Drop ACID and think about data

Bob Ippolito
Mochi Media, Inc.

PyCon 2009 - Chicago (actually Rosemont)

March 28, 2009

Author: Bob Ippolito

Date: March 2009

Venue: PyCon 2009

Bob's Perspective

Startup with lots of data:

- Cofounded Mochi Media in 2005
- MochiBot analytics platform (for Flash)
- MochiAds ad serving platform (for Flash games)
- Other cool services for game developers

Mochi Ad Sales

Hard Sell:

What's ACID?

A promise ring your DBMS wears:
Atomicity: *all or nothing*
Consistency: *no explosions*
Isolation: *no fights*
Durability: *no lying*

ACID Trips

Scalability and reliability:

- Downtime is unacceptable
- Reliable is $>=$ 2 nodes
- Scalable is ... more
- Networks make it hard
- Networks make it hard
- Networks make it hard

What can I have?

CAP theorem says pick two:

- Consistency
- Availability
- Partition tolerance

Turn up the BASE

Write smarter applications:

- Basically Available
- Soft state
- Eventually consistent

BASE jumping

Everyone else is doing it:

- Google
- Amazon
- eBay
- Yahoo!
- Facebook
- ...

BigTable

Google:

- Paxos (Chubby)
- Single-master
- Distributed tablets via GFS
- Row/Column db hybrid
- Compression (BMDiff, Zippy)
- Versioned (Row, Column, Timestamp)
- Bloom filters

BigTable Pros

Pros:

- Compression = Awesome
- Clients are probably simple
- Integrates with map/reduce

BigTable Cons

Cons:

- Proprietary to Google
- Single-master

BigTable Diagram

Single-master:

Dynamo

Amazon:

- Key/Value store
- Consistent hashing
- Vector clocks
- Read repair

Dynamo Pros

Pros:

- No master
- Highly available for write
- Knobs to make it fast to read
- “Simple” (lots of half-baked clones!)

Dynamo Cons

Cons:

- Proprietary to Amazon
- Clients need to be smart
- No compression
- Not suitable for column-like workloads
- Just a Key/Value store

Dynamo Diagram

Smart client:

Cassandra

Facebook -> Apache:

- Open source!
- No master like Dynamo
- Storage model more like BigTable

Cassandra Pros

Pros:

- OPEN SOURCE
- Incrementally scalable
- Minimal administration

Cassandra Cons

Cons:

- Not polished
- No compression yet

Cassandra Diagram

Soul Calibur:

Distributed Musings

New Hotness:

- Distributed databases are the new web framework
- ... except none of them are awesome yet
- I don't think we need another half-baked Dynamo clone

Key-Value Stores

Simple and Fast:

- Similar to a Python dict
- Keys usually bytes, probably limited
- Values usually bytes, often have fewer limits
- Extremely fast, simple

Memcached

Key/Value store as cache:

- No persistence
- RAM only
- Throws data away (on purpose)
- Lightning fast
- “Everyone” uses it

Caching Immutable Data

If only data never changed:

- Immutable is easy, do that

Caching Mutable Data

Invalidation sucks:

- Mutable is hard
- Failed transactions?
- Concurrent writers?
- Dependent cache keys?
- You will get it wrong and it will be hard to debug

Tokyo Cabinet/Tyrant

Not your mom's BerkeleyDB:

- Disk persistent
- Very performant
- Actively developed
- Similar replication strategy to MySQL

Redis

Still very new:

- Not just a Key/Value store
- Matching on key spaces
- Values can be bytes, lists or sets
- Requires full store in RAM
- Might be a nice cache server?

Document Databases

Schema-free:

- Very easy to use
- Document Versioning
- Great for storing documents

CouchDB

Document DB Poster Child:

- Apache project
- Asynchronous replication
- JSON based
- Views materialized on demand (not indexes)
- Neat admin UI

MongoDB

C++'s revenge:

- Fast
- JSON and BSON (binary JSON-ish)
- Asynchronous replication with auto-sharding “soon”
- Index support
- Nested documents
- Advanced queries

Column Databases

Data Warehousing:

- Sequential reads are awesome
- Columns compress better than rows
- Doesn't waste I/O on uninteresting columns

MonetDB

Research project:

- Tried really hard to get it to work
- Crashes a lot and corrupts your data
- Do not waste your time

LucidDB

Sounds interesting:

- Java/C++ open source data warehouse
- No clustering
- No experience yet

Vertica

We paid for it:

- Commercial (based on C-Store)
- Clustered
- Would still prefer open source

Bitmap Indexes

Sequential Scans can be fast:

- 1-N bits per row of data
- Can apply logical operations across indexes
- Can be compressed (BBC, WAH)
- FastBit is a good implementation

Bitmap Index Uses

Big Queries:

- PostgreSQL 8.1+ in-memory for some queries
- Almost a requirement for column stores
- FastBit is a great implementation (WAH)

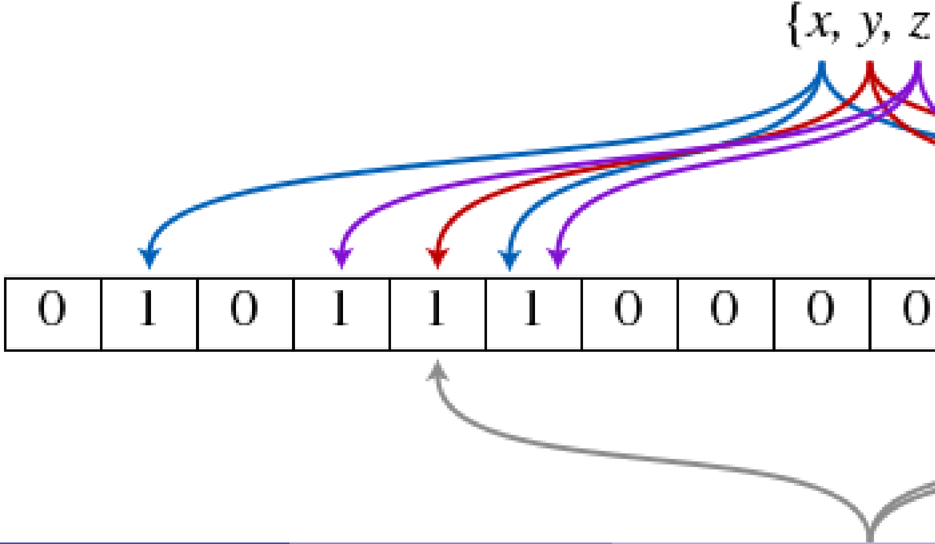
Bloom Filters are Neat

But our Princess is in another castle:

- Probabilistic data structure
- False positives at a known error
- Constant space
- I won't bore you with the math

Bloom Filter Diagram

Actually Relevant:



Bloom Filter Uses

Find stuff, maybe:

- Approximate counting of a large set (e.g. unique IPs from logs)
- Knowing that data is definitely NOT stored somewhere, e.g. remote cache
- Several variants (Counting Bloom Filter, Scalable Bloom Filter, ...)

Questions?

Open Space:

- Open Space TODAY @ 5pm, Lambert. See Jonathan Ellis