A comparison of approaches to large scale data analysis

A. Pavlo, et al., SIGMOD, 2009

Presentation by Atreyee Maiti

Motivation

- MapReduce: A major step backwards?
    - basic control flow of this framework has existed in parallel DBMS for over 20 years
    - parallel DBMS provide a high-level programming environment and parallelize readily
    - possible to write almost any parallel processing task as either a set of database queries or a set of MapReduce jobs
- An attempt to evaluate in terms of performance and development complexity
- Provide a systematic analysis of the design choices made in these two paradigms and the repercussions of those

Approach to analysis

- Benchmark consisting of a collection of tasks run
- Measure each system's performance for various degrees of parallelism on a cluster of 100 nodes

hadoop

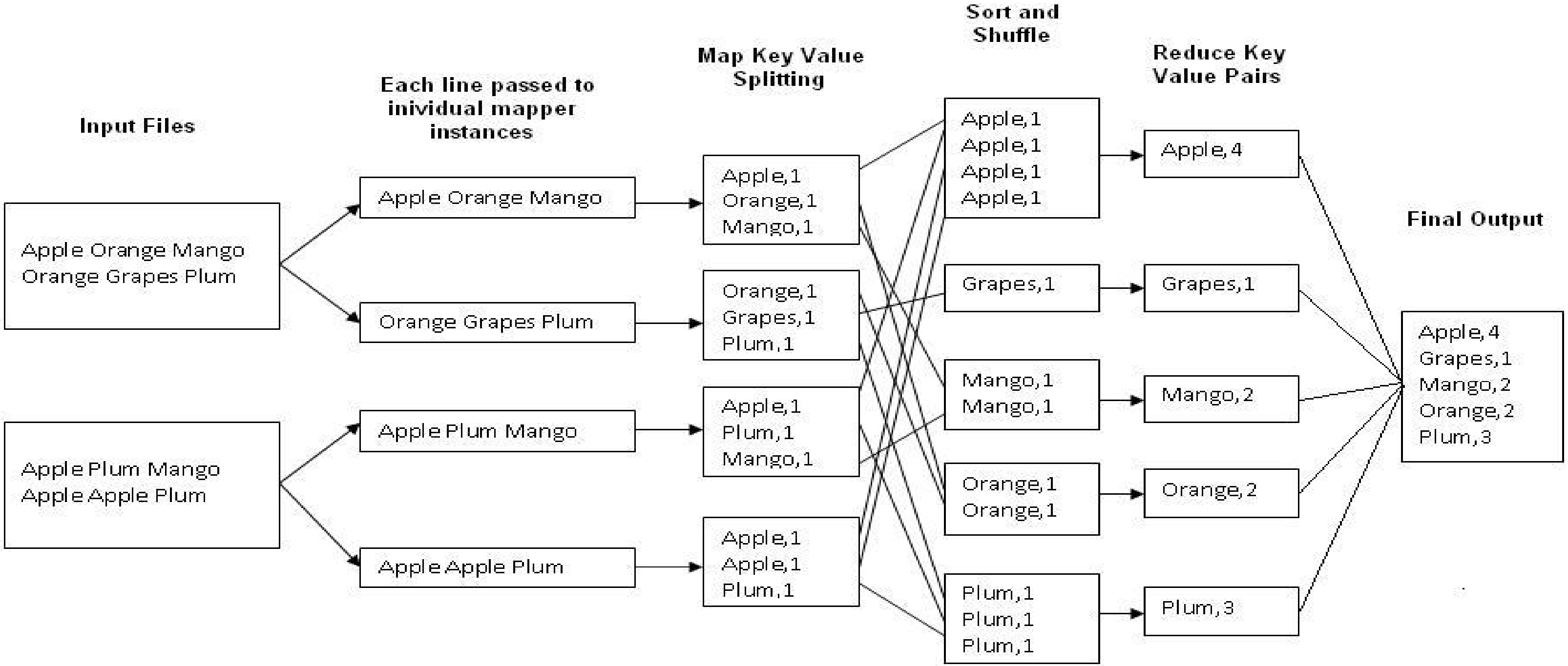
vs

hp

VERTICA

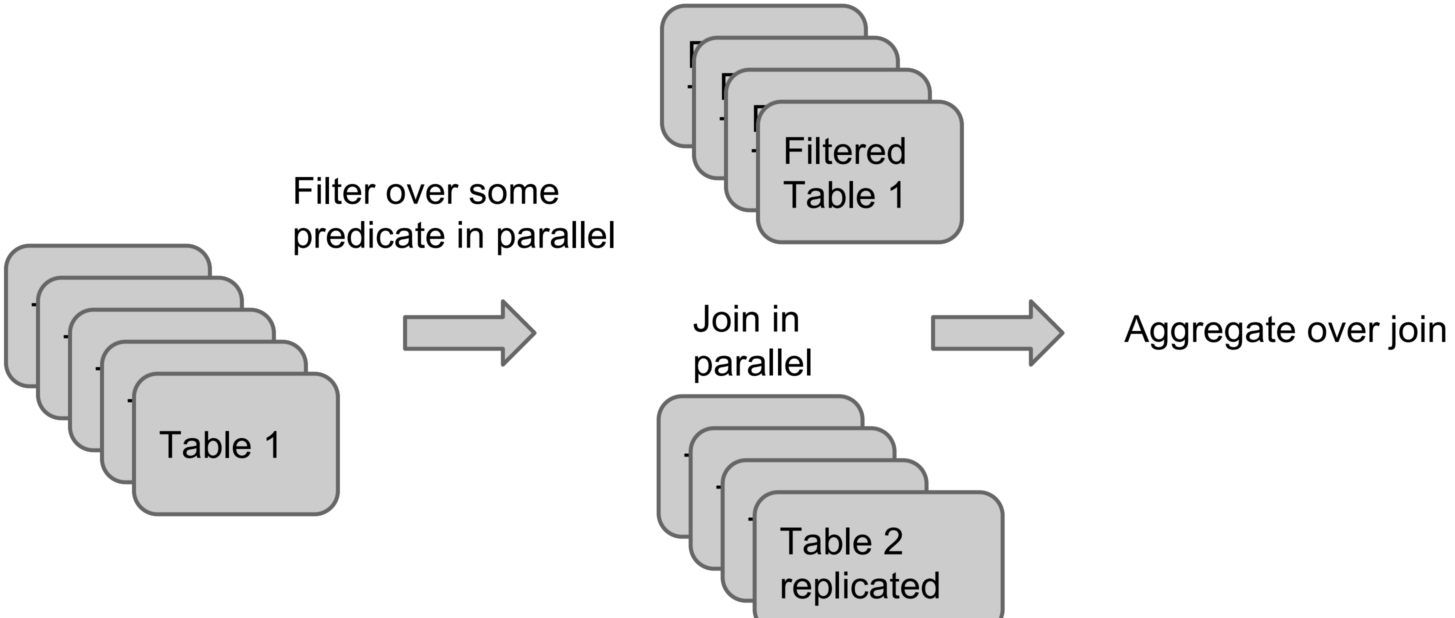
He who
must not
be named
;)

Map Reduce


Input Files
Each line passed to inividual mapper instances
Map Key Value Splitting
Sort and Shuffle
Reduce Key Value Pairs
Final Output
Apple Orange Mango
Orange Grapes Plum
Apple Plum Mango
Apple Apple Plum
Apple Orange Mango
Orange Grapes Plum
Apple Plum Mango
Apple Apple Plum
Apple,1
Orange,1
Mango,1
Orange,1
Grapes,1
Plum,1
Apple,1
Plum,1
Mango,1
Apple,1
Apple,1
Plum,1
Apple,1
Apple,1
Apple,1
Apple,1
Grapes,1
Mango,1
Mango,1
Orange,1
Orange,1
Plum,1
Plum,1
Plum,1
Apple,4
Grapes,1
Mango,2
Orange,2
Plum,3
Apple,4
Grapes,1
Mango,2
Orange,2
Plum,3

Parallel Databases

- Tables are partitioned over the nodes in a cluster
- System uses an optimizer that translates SQL commands into a query plan whose execution is divided amongst multiple nodes



Architectural elements

	Parallel databases	Map reduce frameworks
Schema Support	Data needs to conform to the relational paradigm	Schema-free. need for a custom parser in order to derive the appropriate semantics for their input records. requires discipline. when no sharing is anticipated, the MR paradigm is quite flexible.
Indexing	hash or Btree indexing reduces the scope of the search dramatically. Most database systems also support multiple indexes per table.	do not provide built-in indexes.

	Parallel databases	Map reduce frameworks
Programming Model	State what you want	one is forced to write algorithms in a low-level language in order to perform record-level manipulation. there is widespread sharing of MR code fragments to do common tasks, such as joining data sets. To alleviate the burden of having to re-implement repetitive tasks, the MR community is migrating high- level languages on top of the current interface to move such functionality into the run time.
Data distribution	send the computation to the data	data passed onto the next stages of the computation

	Parallel databases	**Map reduce frameworks**
Execution Strategy	push mechanism to transfer data (no materialization of the split files)	pull mechanism to draw in input files - induces large disk seeks
Flexibility	programming environments like RoR allow developers to benefit from the robustness of DBMS technologies without the burden of writing complex SQL	SQL does not facilitate the desired generality that MR provides.

	Parallel databases	**Map reduce frameworks**
Fault tolerance	larger granules of work (i.e., transactions) that are restarted in the event of a failure.	if a unit of work fails, then the MR scheduler can automatically restart the task on an alternate node.

Experiments carried out

- Original MR task - grep task - representative of MR use cases
  - Loading
  - Execution
- Analytical tasks - HTML documents processing similar to web crawler
  - Loading
  - Selection
  - Aggregation
  - Join
  - UDF Aggregation
- Both DBMS-X and Vertica execute most of the tasks much faster than Hadoop at all scaling levels.

Findings

Loading time

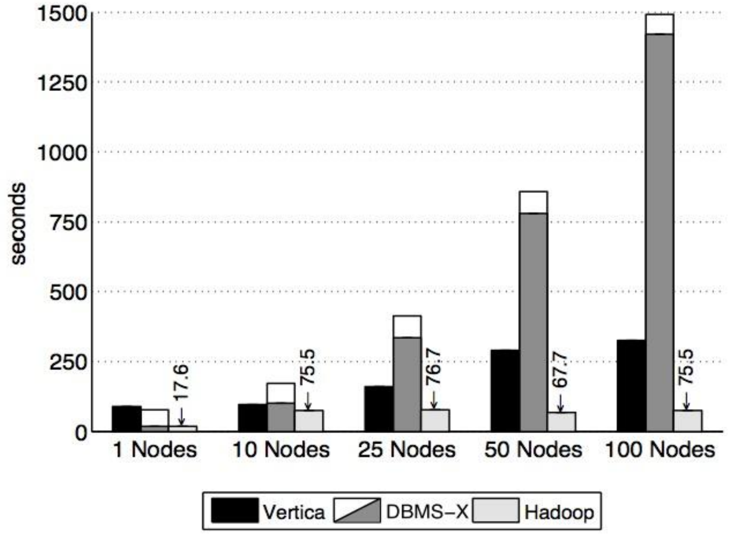


Figure 1: Load Times – Grep Task Data Set (535MB/node)

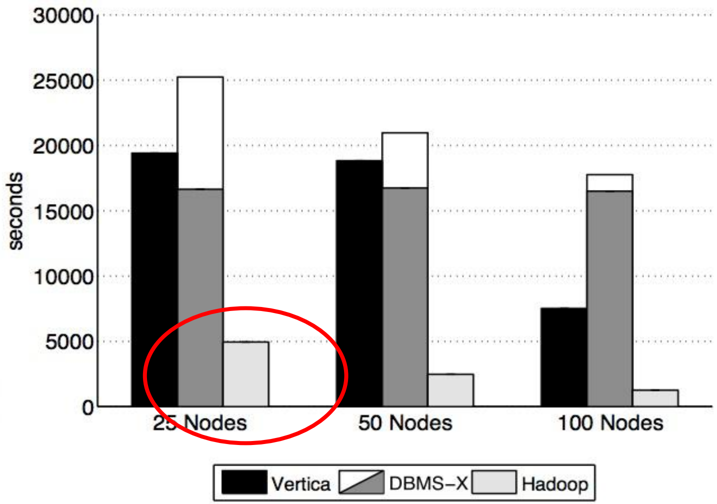


Figure 2: Load Times – Grep Task Data Set (1TB/cluster)

Task execution time

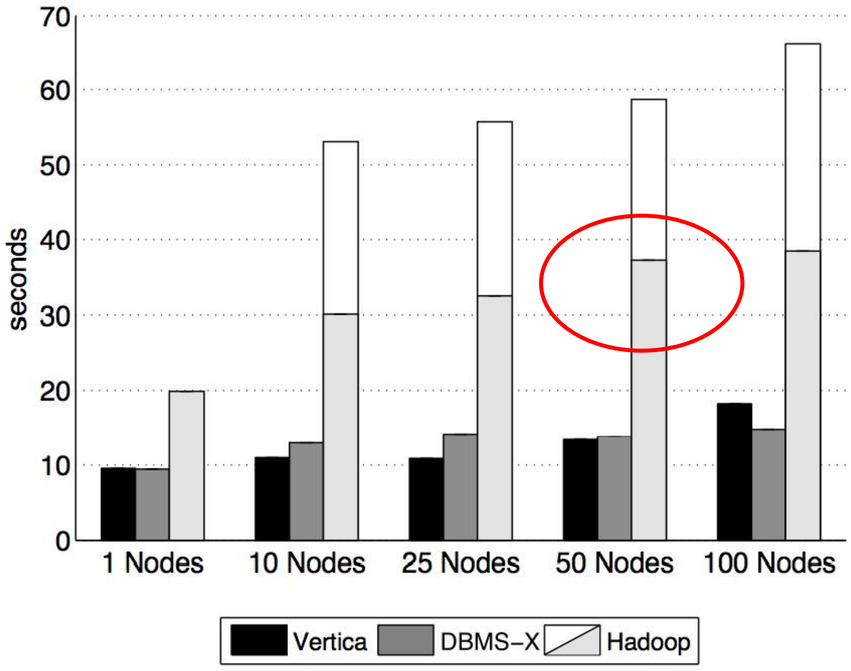


Figure 4: Grep Task Results – 535MB/node Data Set

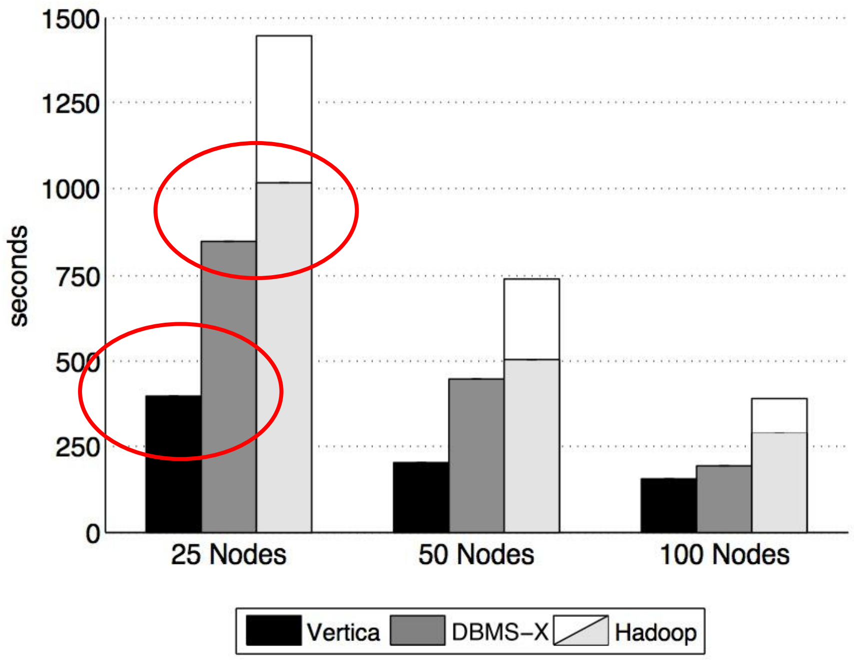


Figure 5: Grep Task Results – 1TB/cluster Data Set

Analytical tasks

Documents, UserVisits and Rankings tables

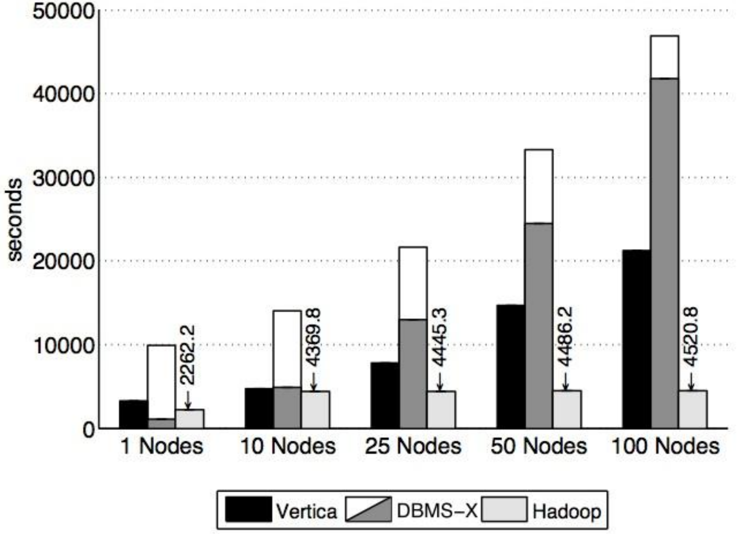


Figure 3: Load Times – UserVisits Data Set (20GB/node)

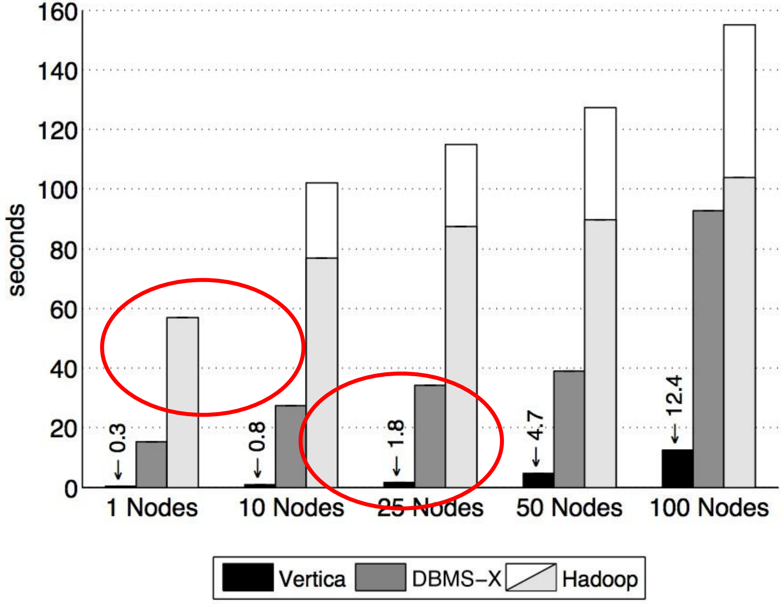


Figure 6: Selection Task Results

Aggregation task

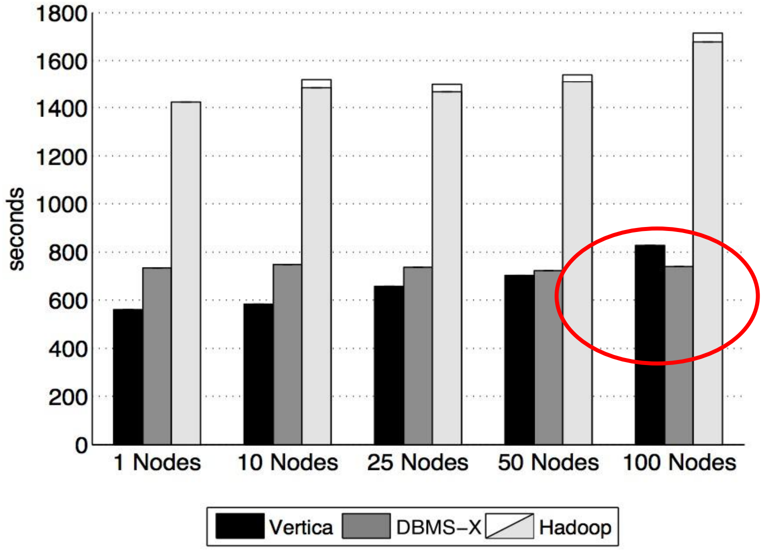


Figure 7: Aggregation Task Results (2.5 million Groups)

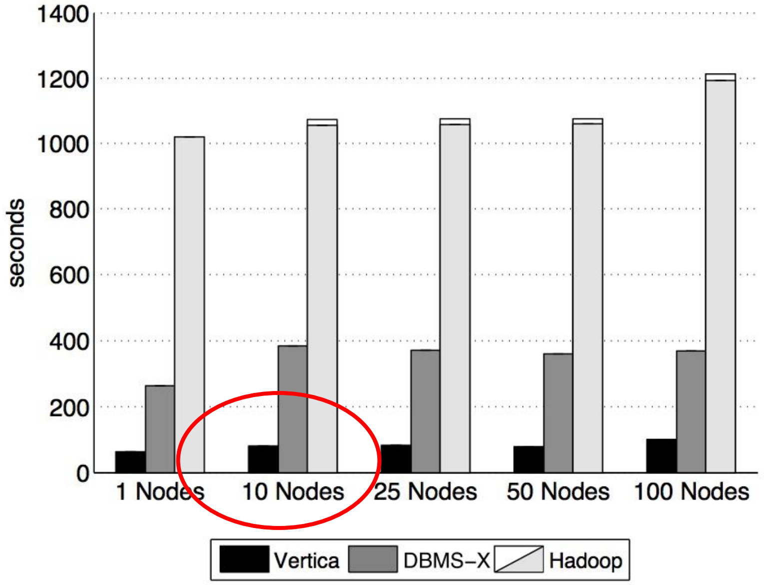


Figure 8: Aggregation Task Results (2,000 Groups)

Join and UDF

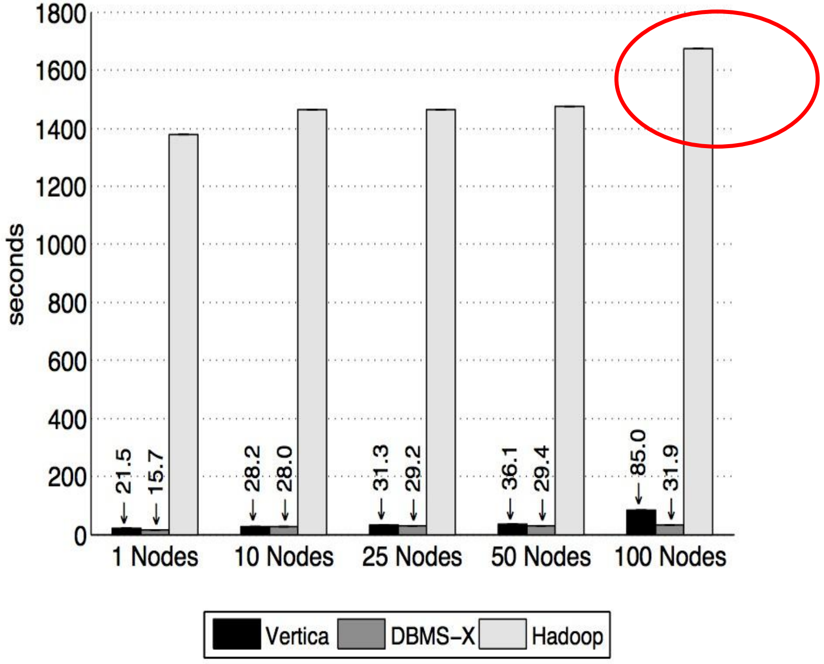


Figure 9: Join Task Results

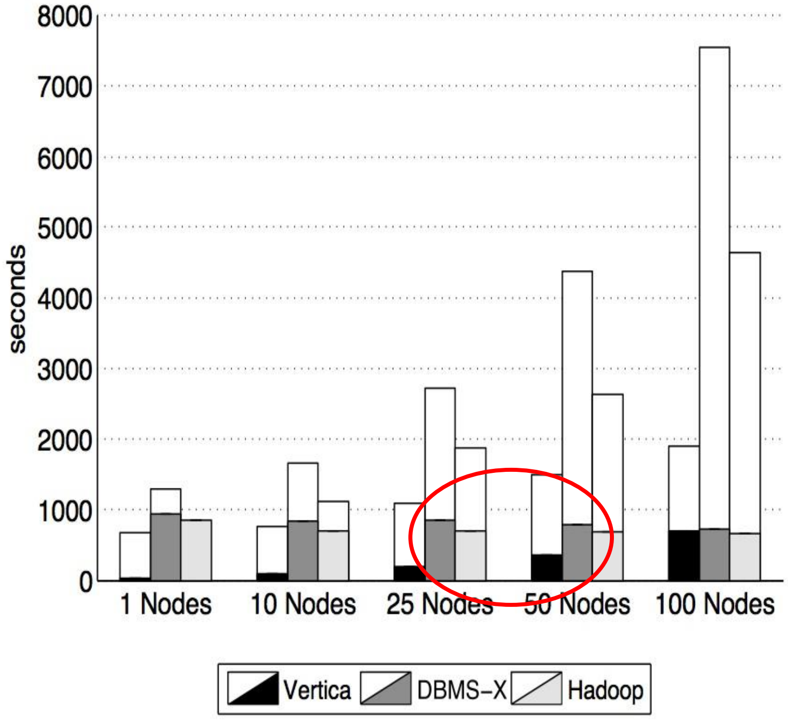


Figure 10: UDF Aggregation Task Results

Analysis of the results

System level aspects

- System Installation, Configuration, and Tuning
- Task Start-up
- Compression
- Loading and Data Layout
- Execution Strategies
- Failure Model

User level aspects

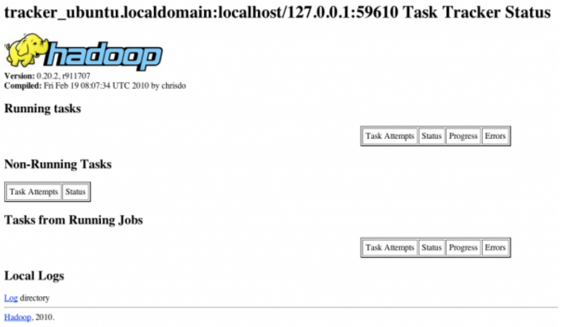
- Ease of use
- Additional tools

- DBMS-X was 3.2 times faster than MR and Vertica was 2.3 times faster than DBMS-X.
- Parallel DBMS-X lesser energy needs.
- B-tree indices, novel storage mechanisms, aggressive compression techniques and sophisticated parallel algorithms for querying large amounts of relational data.
- Hadoop has upfront cost advantage - hence attracted such a large user community.
- Extensibility is USP of MR
- Fault tolerance of MR
- It comes with a potentially large performance penalty, due to the cost of materializing the intermediate files between the map and reduce phases.
- SQL is particularly bad
- MR makes a commitment to a “schema later” or even “schema never” paradigm. But this lack of a schema has a number of important consequences. This difference makes compression less valuable in MR and causes a portion of the performance difference between the two classes of systems.

Where are we now?

Databases with mapreduce support

riak


tracker_ubuntu.localdomain:localhost/127.0.0.1:59610 Task Tracker Status

Version: 0.20.2, r911707
Compiled: Fri Feb 19 08:07:34 UTC 2010 by chrisdo

Running tasks

Task Attempts	Status	Progress	Errors

Non-Running Tasks

Task Attempts	Status

Tasks from Running Jobs

Task Attempts	Status	Progress	Errors

Local Logs

Log directory

Hadoop, 2010.

Better interfaces for MR

Embracing both

SCOPE from Microsoft

Greenplum

Summary

- Different paradigms with areas where each of these shine
- Need for more maturity and tools for MR. Work in progress

References

http://www.cs.cmu.edu/~pavlo/courses/fall2013/static/papers/benchmarks-sigmod09.pdf
http://vgc.poly.edu/~juliana/courses/cs9223/Lectures/paralleldb-vs-hadoop.pdf
http://cacm.acm.org/magazines/2010/1/55743-mapreduce-and-parallel-dbmss-friends-or-foes/fulltext
http://www.datanami.com/datanami/2013-02-05/weighing_mapreduce_against_parallel_dbms.html
http://homes.cs.washington.edu/~billhowe/mapreduce_a_major_step_backwards.html
http://research.google.com/archive/mapreduce-osdi04-slides/index-auto-0032.html