VirginiaTech

CS 4604: Introduction to Database Management Systems

B. Aditya Prakash

Lecture #13: NoSQL and MapReduce

Announcements

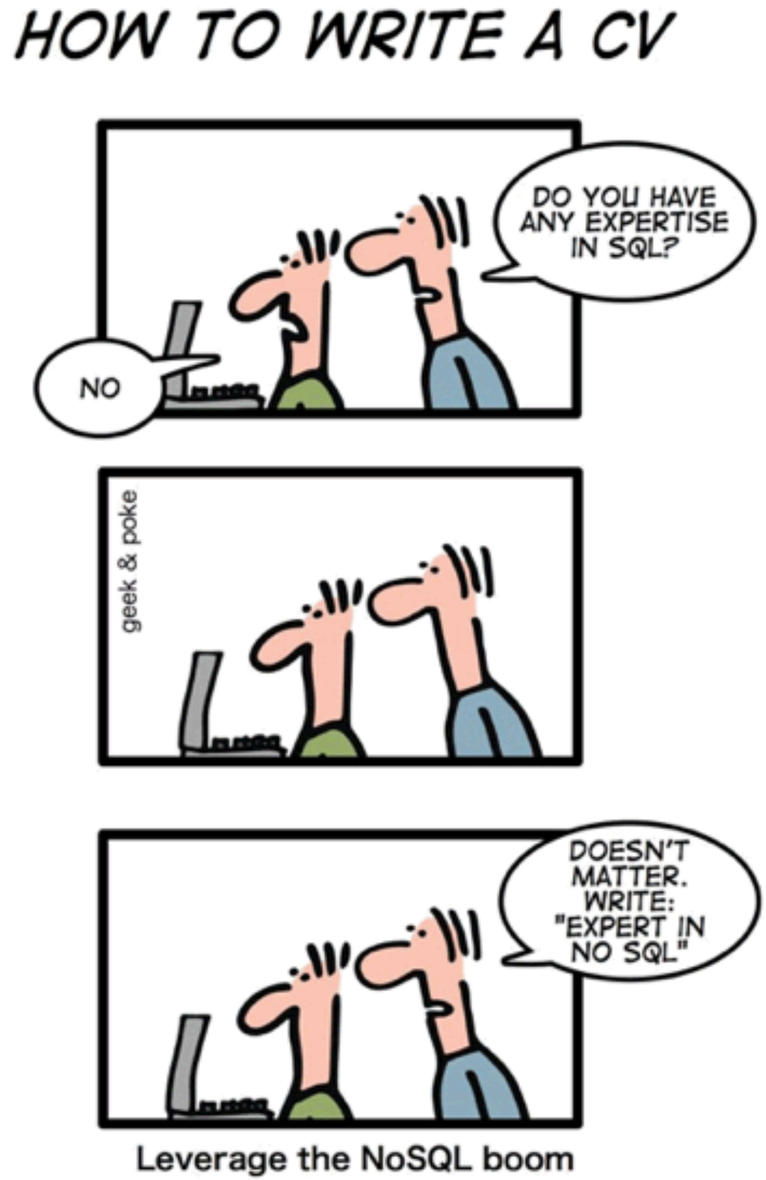
- HW4 is out
  - You have to use the PGSQL server
  - START EARLY!! We can not help if everyone first connects the night before it is due
- On Tuesday 03/25: I will be traveling
  - No office hours. Email me to setup another time, if needed.
  - Lecture will be on XML
    - Prof. Eli Tilevich will substitute
    - HW5 will be released, due on April 1
    - You have to use Amazon AWS, Read directions carefully
    - START EARLY!!
  - Project Assignment will also be released, but will be due April 8
    - START EARLY!! START EARLY!! START EARLY!! START EARLY!! START EARLY!! START EARLY!! START EARLY!! START EARLY!! START EARLY!!

(some slides from Xiao Yu)

NO SQL

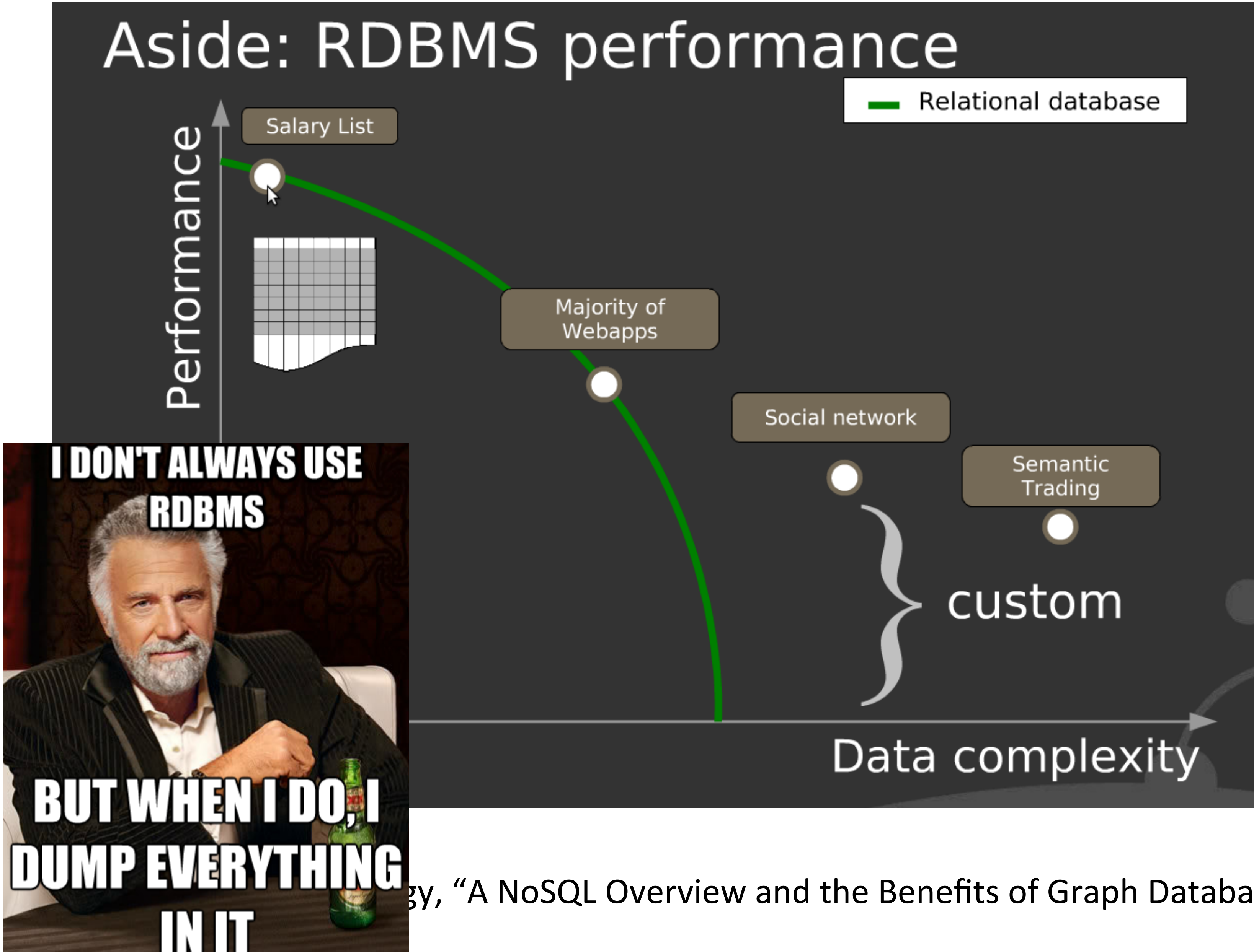
VirginiaTech

Why No SQL?



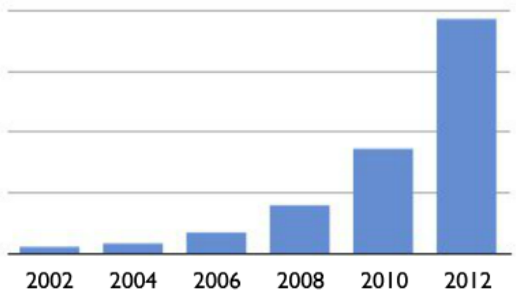
RDBMS

- The predominant choice in storing data
  - Not so true for data miners since we much in txt files.
- First formulated in 1969 by Codd
  - We are using RDBMS everywhere



gy, “A NoSQL Overview and the Benefits of Graph Databases"

When RDBMS met Web 2.0

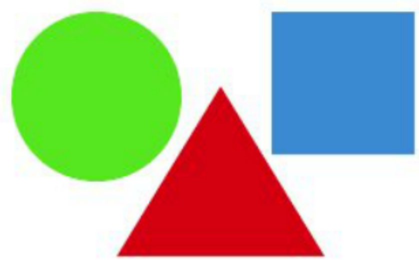


Big data

Connectivity

P2P Knowledge

Concurrency



Diversity

Cloud-Grid

Slide from Lorenzo Alberton, "NoSQL Databases: Why, what and when"

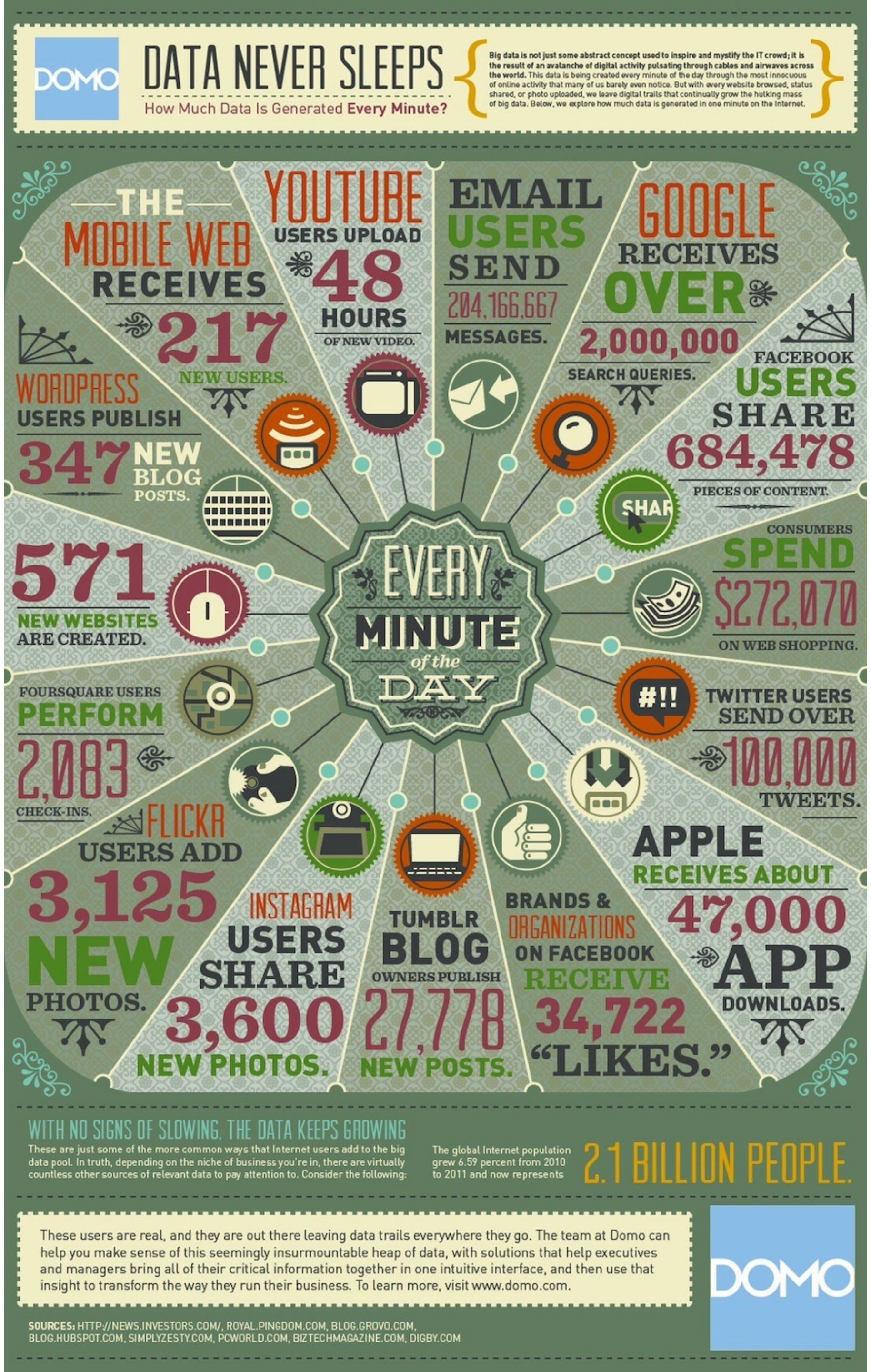
What to do if data is really large?

- Peta-bytes (exabytes, zettabytes …..)
- Google processed 24 PB of data per day (2009)
- FB adds 0.5 PB per day

VirginiaTech

BIG

data



What's Wrong with Relational DB?

- Nothing is wrong. You just need to use the right tool.
- Relational is hard to scale.
  - Easy to scale reads
  - Hard to scale writes

What's NoSQL?

- The misleading term "NoSQL" is short for "Not Only SQL".
- non-relational, schema-free, non-(quite)-ACID
  – More on ACID transactions later in class
- horizontally scalable, distributed, easy replication support
- simple API

Four (emerging) NoSQL Categories

- Key-value (K-V) stores
  – Based on Distributed Hash Tables/ Amazon's Dynamo paper *
  – Data model: (global) collection of K-V pairs
  – Example: Voldemort
- Column Families
  – BigTable clones **
  – Data model: big table, column families
  – Example: HBase, Cassandra, Hypertable

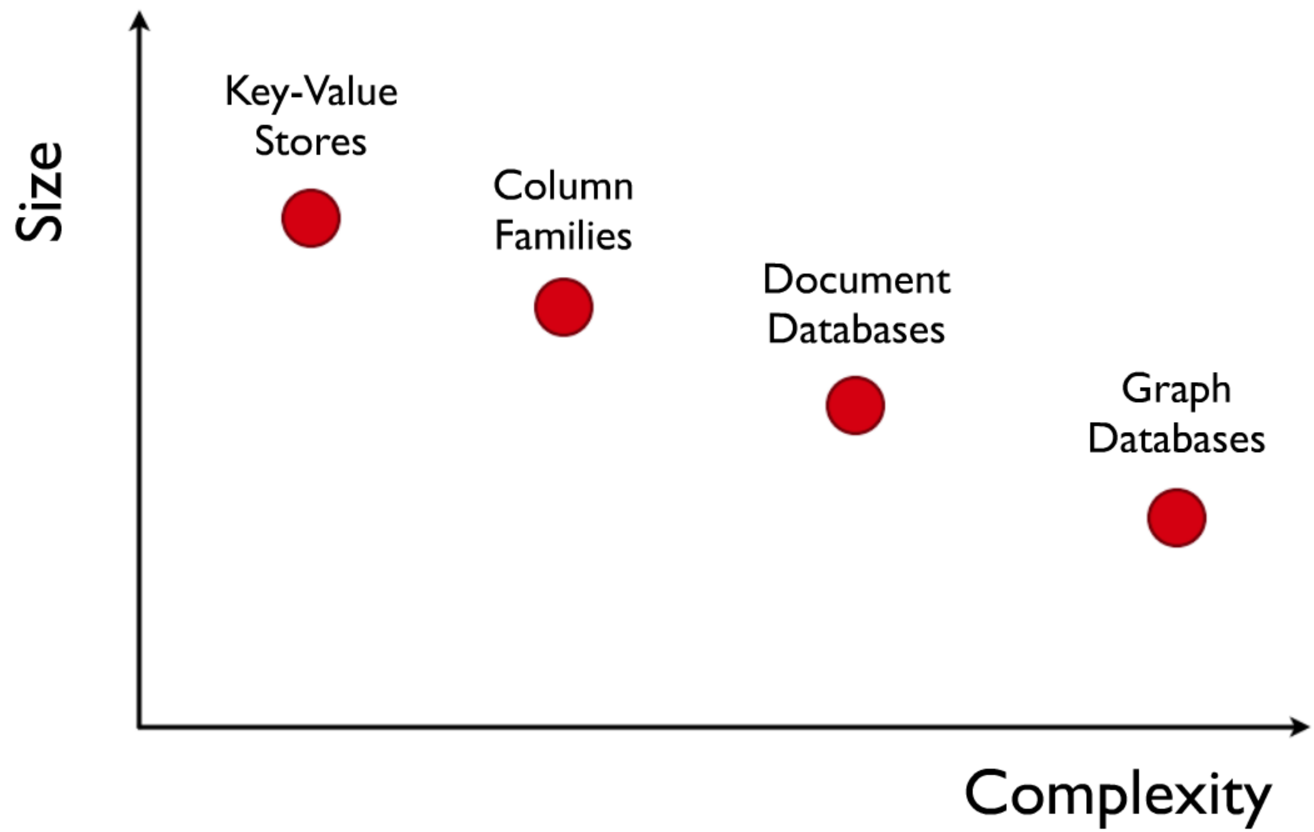
*G DeCandia et al, Dynamo: Amazon's Highly Available Key-value Store, SOSP 07

** F Chang et al, Bigtable: A Distributed Storage System for Structured Data, OSDI 06

Four (emerging) NoSQL Categories

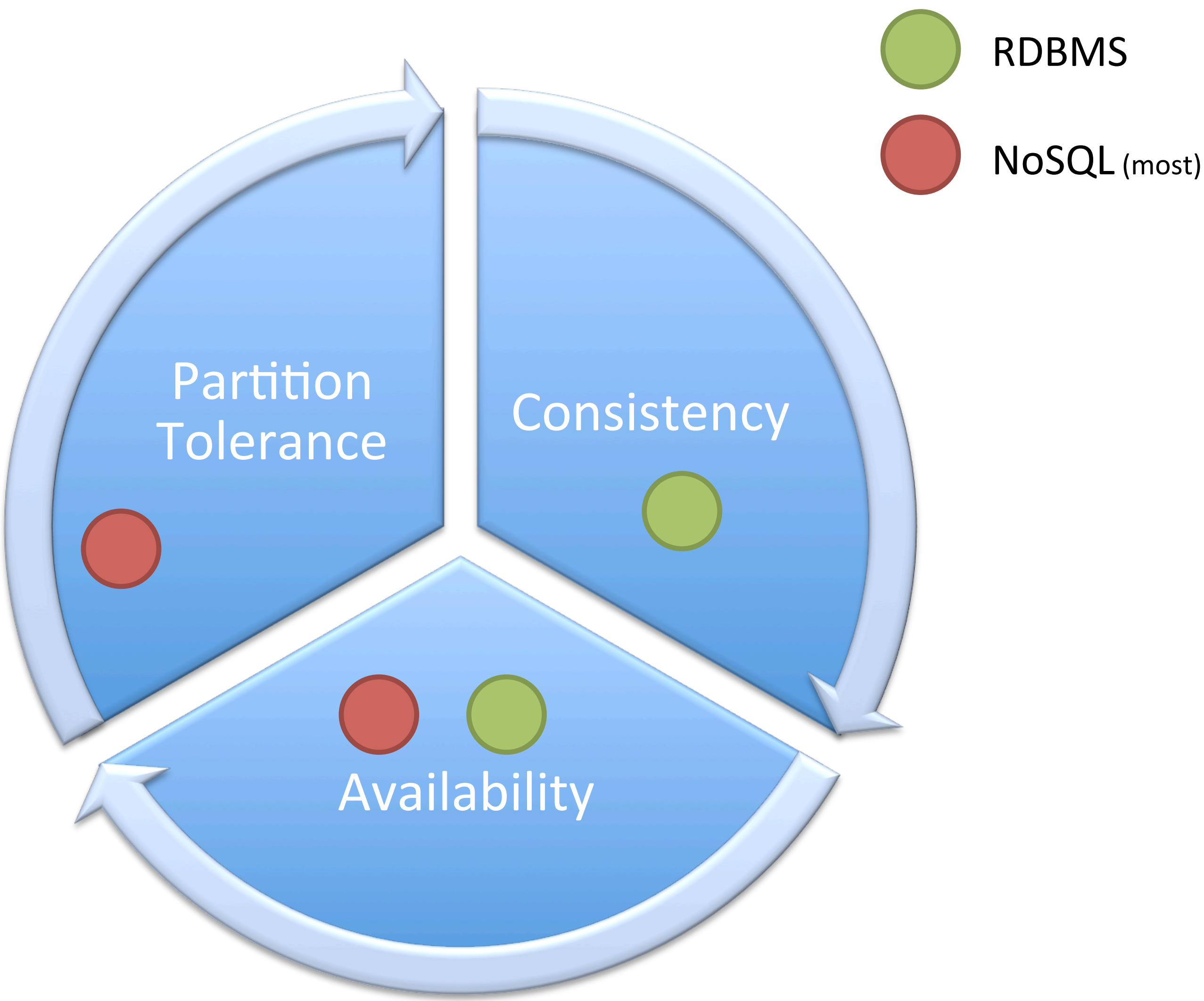
- Document databases
  - – Inspired by Lotus Notes
  - – Data model: collections of K-V Collections
  - – Example: CouchDB, MongoDB
- Graph databases
  - – Inspired by Euler & graph theory
  - – Data model: nodes, relations, K-V on both
  - – Example: AllegroGraph, VertexDB, Neo4j

Focus of Different Data Models



Slide from neo technology, “A NoSQL Overview and the Benefits of Graph Databases"

C-A-P "theorem"


RDBMS
NoSQL (most)
Partition Tolerance
Consistency
Availability

VirginiaTech

When to use NoSQL?

- Bigness
- Massive write performance
  - Twitter generates 7TB / per day (2010)
- Fast key-value access
- Flexible schema or data types
- Schema migration
- Write availability
  - Writes need to succeed no matter what (CAP, partitioning)
- Easier maintainability, administration and operations
- No single point of failure
- Generally available parallel computing
- Programmer ease of use
- Use the right data model for the right problem
- Avoid hitting the wall
- Distributed systems support
- Tunable CAP tradeoffs

from http://highscalability.com/

Key-Value Stores

id	hair_color	age	height
1923	Red	18	6’0”
3371	Blue	34	NA
…	…	…	…

Table in relational db

user1923_color	Red
user1923_age	18
user3371_color	Blue
user4344_color	Brackish
user1923_height	6' 0"
user3371_age	34

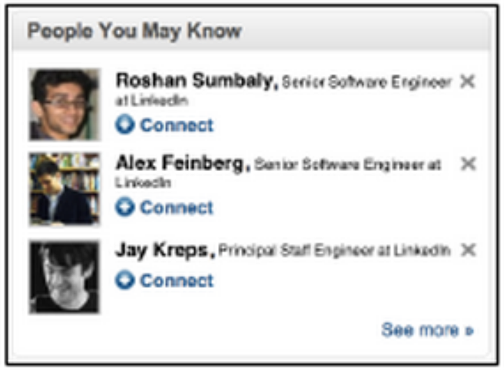
Store/Domain in Key-Value db

Find users whose age is above 18?
Find all attributes of user 1923?
Find users whose hair color is Red and age is 19?
(Join operation) Calculate average age of all grad students?

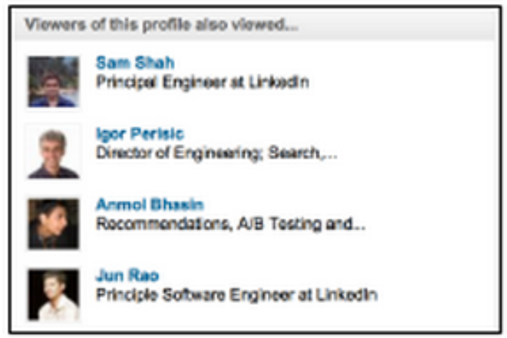
VirginiaTech

Voldemort in LinkedIn

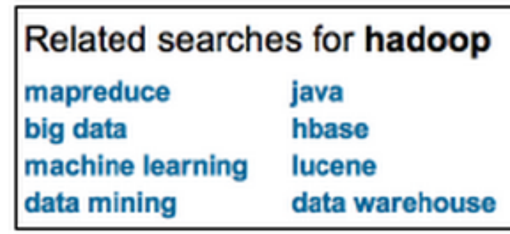
People You May Know



Viewers of this profile also viewed



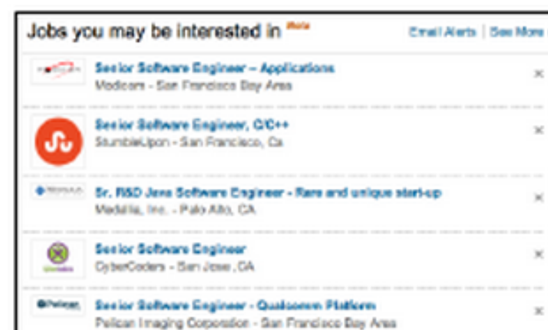
Related Searches



Events you may be interested in

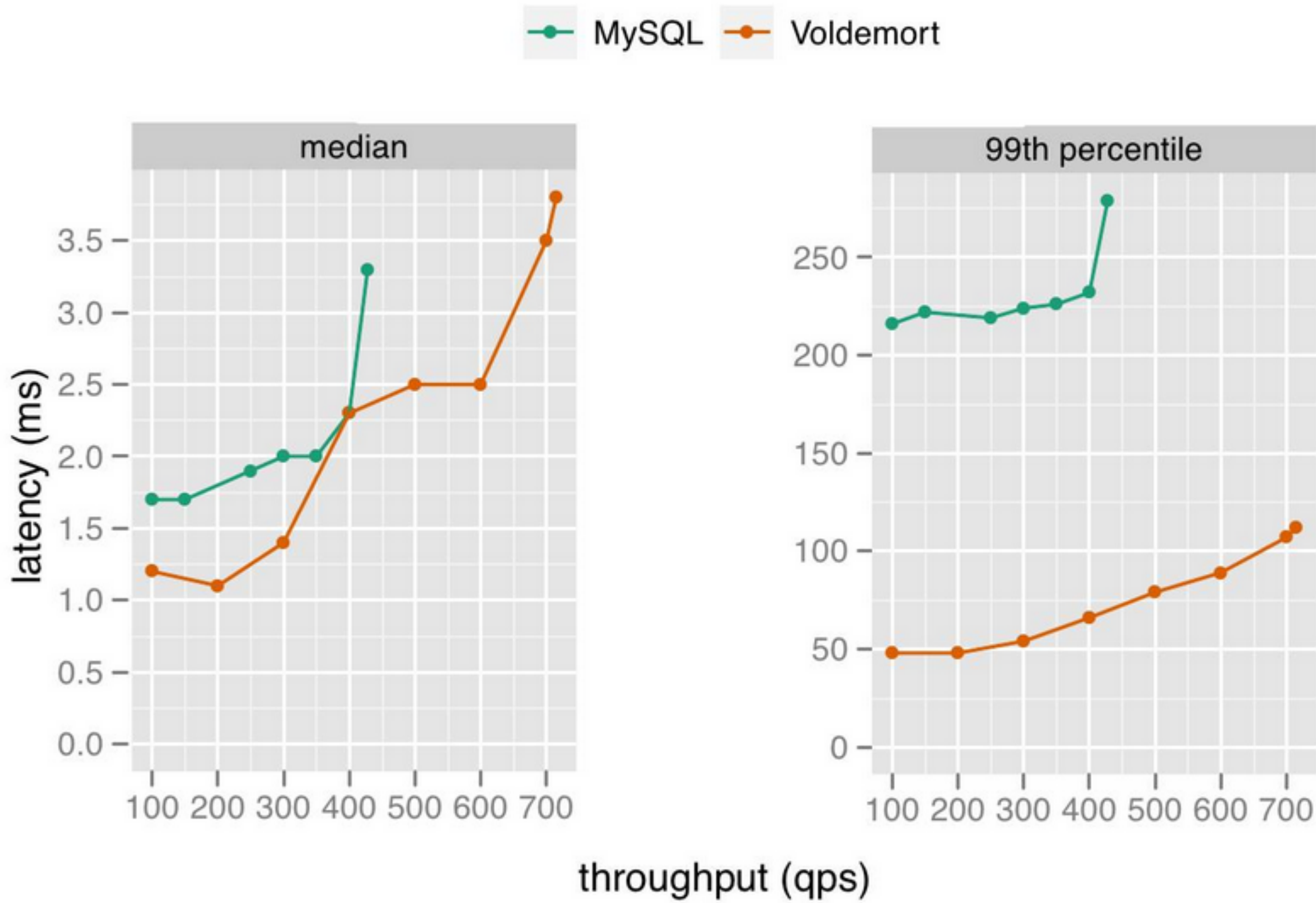
LinkedIn Skills

Jobs you may be interested in



Sid Anand, LinkedIn Data Infrastructure (QCon London 2012)

Voldemort vs MySQL



Sid Anand, LinkedIn Data Infrastructure (QCon London 2012)

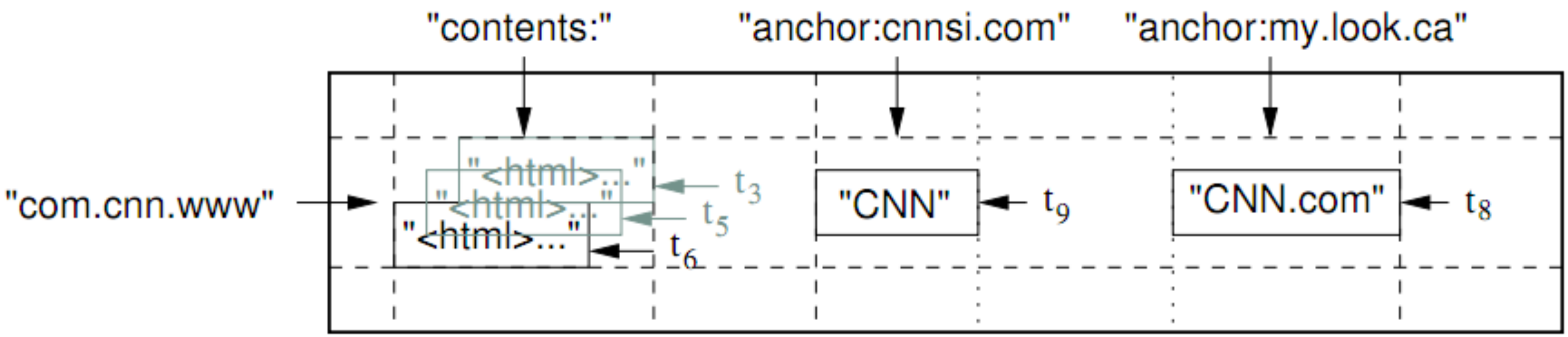
VirginiaTech

Column Families – BigTable like

Sparse, distributed, persistent multi-dimensional sorted map indexed by (*row_key*, *column_key*, *timestamp*)

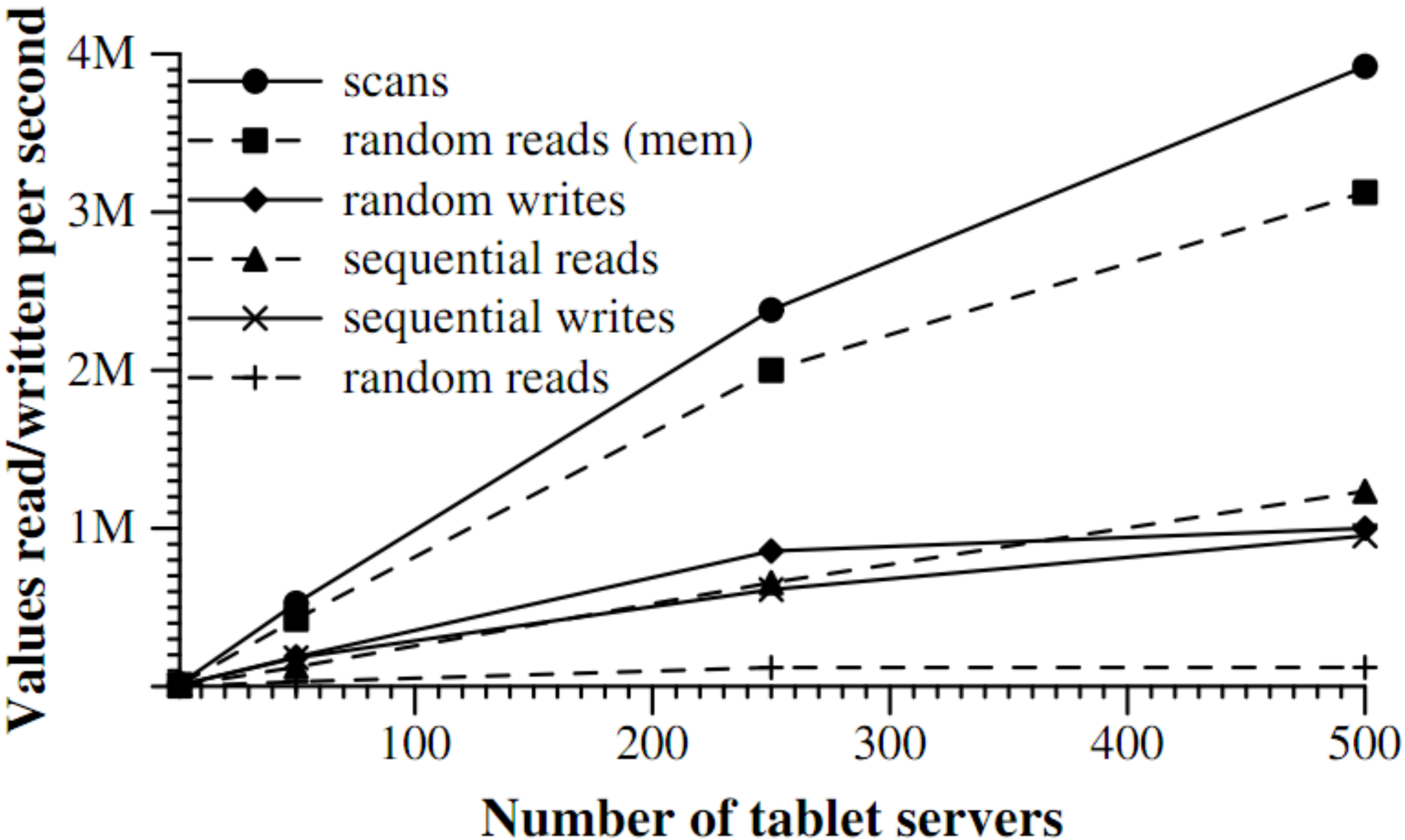
F Chang, et al, Bigtable: A Distributed Storage System for Structured Data, osdi 06

BigTable Data Model



The row name is a reversed URL. The contents column family contains the page contents, and the anchor column family contains the text of any anchors that reference the page.

BigTable Performance



Document Database - mongoDB

Last Name	First Name	Age
DUMONT	Jean	43
PELLERIN	Franck	29
MATTHIEU	Nicolas	51

Table in relational db

```
{
    "_id": ObjectId("4efa8d2b7d284dad101e4bc9"),
    "Last Name": "DUMONT",
    "First Name": "Jean",
    "Age": 43
},
{
    "_id": ObjectId("4efa8d2b7d284dad101e4bc7"),
    "Last Name": "PELLERIN",
    "First Name": "Franck",
    "Age": 29,
    "Address": "1 chemin des Loges",
    "City": "VERSAILLES"
}
```

Documents in a collection

powered by mongoDB

Initial release 2009

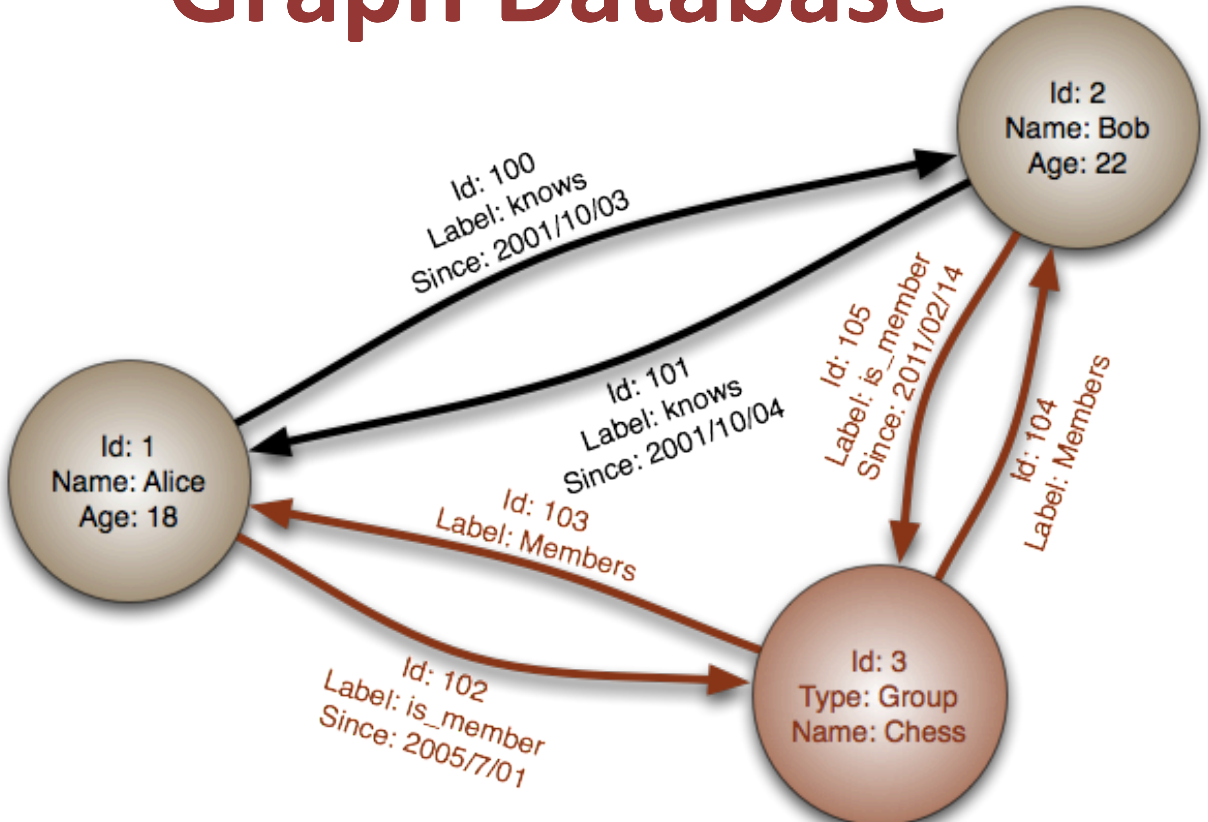
Open source, document db
Json-like document with dynamic schema

mongoDB Product Deployment

the national archives
Pinterest
Springer science+business media
ACTIVESPHERE
淘宝网 Taobao.com
about.me
ATHENA CAPITAL RESEARCH
GLOBAL ADVERTISERS
agentstorm
AdMaster Digital Consulting
bit.ly
The New York Times
sourceforge
github SOCIAL CODING
bump TECHNOLOGIES
Buzzers
Grooveshark
GILT GROUPE
SAP
EA
BuzzFeed
Doodle
foursquare
Disney
Forbes
Eventbrite Events Made Easy
CNN TÜRK .com
craigslist

And much more…

Graph Database



Data Model Abstraction:

- Nodes
- Relations
- Properties

Neo4j - Build a Graph

```
NeoService neo = ... // Get factory


// Create Thomas 'Neo' Anderson
Node mrAnderson = neo.createNode();
mrAnderson.setProperty( "name", "Thomas Anderson" );
mrAnderson.setProperty( "age", 29 );

// Create Morpheus
Node morpheus = neo.createNode();
morpheus.setProperty( "name", "Morpheus" );
morpheus.setProperty( "rank", "Captain" );
morpheus.setProperty( "occupation", "Total bad ass" );

// Create a relationship representing that they know each other
mrAnderson.createRelationshipTo( morpheus, RelTypes.KNOWS );
// ...create Trinity, Cypher, Agent Smith, Architect similarly
```

Slide from neo technology, “A NoSQL Overview and the Benefits of Graph Databases"

A Debatable Performance Evaluation

Conclusion

- Use the right data model for the right problem

THE HADOOP ECOSYSTEM

Single vs Cluster

- 4TB HDDs are coming out
- Cluster?
  - How many machines?
  - Handle machine and drive failure
  - Need redundancy, backup..

3% of 100K HDDs fail in <= 3 months

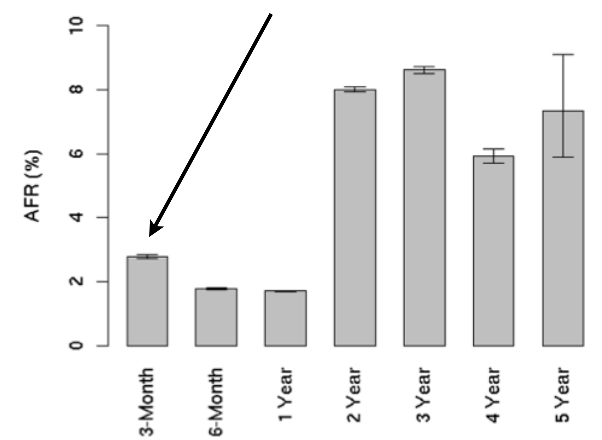


Figure 2: Annualized failure rates broken down by age groups

http://static.googleusercontent.com/external_content/untrusted_dlcp/research.google.com/en/us/archive/disk_failures.pdf

Hadoop

- Open source software
  - Reliable, scalable, distributed computing
- Can handle thousands of machines
- Written in JAVA
- A simple programming model
- HDFS (Hadoop Distributed File System)
  - Fault tolerant (can recover from failures)

Hadoop VS NoSQL

- Hadoop: computing framework
  - Supports data-intensive applications
  - Includes MapReduce, HDFS etc.

    (we will study MR mainly next)

- NoSQL: Not only SQL databases
  - Can be built ON hadoop. E.g. HBase.

Why Hadoop?

- Many research groups/projects use it
- Fortune 500 companies use it
- Low cost to set-up and pick-up

- Its **FREE**!!
  - Well not quite, if you do not have the machines ☺
  - You will be using the Amazon Web Services in HW5 system to 'hire' a Hadoop cluster on demand

Map-Reduce [Dean and Ghemawat 2004]

- Abstraction for simple computing
  - Hides details of parallelization, fault-tolerance, data-balancing
  - MUST Read!

http://static.googleusercontent.com/media/research.google.com/en/us/archive/mapreduce-osdi04.pdf

Programming Model

- VERY simple!
- Input: key/value pairs
- Output: key/value pairs

- User has to specify TWO functions: map() and reduce()
  – Map(): takes an input pair and produces k intermediate key/value pairs
  – Reduce(): given an intermediate pair and a set of values for the key, output another key/value pair

Master-Slave Architecture: Phases

1. **Map phase**

   Divide data and computation into smaller pieces; each machine 'mapper' works on one piece in parallel.

2. **Shuffle phase**

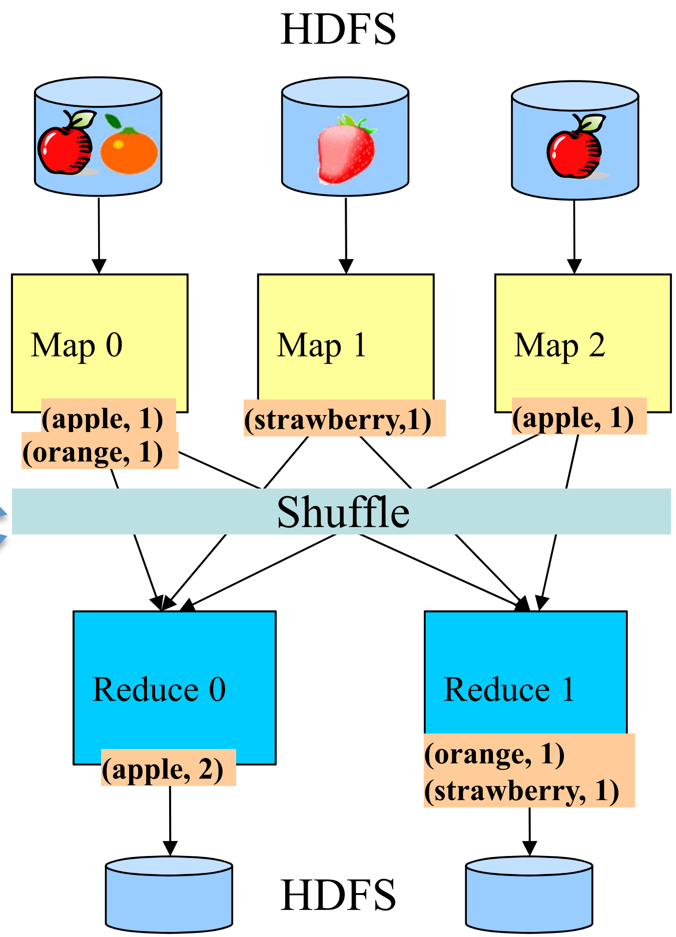
   Master sorts and moves results to reducers

3. **Reduce phase**

   Machines ('reducers') combine results in parallel.

Example: Histogram of Fruit names

Sort the intermediate data on the key (the fruit name here)



This Phase also ensures that *all* tuples of one key end up in only one reducer

```
map( fruit ) {
  output(fruit, 1);
}
```

```
reduce( fruit, v[1..n] ) {
  for(i=1; i <=n; i++)
    sum = sum + v[i];
  output(fruit, sum);
}
```

Source: U Kang, 2013

Map and Reduce

IMPORTANT:

1. each mapper runs the same code (your map function)

2. ditto, for each reducer (your reducer function)

→**Code is independent of the degree of parallelization**

I.E.

Code is independent of the actual number of mappers and reducers used.

Map-Reduce (MR) as SQL

- select count(*) ← **Reducer**

  from FRUITS

  group by fruit-name

Mapper

More examples

- Count of URL access frequency
  - Map(): output <URL, 1>
  - Reduce: output <URL, total_count>
- Reverse Web-link graph
  - Map(): output <target, source> for each target link in a source web-page
  - Reduce(): output <target, list[source]>

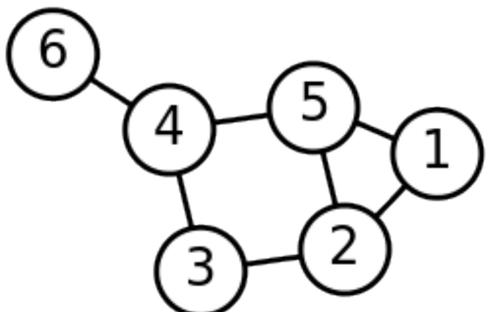
Even more examples given in the MR paper as well

AWS Demo

- Live demo of word-count example on laptop…

Degree of graph Example

- Find degree of every node in a graph



Example: In a friendship graph, what is the number of friends of every person:

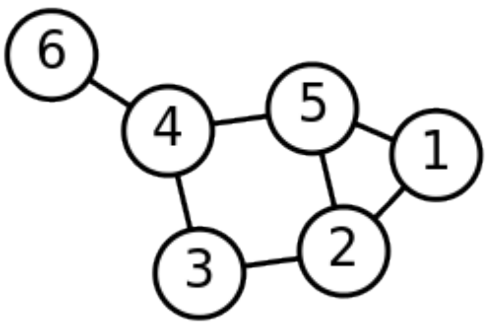
Node 6 = 1   Node 2 = 3

Node 4 = 3   Node 1 = 2

Node 3 = 2   Node 5 = 3

Degree of each node in a graph

- Suppose you have the edge list



===

```
6 4
4 6
4 3
3 4
4 5
5 4
...
```

== a table!

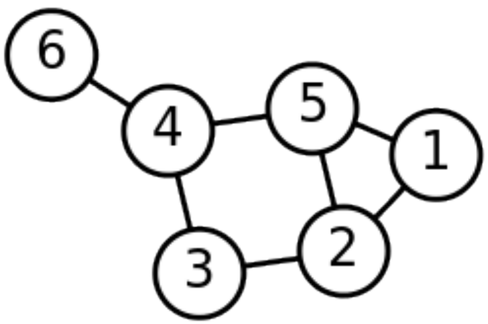
Schema?

Edges(from, to)

* Caveat: these are undirected graphs. In HW5 you will deal with directed graphs.

Degree of each node in a graph

- Suppose you have the edge list



===

```
6 4
4 6
4 3
3 4
4 5
5 4
...
```

== a table!

Schema?

Edges(from, to)

SQL for degree list?

SELECT from, count(*)
FROM Edges
GROUP BY from

Degree of each node in a graph

- So in SQL:

SELECT from, count(*)
FROM Edges
GROUP BY from

- MapReduce?

Mapper:

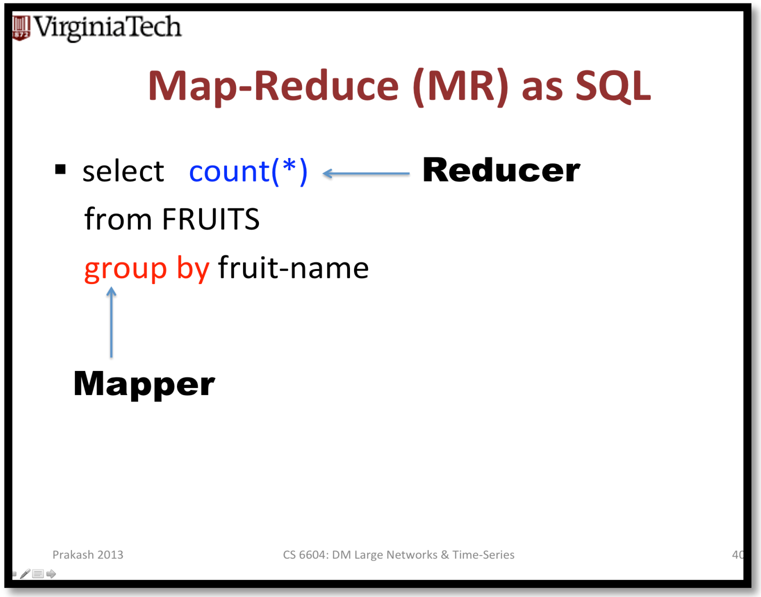
emit (from, 1)

Reducer:

emit (from, count())

6 4
4 6
4 3
3 4
4 5
5 4
. . .

Remember



I.E. essentially equivalent to the ‘fruit-count’ example ☺

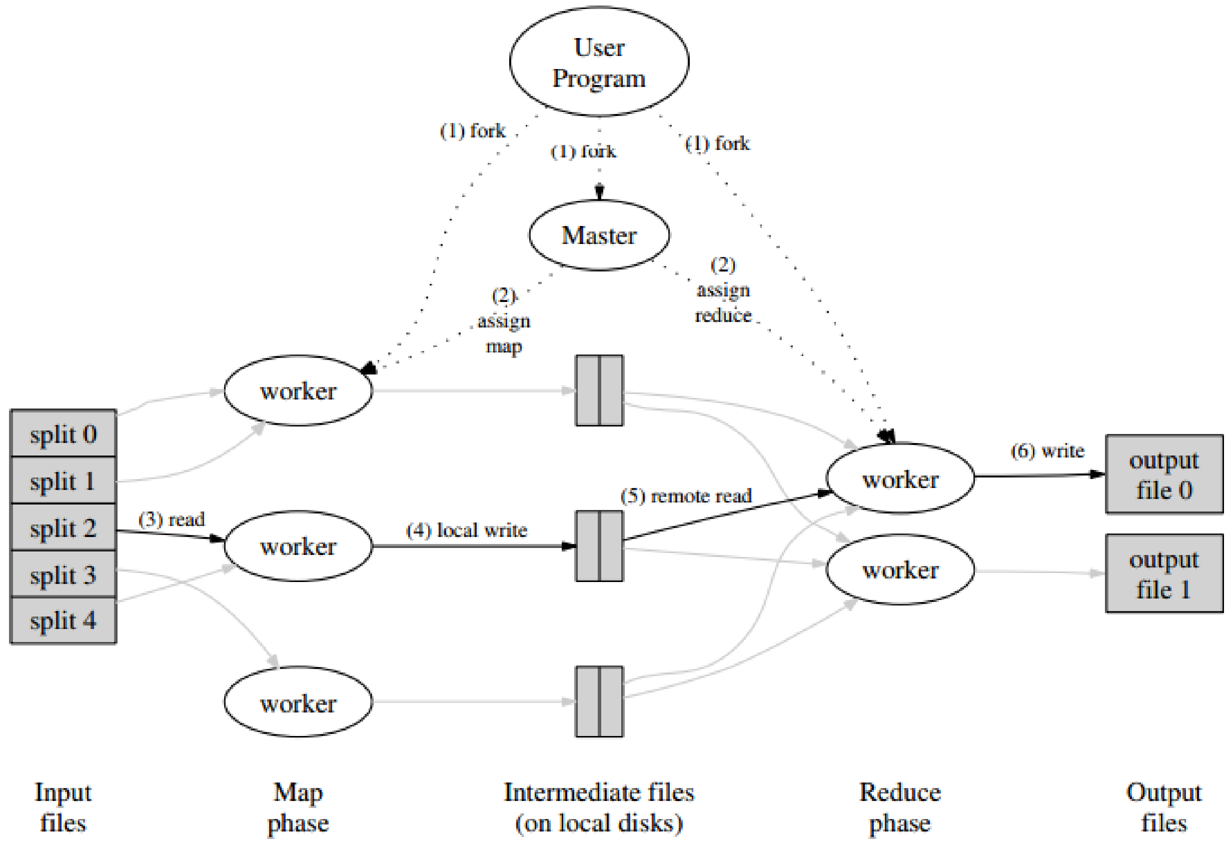
In HW5

- We ask you to get the *in- and out-degree distributions* of a large *directed* graph
  - You will need 2 consecutive MR jobs for this

I.E.:

Input (edges) → Map1(.) → Reduce1(.) → Map2 (.) → Reduce2(.) → Output (= degree distribution)

MapReduce Architecture



Master

- For each map and reduce task, stores
  - The state (idle, in-progress, completed)
  - The identity of the worker machine (for non-idle tasks)

Fault Tolerance

- Master pings each worker periodically
- If response time-out, worker marked as failed
- MR task on a failed worker is eligible for rescheduling

Locality

- GFS (Google File System)
  - – 3-way replication of data

Location information of the input files

Schedule a map task on a machine with replica

Backup Tasks (Speculative Execution)

- The problem of ‘straggler’
  - When one ‘bad’ execution prevents completion
- Solution
  - When a MR operation is close to completion, Master schedules backup executions of the remaining in-progress tasks
  - The task is marked as completed whenever either the primary or backup execution completes
- 44% faster

Refinements

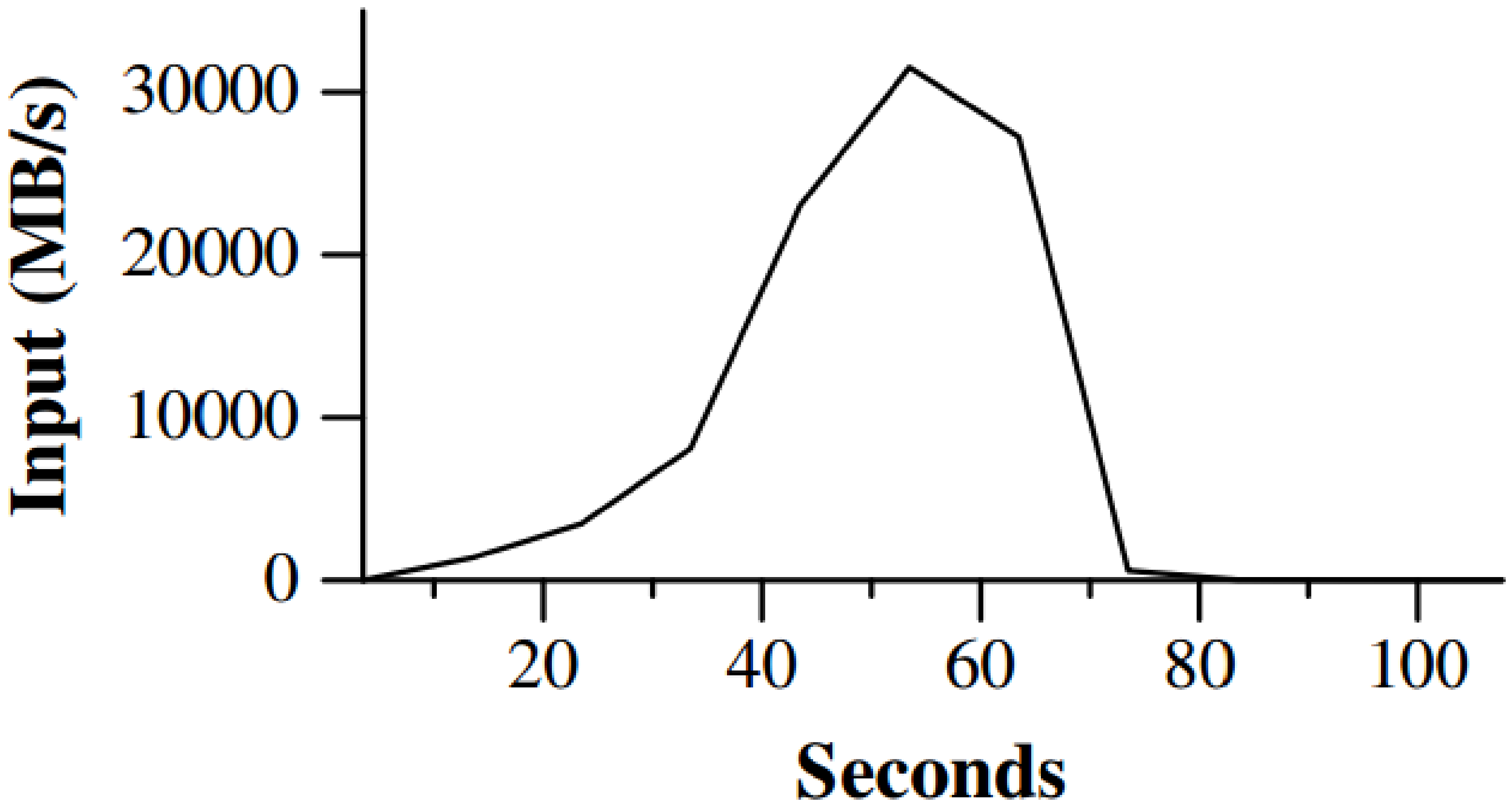
SKIP!

- Partitioning Function
- Ordering Guarantee
- Combiner function
- Status information
- Counter

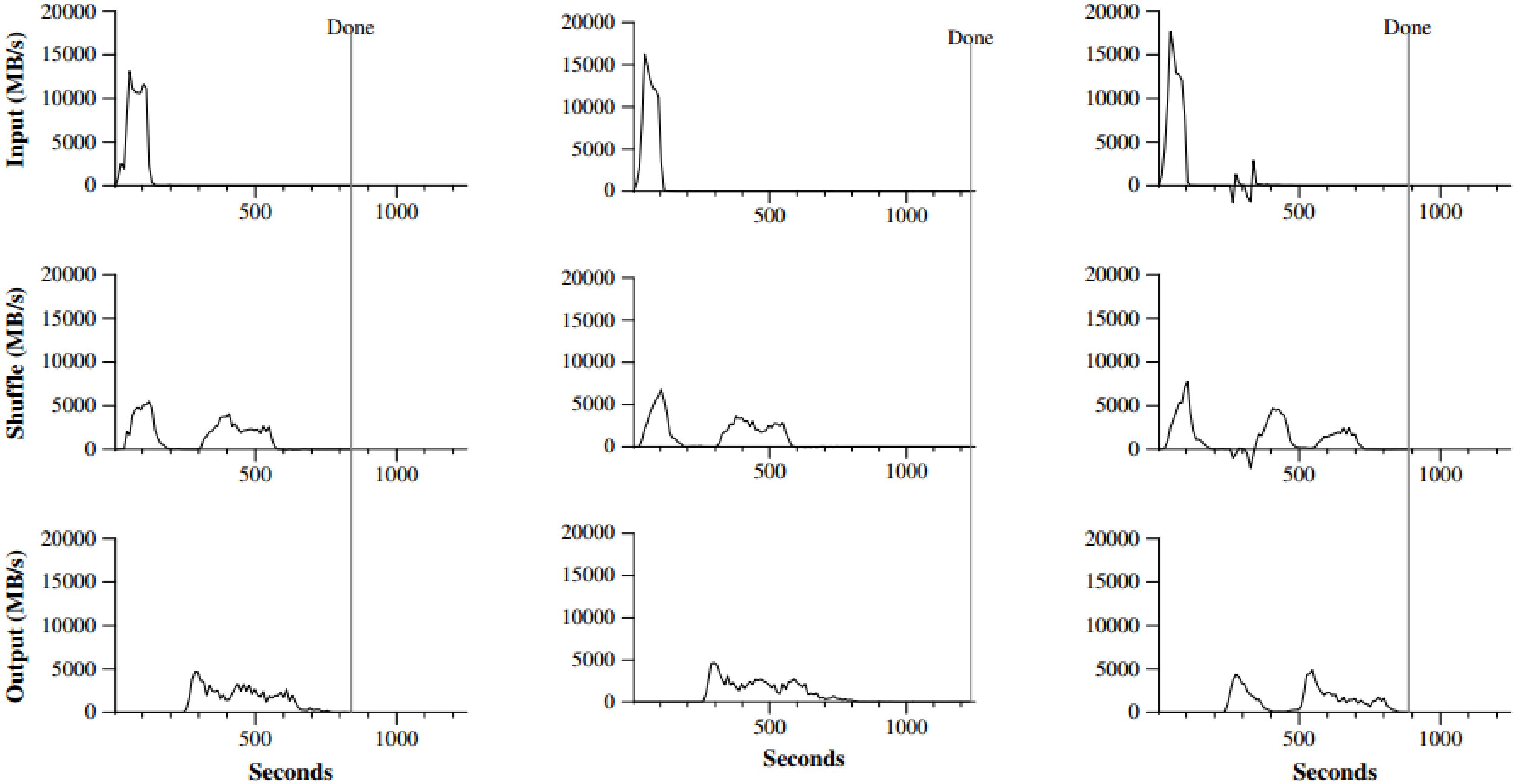
Status information

- Master has an internal HTTP server
  - Exports status web-pages
  - Tells you progress information
    - Are tasks active

Performance---Grep



Performance---Sort



(a) Normal execution

(b) No backup tasks

(c) 200 tasks killed

MR @ Google

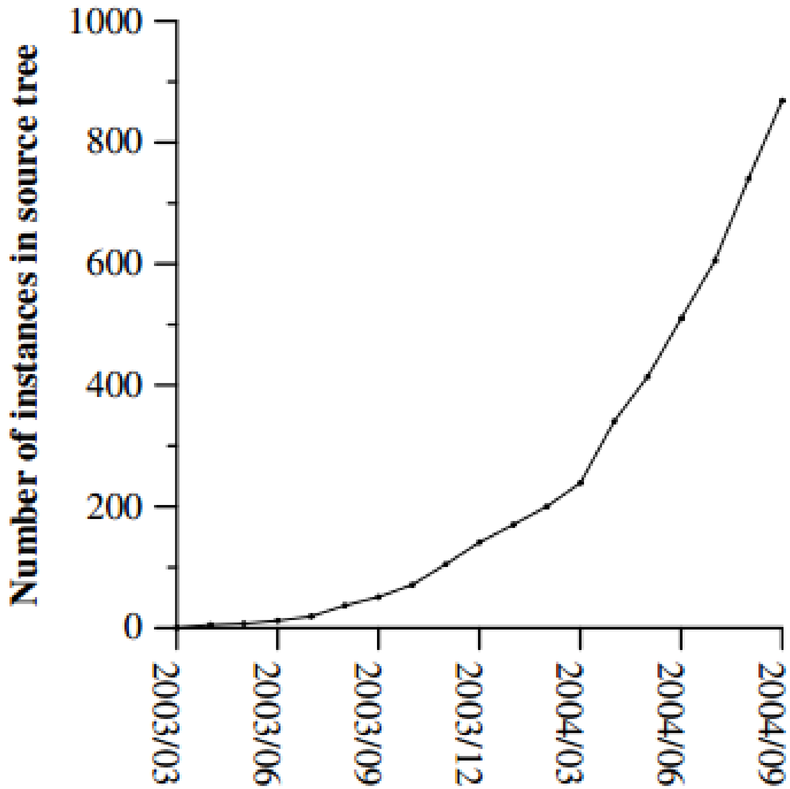


Figure 4: MapReduce instances over time

MR @ Google

Number of jobs	29,423
Average job completion time	634 secs
Machine days used	79,186 days
Input data read	3,288 TB
Intermediate data produced	758 TB
Output data written	193 TB
Average worker machines per job	157
Average worker deaths per job	1.2
Average map tasks per job	3,351
Average reduce tasks per job	55
Unique *map* implementations	395
Unique *reduce* implementations	269
Unique *map/reduce* combinations	426

Table 1: MapReduce jobs run in August 2004

Conclusions

- Hadoop is a distributed data-intesive computing framework
- MapReduce
  - Simple programming paradigm
  - Surprisingly powerful (may not be suitable for all tasks though)
- Hadoop has specialized FileSystem, Master-Slave Architecture to scale-up

NoSQL and Hadoop

- Hot area with several new problems
  – Good for academic research
  – Good for industry

= Fun AND Profit ☺