Big Data Analytics and Visualization
DS5008 Big Data Analytics
CS6011 Data Analysis and Visualization

Omar Usman Khan
omar.khan@nu.edu.pk

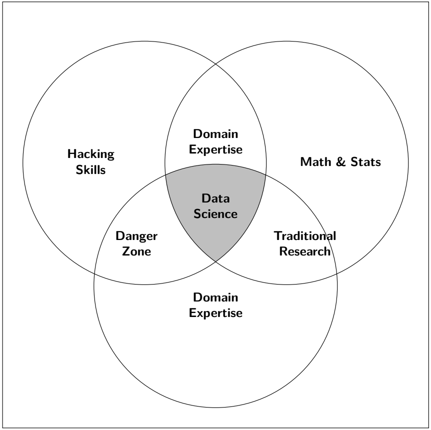
Department of Computer Science
National University of Computer & Emerging Sciences, Peshawar

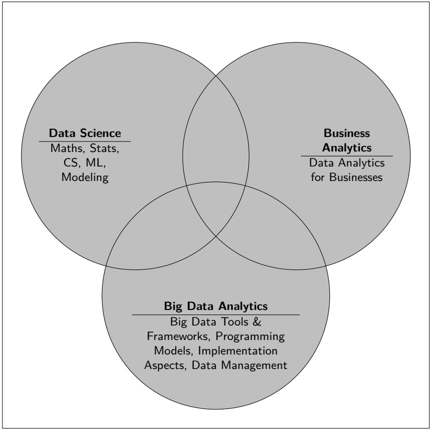
April 22, 2022

Syllabus

1. Overview
   - Industry
   - This Course
2. Analytics
   - Preparedness
   - Big Data Stack
   - R Overview
   - Data Analytics
3. Hadoop
   - HDFS
   - HDFS API's
   - Map Reduce Framework
4. Spark

Big Data in Context of Data Science





W. Cleveland, *Data Science: An Action Plan for Expanding the Technical Areas of the Field of Statistics*, in InternationaL Statistical Review, 69(1), 2001

Overview

Understanding the Scale

- New York Stock Exchange: 1 TB / Day
- Large Hadron Collider: 15 PB / Year
- Internet Archive: 2 PB stored, Growing @ 20 TB / month
- Surveillance Drones (Multiple cameras, several TB / minute)
- Facebook, Google, Twitter, Instagram, Youtube, ...
- ...

- Amount of Data Overwhelms Analysts & Decision Makers

Characteristics

Collections of datasets whose **Volume**, **Velocity**, and **Variety** is so large that it is difficult to **Store**, **Manage**, **Process**, and **Analyze** the data using traditional databases or processing tools.

Job Prospects

P@sha IT Salary Survey 2017

Job Title	Salary (Entry)	Salary (Medium)	Salary (Senior)
(Big) Data Engineer, Business Analyst, Data Analyst	74,110	118,115	144,770
Data Scientist	64,674	119,478	233,470

Sample Entry Requirements

- Technology expertise of solutioning in Hadoop, Hive, Spark / PySpark, SQL, Oozie along Data Modelling in Hive
- Expertise in programming Languages- Java / Python / Scala
- Ability to demonstrate micro / macro designing and familiar with Unix Commands and basic work experience in Unix Shell Scripting
- ...

Job Prospects (cont.)

Companies

- IBM
- Afiniti
- Telecom (Mobilink, Telenor, Zong)
- Banks (Allied)
- Logistics (Careem, Bykea, Keep Trucking)
- 10Pearls, i2c
- ...

Job Prospects (cont.)

Number of Jobs

Job Role	Pakistan	Middle East	Europe	USA
Data Scientist	7	197	6011	6561
Data Analyst	15	160	4845	8564
Data Manager	0	11	1283	1255
Data Architect	4	27	1220	2184
Business Analyst	9	212	9595	15244
Financial Analyst	2	69	1194	6395
Big Data Engineer	3	24	603	674
Data Engineer	6	135	5894	6838
Machine Learning Specialist	1	3	46	14

Table 1: DS Jobs Worldwide (Source: LinkedIn, fetched October 13, 2020)

Job Prospects (cont.)

Figure 1: On a Lighter Note

Course Overview

Learning Outcomes

CLO1	Perform analytics on large scale datasets
CLO2	Deploy massive threading solutions on Parallel systems
CLO3	Deploy massive threading solutions on Distributed systems
CLO4	Be able to visualize Large Scale Multi-Dimensional Datasets
CLO5	Be familiar with algorithms and tools for mining massive datasets

Marks Breakdown

- Sessional I: 15%
- Sessional II: 15%
- Final Examination: 50%
- Assignments: 20% (including Takehome Lab Activities)

- HPC Setup with support for various parallel and distributed computing frameworks will be made available to all students for duration of semester (and beyond upon request)

Course Overview (cont.)

Generating Public Key for Assignments

- On Windows, Use https://winscp.net/eng/docs/ui_puttygen
- On Linux, use **ssh-keygen** command
- Share generated key with me by email.

Probability Distribution of Previous Grades

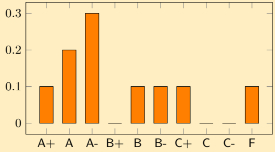


Figure 2: Spring 2019

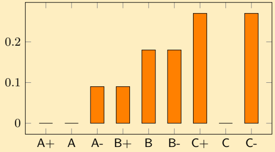


Figure 3: Spring 2020

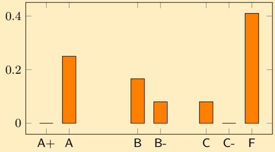
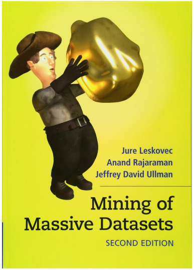
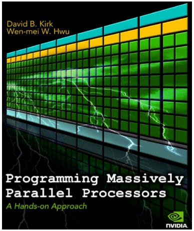
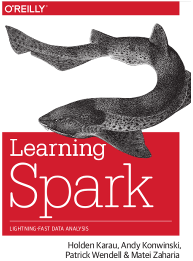


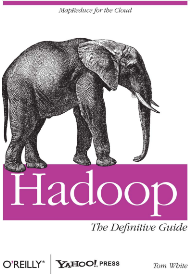
Figure 4: Spring 2021

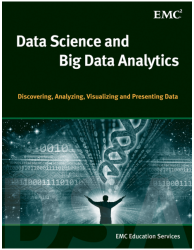
Books


Jure Leskovec
Anand Rajaraman
Jeffrey David Ullman
Mining of Massive Datasets
SECOND EDITION


David B. Kirk
Wen-mei W. Hwu
Programming Massively Parallel Processors
A Hands-on Approach
NVIDIA


O'REILLY
Learning Spark
LIGHTNING-FAST DATA ANALYSIS
Holden Karau, Andy Konwinski, Patrick Wendell & Matei Zaharia


MapReduce for the Cloud
Hadoop
The Definitive Guide
O'REILLY
YAHOO! PRESS
Tom White


EMC²
Data Science and Big Data Analytics
Discovering, Analyzing, Visualizing and Presenting Data
EMC Education Services

Data

Raw Data Sources

- **Logs**: Web applications/servers/daemons
- **Transactions**: E-Commerce/Banking/Financial
- **Social Media**: JSon
- **Databases**: RDBMS
- **Sensor Data**: IoTs/WSNs
- **Clickstream Data**: Patterns of user activities
- **Surveillance**: Sensors/Images/Video Data
- **Healthcare**: Sensors/Hospital Records
- **Network**: Info generated by Network Devices
- ...

Data Storage

- Issues: Disk Density, Access Times, Storage Formats
- Parallel Disk Access & Distributed Storage Solutions

Data (cont.)

Data Processing

- Combining Data (Parallel Access, Distributed Storage)
- HPC and Super-Computing Systems
- MPI, GPU Computing, . . .

Effect of Algorithm Styles

Which code is going to be the fastest?

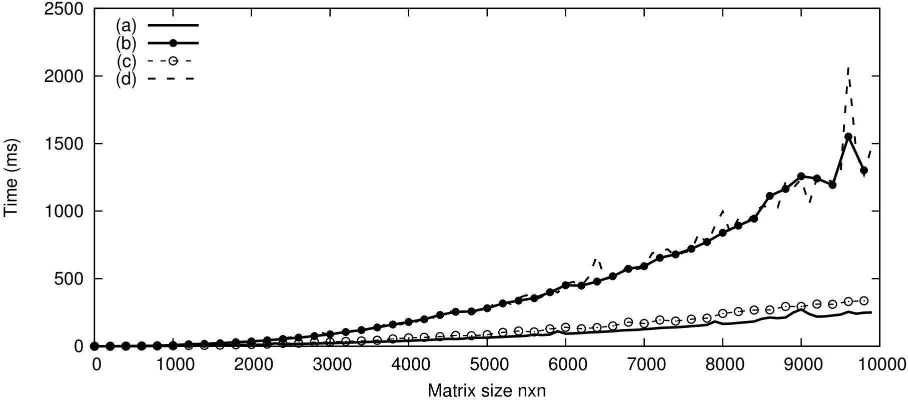
(a)
```
for (i = 0; i < n; i++)
  for (j = 0; j < n; j++)
    sum += a[i][j];
```

(b)
```
for (j = 0; j < n; j++)
  for (i = 0; i < n; i++)
    sum += a[i][j]
```

(c)
```
for (i = 0; i < n; i++)
  for (j = 0; j < n; j++)
    sum += a[i*SIZE+j];
```

(d)
```
for (j = 0; j < n; j++)
  for (i = 0; i < n; i++)
    sum += a[i*SIZE+j]
```

Effect of Algorithm Styles (cont.)



Clock Cycles

- What is a single clock cycle worth?
  - **xchg**: Exchange values in two registers
  - **push**: Push operation using registers
  - **pop**: Pop operation using registers
  - **add**,**sub**: 3 additions/subtractions involving registers
  - **cmp**: 3 comparisons involving registers
  - **mul**: half a fp multiplication involving registers

Where are We Today?

- Clock rates 40 MHz (MIPS R3000 1988) $\rightarrow$ 4.0 GHz (Intel Core-i7-4790K 2015), 4.4 GHz (Intel Xeon X5698 2015)
- Higher processor speed $\propto$ More **heat dissipated**
- Transistor has reached size of 32 nm (Generation 3 Core-i7). **Size limit**: How much more smaller can it get?
- Support for executing multiple instructions per clock cycle, fast cache technologies, superscalar architectures . . .

Ranking of Super Computers

Rank	Site	System	Cores	TFLOPs	Power (kWh)
1	Wuxi, China	TaihuLight: Sunway 1.45Ghz	10,649,600	125,435.9	15,371
2	Guangzhou, China	Tianhe-2: Intel Xeon	3,120,000	54,902	17,808
3	Oak Ridge Lab, USA	Titan: Cray Opteron, NVIDIA K20	560,640	27,112	8,209
4	DoE, USA	Sequoaia:, IBM BlueGene/Q	1,572,864	20,132	7,890
5	RIKEN Ins., Japan	K-Computer: Fujitsu SPARC64	705,024	11,280	12,660
6	DoE, USA	Mira: IBM BlueGene/Q	786,432	8,586.6	3,945
7	DoE, USA	Trinity: Cray XC40	301,056	11,078	-
8	Swiss S.Comp., Switzerland	Cray, Intel Xeon, NVIDIA K20	115,984	7,788	2,325
9	HLRS, Stuttgart, Germany	Hazel Hen: Cray XC40, Intel Xeon	185,088	7,403.5	-
10	KAUST, S. Arabia	Shaheen: Cray XC40, Intel Xeon	196,608	7,235.2	2,834
-	Your Home	Intel Core-i7	4	.026	0.3
-	Your Home	NVIDIA K40	2,880	4.29	0.3
-	Your Home	NVIDIA K80	4,992	8.73	0.3

Table 2: Top 10 Supercomputers: June 2016, top500.org

Ranking of Super Computers (cont.)

Rank	Site	System	Cores	TFLOPs	Power (kWh)
1	RIKEN, Japan	Fugaku: Fujitsu 2.2 GHz	7,630,848	442,010	29,899
2	Oak Ridge Lab, USA	Summit: IBM 3GHz, NVIDIA Volta GV100	2,414,592	148,600	10,096
3	DoE, USA	Sierra: IBM 3.1 GHz NVIDIA Volta GV100	1,572,480	94,640	7,438
4	Wuxi, China,	TaihuLight: Sunway 1.45GHz	10,649,600	125,435	15,371
5	DoE, USA	PerlMutter: Cray 2.4GHz, NVIDIA A100	761,856	70,870	2,589

Table 3: Top 10 Supercomputers: Nov 2021, top500.org

Ranking of Super Computers (cont.)

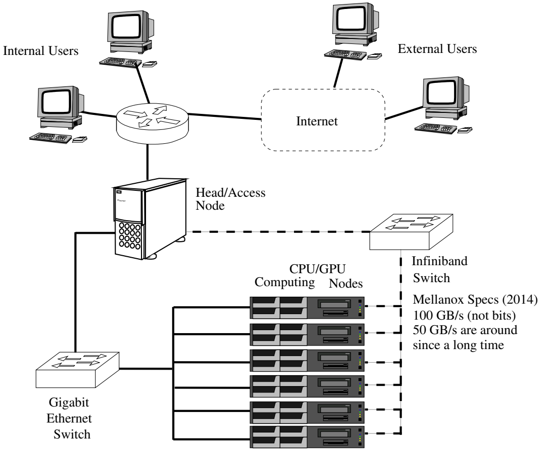


Figure 5: Setup of a typical HPC facility

Moore's Law, 1965

Cramming more components onto integrated circuits

With unit cost falling as the number of components per circuit rises, by 1975 economics may dictate squeezing as many as 65,000 components on a single silicon chip

By Gordon E. Moore

Director, Research and Development Laboratories, Fairchild division of Fairchild Camera and Instrument Corp.

The future of integrated electronics is the future of electronics itself. The advantages of integration will bring about a proliferation of electronics, pushing this science into many new areas.

Integrated circuits will lead to such wonders as home computers—or at least terminals connected to a central computer—automatic controls for automobiles, and personal portable communications equipment. The electronic wristwatch needs only a display to be feasible today.

But the biggest potential lies in the production of large systems. In telephone communications, integrated circuits in digital filters will separate channels on multiplex equipment. Integrated circuits will also switch telephone circuits and perform data processing.

Computers will be more powerful, and will be organized in completely different ways. For example, memories built of integrated electronics may be distributed throughout the

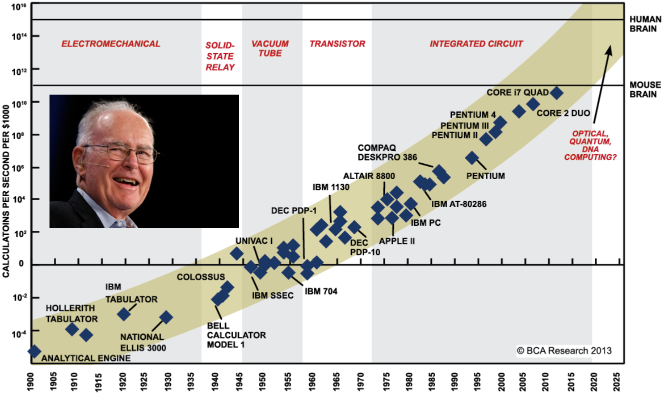
The author

Dr. Gordon E. Moore is one of the new breed of electronic engineers, schooled in the physical sciences rather than in electronics. He earned a B.S. degree in chemistry from the University of California and a Ph.D. degree in physical chemistry from the California Institute of Technology. He was one of the founders of Fairchild Semiconductor and has been director of the research and development laboratories since 1959.

Electronics, Volume 38, Number 8, April 19, 1965

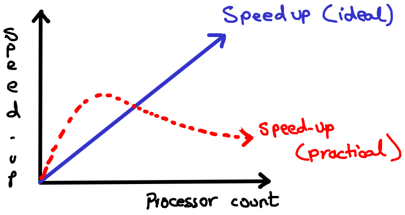
A prediction that would define the pace of digital revolution. Many interpretations:

- Computing would increase in **power** exponentially
- Computing would decrease in relative **cost** exponentially
- **Transistory density** will double every year (revised to double every 18 months)



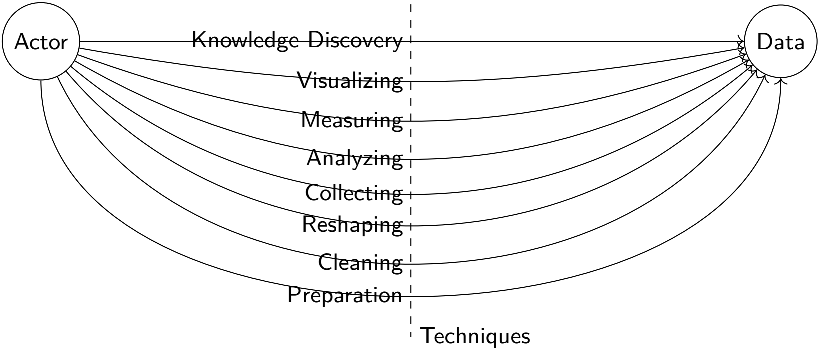
SOURCE: RAY KURZWEIL, "THE SINGULARITY IS NEAR: WHEN HUMANS TRANSCEND BIOLOGY", P.67, *THE VIKING PRESS*, 2006. DATAPOINTS BETWEEN 2000 AND 2012 REPRESENT BCA ESTIMATES.

Premise: Can software exploit this hardware for speed??



- Why ideal speedup is not possible: data transfer (through message exchanges), I/O bottlenecks, race conditions, dependencies, contention (critical section), load balancing, deadlocks, synchronization, node failures, . . .
- **Programmers lacking skills in parallel & distributed regime** . . .

Overview

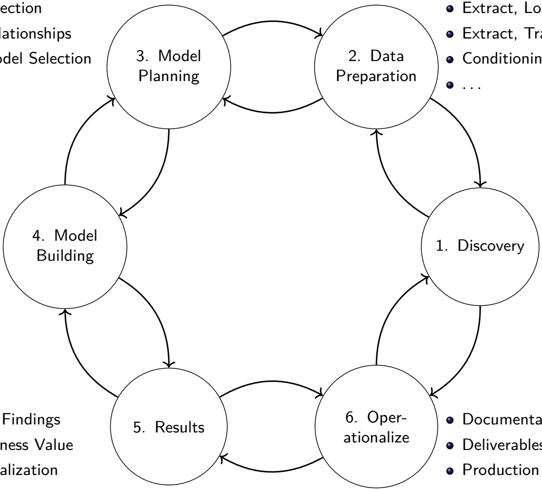


- Data scale may render techniques useless
- New techniques (from other discplines) employed to guarantee functioning of operations on data

Overview (cont.)

- Variable Selection
- Variable Relationships
- Suitable Model Selection
- ...

- Extract, Load, Transform (ELT)
- Extract, Transform, Load (ETL)
- Conditioning
- ...



- Model Execution
- Model Testing
- Model Training
- ...

- Business Domain
- Resource Availability
- Initial hypothesis
- ...

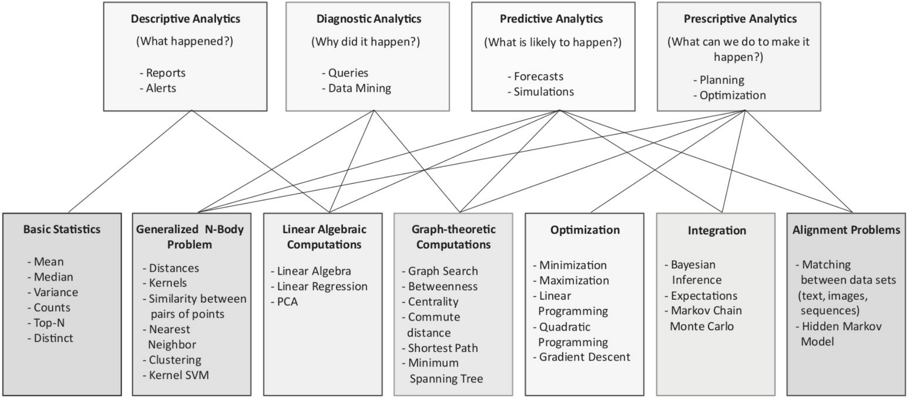
- Key Findings
- Business Value
- Visualization
- ...

- Documentations
- Deliverables
- Production Environment
- ...

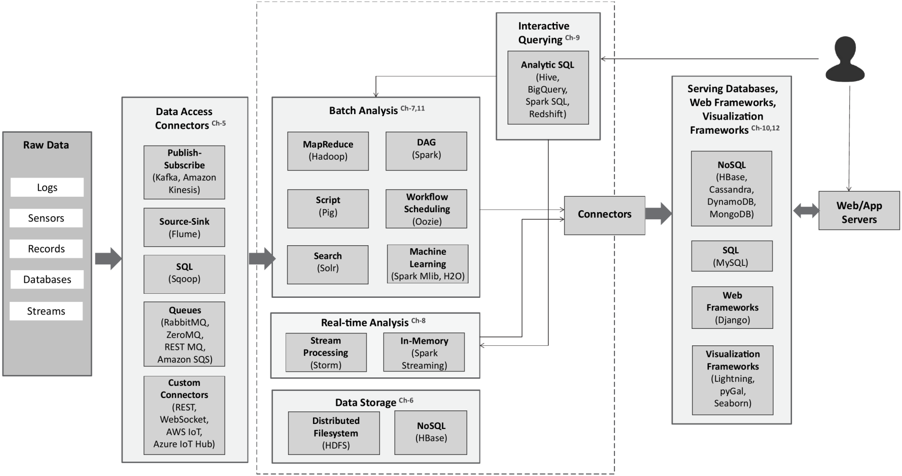
Overview (cont.)

Analysis Techniques

- Batch Analysis
- Real-Time Analysis



Big Data Stack



Big Data Stack (cont.)

Hadoop

- Open Source Framework by Apache for Distributed Batch Processing
- Uses Map-Reduce Programming Model
- Data Stored on Hadoop Distributed File System (HDFS)
- Data processed in Hadoop Clusters
- Contains other tools for job scheduling, and machine learning
- Available directly from Apache, or Cloudera, or HortonWorks

Spark

- Open Source Framework by Apache for Distributed Parallel Processing
- Uses Map-Reduce Programming Model
- Data Stored in Memory (RAM)
- Visualizes operations by constructing Directed Acyclic Graphs
- 100s of times faster than Hadoop for sorting, machine learning operations

R Overview

- Software Framework for Statistical Analysis and Graphics
- GNU General Public License
- R, Rscript, Rstudio

Sample Code

```r
#Command to Run: Rscript file.R

data <- read.csv("diabetes.csv")              # Read CSV File

head(data)                                    # CSV Header
summary(data)                                 # Descriptive Analytics

plot(data$Age, data$Glucose)                  # Plot 1 Col against Other

results <- lm(data$Age, data$Glucose)         # Apply Linear Regression (e.g.)
summary(results)                              # Descriptive Analytics again
hist(results$residuals)                       # Plot Histogram
```

R Overview (cont.)

Quick Commands

Get Help	`help(command)`
Set working directory	`setwd("/path")`
Write data	`write.table(data, "/path/data.txt", sep="/t", row.names=FALSE)`
Install Packages	`install.packages("RODBC")`
Load Package	`library(RODBC)`
Save Image	(1) `jpeg(file="/path/test.jpg")` (2) `hist(results$residuals)` (3) `dev.off()`
Get Class and Typeof variables	`class(var)` and `typeof(var)`
Type forcing	`j <- as.integer(var)`
Count of elements	`length(var)`

R Overview (cont.)

Function	Headers	Separator	Decimal Point
`read.table()`	False		.
`read.csv()`	True	,	.
`read.csv2()`	True	;	,
`read.delim()`	True	\t	.
`read.delim2()`	True	\t	,

Table 4: Different ways of Opening CSV/Text Files

Connect Through ODBC Driver

```
conn <- odbcConnect("dbName", "user", "password")
data <- sqlQuery(conn, "select * from table")
```

R Overview (cont.)

Vector Datatype

```
u <- c("red", "yellow", "blue")     # create a vector "red" "yellow" "blue"
u                                   # returns "red" "yellow" "blue"
u[1]                                # returns "red" (1st element in u)
v <- 1:5                            # create a vector 1 2 3 4 5
sum(v)                              # returns 15
w <- v * 2                          # create a vector 2 4 6 8 10
z <- v + w                          # sums two vectors element by element
z > 8                               # returns FALSE FALSE TRUE TRUE TRUE
z[z > 8]                            # returns 9 12 15
z[z > 8 | z < 5]                    # returns 3 9 12 15
```

R Overview (cont.)

Array and Matrix Types

```
data <- array(0, dim=c(3,4,2))
data[1,1,1] <- 22

data <- matrix(0, nrow=3, ncol=4)
data[1,1] <- 22

data <- matrix(c(1,2,3,4,5,6,7,8,9), nrow=3, ncol=3)
data_transpose <- t(data)
data_inverse <- matrix.inverse(data)
data_matrix_mul <- data %*% data_inverse

data[,1]       # Show 1st column
data[1,]       # Show 1st row
data[1:2,]     # Show 1st two rows
data[,c(1,3)] # Show 1st and 3rd col
```

R Overview (cont.)

DataFrame and List Types

- Multiple Type/Object Support

```
data[c("Glucose")]

list1 <- list("basketball", "cricket")
list2 <- list("cricket", 1, 2, 3)

data <- as.data.frame(cbind(list1, list2)) # Numerical binding
```

Applying a Function to Multiple Values

```
apply(data[,c(1:2)], MARGIN=2, FUN=sd)
```

R Overview (cont.)

Functions

```
doubleUpOne <- function(v)
{
  return( 2 * v[1:length(v)])
}
doubleUpTwo <- function(v)
{
  test <- 1:length(v)
  for(x in 1:length(v)) {
    test[x] = 2 * v[x]
  }
  return(test)
}
doubleupOne(v <- 1:5)
doubleupTwo(v <- 1:5)
```

Data Analytics

Analytics

- Processes, Technologies, Frameworks, and Algorithms that **extract meaningful insights from (raw) data**
- Insights organized and structured to infer knowledge

Case 1: Descriptive Analytics

- Present Data in summarized form (what has happened so far)
- Descriptive Statistics: Counts, Mins, Maxs, Means, Top-N, Percentages, ...

- Finding Correlation between News Items and Stock Prices
- Which pages are popularly visited on a website
- What is the average rainfall per year in Peshawar
- Top 10 coldest days in a year

Data Analytics (cont.)

Case 2: Diagnostic Analytics

- Analysis of Past Data to Diagnose why certain events Happen (why did something happen)
- Example: Causes of fault occurence in light of data from sensors

- Why did patient X die in hospital when patient Y survived?
- Which candidate is suitable to hire given certain parameters?

Case 3: Predictive Analytics

- Predict occurence of an event likely to happen (What is likely to happen)
- Example, when a fault will occur, tumor is benign or malignant, predicting pollution levels, weather, natural disasters

- Is a transaction fraudulent?
- Is a tumor malignant or benign?
- Is it going to rain on a particular day?

Data Analytics (cont.)

Case 4: Prescriptive Analytics

- Uses multiple prediction models to predict various outcomes and the best course of action for each outcome (What can we do to make it happen)

- Finding similar news items (similar patients, or similar products)
- Finding similar images in an image search
- Best medicine for treatment of patients (based on outcomes of medicines for similar patients)
- Best mobile data plan given customers usage

Descriptive Analytics

Data Summarization

Definition 1 (Mean)

$$\bar{y} = \frac{1}{N} \sum_{i=1}^{n} y_i \tag{1}$$

- Easily influenced by outliers (may be remedied as **trimmed** mean and/or **winsorized** mean)

$$\bar{y}_k = \frac{1}{N - 2k} \sum_{i=1+k}^{n-k} y_i \tag{2}$$

Descriptive Analytics (cont.)

Data Summarization

Definition 2 (Median)

- 50% less, 50% more
- Data must be sorted before obtaining m
- Not influenced by outliers
- If n is odd:

$$m = y_{\frac{n+1}{2}} \tag{3}$$

- If n is even:

$$m = \frac{1}{2}\left(y_{\lfloor \frac{n+1}{2} \rfloor} + y_{\lceil \frac{n+1}{2} \rceil}\right) \tag{4}$$

- the **Q2**

Descriptive Analytics (cont.)

Data Summarization

Definition 3 (Q1)

- If n is odd:

$$Q1 = y_{\frac{n+1}{4}} \tag{5}$$

- If n is even:

$$Q1 = \frac{1}{2}\left(y_{\lfloor \frac{n+1}{4} \rfloor} + y_{\lceil \frac{n+1}{4} \rceil}\right) \tag{6}$$

Definition 4 (Q3)

- If n is odd:

$$Q3 = y_{3\frac{n+1}{4}} \tag{7}$$

- If n is even:

$$Q3 = \frac{1}{2}\left(y_{\lfloor 3\frac{n+1}{4} \rfloor} + y_{\lceil 3\frac{n+1}{4} \rceil}\right) \tag{8}$$

Descriptive Analytics (cont.)

Data Summarization

Definition 5 ($\min, \max$)

$$y_{\min} = y_0 \tag{9}$$

$$y_{\max} = y_{n-1} \tag{10}$$

Definition 6 (Mode)

- Most frequently occuring data in dataset
- Can be multiple modes if same frequency applies to terms
- Can be zero mode if frequency of all terms is not more than 1

Descriptive Analytics

Data Spread Measurements

Definition 7 (Range)

$$R = y_n - y_1 \tag{11}$$

- Distance between two numbers (i.e., resembles deviation)
- Sign Affected by quadrant of operations

$y =$	{	7.5	8.0	8.0	8.5	9.0	11.0	19.5	19.5	28.5	31.0	36.0	} , $\bar{y} = 17$
$y - \bar{y} =$	{	-9.5	-9.0	-9.0	-8.5	-8.0	-6.0	2.5	2.5	11.5	14.0	19.0	}

- Average Deviation:

$$\frac{1}{N}\sum_{i=0}^{N-1}(y_i - \bar{y}) = 0 \tag{12}$$

Descriptive Analytics (cont.)

Data Spread Measurements

- Average Absolute Deviation

$$\frac{1}{N}\sum_{i=0}^{N-1}|y_i - \bar{y}| = 9 \tag{13}$$

Definition 8 (Variance)

$$\mathrm{var}(y) = \sigma^2 = \frac{1}{N-1}\sum_{i=1}^{n}(y_i - \bar{y})^2 \tag{14}$$

- Averaged square deviation from mean $\bar{y}$
- Square takes care of sign change problem, but it gives more weightage to data far from mean whereas closer data contribution is negligeable

$y =$	$\{$	7.5	8.0	8.0	8.5	9.0	11.0	19.5	19.5	28.5	31.0	36.0	$\}$, $\sigma_y^2 = 113$

Descriptive Analytics (cont.)

Data Spread Measurements

Definition 9 (Standard Deviation)

$$\text{std}(y) = \sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{15}$$

$y =$	$\{$	7.5	8.0	8.0	8.5	9.0	11.0	19.5	19.5	28.5	31.0	36.0	$\}$, $\sigma_y = 11$

Descriptive Analytics

Similarity Measures

Euclidean Distance

$$d(X,Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{16}$$

Manhattan (Taxicab) Distance

$$d(X,Y) = \sum_{i=1}^{n} |x_i - y_i| \tag{17}$$

Chebyshev Distance

$$d(X,Y) = \max \{|x_i - y_i|\} \tag{18}$$

Descriptive Analytics (cont.)

Similarity Measures

Minkowski Distance

$$d(X,Y) = \sqrt[p]{\sum_{i=1}^{n} (x_i - y_i)^p} \tag{19}$$

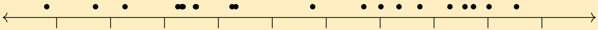
Other similarity measures: Cosine Similarity, Pearson Correlation Coefficient, Mahalanobis Measure, Jaccard Similarity, Levenshtein Distance, Hamming Distance, Chi-Square

Descriptive Analytics

Visualization

Dot / Scatter Plot

- Place dots at appropriate location on a number line



- Dot Chart in R: `dotchart(data$colx)`
- Extended as a 2D dot plot where point cloud / scatter plot can be constructed
- Scatter Plot in R: `plot(data$colx, data$coly)`

pch	pointer change	1, 2, 3, ...
col	colour	blue, red, green, ...
main	Title	string
xlab	X title	string
ylab	Y title	string

Table 5: Additional Arguements to plot() in R

Descriptive Analytics (cont.)

Visualization

Trend Line

- $y = mx + b$
- `abline(lm(data$coly ~ data$colx), col="red")`

- Slope

$$m = r\frac{\sigma_y}{\sigma_x} \tag{20}$$

- Intercept

$$b = \bar{y} - m\bar{x} \tag{21}$$

- Pearson's Correlation Coefficient

$$r = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}} \tag{22}$$

Descriptive Analytics (cont.)

Visualization

- Standard Deviation

$$\sigma_x = \sqrt{\frac{1}{N-1}\sum_{i=0}^{N-1}(x_i - \bar{x})^2} \tag{23}$$

Curve Fit

- $y = m_0 x^p + m_1 x^{p-1} + \ldots + m_{p-1} x^0 + \epsilon$
- Use Polynomial Regression Fitting

```
fit <- predict(loess(data$colx ~ data$coly))
points(data$colx, fit, col="blue")
```

Bar Plot

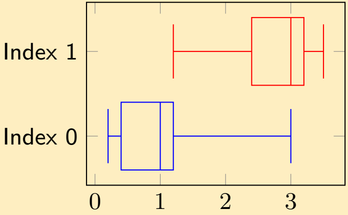
- `barplot(sales_delim$Glucose)`

Descriptive Analytics (cont.)

Visualization

Box Plot

- Sort Data
- Get $\min$ and $\max$
- Get Q1, Q2, and Q3



- in R: `boxplot(data$col1)`

Descriptive Analytics (cont.)

Visualization

Stem / Leaf Diagram

- Decide units for Stem (column 1) and Leaves (column 2)
- Prepare linearly increasing sequence of stems (with no gaps)
- For each variable, assign its units of leaves to respective bin of stem (bins must be of equal size)

stem	leave
1.2	3
1.3	1, 4
1.4	2, 7, 8
1.5	4, 1
1.6	4

- **Frequency Table**
  - Identify bins (not necessarily of equal size)
  - Place each item in respective bin and increment counter on its inclusion

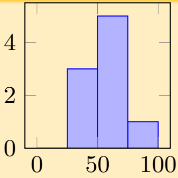
Bin	Count
$0 < x \leq 25$	0
$25 < x \leq 50$	3
$50 < x \leq 75$	5
$75 < x \leq 100$	1

Descriptive Analytics (cont.)

Visualization

Histogram from Frequency Table

- Horizontal axis representing bins
- Rectangular bars corresponding to count (frequency)
- No gaps if bins are continuous
- Represents **Density**



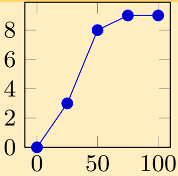
- In R: `hist(data$colx, breaks=50)` for histogram
- In R: `plot(density(data$colx))` for kernel density with kernel options of gaussian, triangular, cosine, rectangular, etc.

Descriptive Analytics (cont.)

Visualization

Cummulative Frequency from Frequency Table

- Horizontal axis representing bins
- Cummulative Frequency on Vertical axis
- Plot and join all the points



Example

- Sorted results for the Density of Earth (unknown units) by Cavendish (1798):
- 4.88 5.07 5.10 5.26 5.27 5.29 5.29 5.30
  5.34 5.34 5.36 5.39 5.42 5.44 5.46 5.47
  5.50 5.53 5.55 5.57 5.58 5.61 5.62 5.63
  5.65 5.68 5.75 5.79 5.85

Normal Datasets

- Highest at Middle Interval $\bar{y}$
- Histogram is symmetric
- 68% of observations are within $\bar{y} \pm \sigma$
- 95% of observations are within $\bar{y} \pm 2\sigma$
- 99.7% of observations are within $\bar{y} \pm 3\sigma$

Skewness

$$g_1 = \frac{\frac{1}{N}\left(Y_i - \bar{Y}\right)^3}{\sigma^3} \tag{24}$$

- Approximately Symmetric: $-0.5 < g_1 < 0.5$, Highly Skewed: $g_1 < -1$, $g_1 > +1$

Statistical Approaches to Outlier Detection (1D)

Classical Approach (|Z-Score|)

$$\frac{\left|X - \bar{X}\right|}{\sigma} \geq \{1.5, 2, 2.5, 3\} \tag{25}$$

$y =$	{	2	2	3	3	4	4	1000	}

$$\frac{|1000 - 145.43|}{376.83} \geq 2 \implies 2.26 \geq 2 \tag{26}$$

$y =$	{	2	2	3	3	4	4	1000	1000	}

$$\frac{|1000 - 252.25|}{461.52} \geq 2 \implies 1.62 \geq 2 \tag{27}$$

Boxplot Approach

$$X < Q_1 - 1.5\,(Q_3 - Q_1) \tag{28}$$

$$X > Q_3 + 1.5\,(Q_3 - Q_1) \tag{29}$$

Statistical Approaches to Outlier Detection (1D) (cont.)

$y =$	{	2	2	3	3	4	4	1000	}

$$1000 > 4 + 1.5\,(4 - 2) \implies 1000 > 7 \tag{30}$$

$y =$	{	2	2	3	3	4	4	1000	1000	}

$$1000 > 502 + 1.5\,(502 - 2.5) \implies 1000 > 1251.2 \tag{31}$$

Measuring Associations

Definition 10 (Covariance)

$$\operatorname{cov}(X,Y) = \frac{1}{N}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}) \tag{32}$$

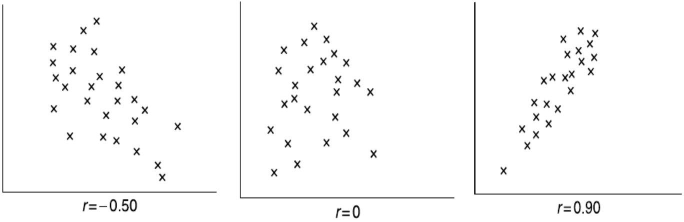
- Joint variability of two variables

Definition 11 (Correlation)

$$\operatorname{corr}(X,Y) = \frac{\operatorname{cov}(X,Y)}{\sqrt{\operatorname{var}(X)\operatorname{var}(Y)}} \tag{33}$$

$$\operatorname{corr}(X,Y) = \frac{\operatorname{cov}(X,Y)}{\sqrt{\frac{1}{N^2}\sum_{i=1}^{n}(x_i - \bar{x})^2(y_i - \bar{y})^2}} = \frac{\operatorname{cov}(X,Y)}{\operatorname{std}(X)\operatorname{std}(Y)} \tag{34}$$

Measuring Associations (cont.)



- If $\text{corr}(X, Y) = +1$ represents points on straight line with positive slope, strong correlation
- If $\text{corr}(X, Y) = -1$ represents points on straight line with negative slope, weak correlation
- If $\text{corr}(X, Y) < +1$ represents scattered points

Overview

Hadoop common

For Storage

Hadoop File System

For Processing

Map Reduce

Implementations

- Cloudera
- Amazon
- Ali Baba
- or Direct Installation

Overview (cont.)

Topics Covered

- Software Stack
- Distributed File System (HDFS)
- Physical Organization of Compute Nodes (Compute Nodes, Redundant Nodes, etc.)
- Map Reduce Framework
  - Key Value Pairs
  - Concept of Grouping Keys
  - Mappers & Reducers
- Sample Algorithms (See sample questions given)

Hadoop File System

Design Consideration

- Break files into blocks $B = \{b_1, b_2, \ldots, b_n\}$ of certain size s
- Distribute blocks across multiple (data) nodes $N = \{n_1, n_2, \ldots, n_m\}$ within a cluster
- Challenges:
  - For large m, node failures are quite probable. So introduce redundancy.
  - For large n, high throughput is required. So introduce concepts such as write once read many, and move computation closer to data.

Hadoop File System (cont.)

Performance Impact

- File Size, Block Length, Block Quantity, all affect performance
- Block quantity $\leftrightarrow$ Number of Threads
  - Queues
  - Thread Creation/Deletion Time
  - I/O
  - Messages exchanges between threads
- Strategies: Merge Files, Load Files in Sequence

Hadoop Common Architecture

Name Node (Master Server)

- Manages File System Namespace
- Regulates/Control access to files
  - Read/write requests from client
  - Create/Delete/Replicate blocks on data nodes

Data Nodes

- Manages Physical Storage of Blocks
- Serve Read/Write requests of Clients
- Serve Create/Delete/Replicate block requests of Name Node

Node Failures

Data Node Failure

- Server Crash / Disk Crash / Data Corruption / Network Failure
- Name node sends periodic heart-beats
- If true, mark dead, re-replicate block copy

Network Failure

- Denial of Service
- Physical Network failure

Namenode Failure

- Server Crash / Disk Crash / Data Corruption
- Send data/metadata to secondary nameserver (if configured)

HDFS Tuning Parameters

(Name,Value) properties in `/etc/hadoop/hdfs-site.xml`

- `dfs.replication` : 3 for Replication Factor
- `dfs.namenode.name.dir`: `/var/lib/hadoop/hdfs/name` for Name Node
- `dfs.datanode.data.dir`: `/var/lib/hadoop/hdfs/data` for Data Node
- `dfs.namenode.secondary.http-address` : `hdfs://localhost:50090` For Secondary Name node
- `dfs.permissions.superusergroup` : `hadoop` for User Permissions (must belong to this group)
- `dfs.block.size` : 134217728 for changing block size (across all clusters)
- `dfs.datanode.handler.count` : 10 for changing threads per data node.
- `dfs.namenode.fs-limits.max-blocks-per-file` : 100 for fixing maximum allowable blocks per file

- Dozens of other parameters specified in `hdfs-default.xml` (search online)

Specific Adjustments

- `hdfs dfs -D dfs.blocksize=134217728 -put test_128MB.csv /user`

HDFS Commands

- `hdfs dfs -ls /`
- `hdfs dfs -lsr /` ls with recursive display
- `hdfs dfs -du` Disk usage
- `hdfs dfs -dus` Disk usage summary
- `hdfs dfs -mv src dest`
- `hdfs dfs -rm xyz` Remove file or empty directory
- `hdfs dfs -rmr xyz` Recursive remove file or directory
- `hdfs dfs -put local remote`
- `hdfs dfs -get remote local`
- `hdfs dfs -cat file`
- `hdfs dfs -tail file`
- `hdfs dfs -head file`
- `hdfs dfs -chmod 777 file`
- `hdfs dfs -chown group file`
- `hdfs dfsadmin -report` Shows utilization of HDFS

Using C

libhdfs

- Part of the Hadoop distribution (Located pre-compiled in `$HADOOP_HDFS_HOME/lib/native/libhdfs.so`)
- Compatible with both Linux/Windows
- Java Native Interface (JNI) to Core Interface API in Java
- Thread Safe
- To compile (gcc)

```
gcc -I /opt/hadoop-3.2.1/include hdfsC.c
    -L /opt/hadoop-3.2.1/lib/native -lhdfs
    -L /opt/oracle-jdk-bin/jre/lib/amd64/server -ljvm|
```

- To run

```
CLASSPATH=$CLASSPATH:$(/opt/hadoop-3.2.1/bin/hadoop classpath --glob)
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/oracle-jdk-bin/jre/lib/amd64/server: \
                                 /opt/hadoop-3.2.1/lib/native
./a.out
```

Using C (cont.)

```c
#include "hdfs.h"
#include <string.h>
#include <stdio.h>

int main(int argc, char **argv) {
  hdfsFS      fs           = hdfsConnect("default", 0);

  const char  *writePath  = "/testfile.txt";
  hdfsFile    writeFile    = hdfsOpenFile(fs, writePath, O_WRONLY|O_CREAT, 0, 0, 0);

  char* buffer             = "Hello, World!";
  tSize num_written_bytes  = hdfsWrite(fs, writeFile, (void*)buffer, strlen(buffer)+1);

  hdfsFlush(fs, writeFile);
  hdfsCloseFile(fs, writeFile);
}
```

HTTP Rest API

- Support for HTTP Get, Put (-X PUT), Post (-X POST), Delete (-X DELETE)
- Configuration for `dfs.webhdfs.enabled` required in `hdfs-site.xml` (default is true)
- General Usage:

```
curl -i http://localhost:port/webhdfs/v1/[path|file]?
                                        [user.name=<user>&]
                                        op=[<options>]
```

- Default port: 50090 → 9870
- Responses in JSON

HTTP Rest API (cont.)

Sample Operations

- `op=GETFILESTATUS` Get Information about files
- `op=MKDIRS` for creating directory
- `permission=755` for specifying Linux permissions
- `op=CREATE` for creating a (blank) file. For copying contents of an existing file, (use with -T <LOCAL_FILE>
- `blocksize=<LONG>` for Block Size
- `replication=<SHORT>` for replication factor
- `op=APPEND` for appending to a file
- `op=OPEN` for opening and reading a file
- `op=RENAME&destination=<PATH>` for renaming a file
- `op=DELETE` for deleting a file/directory, (Use with -X DELETE)
- . . . and others (See hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/WebHDFS.html

Map Reduce Framework

- Mapper Function
- Reducer Function
- Shuffle and Sort

Map Reduce Framework (cont.)

Example: Word Count

- <word, 1>
- Tokenize text and produce key-value pairs in a stream.
- Get new word from stream. If new word is same as previous word, increment a counter, else reset the counter.

```
import sys

for line in sys.stdin:
  line  = line.strip()
  words = line.split()
  for word in word:
    print ("%s\t1" % word)
```

```
import sys

current_word = None
current_count = 0
word = None

for line in sys.stdin:
    line = line.strip()
    word, count = line.split('\t', 1)
    count = int(count)
        if current_word == word:
        current_count += count
    else:
        print ('%s\t%s' % (current_word, current_count))
        current_count = count
        current_word = word
```

Map Reduce Framework (cont.)

To run Job

```
hadoop jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming-3.2.1.jar
            -file /home/omar/work/codes/hadoop/mapper.py
            -file /home/omar/work/codes/hadoop/reducer.py
            -mapper mapper.py
            -reducer reducer.py
            -input /4300-folder/*
            -output /4300-output
```

- Reducer Threads controlled using `-D mapred.reduce.tasks=16`

Spark Overview

- Cluster computing platform designed for speed (mostly in-memory operations rather than file I/O)
- Support for Stream Processors (GPU's)
- API's available in Python, Java, Scala, and SQL

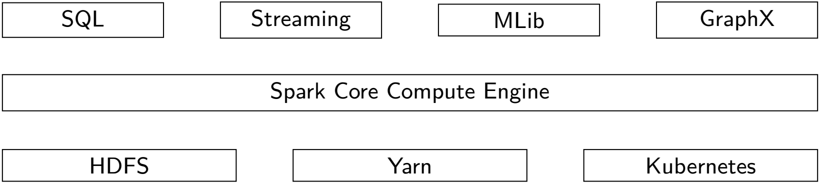
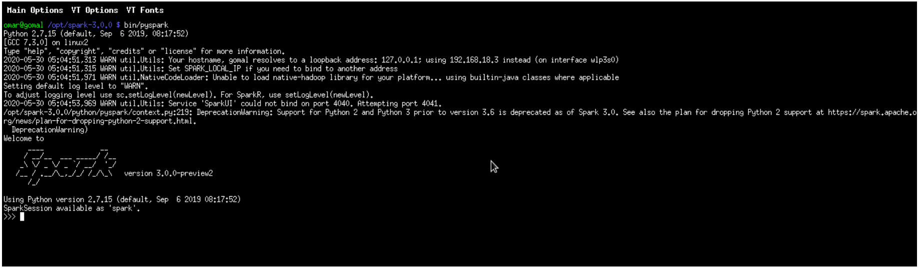


Figure 6: Spark System Architecture

- Web based Monitoring Interface on http://localhost:4040

Spark Shells

- `bin/pyspark` for Python
- `bin/spark-shell` for Scala
- `bin/sparkR` for R



- Can also be interfaced with Python Notebooks (e.g. Jupyter or iPython)

Hello World (Word Count)

```
lines = sc.textfile("README.md") # Must be on HDFS
                                 # lines is an RDD Object
lines.count()                    # Returns number of items
lines.first()                    # First line in RDD (or file)
```

Resilient Distributed Dataset RDD

- Computations performed on *distributed collections* that are automatically parallelized across a cluster.
- Immutable Objects (always new RDD returned)
- These Collections are the RDD's (more in coming slides)
- Created using `parallelize()` or `textfile()` methods of spark context.

Hello World (Word Count) (cont.)

Standalone Application

- Same API, but have to create Spark Contxt yourself
- Run using bin/spark-submit

```python
from pyspark import SparkConf, SparkContext

conf = SparkConf().setMaster("local").setAppName("My App")
sc   = SparkContext (conf = conf)

# User program from here onward

lines = sc.textFile("/4300-folder/4300-0.txt")
count = lines.count()

def hasBook(line):
    return "Book" in line

bookLines = lines.filter(hasBook)
bookcount = bookLines.count()
```

Core Spark Concepts

Driver Program (e.g., pyspark shell)

- Launches various parallel operations on a cluster
- Contains your applications main function
- Contains your applications distributed datasets
- Accesses Spark through a **Spark Context** object. This context is automatically created as object `sc`.

  `<SparkContext master=local[*] appName=PySparkShell>`
- RDD's created from Context

Executors

- Present on each Worker Node (computer)
- Managed by Driver

Core Spark Concepts (cont.)

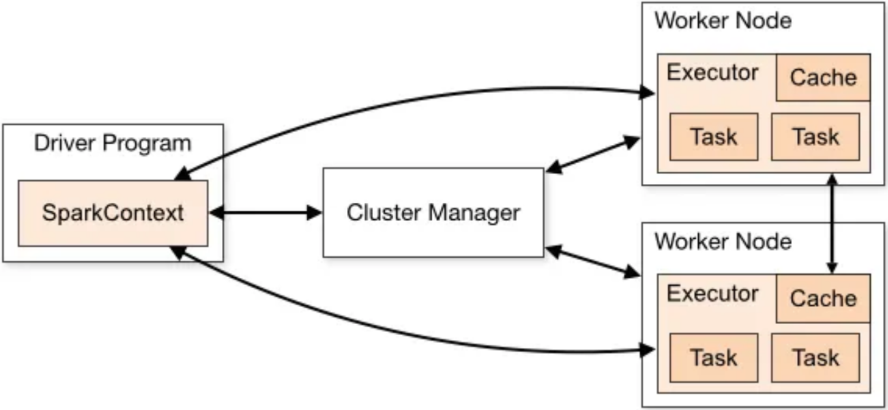


Figure 7: Components for Distributed Execution in Spark IMG: Apache Spark

Resilient Distributed Dataset (RDD)

Creation Approaches in Driver Program

- Loading an External Dataset from File `rddObj = sc.textFile("myFilename")`
- Collection of List objects `rddObj = sc.parallelize(["FAST", "I Like FAST"])`
  `rddObj = sc.parallelize([1, 2, 3, 4])`

Creation of DataFrame from RDD Object

```
dataCol = Seq("key", "value")
dataObj = Seq(("k1", v1), ("k2", v2))

rddObj  = sc.parallelize(dataObj)
dfRddObj = rddObj.toDF(dataCol)

dfRddObj.printSchema()
dfRddObj.show()
dfRddObj.select("key").show()
dfRddObj.filter(dfRddObj("value") > 100).show()
```

Resilient Distributed Dataset (RDD) (cont.)

Operations on RDD

- Actions (operations directly on RDD; Output displayed to Driver program, or to HDFS storage). For example: count(), first(), take(), collect()
- Transformations (new RDD from existing one). For example: Filtering(), Union(), Map(), FlatMap()

```
inputRDD  = sc.textFile("log.txt")

inputRDD.count()                                              # Action
inputRDD.first()                                              # Action

errorRDD   = inputRDD.filter(lambda x: "Error" in x)          # Transformation
warningRDD = inputRDD.filter(lambda x: "Warning" in x)        # Transformation

badLinesRDD = errorRDD.union(warningRDD)                      # Transformation

print("Bad Lines: " + badLinesRDD.count())                    # Action
for line in badLinesRDD.take(10):                             # Action
```

Resilient Distributed Dataset (RDD) (cont.)

```
      print(line)                                            # or file write

for line in badlinesRDD.collect():                           # Danger Action
      print(line)
inputRDD  = sc.parallelize([1, 2, 3, 4])
squareRDD = inputRDD.map(lambda x: x * x).collect()          # Map Collect
for num in squareRDD:
      print("%d" % num)

inputRDD  = sc.parallelize(["Coffee Panda", "Happy Panda"])
outputRDD = inputRDD.map(lambda line: line.split(" "))
outputRDD.take(2)
# Displays: [['Coffee', 'Panda'], ['Happy', 'Panda']

outputRDD = inputRDD.flatMap(lambda line: line.split(" "))
outputRDD.take(4)
# Displays: ['Coffee', 'Panda', 'Happy', 'Panda']
```

Resilient Distributed Dataset (RDD) (cont.)

Lazy Loading Principle

- Compute/Retrive RDD only when required (determined through internal metadata)
- For transformation RDD's, maintain **Lineage Graph**
- Recompute again and again, any time you need it, with certain degree of caching (makes sense for large datasets). To override:

```
lines.persist()    # Hold data in memory
lines.count()
lines.first()
```

Other Transformation Operations

- `RDD1.distinct()` returns unique members
- `RDD1.union(RDD2)` returns Union
- `RDD1.intersection(RDD2)` returns Intersection
- `RDD1.subtract(RDD2)` returns RDD1 - RDD2
- `RDD1.cartesian(RDD2)` returns RDD1 $\times$ RDD2

Resilient Distributed Dataset (RDD) (cont.)

Saving RDD's (to HDFS)

- `lines.saveAsTextFile("Directory")`

Spark with Key Value Pairs

```
from pyspark import SparkContext, SparkConf

conf = SparkConf().setMaster("local").setAppName("My App")
sc = SparkContext(conf=conf)

words = sc.textFile("/documenttxt").flatMap(lambda line: line.split(" "))

wc    = words.map(lambda word: (word, 1)).reduceByKey(lambda a,b : a +b)

wc.saveAsTextFile("/sparktest")
```