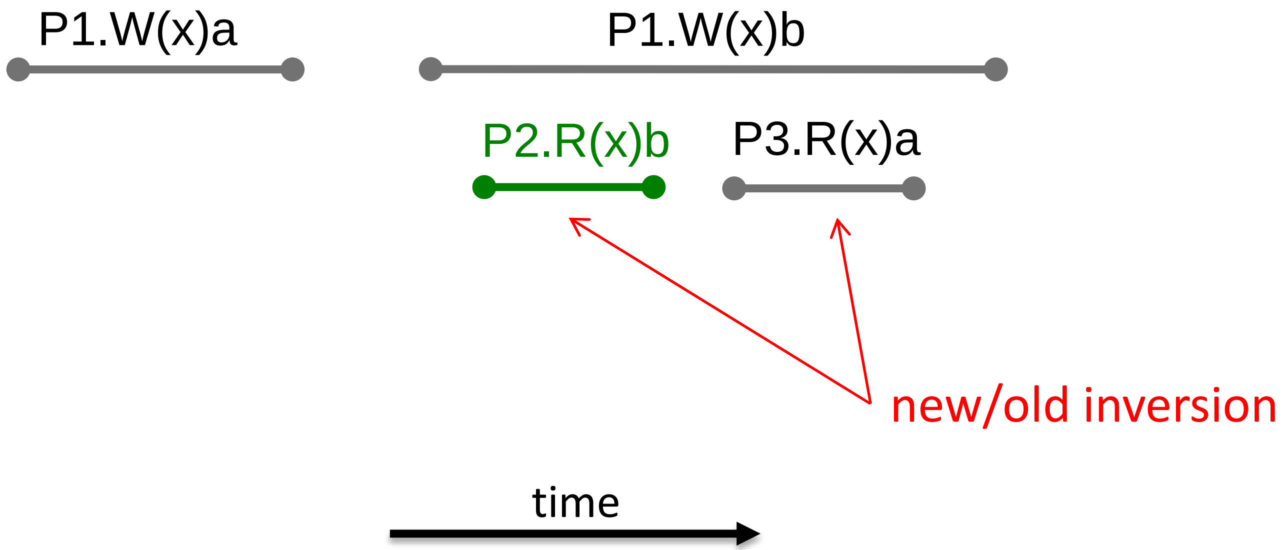
regular

time →

not regular

based on Lamport (1986)

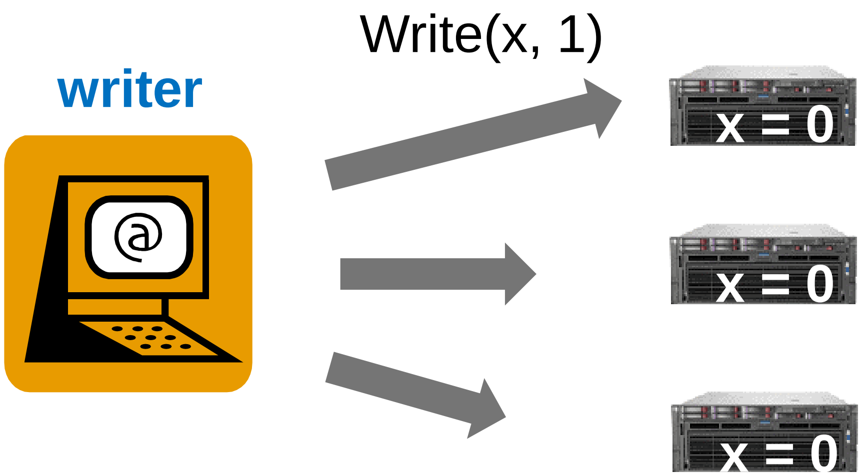
EXAMPLE OF HISTORY THAT IS REGULAR BUT NOT LINEARIZABLE



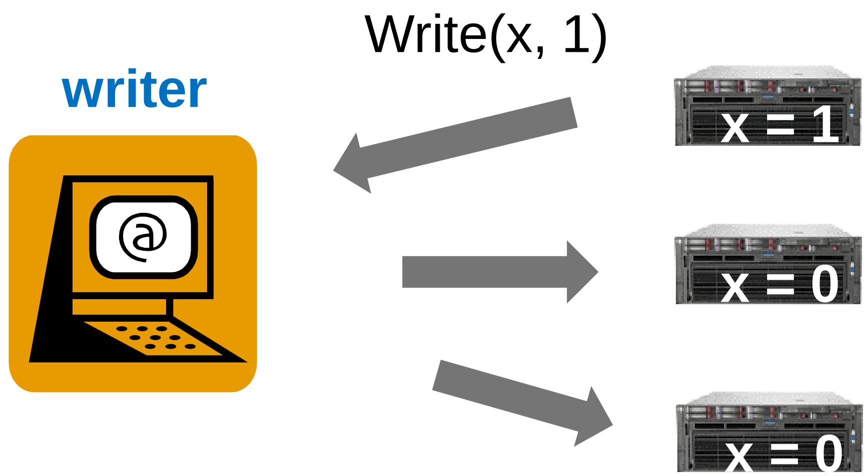
ABD SIMULATION

- Attiya, Bar-Noy, Dolev (1990)
- single-writer multi-reader register simulation on top of message passing
- asynchronous model with process crash failures and dynamic link failures
- majority of processes must be correct
- ensures linearizability
- 1 roundtrip for writer, 2 roundtrips for reader

ABD SIMULATION

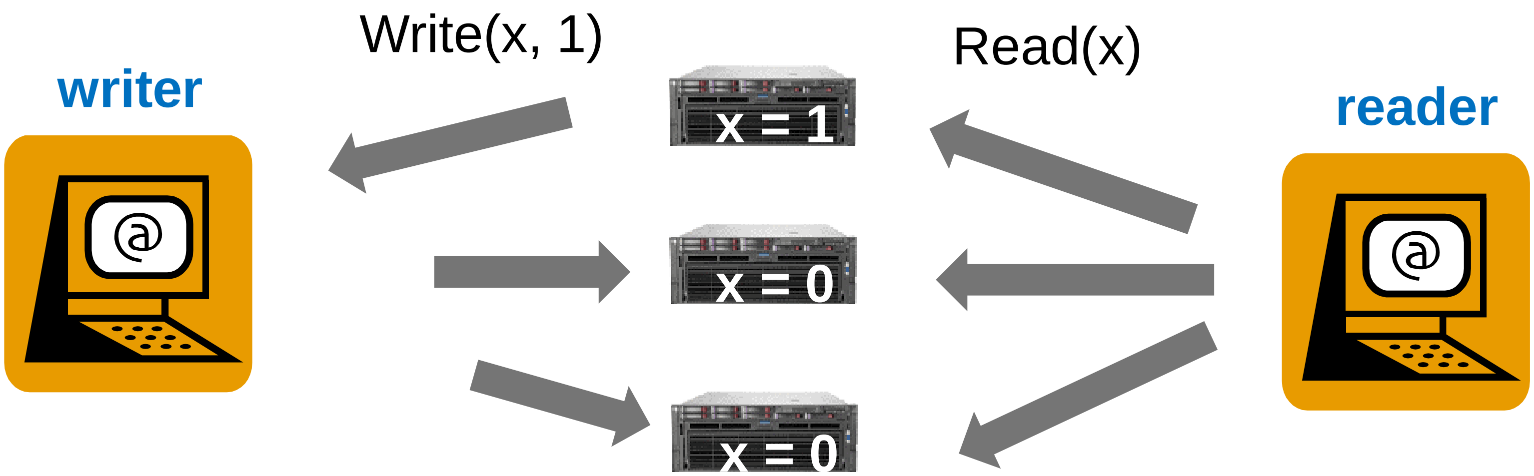


ABD SIMULATION



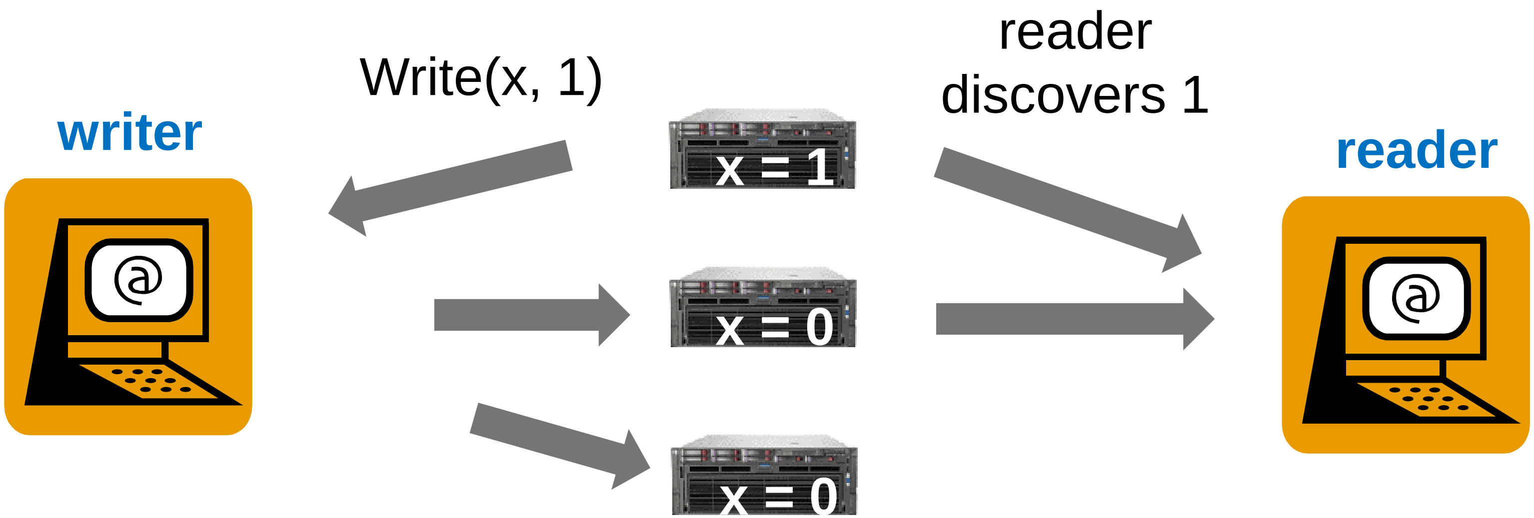
(waiting for majority to respond)

ABD SIMULATION



(waiting for majority
to respond)

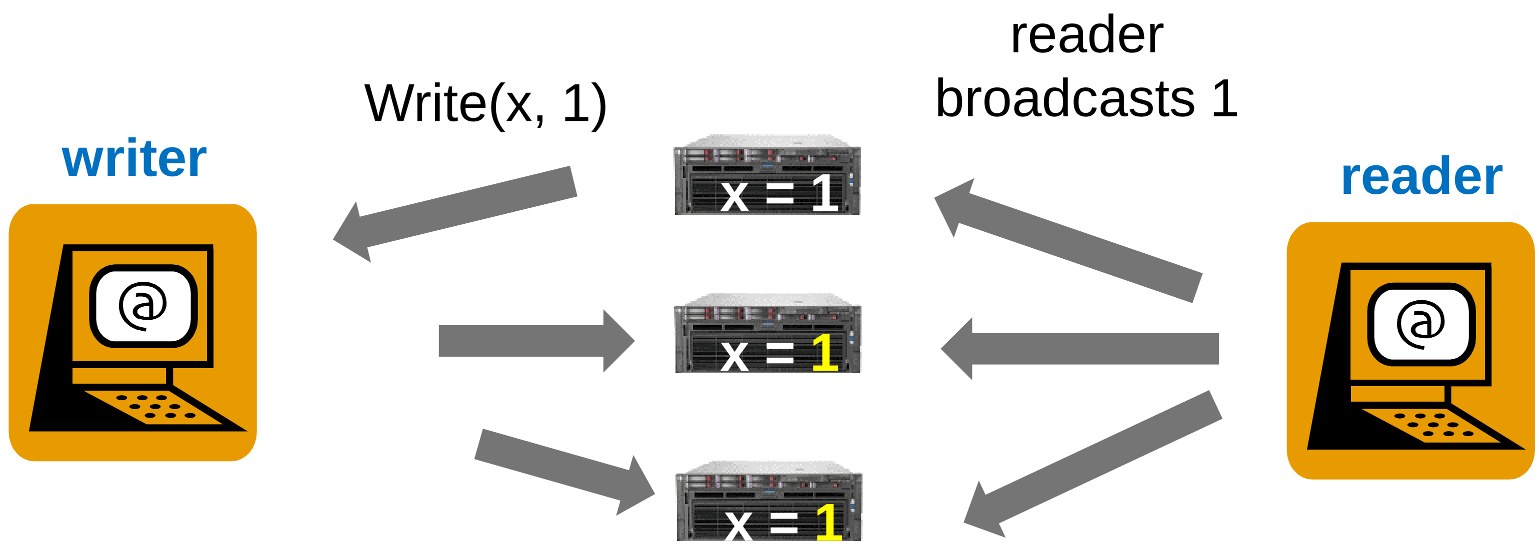
ABD SIMULATION



(waiting for majority
to respond)

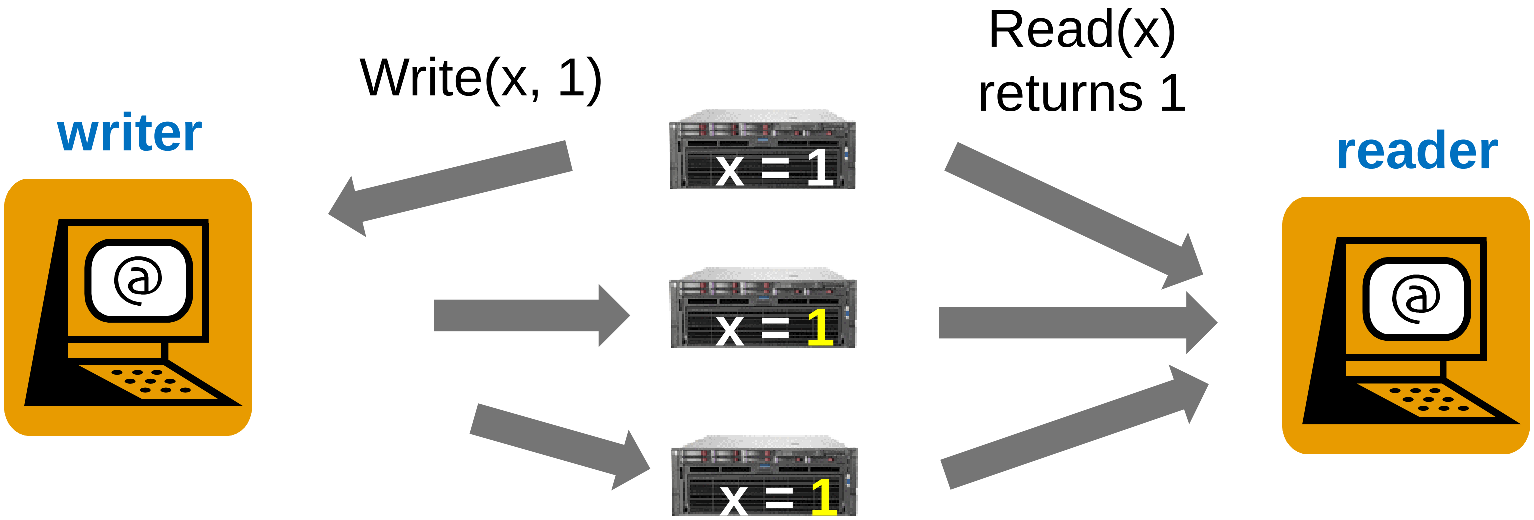
(waiting for majority
to respond)

ABD SIMULATION



(waiting for majority to respond)

ABD SIMULATION



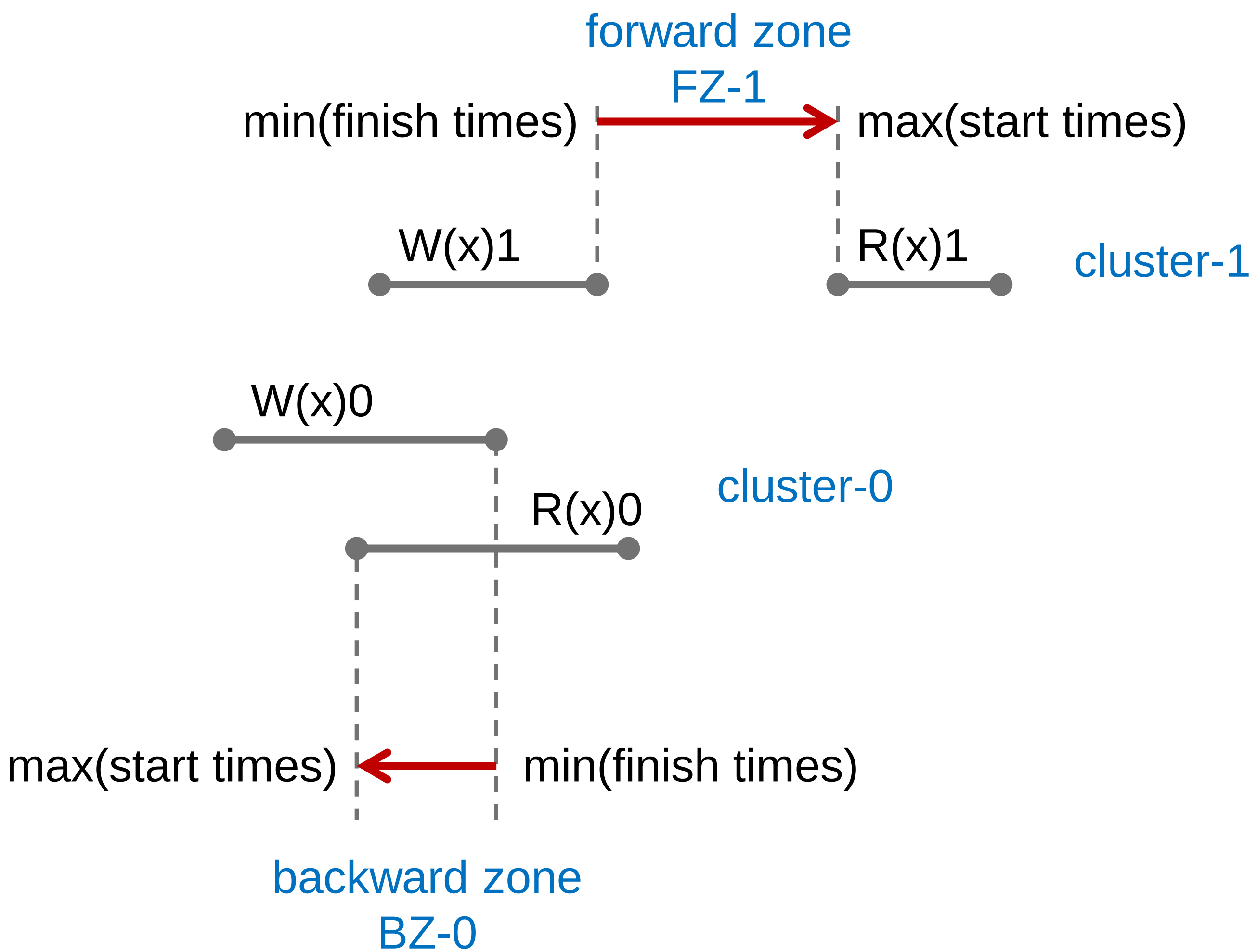
(waiting for majority
to respond)

(two round trips)

DECIDING LINEARIZABILITY

- Gibbons and Korach (1997)
- algorithm works for histories over Read and Write operations
- assumes the "reads-from" mapping is known, for example because all Write operations on a given object assign distinct values
- $O(N \log N)$ steps for a history of N operations

DECIDING LINEARIZABILITY

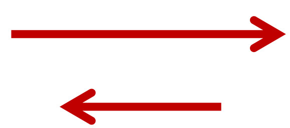


DECIDING LINEARIZABILITY

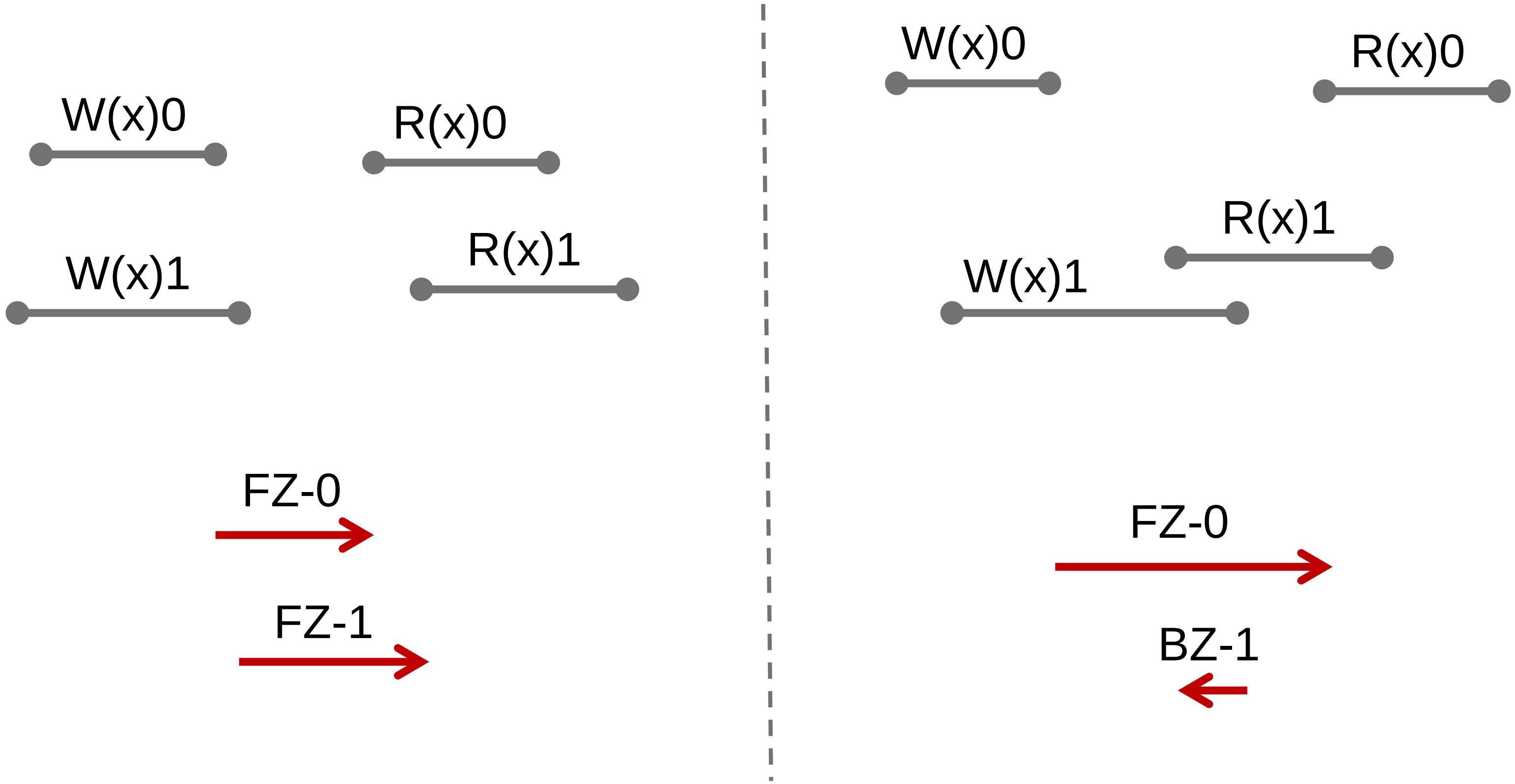
A history of Read and Write operations is linearizable if every Read returns the value of some Write, and no two zones conflict.

Two forward zones conflict if they overlap:

A forward zone conflicts with a backward if the former is a superset of the latter:



DECIDING LINEARIZABILITY



PROBABILISTIC QUORUM SYSTEMS

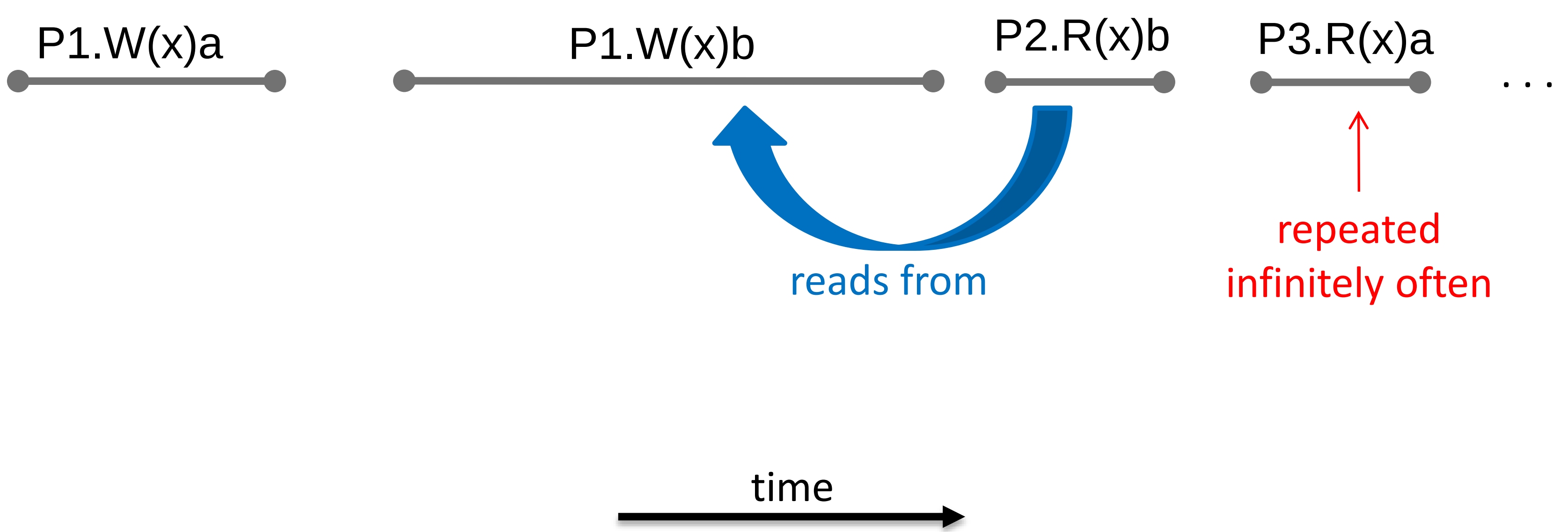
- Malkhi, Reiter, Wool, and Wright (2001)
- ε-intersecting quorum system: any two "quorums" must overlap with probability at least $1 - \varepsilon$ with respect to an access strategy
- example:
  - » $N = 2$ processes
  - » Read and Write operations access one server chosen uniformly at random
  - » $\varepsilon = 1/2$

RANDOMIZED REGISTERS

- Lee and Welch (2004)
- random register satisfies three conditions:
  1. every operation terminates
  2. every read operation reads from some write
  3. for any given write, the probability that this write is read from infinitely often is 0 if there are infinitely many writes
- relaxation of Lamport's regularity property for single-writer multi-reader registers
- implementable using probabilistic quorums
- alternative definitions: *P*-bounded and monotone random registers

RANDOMIZED REGISTERS

Possible behavior:



RANDOMIZED REGISTERS

k = quorum size (uniform access strategy)

l-outdated read: returned value is not allowable but is the value of the *l*-th write preceding the beginning of the read

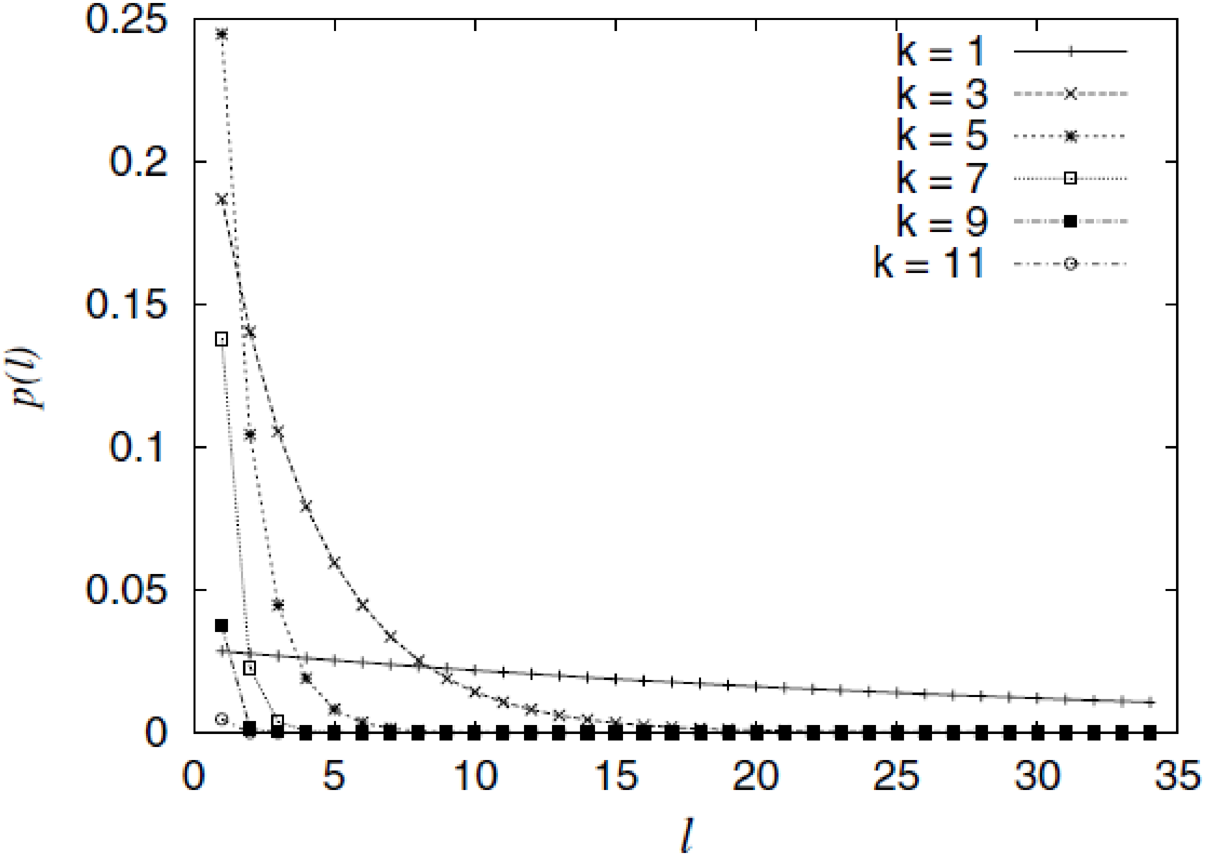


Fig. 2. Probability distribution $p(\ell)$: outdatedness level ℓ vs. probability of being ℓ-outdated

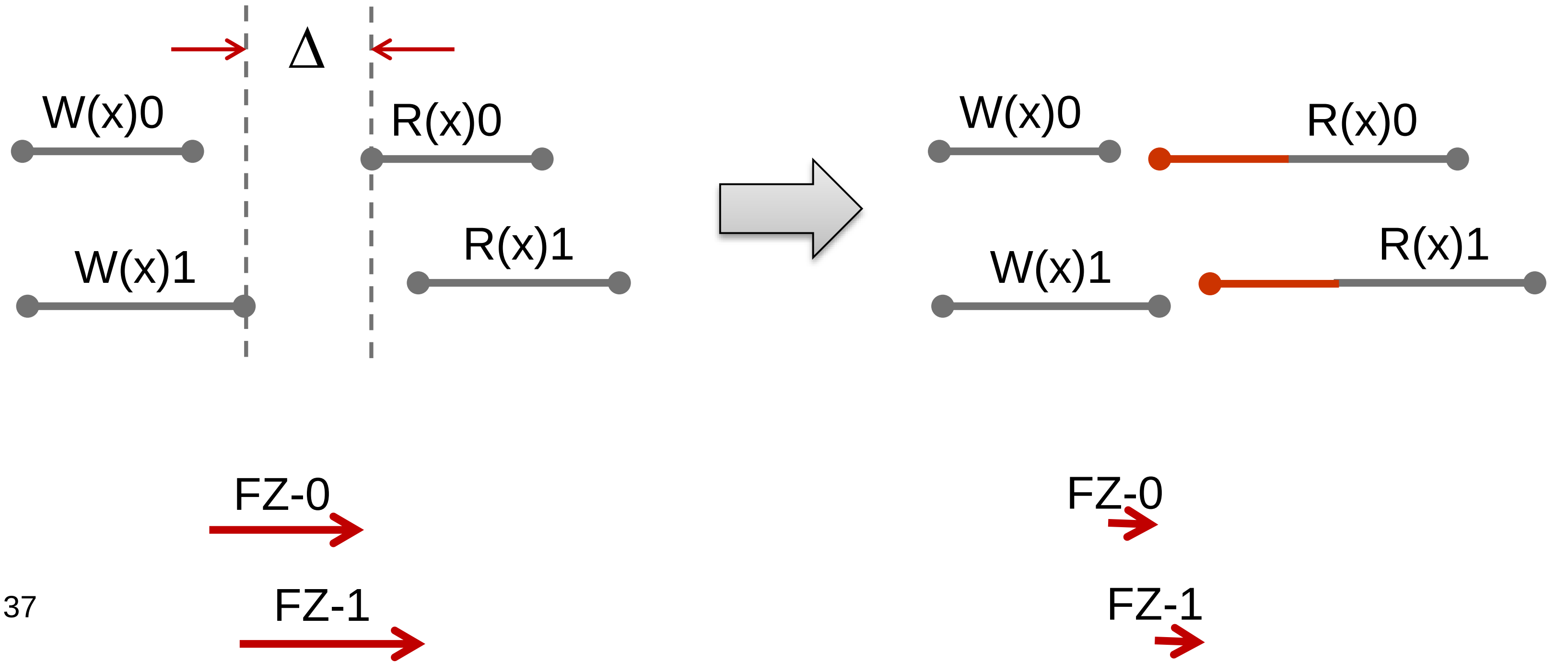
Lee and Welch (2004)

QUANTIFYING STALENESS

- Golab, Li, and Shah (2011)
- techniques for quantifying both the severity and frequency of linearizability anomalies
- builds on Gibbons and Korach (1997)
- anomalies counted at the granularity of "clusters" (subsets of operations applied to one object that access the same value)

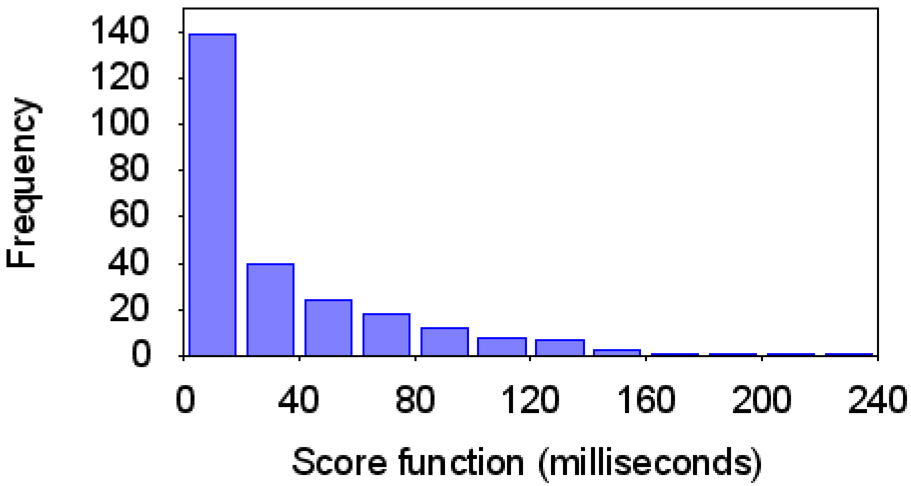
QUANTIFYING STALENESS: SEVERITY

The linearizability anomalies in a history have maximum severity at most Δ time units if decreasing the start time of every Read operation by Δ makes the history linearizable.



QUANTIFYING STALENESS: SEVERITY

Severity is quantified by a score function $F_x(v, w)$ that defines how far the start times of reads on object *x* must be shifted to resolve any conflict between the zone for *v* and the zone for *w*.



Rahman, Golab, AuYoung, Keeton, Wylie (2012)

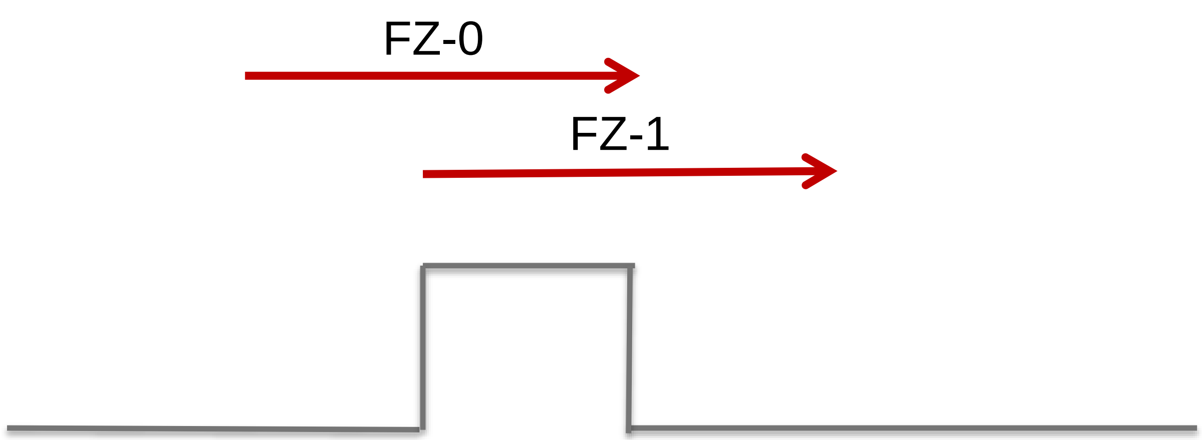
QUANTIFYING STALENESS: FREQUENCY

Frequency is quantified as the proportion of values that participate in linearizability violations for object *x*:

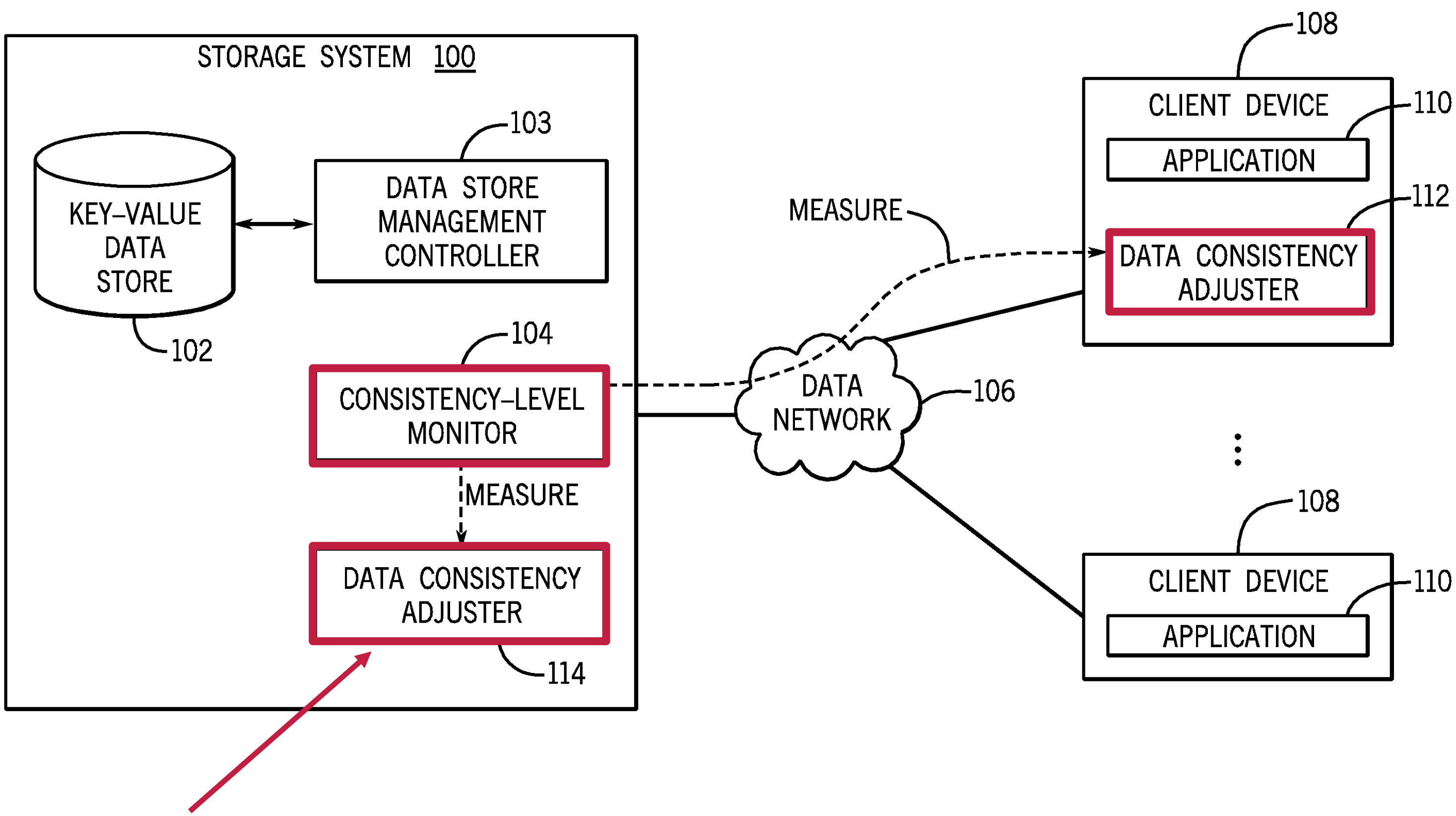
$$\frac{\text{number of values } v \text{ for which } F_x(v, \ \cdot\ ) > 0}{\text{total number of distinct values accessed}}$$

PROVIDING A MEASURE OF INSTANTANEOUS CONSISTENCY

- Golab and Wylie (2012)
- builds on Golab, Li, and Shah (2011)
- instantaneous staleness at time *t* with respect to object *x:* maximum of the score function $F_x(v, w)$ for any pair of values *v* and *w* whose zones overlap at time *t*.



PROVIDING A MEASURE OF INSTANTANEOUS CONSISTENCY



artificial delay

US Patent 9292566

PROVIDING A MEASURE OF INSTANTANEOUS CONSISTENCY

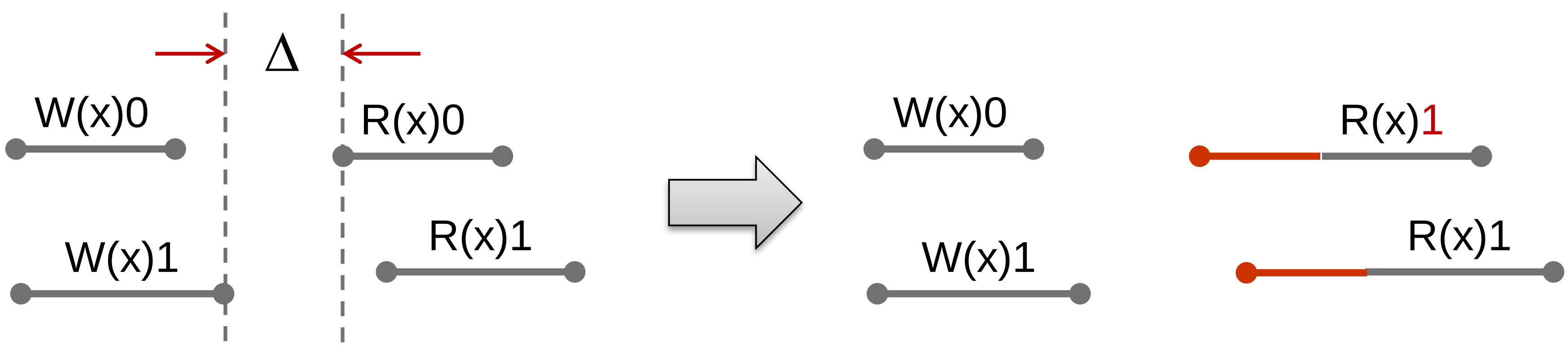
Example of service level agreement (SLA):

X% of the time
the instantaneous staleness is $\leq$ Y ms

(+ bound on latency, for example 95%-ile)

PROVIDING A MEASURE OF INSTANTANEOUS CONSISTENCY

Tuning technique: artificial delay

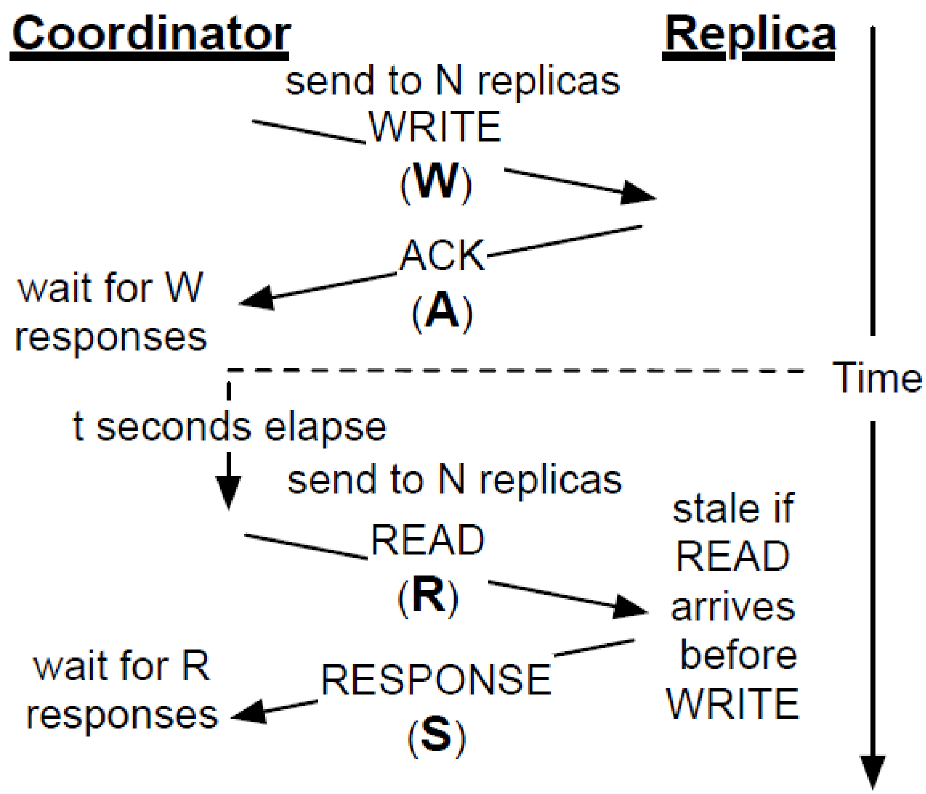


PROBABILISTICALLY BOUNDED STALENESS (PBS)

- Bailis, Venkataraman, Franklin, Hellerstein, and Stoica (2012)
- mathematical model of weak consistency based on probabilistic quorums
- *t*-visibility: probability that a Read invoked *t* time units after the completion of a Write returns the value assigned by that Write
- concurrent reading and writing outside the scope of the model

PROBABILISTICALLY BOUNDED STALENESS (PBS)

Write-Ack-Read-Response (WARS) model:



Bailis et al. (2012)

PROBABILISTICALLY BOUNDED STALENESS (PBS)

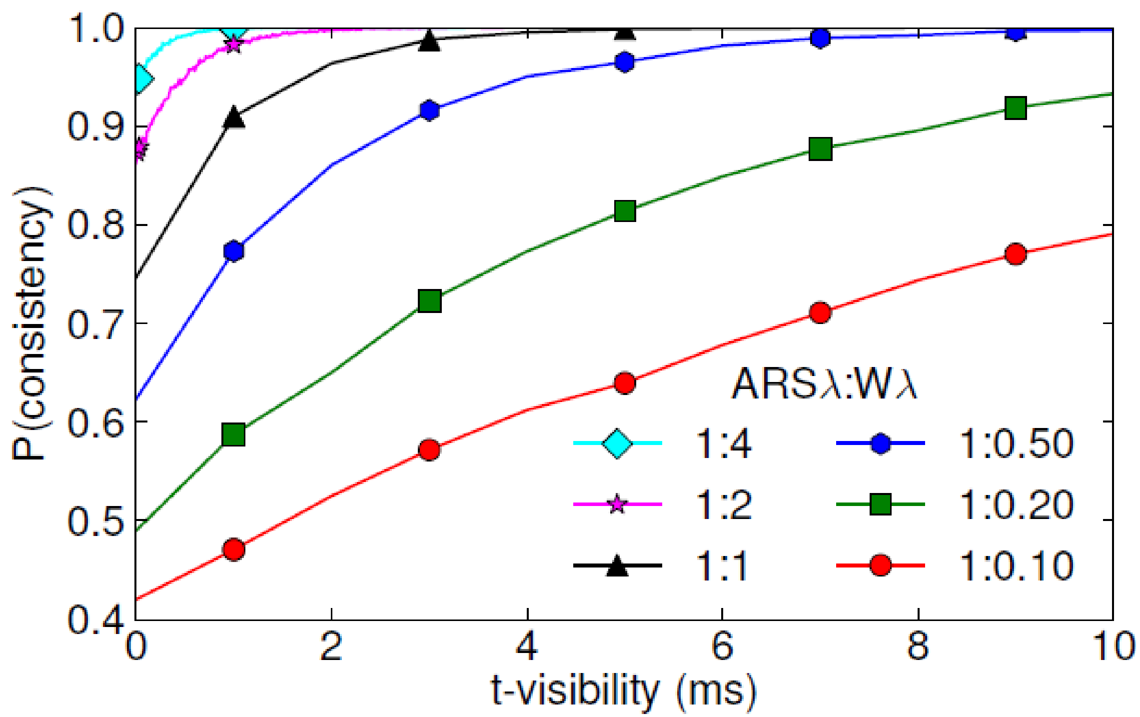


Figure 4: t-visibility with exponential latency distributions for `W` and `A`=`R`=`S`. Mean latency is $1/\lambda$. N=3, R=W=1.

Bailis et al. (2012)

PROBABILISTICALLY BOUNDED STALENESS (PBS)

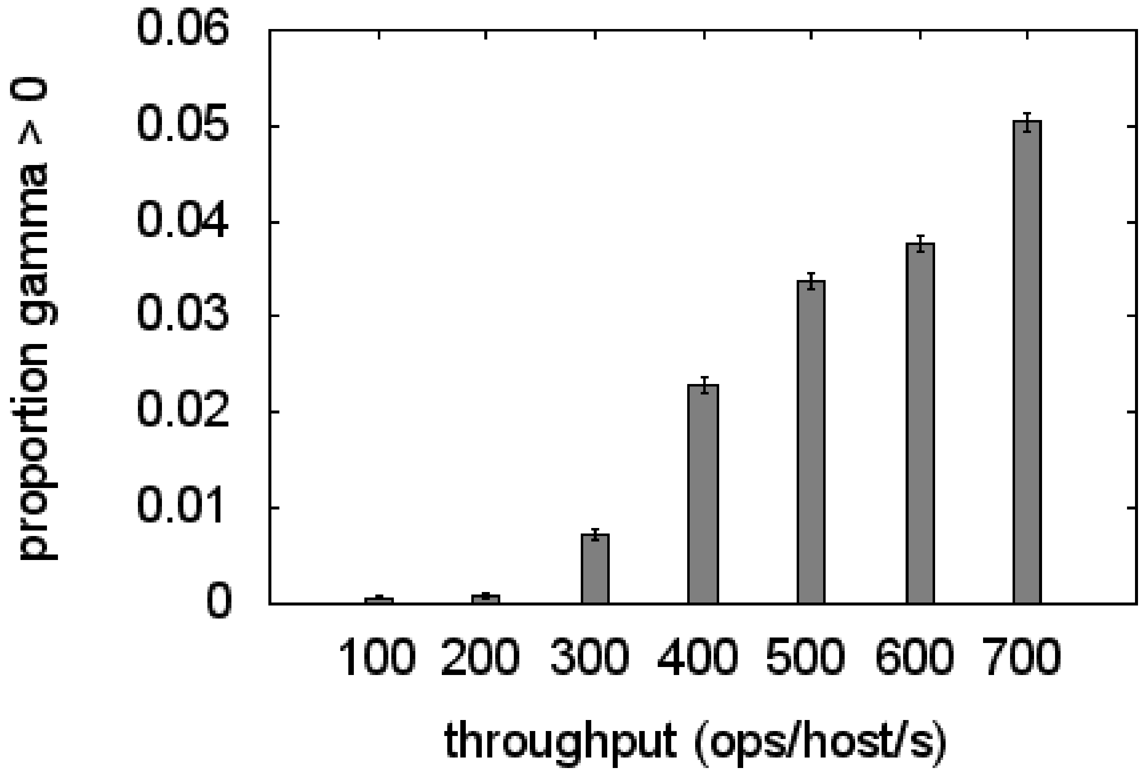
<live demo>

http://pbs.cs.berkeley.edu/

BENCHMARKING EVENTUAL CONSISTENCY

- Golab, Rahman, AuYoung, Keeton, Gupta (2014)
- evaluated effect of system and workload parameters on staleness measurements
- staleness quantified using a score function (gamma) similar to the one introduced by Golab, Li, and Shah (2011)

BENCHMARKING EVENTUAL CONSISTENCY



theoretical worst-case = 1.00

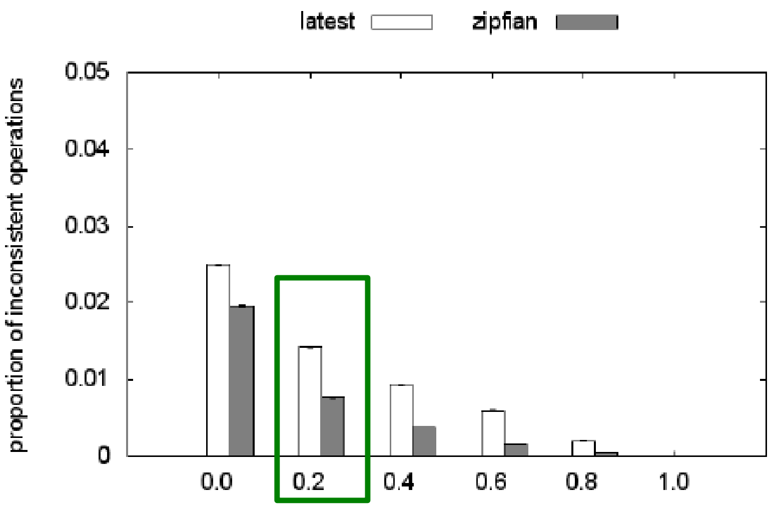
(write at one server, read immediately at another)

Golab et al. (2014)

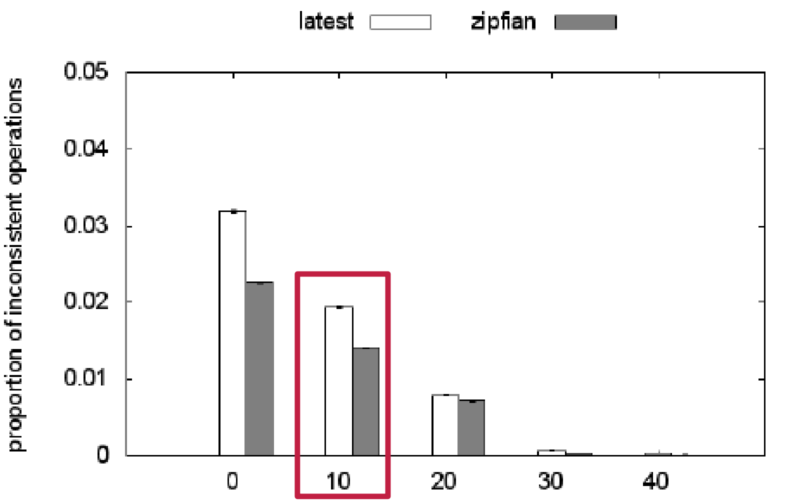
FINE-TUNING THE CONSISTENCY-LATENCY TRADE-OFF

- McKenzie, Fan, and Golab (2015)
- technique #1: artificial delay (AD)
- technique #2: continuous partial quorums (CPQ)
- observation: AD works best when network delay is constant, CPQ better when distribution of network delays has long tail

FINE-TUNING THE CONSISTENCY-LATENCY TRADE-OFF



(a) consistency vs. probability of quorum level

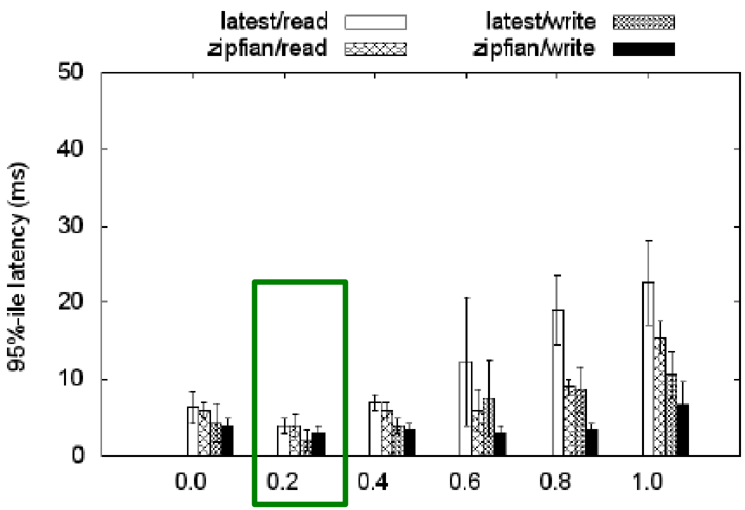


(a) consistency vs. artificial delay (ms)

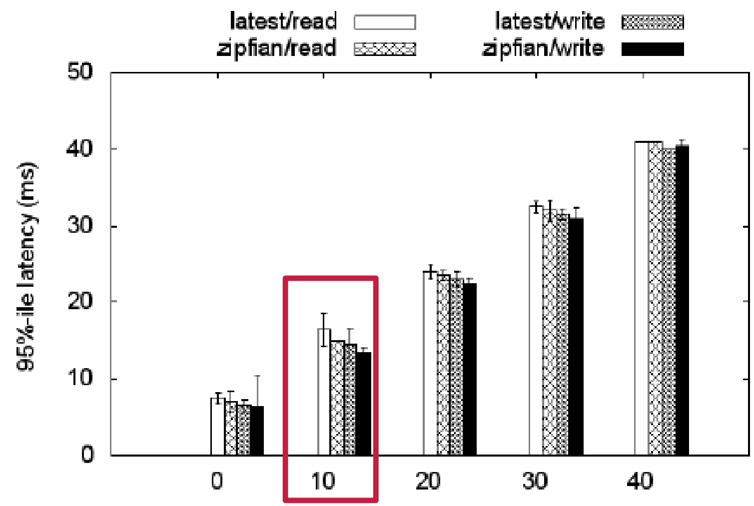
(in)consistency plots – CPQ (left) and AD (right)

McKenzie, Fan, Golab (2015)

FINE-TUNING THE CONSISTENCY-LATENCY TRADE-OFF



(b) 95th %-ile latency vs. probability of quorum level



(b) 95th %-ile latency vs. artificial delay (ms)

latency plots – CPQ (left) and AD (right)

McKenzie, Fan, Golab (2015)

WATCA: THE WATERLOO CONSISTENCY ANALYZER

- Fan, Chatterjee, Golab (2016)
- real-time consistency metric computation and visualization
- built-in support for CPQ and AD
- open-source software: https://github.com/wgolab/WatCA

PROBABILISTIC CAP (PCAP)

- Rahman, Tseng, Nguyen, Gupta, Vaidya (2016)
- mathematical model of consistency-latency trade-off + adaptive tuning framework
- staleness quantified similarly to Golab, Li, and Shah (2011) under the assumption that a Write takes effect at its invocation (model ignores write latency)
- (t_c, p_{ic})-consistency: fraction of Reads returning values $>t_c$ time units stale is at most p_{ic}

PROBABILISTIC CAP (PCAP)

Impossibility result for consistency-latency trade-off:

- t_c: upper bound on staleness
- t_a: upper bound on operation latency
- t_p: upper bound on message delay

Theorem 1: $t_c + t_a \geq t_p$

PROBABILISTIC CAP (PCAP)

If $t_c = 0$ then Theorem 1 resembles the lower bound of Lipton and Sandberg (1988):

Any implementation of a sequentially consistent read-write register must satisfy $|r|+|w| \geq d$, where $|r|$ is the latency of a Read, $|w|$ is the latency of a Write, and d is the network delay.

PROBABILISTIC CAP (PCAP)

Probabilistic variation:

- p_{ic}: proportion of reads with staleness > t_c
- p_{ua}: proportion of operations with latency > t_a
- p_{α}: proportion of messages with delay > t_p

Theorem 2: if $t_c + t_a < t_p$ then $p_{ic} + p_{ua} \geq p_{\alpha}$.

PROBABILISTIC CAP (PCAP)

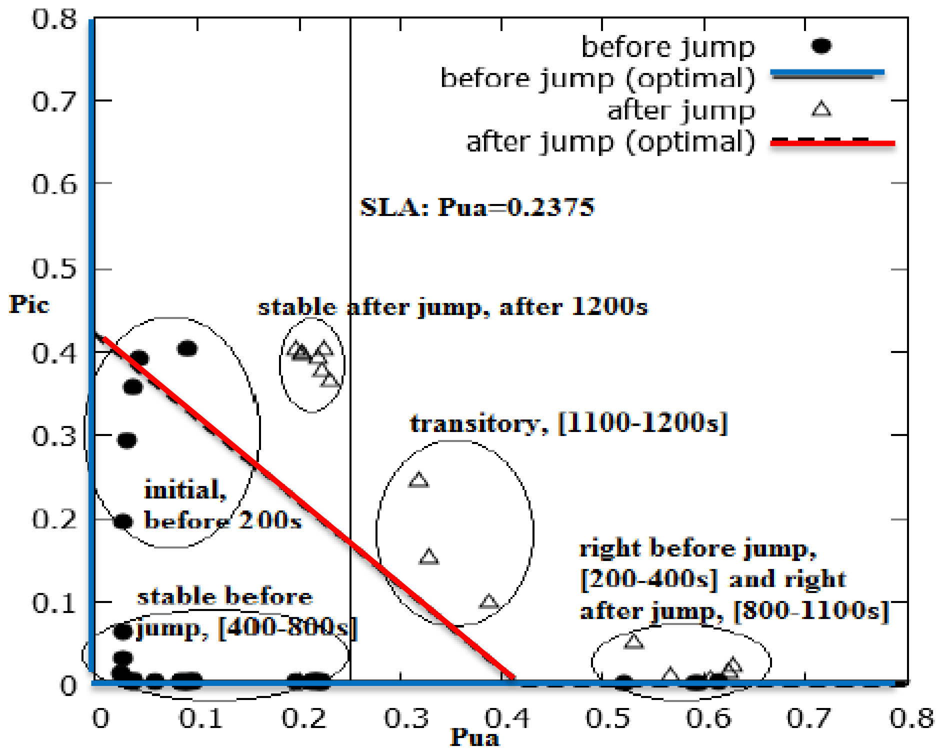


Figure 15: *Latency SLA with PCAP Cassandra under Sharp Network Jump: Consistency-Latency Scatter plot.*

Rahman, Tseng, Nguyen, Gupta, Vaidya (2016)

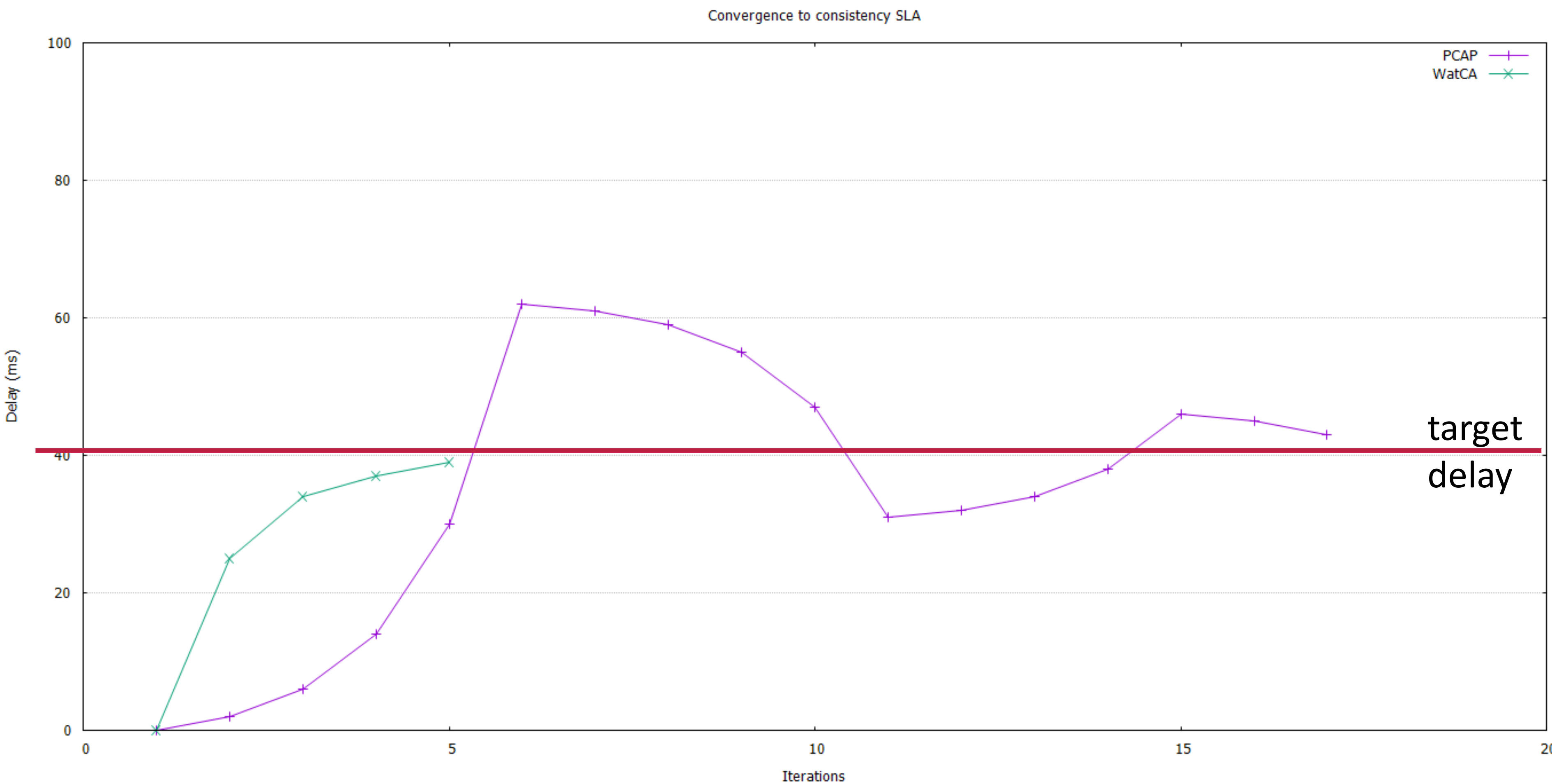
Ongoing Work at Waterloo

MATHEMATICAL MODEL OF EVENTUAL CONSISTENCY

Prior work does not answer the question posed earlier:

- analysis of probabilistic quorums does not account for eventual consistency
- PBS focuses on a single Write/Read pair
- PCAP describes worst-case behavior

IMPROVED ADAPTIVE CONSISTENCY-LATENCY TUNING



Experimental analysis by Shankha Chatterjee (MASc candidate)

Preguntas y Respuestas (Q&A)