BigTable

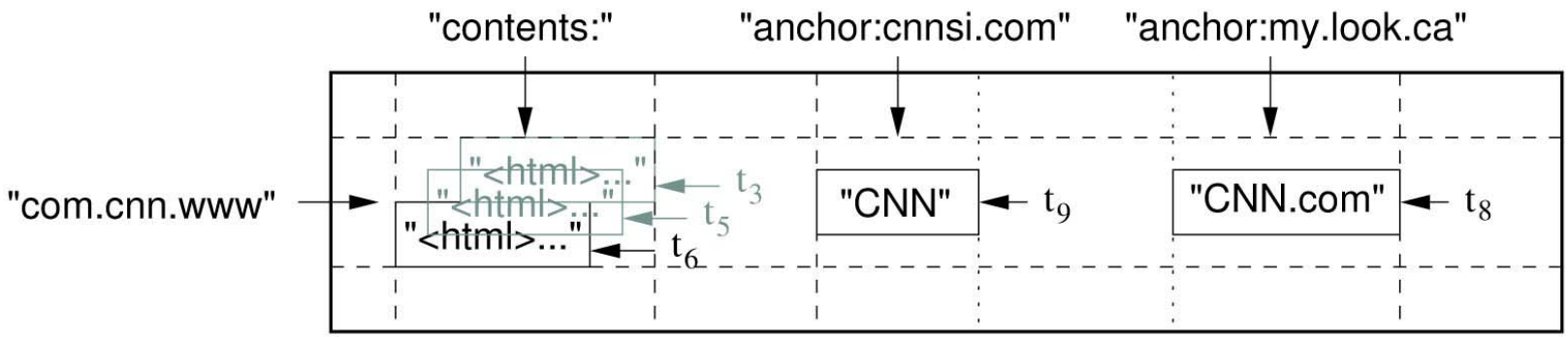
CSE 490h, Autumn 2008

What is BigTable?

- "A BigTable is a sparse, distributed, persistent multidimensional sorted map. The map is indexed by a row key, a column key, and a timestamp; each value in the map is an uninterpreted array of bytes."

```
(row:string, column:string, time:int64) -> string
```

Motivating example



- Lots of semi-structured data
- Enormous scale

Why?

- You often need more than a file system
  - Cf. Windows Vista
- Do you need a full DBMS?
  - Few DBMS's support the requisite scale
  - Most DBMS's require very expensive infrastructure
  - Google had highly optimized lower-level systems that could be exploited
    - GFS, Chubby, MapReduce, job scheduling
  - DBMS's provide more than Google needed
    - E.g., full transactions, SQL

Relative to a DBMS, BigTable provides …

- - Simplified data retrieval mechanism
  - `(row, column, time:) -> string` only
  - No relational operators
- - Atomic updates only possible at row level
- + Arbitrary number of columns per row
- + Arbitrary data type for each column

- Designed for Google's application set
- Provides extremely large scale (data, throughput) at extremely small cost

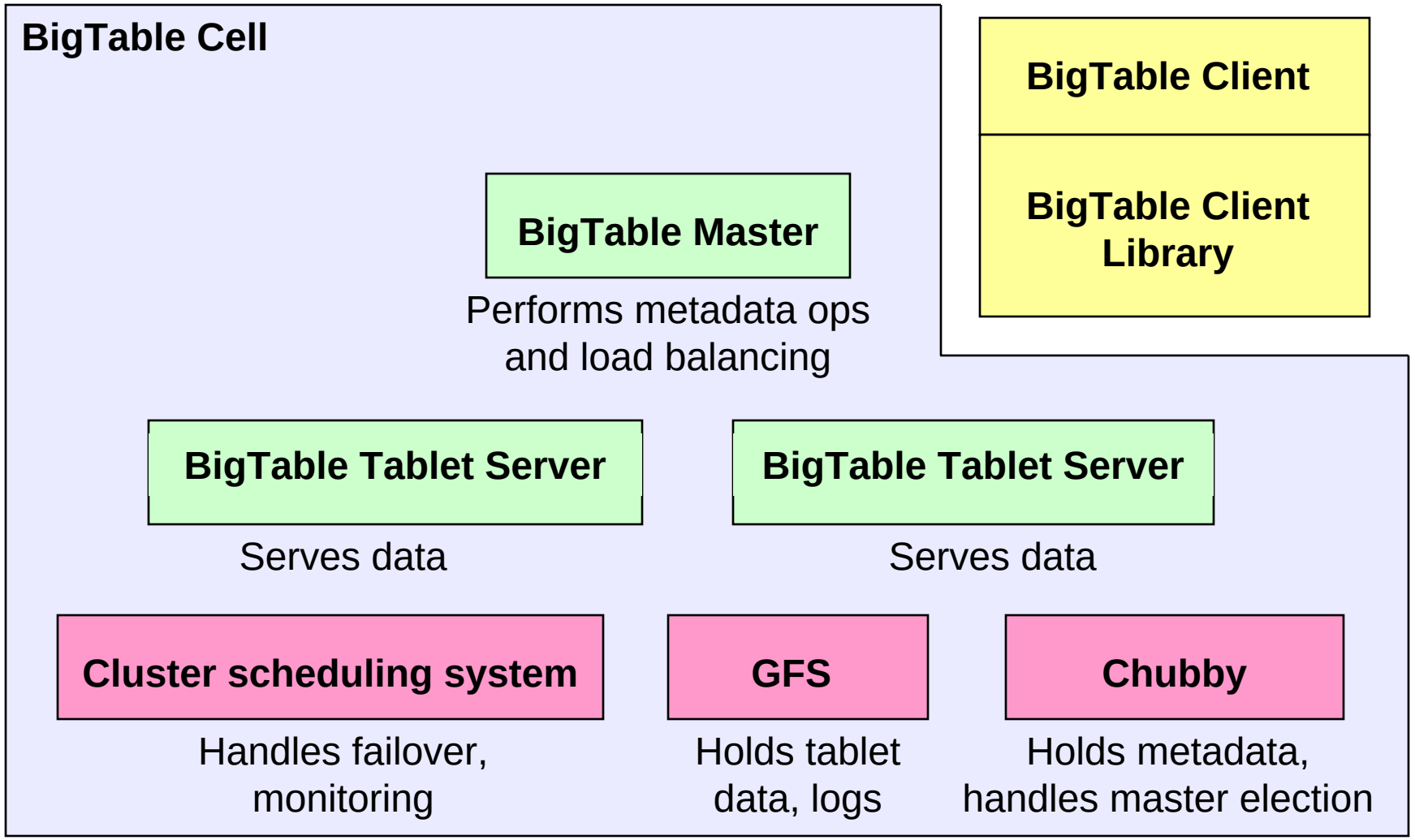
Physical representation of data

- A logical "table" is divided into multiple *tablets*
  - Each tablet is one or more SSTable files in GFS
- Each tablet stores an interval of table rows
  - Rows are lexicographically ordered (by key)
  - If a tablet grows beyond a certain size, it is split into two new tablets

Software structure of a BigTable cell

- One master server
  - Communicates only with tablet servers
- Multiple tablet servers
  - Perform actual client accesses
- Chubby lock service holds metadata (e.g., the location of the root metadata tablet for the table), handles master election
  - How?
- GFS servers provide underlying storage

High-level structure



A few implementation notes

- Finding the tablet holding the row you want
  - Hierarchical metadata tablets
- Writing
  - Append-only log held in GFS
- A million and one details
  - Locality groups
  - Compression
  - Caching
  - Bloom filters
  - Commit log implementation
  - Tablet recovery