Department of Computer Applications

Key value store

Course: NoSQL Database system

Class / Semester: II MCA / III Semester

Key-Value stores

- simple data model that maps keys to a list of values

. Easy to achieve

  - Performance
  - Fault tolerance
  - Heterogeneity
  - Availability

due to its schema-less data model and fine granularity partitioning of the data

Goog BigTable

- Google use the key-value paradigm to map URLs to multidimensional data, such as:
  - Timestamps/Versions
  - Rank
  - Keywords
  - Links
- No explicit ordering is needed on keys since a hash function is used

MapRed

- Many data distributed over hundreds or thousands of machines
- These data need to be processed and produce new data (usually the process is a simple task)
  - ex. aggregate functions, filtering, etc.

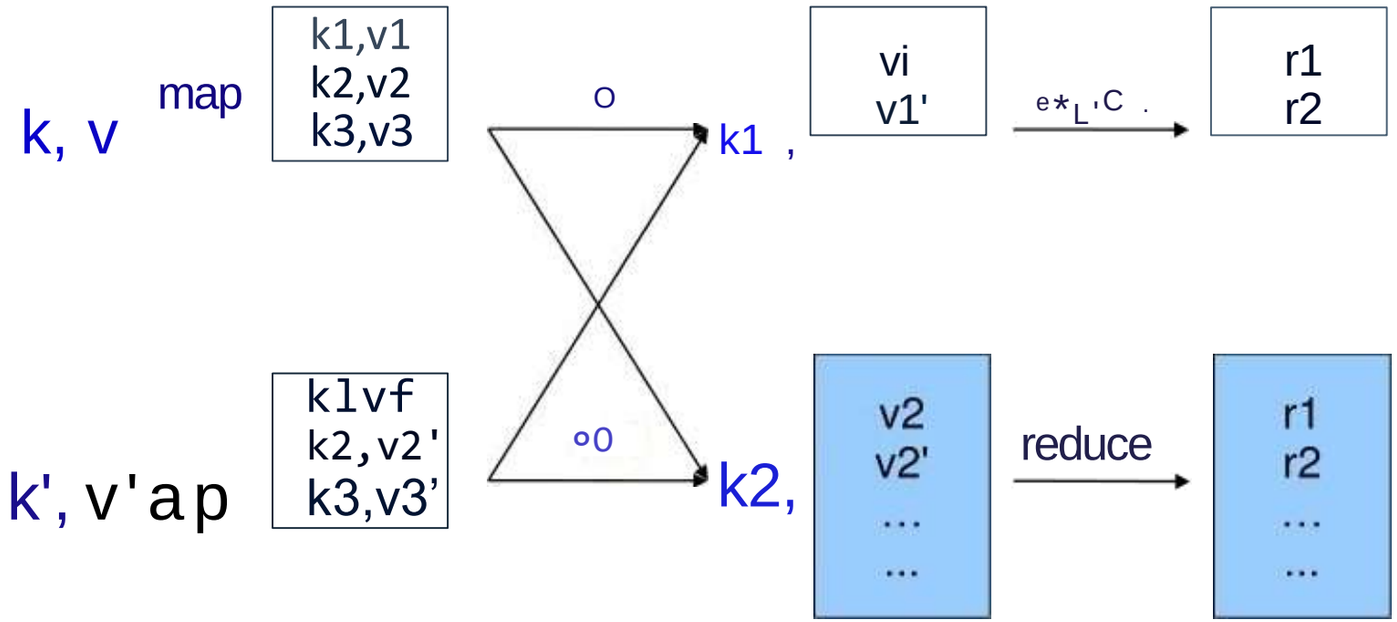
MapReduce

- Partition the data according to pre-defined key
  - ex. words, URLs, etc.
- A master node assigns to workers specific partitions (=keys, =mappings of data to keys)
- The worker will produce a new list of key—value, corresponding to the new intermediate processed data (Map phase)
- Workers then will gather all intermediate data belonging to a specific key, and reduce them to the requested output ordered by key (Reduce phase)

M apRed

map (in key, in value) -> list(out key, intermediatevalue)

reduce (out key, list(intermediate_value)) -> list(out_value)

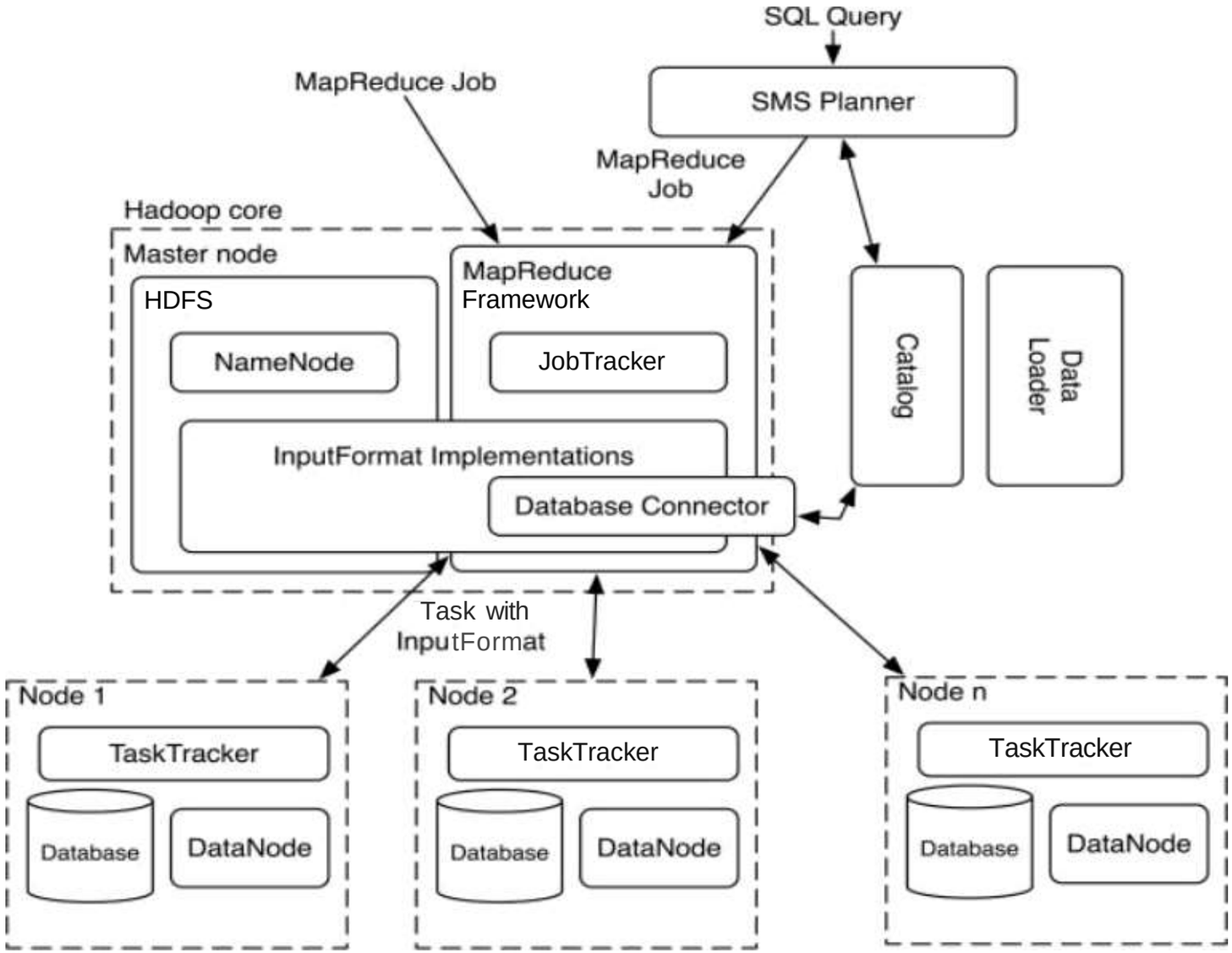


The dirty little secret of Google that is too obvious is

Hadoop Map/Reduce

- open source implementation of the map/reduce idea
- takes care of scheduling tasks, monitoring them and re-executes the failed tasks
- a single master JobTracker and
- one slave TaskTracker per cluster-node

HadoopDB


SQL Query
SMS Planner
MapReduce Job
MapReduce Job
Hadoop core
Master node
HDFS
NameNode
MapReduce Framework
JobTracker
InputFormat Implementations
Database Connector
Catalog
Data Loader
Task with InputFormat
Node 1
TaskTracker
Database
DataNode
Node 2
TaskTracker
Database
DataNode
Node n
TaskTracker
Database
DataNode

HadoopDB: Database Connector

- extends Hadoop's InputFormat class
- connects to a database, executes the SQL query and returns results as key-value pairs
- “should” suppoi1 any JDBC-compliant database that resides in the cluster

HadoopDB: Catalog

- The catalog maintains meta-information about the databases
  - connection parameters such as database location, driver class and credentials
  - metadata such as data sets contained in the cluster, replica locations, and data partitioning properties

HadoopDB: Data Loader

- responsible for:
  - globally repartitioning data on a given partition key upon loading
  - breaking apart single node data into multiple smaller partitions or chunks and
  - finally bulk-loading the single-node databases with the chunks

The [not so] easy way to do the labwork...

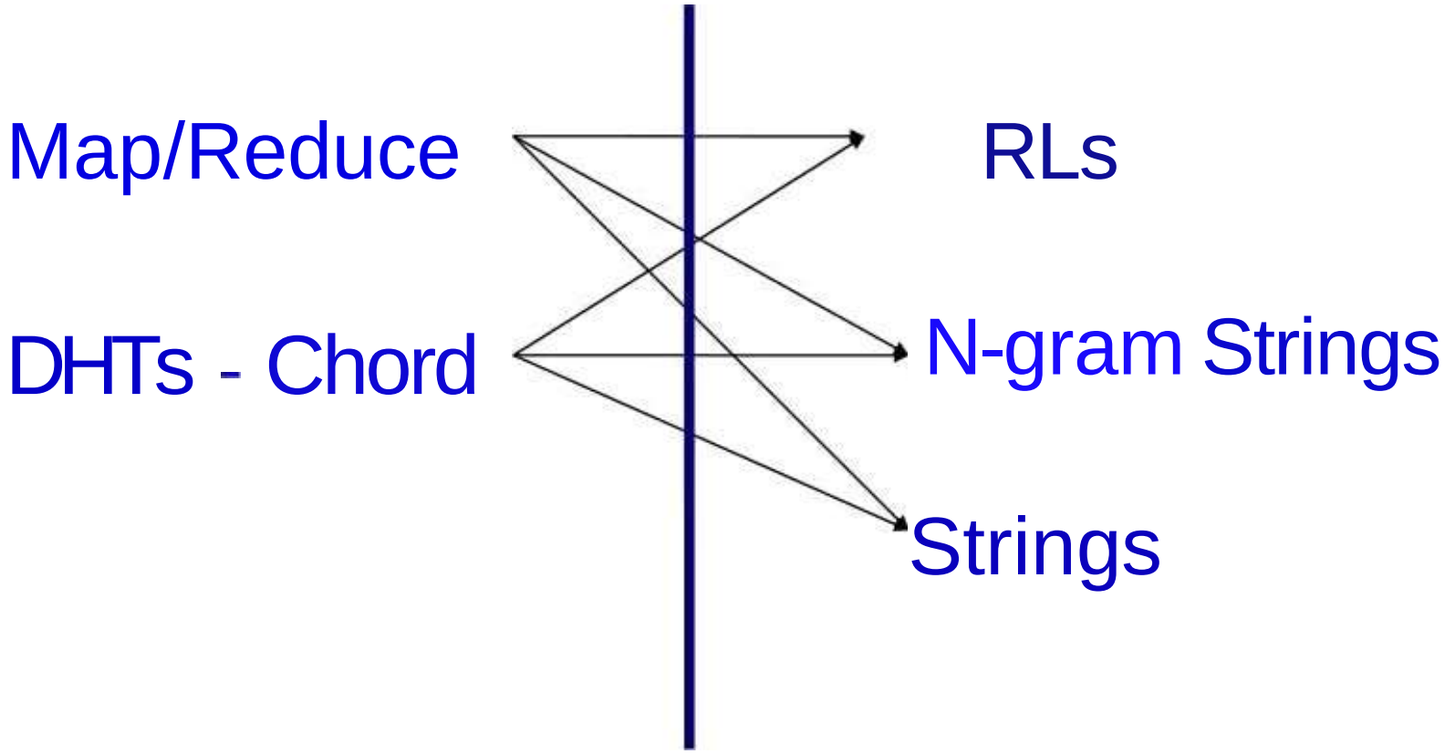
. HadoopDB with MonetDB instead of PostgreSOL

- read the HadoopDB paper
- download HadoopDB
- hook in MonetDB
- define a benchmark
- do experiments
- write an excellent report!

  implementation details+experiments

The way do the labwork...

- Combine something of the following:



M5 module

DHTs

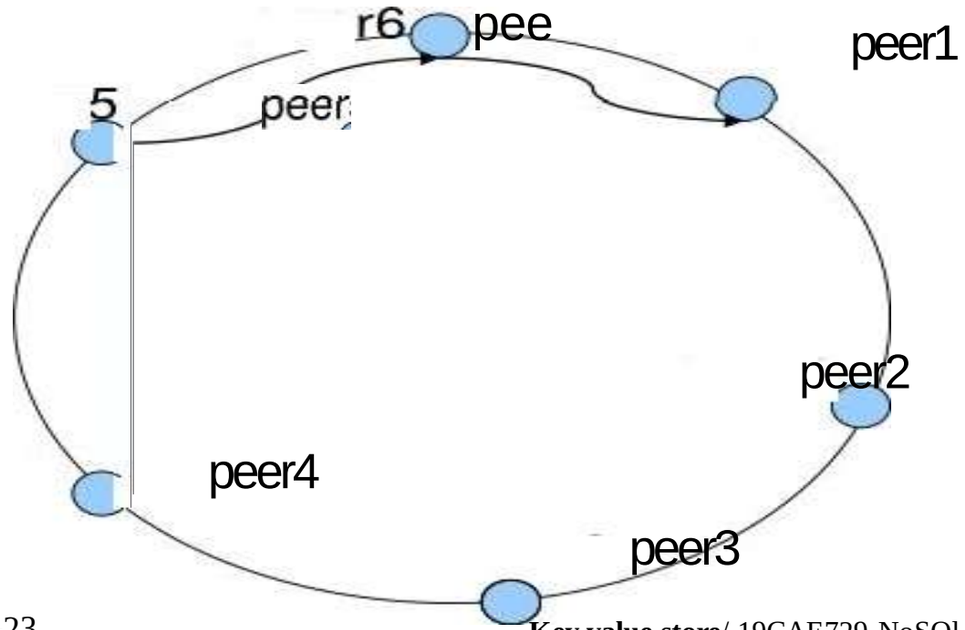
- a traditional key-value store based on a distributed hash table
- there are no master nodes
- keys are distributed to nodes according to a hash function
- values are retrieved with O(logN) messages by employing routing tables

Chord protocol

- keys are assigned an identifier: hash(key)
- peers are assigned an identifier: hash(IP)
- store and retrieve pairs of (key, data): lookup(key)

Each peer maintains a routing table (finger table) to route lookups

V+2^0	peer2
V+2^1	peer4
V+2^2	peer5



Chord Ring modulo 2^m

The [not so] fun way to do the work

- Design the BAT representation of distributed Key-Value store module
- Develop a wrapper around it to (simulate) parallel behavior
- define a benchmark
- do experiments
- write an excellent report!

  implementation details + experiments

SiteStat Basic event

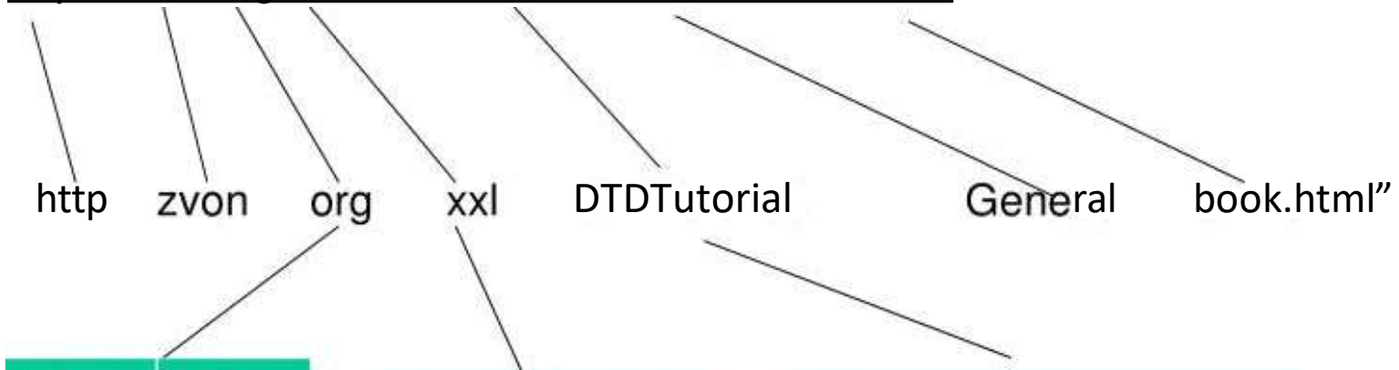
ns utc=1117835999527&
Time=86399527&
type=view&
ns m2 no& name—
statistics.basics& ip
213.46.1 53.0&
ns site cookie=426EA62C01 D400FA&
agent=Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1 ; SV1 ; .NET CLR 1.1.4322)&
ns pageurl—http://www.nedstatbasic.net/s%3Ftab%3D1%26link%3D1%26id^O 3D3165999
&ns_js=yes&
referrer=http%3A//www.nedstatbasic.net/s%3Ftab%3D1%26link%3D1%26id%3D3165999
&lang=nl
&secure-no
&id=3165999&
Pv—2& cntry—
NL&
language=NL

Dealing with formatted key strings

http://zvon.org/HowTo/Output/index.html"
http://zvon.org/index.php"
http://zvon.org/meta/RSS/Output/zvon.html"
http://zvon.org/other/charSearch/PHP/search.php"
http://zvon.org/other/PerlTutorial/Output/index.html"
http://zvon.org/other/python/PHP/search.php"
http://zvon.org/search.php"
http://zvon.org/xxl/ATAG1.0/Output/index.html"
http://zvon.org/xxl/DOM2reference/Output/index.html”
http://zvon.org/xxl/DTDTutorial/General/book.html"

rl box

http://zvon.org/xxl/DTDTutorial/General/book.html"



Oid	Value
1	org
2	com
3	net

Oid	Value
1	xxl

Oid	value
1	DTDtutorial
1	DOM2reference

Insert 1000 urls -> 2 seconds

```
[ 0,  "urlbox 0",   1,   1   ]
[ 1,  "urlbox 1",   270,     270     ]
[ 2,  ”urlbox 2",   236,     219     ]
[ 3,  ”urlbox 3",   180,     173     ]
[ 4,  "urlbox 4",   136,     122     ]
[ 5,  "urlbox S",   56,  48  ]
[ 6,  "urlbox 6",   34,  32  ]
[ 7,  "urlbox 7",   5,   5   ]
```

N-gram indexing

http://zvon.org/xxl/DTDTutorial/General/book.html"

Break the string in overlapping n-grams

Oid	value
1	Tuto
1	utor
1	tori

Use a single table per n-gram + wildcards

Oid	value
1	Tuto

Oid	value
1	utor

Oid	value
1	tori


ThankYou