Eventual Consistency In the real world

or
Why you already know Eventual Consistency

/usr/bin/whoami

Chris Molozian

Client Services Engineer, Basho EMEA

Basho Technologies

cmolozian@basho.com

Basho Technologies

- Founded in 2008 by engineers and executives from Akamai Technologies, Inc.
- Design large scale distributed systems
- Develop Riak, open-source distributed database
- Specialize in storing critical information, with data integrity
- Offices in US, Europe (London) and Japan


riak

What is Riak?

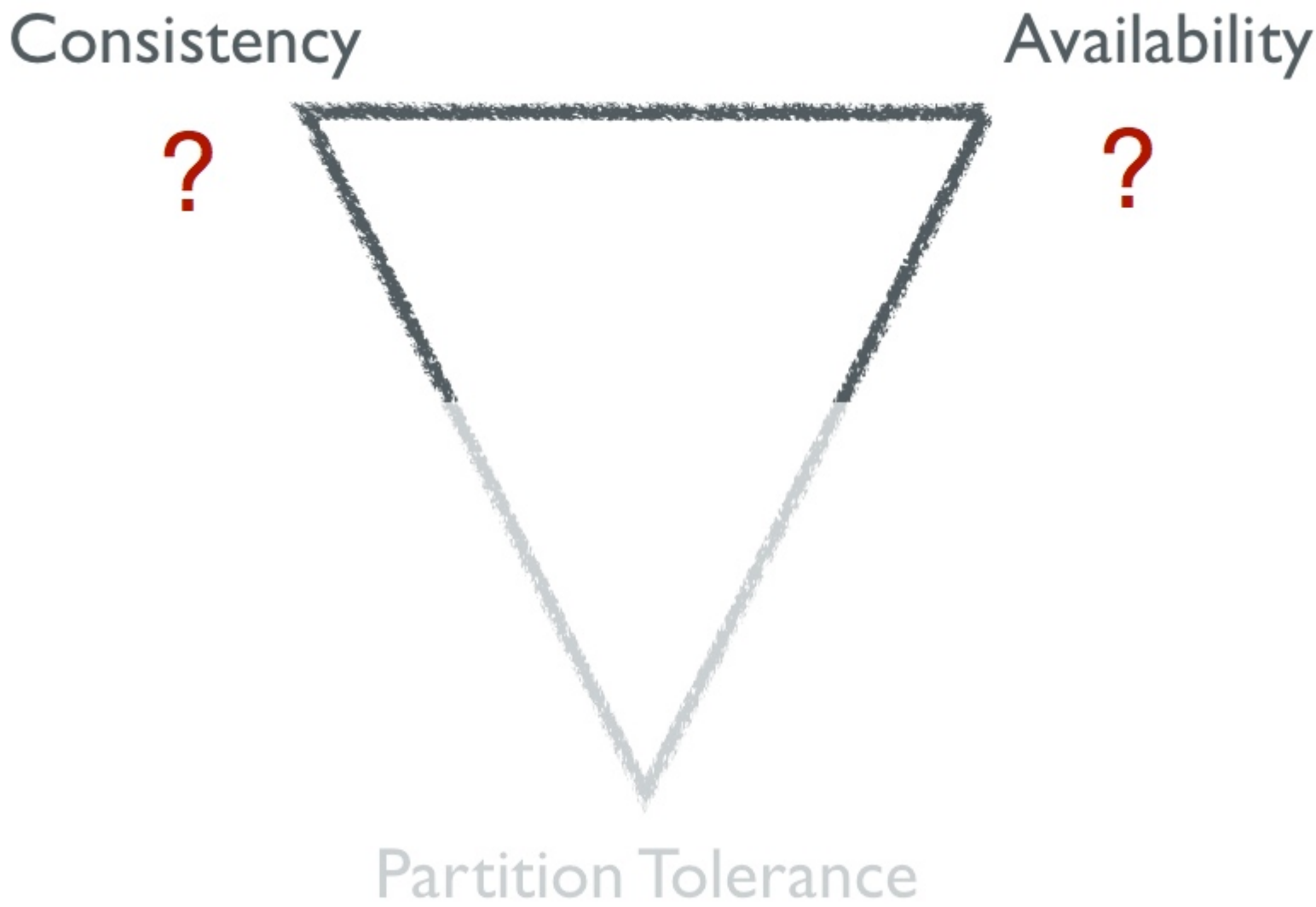
- Key/Value Store + Extras
- Distributed, horizontally scalable
- Fault-tolerant
- Highly-available
- Built for the Web
- Inspired by Amazon's Dynamo

CAP Theorem

- Brewer's Conjecture (2000)

  Symposium on Principals of Distributed Computing
- Formally proven in 2002

  Seth Gilbert and Nancy Lynch, MIT
- Impossible for a distributed system to guarantee:
  - Consistency
  - Availability
  - Partition Tolerance


Consistency
Availability
?
?
Partition Tolerance

Amazon Dynamo

- Amazon analyzed their visitors purchasing habits
- Determined that `High latency == Lost revenue`
- Researched low latency & high availability for their data
- Developed a new database model
- Released a research paper in 2007

What is Consistency?

... when we say ”data is consistent”
what do we mean?

Strong Consistency (SC)

> *Replicas update linearly in the same total order.*

- As application developers, Strong Consistency is what we're used to
- All ACID-compliant databases are Strongly Consistent
- Distributed + ACID = "Consensus"
  - Well known limitations...
  - Serialization bottlenecks.
  - Not tolerant beyond `n / 2` faults.

Eventual Consistency (EC)

> *Replicas update in the background and may not converge to the same total order.*

- Many NoSQL databases are Eventually Consistent
- Update is accepted by local node
- Local node propagates update to replica nodes
- No synchronization phase:
  - No synchronization phase
  - Eventually, all replicas are updated
  - Data can diverge, arbitrate or rollback?

Life is full of

tradeoffs

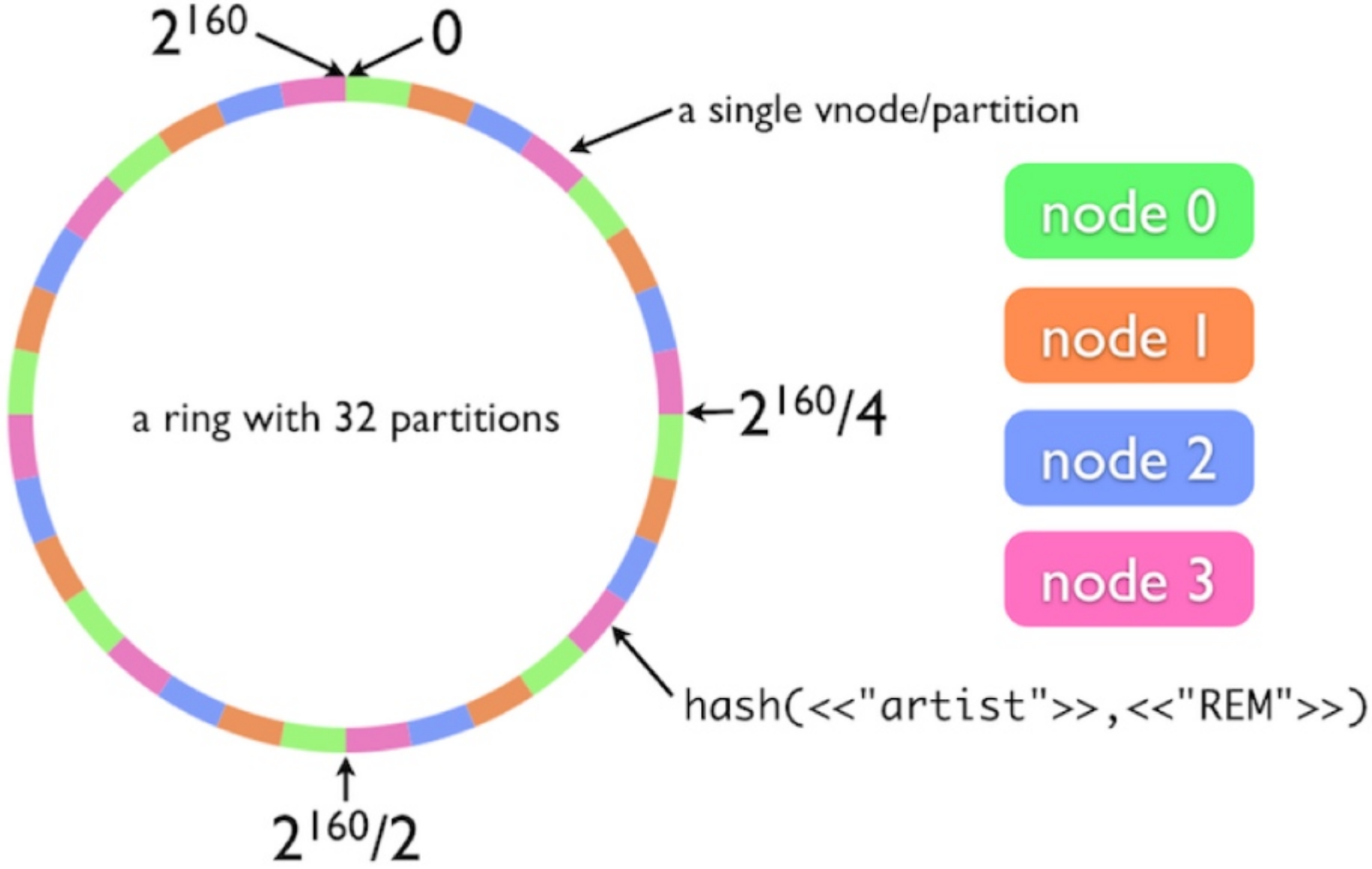
Consistency Tradeoffs

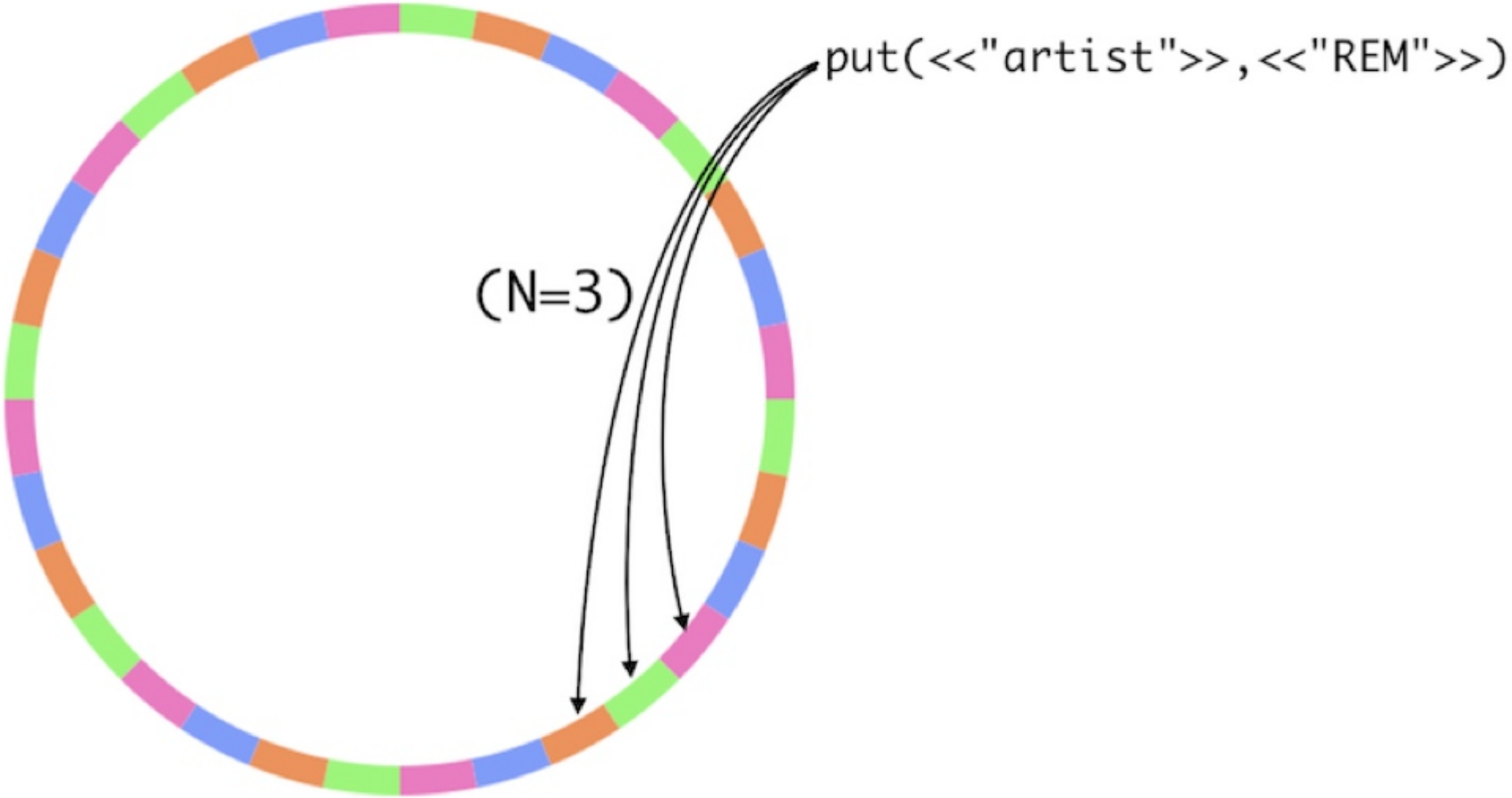
Strong Consistency is too slow in a distributed system

Eventual Consistency can introduce data conflicts

- Strong Eventual Consistency is the target
- What would this look like?
- Replicas that execute the same updates **in any order** have the same total order.

Back to Riak


2^{160}
0
a single vnode/partition
node 0
node 1
node 2
node 3
a ring with 32 partitions
$2^{160}/4$
hash(<<"artist">>,<<"REM">>)
$2^{160}/2$


put(<<"artist">>,<<"REM">>)
(N=3)

Riak's tools for Eventual Consistency

- Concurrenct actors modifying the same data (k/v pair) cause data divergence
- Riak tracks these occurrences
- Riak provides two solutions to manage this:
  - **Last Write Wins**

    Naive approach but works for some use cases (i.e. immutable data)
  - **Vector Clocks**

    Retain ”sibling” copies of data for merging.

Vector Clocks (tracking divergence)

- Every node has an ”actor” ID.
- Send ”last seen” vclock in every PUT or DELETE request.
- Auto-resolves stale versions.
- Lets you decide how to handle conflicts.

Siblings

- Siblings are created when:
- Simultaneous requests write to the same object ID
- Network partitions, ”split brain” in a cluster of Riak nodes
- Writes to an existing key without a vclock

How Riak Developers handle siblings

We don't ever do conflict resolution by picking a random sibling.

For an array property, we often take the union of all values in all siblings.

This works great for array properties that we only ever add to.

We often take the maximum sibling value or the minimum sibling value,

depending on the semantics of that attribute

Myron Marston, SEOMoz


bump
Scaling to
80 million users
and beyond!
Powered by
basho
CBS

How Riak Developers handle siblings

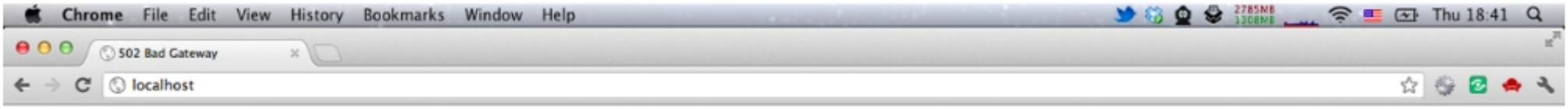
> *Storing a communication between two users[...]will be written once[...]but it can be updated multiple times. The updates are resolved as a time sorted list.*

> *For every photo (or other large data item) sent via Bump we back it up to S3, but keep a little metadata about the item.[...] Resolutions are simply a matter of doing a set union between these two values.*
>
> ***Will Moss, Bump***

Eventual

Availability


LOADING



502 Bad Gateway

---

nginx/0.8.54

In the real

world...


HSBC
HSBC


HSBC
PLUS FROM HSBC
MORE THAN A CHECKING ACCOUNT
STOP IN AND ASK US ABOUT
THE NEW MATH OF BANKING
800-975-HSBC
FEE NOTICE


JCB
JS220
nuttall


URBAN
GAP
GAP
CASH

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

quantitatively demonstrate why eventual consistency is "good enough" for many users

Questions?

Want to know more?

We will come and give a Riak tech talk at your organisation or group:

bit.ly/RiakTechTalk