Popular Open Source Data Processing Frameworks

Giri Iyengar

Cornell University

gi43@cornell.edu

April 23, 2018

Agenda for the week

- Pig
- Spark
- Storm
- BlinkDB
- Druid

Overview

Hadoop

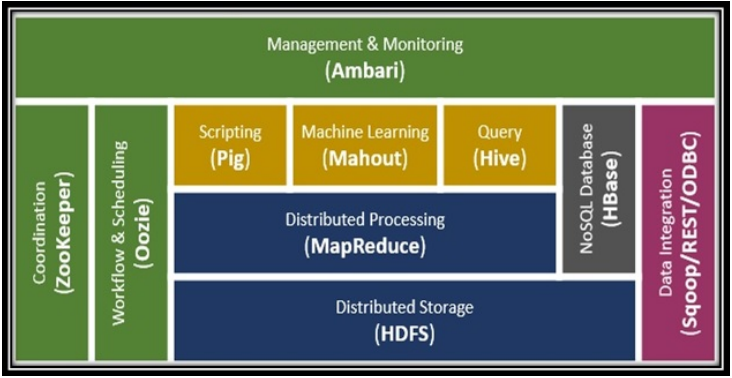


Figure: Hadoop Ecosystem

Hadoop

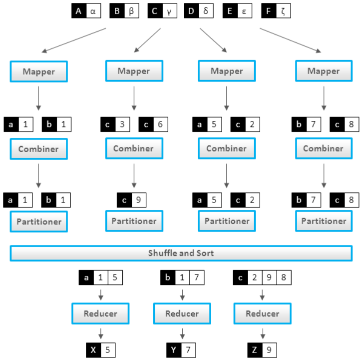


Figure: MR Framework

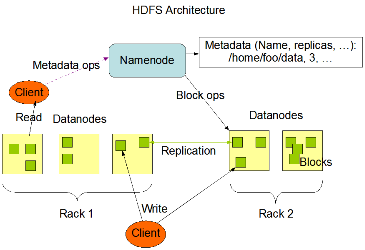


Figure: HDFS

Pig Demo

Apache > Hadoop > Pig > docs > r0.17.0

Project | Wiki | **Pig 0.17.0 Documentation**

- **Pig**
  - Overview
  - Getting Started
  - Pig Latin Basics
  - Built In Functions
  - User Defined Functions
  - Control Structures
  - Shell and Utility Commands
  - Performance and Efficiency
  - Testing and Diagnostics
  - Visual Editors
  - Administration
  - Index
- **Miscellaneous**

Getting Started

- Pig Setup
  - Requirements
  - Download Pig
  - Build Pig
- Running Pig
  - Execution Modes
  - Interactive Mode
  - Batch Mode
- Running jobs on a Kerberos secured cluster
  - Short lived jobs
  - Long lived jobs
- Pig Latin Statements
  - Loading Data
  - Working with Data
  - Storing Intermediate Results
  - Storing Final Results
  - Debugging Pig Latin
- Pig Properties
- Pig Tutorial
  - Running the Pig Scripts in Local Mode
  - Running the Pig Scripts in Mapreduce Mode, Tez Mode or Spark Mode
  - Pig Tutorial Files
  - Pig Script 1: Query Phrase Popularity
  - Pig Script 2: Temporal Query Phrase Popularity

Figure: Pig Reference Manual

Hadoop

- Use commodity hardware to achieve super stable, reliable, data processing
- Reliable data storage via replication - HDFS
- Generic Computation approach - Map Reduce
- Extremely well-suited for batch processing on 1000s of nodes handling Petabytes of data

Since the advent of Hadoop

- Speed and sophistication required for data processing has grown tremendously
- Complex algorithms like Machine Learning and Graph Analysis are much more common
- E.g. ML requires multiple passes over the data – not suited for Map Reduce style computing
- Streaming analysis of real-time data is increasingly important
- Both one-pass aggregations and multi-pass analysis applications need to be supported

Overview

1. Pig
2. Spark

Spark

Apache Spark

Created by Matei Zaharia as part of his Doctoral work at UC Berkeley. Designed to address some of the limitations observed with Hadoop.

Stated goal is to scale to 10s of thousands of compute nodes

Spark Stack

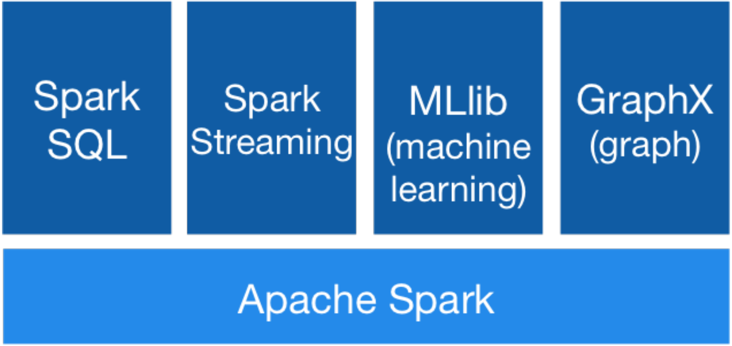


Figure: Spark Stack

Spark Cluster Overview

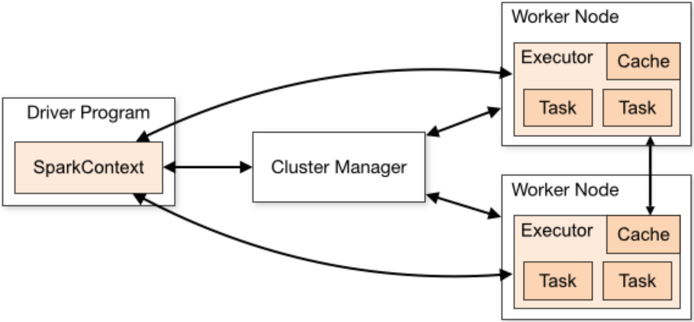


Figure: Spark Stack

Spark RDD

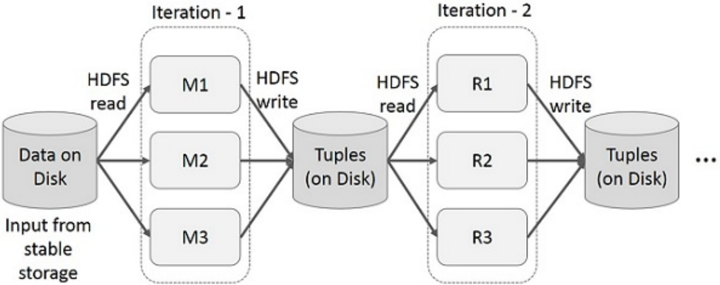
Resilient Distributed Dataset

Spark revolves around the concept of a resilient distributed dataset (RDD), which is a fault-tolerant collection of elements that can be operated on in parallel. There are two ways to create RDDs: parallelizing an existing collection in your driver program, or referencing a dataset in an external storage system, such as a shared filesystem, HDFS, HBase, or any data source offering a Hadoop InputFormat.

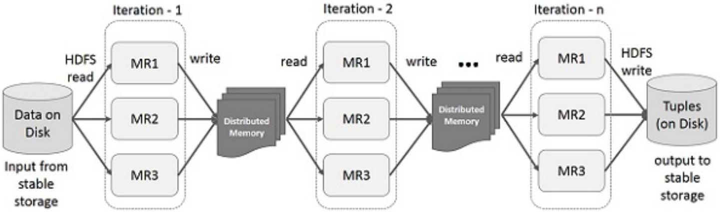
RDD

- Basic Abstraction in Spark
- Immutable, Partitioned collection of elements that can be operated in parallel
- Supports Lazy evaluations

Map Reduce Intermediate Data



Spark Intermediate Data



Properties of RDD

- In-Memory
- Lazy
- Fault-Tolerant
- Immutability
- Partitioned
- Persistent
- Parallel

- Location-Stickiness
- Typed
- Coarse-Grained Operations (whole RDD and not individual elements)
- No limitations (bound by available system memory)

Spark DataFrames

Spark DataFrames

This API is inspired by data frames in R and Python (Pandas), but designed from the ground-up to support modern big data and data science applications. It is an extension to the existing RDD API.

Spark DataFrames

- Ability to scale from kilobytes of data on a single laptop to petabytes on a large cluster Support for a wide array of data formats and storage systems
- State-of-the-art optimization and code generation through the Spark SQL Catalyst optimizer
- Seamless integration with all big data tooling and infrastructure via Spark
- APIs for Python, Java, Scala, and R


рахмат
danke
謝謝
ngiyabonga
teşekkür ederim
Баярлалаа
спасибо
thank you
gracias
tapadh leat
dank je
mersi
barka
welalin
tack
spas
vinaka
blagodaram
misaotra
matondo
paldies
grazzi
mahalo
хвала
asante
manana
obrigada
murakoze
tenki
faafetai lava
kiitos
dankie
dhanyavad
bedankt
nanni
nandi
bayarlalaa
gracie
hvala
mauruuru
köszönöm
enkosi
dziękuję
mochchakkeram
djiere dieuf
tau
mamnun
дякую
chnorakaloutioun
gratias ago
gràcies
sulpày
taiku
go raibh maith agat
obrigado
sobodi
děkuji
mesi
sagolun
sukriya
kop khun krap
arigatô
takk
dakujem
trugarez
didi madloba
najis tuke
ありがとう
tanemirt
grazie
shukriya
мерси
kam sah hamnida
terima kasih
rahmet
diolch
dhanyavadagalu
merce
rahmat
감사합니다
xièxie
ευχαριστώ
merci