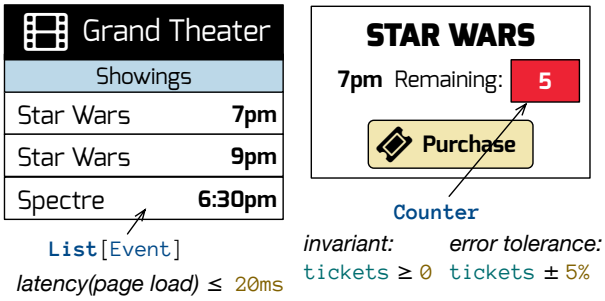
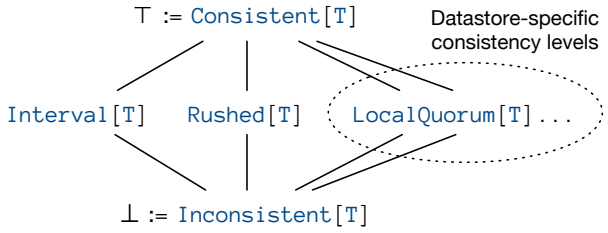




T

×

Counter

×

SetTSet[T] with Consistency(Strong)SetSet[T] with LatencyB

LatencyBound(x)
ErrorTolerance(x%)sizeSet

Setsizecontains
ErrorTolerance(5%)List

TInconsistent[T]Consistent[T]T≺

$$\frac{\tau\tau'}{\tau'[T] \prec \tau[T]}$$

BoundedCounter

Consistent[T]T
Inconsistent[T]Inconsistent[T]Consistent(x)Consistent[T]

$$\frac{\Gamma \vdash e_1 : \tau[T] T \prec \tau[T]}{\Gamma \vdash \text{Consistent}(e_1) : T}$$

Set[T]

Inconsistent[T]

Rushed[T]LatencyBound(x)Rushed[T]LatencyBoundRushed[T]
Rushed[T]
  Rushed[T]Consistent[T]LocalQuorum[T]Inconsistent[T]

```
set.contains() match {
  case Consistent(x) => print(x)
  case LocalQuorum(x) => print(x + ", locally")
  case Inconsistent(_) => print("unknown")
}
```

Inconsistent(_)

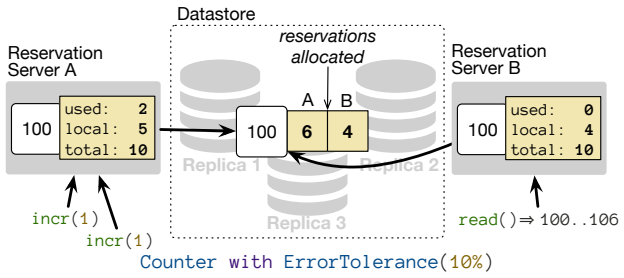
Interval[T]ErrorTolerance(x%)Interval[T]
  Interval[T]Setsizeaddsizesize$[95, 105]$$[100, 107]$add

  Interval[Int]

$W + R > N(N+1)/2$QUORUMNALLONELOCAL_QUORUM

QUORUMQUORUM

  Counter
  ALL
$[100, 106]$



  Setaddremovesize

```
Interval(min = v - removePool.delta()
         max = v + addPool.delta())
```

vdelta

```
delta(): pool.total - (pool.local - pool.used)
```

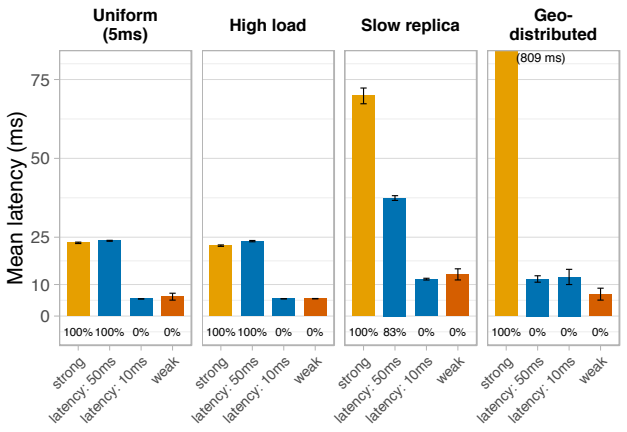
LatencyBound

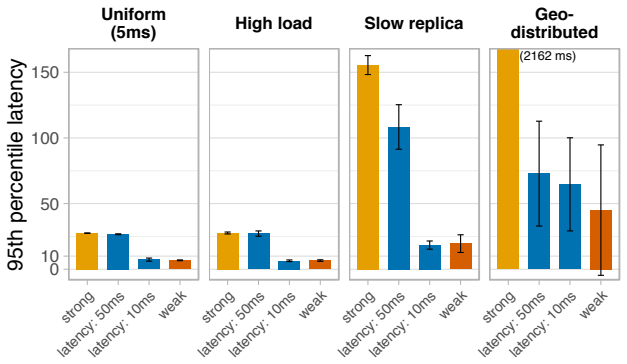
  Counter
  Setaddremovecontainssizesize
  BoundedCounter
  UUIDPoolBoundedCounter
  List

  tc netemnetem

incr(1)read

```
trait LatencyBound {
  // execute readOp with strongest consistency possible
  // within the latency bound
  def rush[T](bound: Duration,
              readOp: ConsistencyLevel => T): Rushed[T]
}

/* Generic reservaton pool, conceptually one per
 * ADT instance. `max` recomputed as needed
 * (e.g. for percent error) */
abstract class ReservationPool(max: () => Int) {
  def take(n: Int): Boolean // try to take tokens
  def sync(): Unit      // sync to regain used tokens
  def delta(): Int      // # possible ops outstanding
}

/* Counter with ErrorBound (simplified) */
class Counter(key: UUID) with ErrorBound {
  def error: Float // error bound
  def computeMax(): Int = (cass.read(key) * error).toInt

  val incrPool = ReservationPool(computeMax)
  val decrPool = ReservationPool(computeMax)

  def value(): Interval[Int] = {
    val v = cass.read(key)
    Interval(v - decrPool.delta,
             v + incrPool.delta)
  }

  def incr(n: Int): Unit = {
    waitFor(incrPool.take(n)) {
      cass.incr(key, n)
    }
  }
}
```





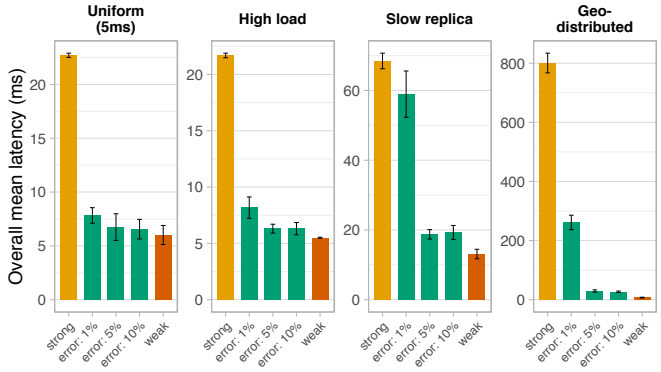
QUORUM
  weakALLQUORUM

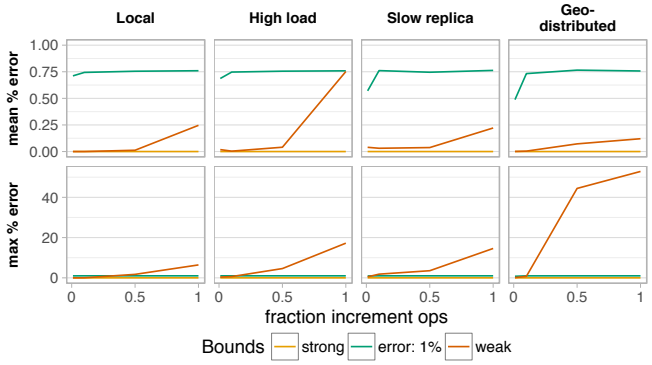
  ALL

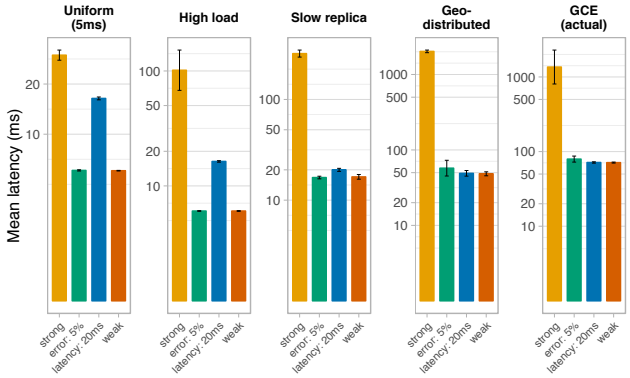
us-eastus-centraleurope-westasia-east

  browse
  viewEvent
  purchase
  addEvent

ListUUIDPoolBoundedCounter
  BoundedCounter



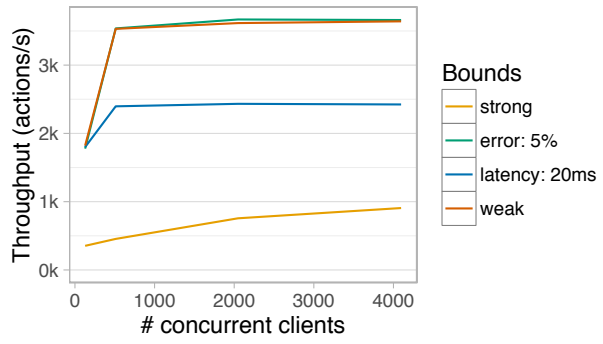




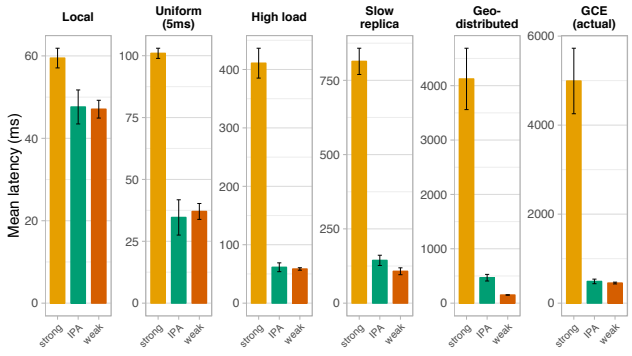
×Pool

```
User(id: UserID,
  name: String,
  followers: Set[UserID] with LatencyBound(20 ms),
  timeline: List[TweetID] with LatencyBound(20 ms)
)

Tweet(id: TweetID,
  user: UserID,
  text: String,
  posted: DateTime,
  retweets: Set[UserID] with Size(ErrorTolerance(5%))
)
```

size



tweet



×

Interval timelinetweetretweetfollownewUser

viewEventbrowsepurchaseaddEvent×purchaseBoundedCounter××viewEventbrowse

PoolCounter

SetSetListListSet

Rushed[T]

http://hyperdex.org/

https://www.usenix.org/conference/osdi12/technical-sessions/presentation/li

http://glinden.blogspot.com/2006/12/slides-from-my-talk-at-stanford.html

https://www.usenix.org/conference/nsdi14/technical-sessions/presentation/liu_jed

http://www.wired.com/2015/11/how-instagram-solved-its-justin-bieber-problem/

http://www.buzzfeed.com/daozers/what-its-like-to-work-on-buzzfeeds-tech-team-during-record-t

http://outworkers.github.io/phantom/

http://fusionticket.org

https://aws.amazon.com/ec2/

http://cassandra.apache.org/

http://www.reuters.com/article/2014/03/03/us-oscars-selfie-idUSBREA220C320140303

http://docs.basho.com/riak/latest/

<>

http://redis.io/

http://redis.io/topics/twitter-clone

http://www.forbes.com/sites/hayleycuccinello/2015/10/20/star-wars-presales-crash-ticketing-sites-sets-record-for-fandango/

http://docs.datastax.com/en/cassandra/3.x/cassandra/dml/dmlAboutDataConsistency.html

http://www.linuxfoundation.org/collaborate/workgroups/networking/netem

https://twitter.github.io/finagle/

https://www.usenix.org/conference/osdi14/technical-sessions/presentation/xie

https://www.docker.com/

http://radar.oreilly.com/2009/06/bing-and-google-agree-slow-pag.html

https://cloud.google.com/compute/