Relational Joins in Hadoop:
A module for Cloud9lib MapReduce library.

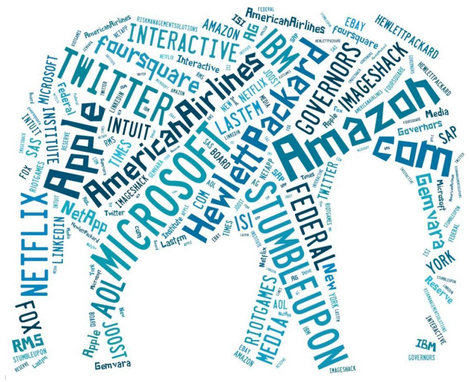
Mohamed Mohideen Abdul Rasheed

Big Data

Data is Everywhere: Information is accumulated at a staggering rate everywhere from small startups to massive corporations.

More Data = Better Insights: Research provides evidence that simple algorithms running on large data outperforms sophisticated algorithms running on smaller data set.

Business Intelligence: It is a widely accepted fact that these accumulated data (such as user logs) has the key to identify crucial business opportunities and will guide the companies make right decisions.



Hadoop

Hadoop is an open source implementation of the **MapReduce** – a framework for tackling data-intensive applications.

Widely adopted Big Data tool.
(Facebook, Amazon, Microsoft,…)
Active community for getting support

Less Investment
Open Source
Runs on commodity hardware

Great Data Capabilities
Can handle data in the PB scale
Support for unstructured and semi-structured data sets.

Deliverables

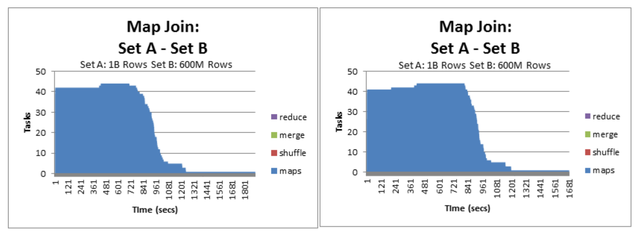
Working Relational Join Module
- Availble for download at https://github.com/mohideen/cloud9

Experimentation Report & Documentation
- Available at http://mohideen.github.com/Cloud9/

Performance Tuning

Hadoop job configuration can be tuned to set internal sort properties appropriate to application characterstics to gain perfomance.

Chart showing time taken before (left) and after (right) perfomance tuning.



GOALS

- Learn Map-Reduce programming
- Understand Hadoop Framework
- Contribute meaningfully to a project

To deliver a relational join module for UMD's Cloud9lib, an open source Hadoop library, developed by Dr. Jimmy Lin.

Relational Joins

"Relational Joins combine different data structures together and gain meaningful information from them."

Relational Joins are essential for any **data warehousing** application.

Hadoop Relational Join Techniques:
- Reduce-Side Joins
- Map-side Joins
- Memory Backed Joins
- Memcached Joins

Techniques Implemented

Reduce-Side Joins – Caters applications with unorganized data. Less efficient. Suitable for Ad-hoc necessities.

Map-side Joins – Efficient technique but requires sorted-partitioned input data.

Experimentation on the cluster

Bespin: A 13-node UMIACS Hadoop Cluster with 78 Map Slots and 52 Reduce Slots.

Test Data:
Size ranging from 10 Milllon to 1 Billion rows.
Uniform and Zipf distributions

Map Join vs Reduce Join

Set X: 1 Billion Rows (2 Integer & 3 String Fields) Uniform Distribution
Set Y: 600 Million Rows (3 Integer Fields) Uniform Distribution

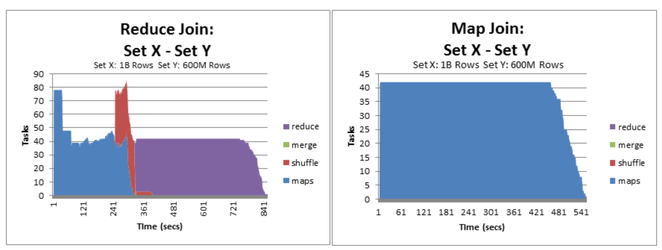


Chart showing job durations for joins on same dataset pairs with increasing the number of tasks.