Data Management with Google BigTable

Pramod Bhatotia
http://homepages.inf.ed.ac.uk/pbhatoti/

Credits for the lecture material:
BigTable OSDI paper, and HBase

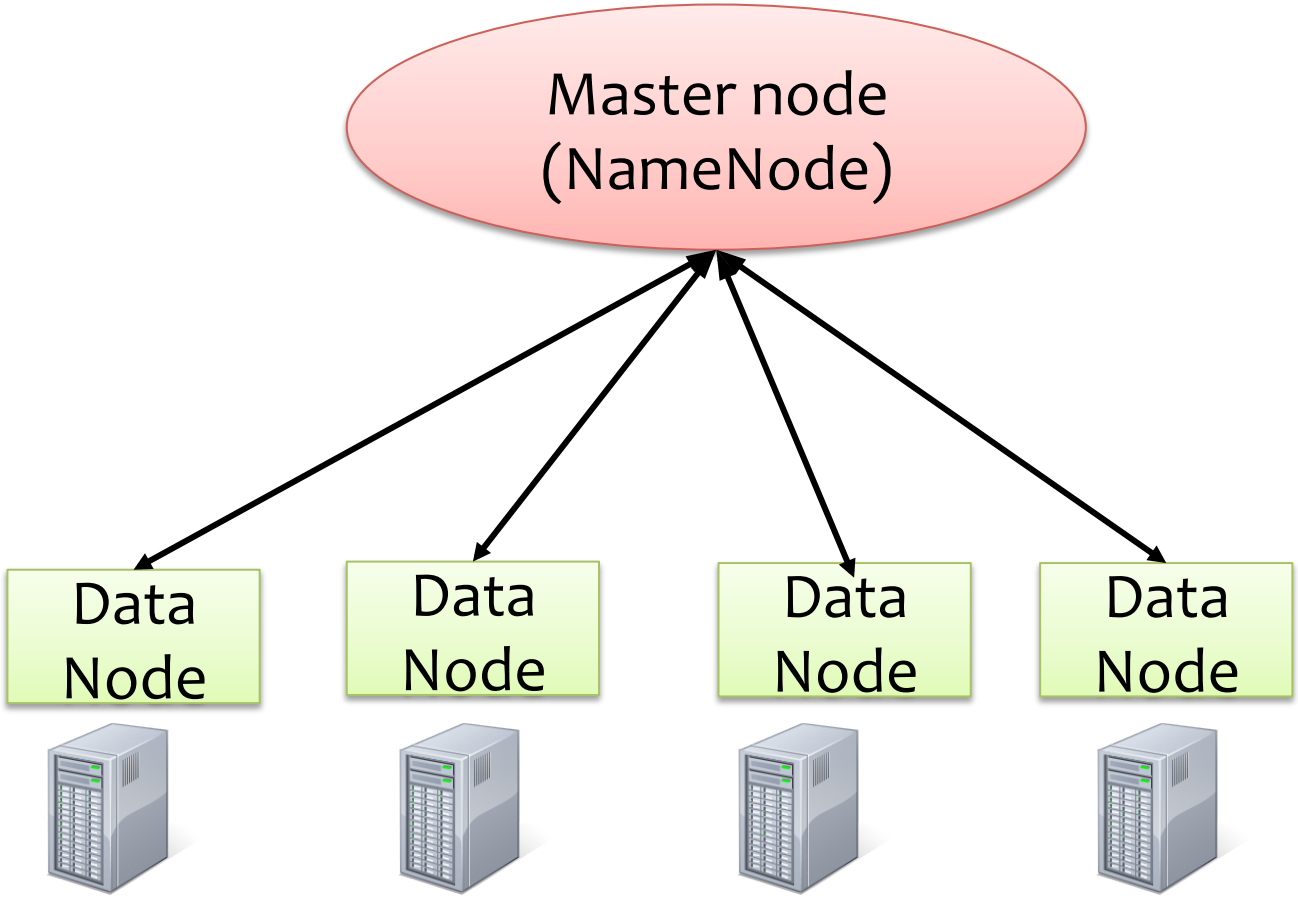
THE UNIVERSITY *of* EDINBURGH

In today's class

How to manage "Big data" in distributed setting?

- Google File System (HDFS)

Architecture



Limitations of GFS/HDFS

- Designed for unstructured data
  - Sequential reads (less random reads)
  - High throughput, not so latency sensitive
  - Unstructured data

(Semi-) Structured data

- Web data
  - Contents, crawl metadata, links, pagerank, …
- Per-user data
  - User preference settings, recent queries/search results, …
- Map data
  - Physical entities (shops, restaurants, etc.), roads, satellite image data, user annotations, …

Why not to use DB?

DBMS are not scalable! ☺

BigTable/HBase
A distributed storage system for structured data

BigTable features

- Scalable
  - Thousands of servers
  - Terabytes of in-memory data
  - Petabyte of disk-based data
  - Millions of reads/writes per second, efficient scans
- Self-managing
  - Servers can be added/removed dynamically
  - Servers adjust to load imbalance
- Extremely popular at Google (as of 2008)
  - Web indexing, personalized search, Google Earth, Google Analytics, Google Finance, …

Design goal #1: Scalable

- Billions of Web pages, many versions/page (~20K/ version)
- Hundreds of millions of users, thousands of q/sec
- 100TB+ of satellite image data

Design goal #2: Efficient

- Very high read/write rates (millions of ops per second)
- Efficient retrieval of small subsets of the data
- Efficient scans over entire or subsets of the data

What is BigTable/Hbase?

A BigTable is
sparse, distributed, persistent multi-dimensional sorted map

Key:(row, column, time) → value

Data Model

- A BigTable of data with rows and columns
- Rows are uniquely identified by a key
- Columns are organized column families
  - Each family has a set of related columns identified by the column qualifier
- Values in each cell are versioned based on timestamp

Row

- Row
  - Row name/key is an arbitrary string
  - Sorted lexicographically
  - Access to data in a row is atomic
  - Does not support relation model (transactions)
  - **Tablet:** a range of rows

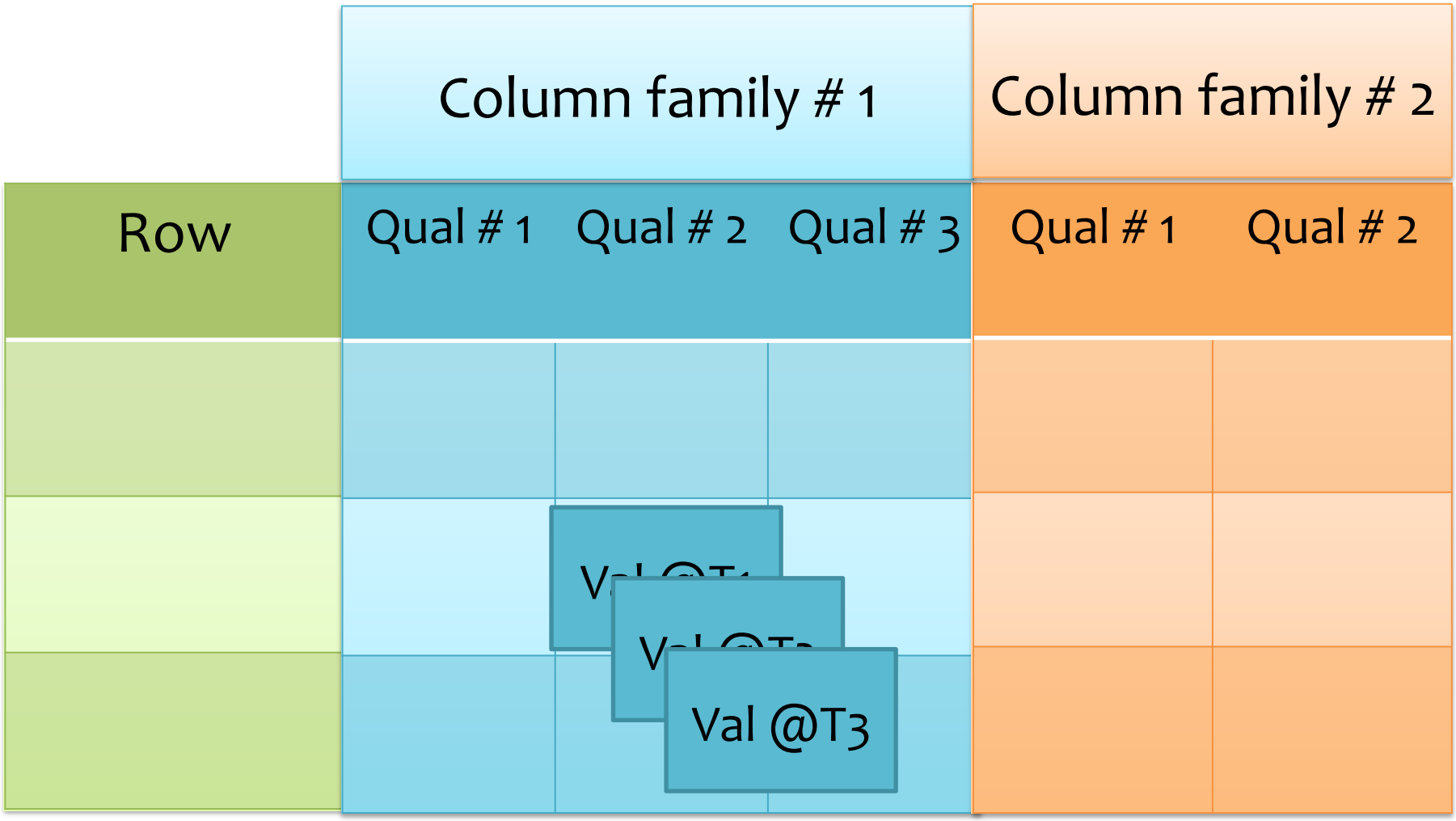
Column

- Columns are organized hierarchical in families
  - Each column family has a set of columns
- Column_family: column_qualifier
- Column family
  - Unit of access control
  - Stored/compressed together

Time stamp

- Versioning: used to store different versions of data in a cell
  - 64-bits integer (UNIX timestamp)
  - New writes default to current time, but timestamps for writes can also be set explicitly by clients
- Lookup options
  - "Return most recent K values"
  - "Return all values in timestamp range (or all values)"
- Garbage collection:
  - "Only retain most recent K values in a cell"
  - "Keep values until they are older than K seconds"

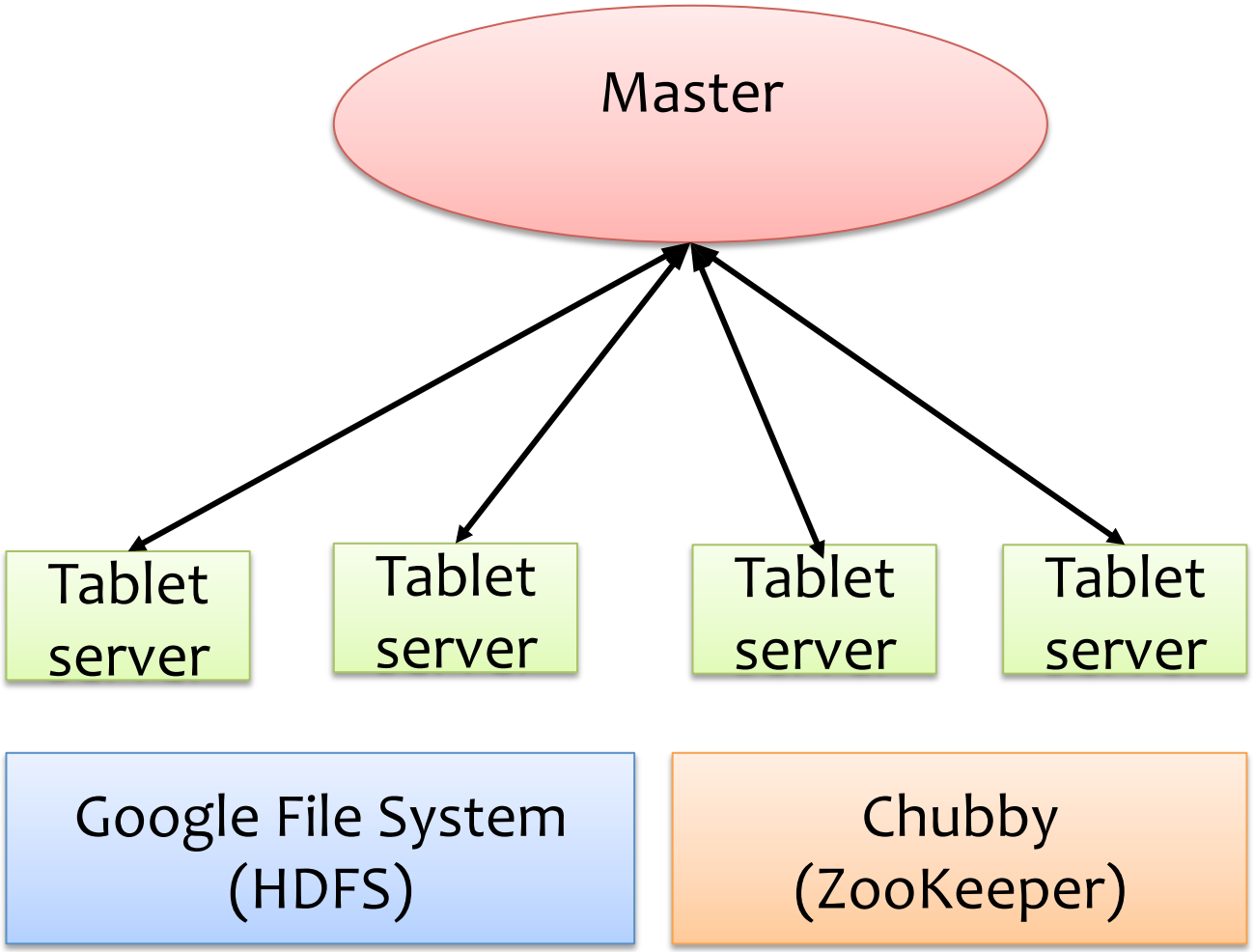
Example


Column family # 1
Column family # 2
Row
Qual # 1
Qual # 2
Qual # 3
Qual # 1
Qual # 2
Val @T3

Programming API

- Create/delete/manage tables (Of course!)
- CRUD:
  - Create a row via PUT
  - Read a row via GET
  - Write/Update a row (atomically) via PUT
  - Updates: Can be conditional
  - Delete a row via DELETE
  - Transaction: single row read-modify-write
- Scan: Iterator
  - Sequential read over ranges of rows (sorted)

Architecture



Tablets

- A Bigtable table is partitioned into many tablets based on row keys
  - Tablets (100-200MB each) are stored in a particular structure in GFS
  - Each tablet is served by one tablet server

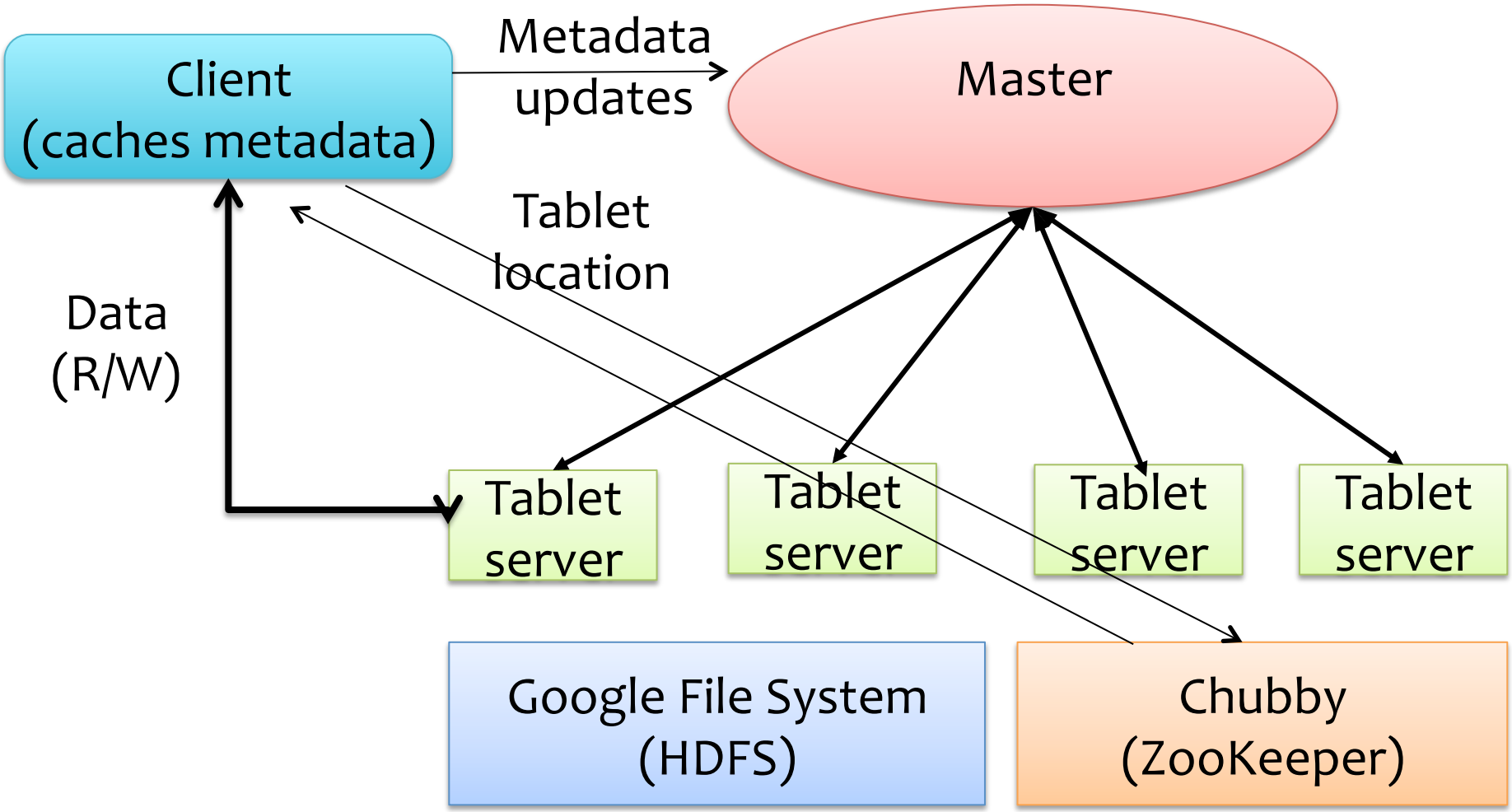
Master

- Assigns/load-balances tablets to tablet servers
- Detects up/down tablet servers
- Garbage collects deleted tablets
- Coordinates metadata updates
- Does NOT provide tablet location

Tablet servers

- Tablet servers handle R/W requests to their tablets
- Split tablets that have grown too large
- Tablet servers are also stateless – their state is in GFS

Read/write operations



Reference

- BigTable [OSDI'06]
  - Original paper for BigTable
- Spanner [OSDI'12]
  - Distributed databases with serializable transactions

Summary

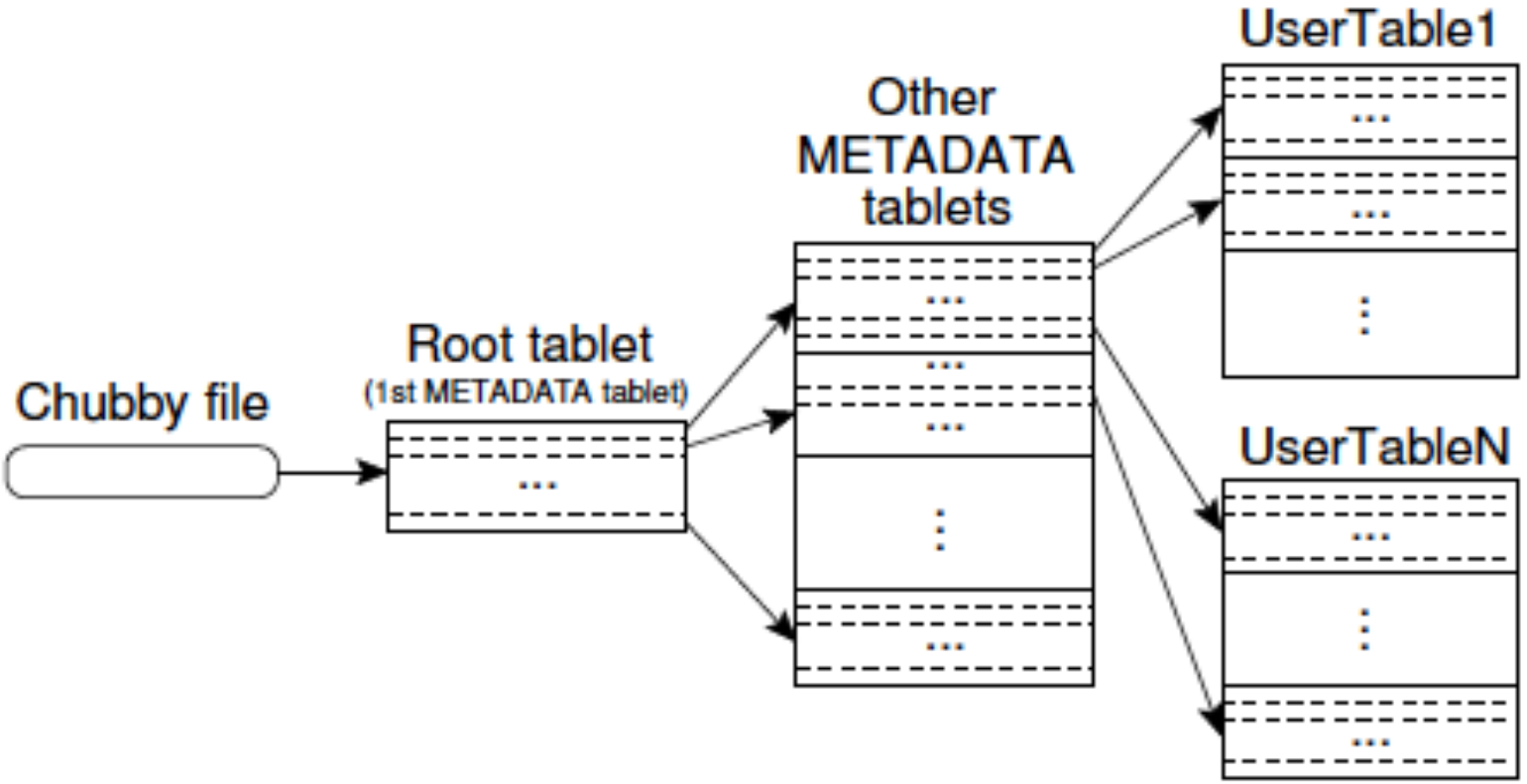
- Data-management with BigTable
  - Distributed storage architecture
  - BigTable: A database system for structured data

- Resources:
  - HDFS: http://hadoop.apache.org/
  - Hbase: http://hbase.apache.org/
  - Tachyon: http://tachyon-project.org/

Thanks!

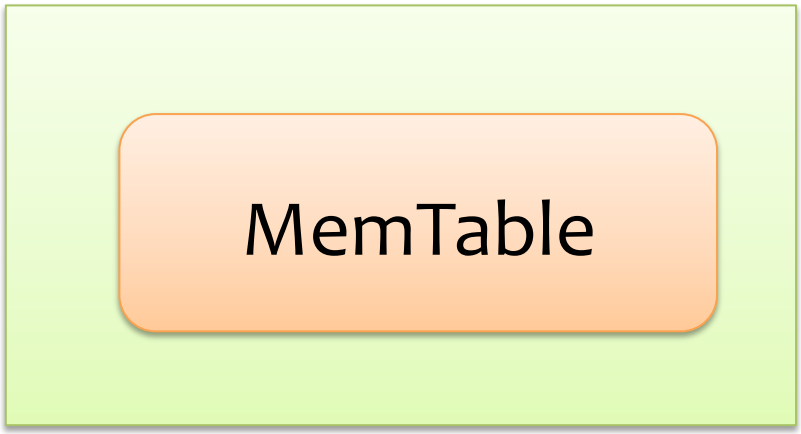
http://homepages.inf.ed.ac.uk/pbhatoti/

Tablet location



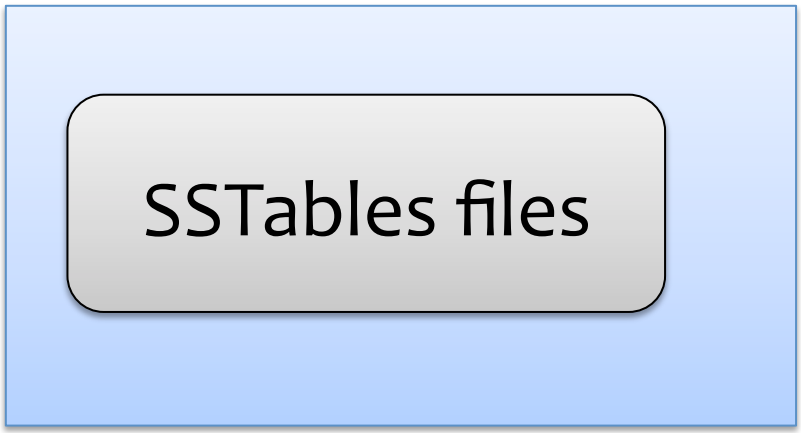
Tablet server

Memory



In-memory
Key-value store

Disk/GFS



Immutable,
ordered subset of
table

HBase Architecture