MapReduce:
A Flexible Data Processing Tool

Nick Petty

Original paper: Jeffrey Dean and Sanjay Ghemawat

Agenda

MapReduce defined

Basic example

In use

Applications

Strengths and Weaknesses

Summary of paper

Compared to parallel DBMS

Personal experience

Conclusion

Discussion

What is MapReduce?

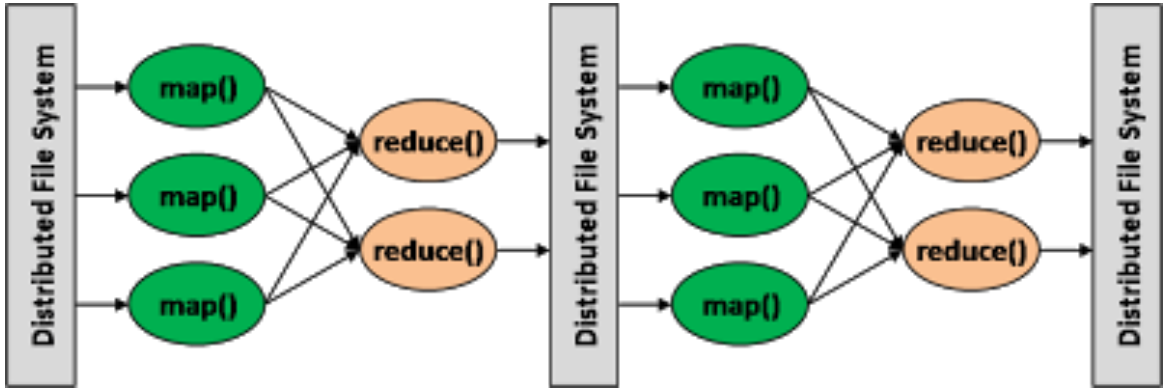
A programming model for processing and generating large data sets

Two main components:

1. Map function – process a key-value pair to generate a set of intermediate key-value pairs
2. Reduce function – merge all intermediate values associated with the same intermediate key

Shuffling and sorting – intermediate steps where data is moved from Map to Reduce

Designed for parallel, distributed queries on big data



Example pseudocode

MapReduce program for counting the number of occurrences of each word in a large collection of documents:

```
map(String key, String
value):
// key: document name
// value: document contents
for each word w in value:
    EmitIntermediate(w, “1”);
```

```
reduce(String key, Iterator
values):
// key: a word
// values: a list of counts
int result = 0;
for each v in values:
    result += ParseInt(v);
Emit(AsString(result));
```

MapReduce in use

Large-scale graph, image, and text processing

- Querying social network data sets
- Stitching satellite images together and removing seams

Machine learning

- Large data sets are needed for pattern recognition

Inverted indices

- Maps content to its location in a database
- Fast to search, slow to add and update

Logging

- Applications record events in timestamped logs
- Tracing issues requires navigating millions of such events

MapReduce applications

Hadoop

- Popular open-source implementation
- Currently managed by Apache Software Foundation
- Includes Hadoop Distributed File System (HDFS) and other management tools

Google

- Previously used MapReduce to index the web
- More than 10,000 programs at Google used MapReduce
- Company has moved on to other technologies

Other MapReduce applications

CouchDB

- Web-focused with JSON, JavaScript, HTTP, and concurrency
- Database is a collection of documents, not relational tables
- A Map function creates “views” which are indexed for queries

Riak

- Fault-tolerant distributed data storage
- MapReduce in JavaScript and Erlang for queries
- Enterprise versions supported by Basho Technologies

Considerations

Strengths

Easy, cost-effective deployment

Large data sets

Complicated queries

Fault-tolerance

Storage independence

Weaknesses

Flexibility: only Map → Reduce

No complex schema

Not for data storage

Not designed for speed

Summary of the paper

Criticism of another paper that compared MapReduce to parallel databases

Primarily asserts that previous research was not done correctly

Primary counterpoints:

- Indices can be used
- Input and output is not limited to files and textual data
- Implementation was not optimized
- Data loading is not a MapReduce feature
- Complicated expressions are often easier with MapReduce
- Push vs. pull model

Comparing MapReduce and parallel databases

MapReduce

Data is only processed

Easier for complicated queries

Functions are adapted to datasets

Parallel database

Data is processed and stored

Faster for simple queries

Schemas allow data sharing across applications

My work with MapReduce technology

Circuit by Unify

Hadoop

Solr

Cassandra

Elasticseach

Closing remarks

MapReduce works on a simple key-value pair system

1. Map function breaks up and organizes data
2. Supporting systems distribute and move the data and tasks
3. Reduce function combines results into query response

Best for complicated queries on large datasets

Designed for low-cost, high-fault hardware

Not a database, just an analytic process

Popularized by Google, now a major open-source project

Questions and Answers

Thank you