## 1. Big Data Fundamentals

### Definition

Big Data refers to extremely large datasets that cannot be processed, stored, or analyzed using traditional computing methods due to their volume, velocity, and variety.

### The 5 V's of Big Data

### Volume - Scale of Data

- **Definition**: The sheer amount of data generated
- **Scale**: Terabytes, petabytes, exabytes
- **Real-world examples**:
    - **Facebook**: Processes 4+ petabytes of data daily
    - **Walmart**: Handles 2.5 petabytes of customer transaction data hourly
    - **Netflix**: Stores over 15 petabytes of content and user data

### Velocity - Speed of Data

- **Definition**: The speed at which data is generated and processed
- **Real-world examples**:
    - **Twitter**: Processes 500 million tweets per day (6,000 tweets/second)
    - **Stock Market**: NYSE processes 1 billion transactions daily

    - **Uber**: Processes location data from millions of drivers in real-time

**Variety - Types of Data**

- **Definition**: Different formats and types of data (structured, semi-structured, unstructured)
- **Real-world examples**:
    - **Amazon**: Product data (structured), customer reviews (unstructured), clickstream data (semi-structured)
    - **Healthcare**: Medical records (structured), medical images (unstructured), sensor data (structured)

**Veracity - Quality of Data**

- **Definition**: Trustworthiness and quality of data
- **Real-world examples**:
    - **Google**: Filters spam and fake reviews using ML algorithms
    - **Financial institutions**: Detect fraudulent transactions in real-time

**Value - Business Value**

- **Definition**: The worth derived from data analysis
- **Real-world examples**:
    - **Target**: Predicts customer pregnancy to send relevant coupons
    - **Spotify**: Creates personalized playlists increasing user engagement by 30%

## Components of Big Data

### Storage Components

#### Hadoop Distributed File System (HDFS)

- **Purpose**: Distributed storage across commodity hardware
- **Real-world usage**:
    - **Yahoo**: Stores 40+ petabytes across 40,000+ nodes
    - **LinkedIn**: Manages member profiles and connection data

#### NoSQL Databases

- **MongoDB**: Document-based storage
- **Cassandra**: Column-family database
- **HBase**: Wide-column store on top of HDFS

**Real-world examples**:

- **Instagram**: Uses Cassandra for photo metadata
- **Airbnb**: Uses MongoDB for user profiles and listings

### Processing Components

#### MapReduce

- **Purpose**: Distributed processing framework
- **Real-world usage**: Google's original web indexing system

#### Apache Spark

- **Purpose**: Fast, in-memory processing
- **Advantages**: 100x faster than MapReduce for certain workloads

**Real-world examples**:

- **Uber**: Real-time pricing and driver matching
- **Netflix**: Real-time recommendation engine

### Analytics Components

#### Apache Hive

- **Purpose**: SQL-like queries on big data
- **Real-world usage**: Facebook's data warehousing

### Architecture

#### Lambda Architecture

Combines batch and real-time processing for comprehensive data analysis.

**Components:**

1. **Batch Layer**: Processes historical data (Hadoop/Spark)
2. **Speed Layer**: Handles real-time data (Storm/Kafka)
3. **Serving Layer**: Stores processed results (NoSQL databases)

**Real-world example**: Netflix uses Lambda architecture for:

- **Batch Layer**: Nightly processing of viewing history
- **Speed Layer**: Real-time recommendation updates
- **Serving Layer**: Personalized content delivery

#### Kappa Architecture

Simplified architecture using only stream processing.

**Real-world example**: LinkedIn's activity feed processing

## Benefits

### Business Intelligence and Analytics

- **Walmart**: Analyzes 2.5 petabytes of data to optimize inventory, reducing costs by $2 billion annually
- **American Express**: Detects fraud in real-time, saving millions in fraudulent transactions

### Personalization

- **Amazon**: Product recommendations generate 35% of total revenue
- **Spotify**: Discover Weekly increases user engagement by 30%

### Operational Efficiency

- **UPS**: ORION system analyzes delivery routes, saving 100 million miles annually
- **General Electric**: Predictive maintenance on jet engines saves airlines $2 billion yearly

### Cost Reduction

- **Capital One**: Moved from traditional data warehouses to cloud-based big data solutions, reducing costs by 40%

## Challenges

### Data Quality and Governance

- **Challenge**: Inconsistent, incomplete, or inaccurate data

- **Real-world example**: IBM estimates that poor data quality costs US businesses $3.1 trillion annually
- **Solution**: Implement data quality frameworks and governance policies

**Privacy and Security**

- **Challenge**: Protecting sensitive information while enabling analytics
- **Real-world example**: GDPR fines totaling €1.2 billion in 2021
- **Solution**: Data anonymization, encryption, access controls

**Scalability**

- **Challenge**: Systems must handle exponentially growing data
- **Real-world example**: Facebook's data centers process 4 petabytes daily
- **Solution**: Distributed computing, cloud auto-scaling

**Skills Gap**

- **Challenge**: Shortage of big data professionals
- **Statistics**: 39% of companies report difficulty finding qualified data scientists
- **Solution**: Training programs, automated ML tools

**Integration Complexity**

- **Challenge**: Connecting disparate data sources and systems
- **Solution**: Data lakes, API management, ETL pipelines

**Data Lifecycle Stages**

**1. Data Generation**

- **Sources**: IoT sensors, web applications, mobile apps, social media
- **Real-world example**: Tesla vehicles generate 25GB of data per hour

**2. Data Collection**

- **Methods**: APIs, web scraping, sensors, databases
- **Tools**: Apache Flume, Logstash, Beats

**3. Data Storage**

- **Options**: HDFS, cloud storage (S3, Azure Blob), NoSQL databases
- **Strategy**: Data lakes for raw data, data warehouses for processed data

**4. Data Processing**

- **Batch Processing**: Historical analysis, reporting
- **Stream Processing**: Real-time analytics, monitoring
- **Tools**: Spark, Flink, Storm

**5. Data Analysis**

- **Descriptive**: What happened?
- **Diagnostic**: Why did it happen?
- **Predictive**: What will happen?
- **Prescriptive**: What should we do?

**6. Data Visualization**

- **Tools**: Tableau, Power BI, D3.js, Plotly
- **Real-world example**: Uber's real-time city analytics dashboard

**7. Data Archiving**

- **Strategy**: Move old data to cheaper storage
- **Implementation**: Automated lifecycle policies

**Types of Data: Structured, Semi-Structured, Unstructured**

**Structured Data**

- **Definition**: Organized in predefined format (tables, rows, columns)
- **Characteristics**: Easy to search, analyze, and store
- **Examples**: SQL databases, spreadsheets, financial records

**Real-world examples**:

- **Bank transactions**: Account number, amount, date, merchant
- **E-commerce orders**: Order ID, customer ID, product ID, quantity, price

- **Sensor readings**: Timestamp, temperature, humidity, location

**Semi-Structured Data**

- **Definition**: Contains organizational properties but doesn't fit rigid structure
- **Formats**: JSON, XML, CSV with varying schemas
- **Characteristics**: Flexible schema, self-describing

**Real-world examples**:

- **Web APIs**: REST API responses, configuration files
- **Log files**: Web server logs, application logs
- **Email**: Headers (structured) + body (unstructured)

**Unstructured Data**

- **Definition**: No predefined structure or organization
- **Types**: Text, images, videos, audio, social media posts
- **Challenges**: Requires specialized tools for analysis

**Real-world examples**:

- **Social media**: Tweets, Facebook posts, Instagram images
- **Documents**: PDFs, Word documents, emails
- **Multimedia**: Videos, audio files, medical images

**File Types**

**Structured File Types**

**CSV (Comma-Separated Values)**

- **Use case**: Simple tabular data exchange
- **Pros**: Human-readable, widely supported
- **Cons**: No schema enforcement, large file sizes

**Parquet**

- **Use case**: Analytics workloads, data warehousing
- **Pros**: Columnar storage, compression, schema evolution
- **Cons**: Not human-readable

**ORC (Optimized Row Columnar)**

- **Use case**: Hive data warehousing
- **Pros**: High compression, predicate pushdown
- **Real-world**: Used extensively in Hadoop ecosystems

**Semi-Structured File Types**

**JSON (JavaScript Object Notation)**

- **Use case**: Web APIs, configuration files, document stores
- **Pros**: Flexible schema, human-readable
- **Cons**: Verbose, parsing overhead

**XML (eXtensible Markup Language)**

- **Use case**: Legacy systems, SOAP APIs, configuration
- **Pros**: Self-describing, hierarchical
- **Cons**: Verbose, complex parsing

**Avro**

- **Use case**: Data serialization, schema evolution

- **Pros**: Schema evolution, compact binary format
- **Real-world**: Kafka message serialization

**Unstructured File Types**

**Text Files**

- **Use case**: Logs, documents, free-form text
- **Processing**: Natural Language Processing (NLP)

**Binary Files**

- **Types**: Images (JPEG, PNG), Videos (MP4, AVI), Audio (MP3, WAV)
- **Processing**: Computer vision, audio processing

**Big Data Optimized Formats**

**Delta Lake**

- **Features**: ACID transactions, time travel, schema enforcement
- **Use case**: Data lakes with reliability requirements

**Apache Iceberg**

- **Features**: Schema evolution, partition evolution, time travel
- **Use case**: Large-scale analytics with evolving schemas

Performance Comparison of File Formats

| Format | Read Speed | Write Speed | Compression | Schema Evolution | Query Performanc |
|---|---|---|---|---|---|
| CSV | Slow | Fast | None | No | Poor |
| JSON | Moderate | Moderate | Moderate | Yes | Moderate |
| Parquet | Fast | Moderate | High | Yes | Excellent |
| ORC | Fast | Moderate | High | Yes | Excellent |
| Avro | Moderate | Fast | Moderate | Yes | Good |
| Delta | Fast | Moderate | High | Yes | Excellent |

**Choosing the Right Format**

- **CSV**: Quick data exchange, small datasets
- **JSON**: APIs, flexible schemas, moderate sizes
- **Parquet**: Analytics, large datasets, columnar queries
- **ORC**: Hive-based workflows, heavy compression needs
- **Avro**: Schema evolution, Kafka streaming
- **Delta/Iceberg**: Enterprise data lakes, ACID requirements

---

Hadoop is an open-source software framework that is used for storing and processing large amounts of data in a distributed computing environment. It is designed to handle big data and is based on the MapReduce programming model, which allows for the parallel processing of large datasets. Its framework is based on Java programming with some native code in C and shell scripts.

Hadoop is designed to process large volumes of data (Big Data) across many machines without relying on a single machine. It is built to be scalable, fault-tolerant and cost-effective. Instead of relying on expensive high-end hardware, Hadoop works by connecting many inexpensive computers (called nodes) in a cluster.

**Hadoop Architecture**

Hadoop has two main components:

- **Hadoop Distributed File System (HDFS):** HDFS breaks big files into blocks and spreads them across a cluster of machines. This ensures data is replicated, fault-tolerant and easily accessible even if some machines fail.
- **MapReduce:** MapReduce is the computing engine that processes data in a distributed manner. It splits large tasks into smaller chunks (map) and then merges the results (reduce), allowing Hadoop to quickly process massive datasets.

Apart from the above-mentioned two core components, Hadoop framework also includes the following two modules **Hadoop Common** which are Java libraries and utilities required by other Hadoop modules and **Hadoop YARN** which is a framework for job scheduling and cluster resource management.

Hadoop Architecture

**Hadoop Distributed File System (HDFS)**

HDFS is the storage layer of Hadoop. It breaks large files into smaller blocks (usually 128 MB or 256 MB) and stores them across multiple DataNodes. Each block is replicated (usually 3 times) to ensure fault tolerance so even if a node fails, the data remains available.

**Key features of HDFS:**

- **Scalability:** Easily add more nodes as data grows.
- **Reliability:** Data is replicated to avoid loss.
- **High Throughput:** Designed for fast data access and transfer.

**MapReduce**

MapReduce is the computation layer in Hadoop. It works in two main phases:

1. **Map Phase:** Input data is divided into chunks and processed in parallel. Each mapper processes a chunk and produces key-value pairs.
2. **Reduce Phase:** These key-value pairs are then grouped and combined to generate final results.

This model is simple yet powerful, enabling massive parallelism and efficiency.

**How Does Hadoop Work?**

Here's a overview of how Hadoop operates:

1. *Data is loaded into HDFS, where it's split into blocks and distributed across DataNodes.*
2. *MapReduce jobs are submitted to the ResourceManager.*
3. *The job is divided into map tasks, each working on a block of data.*
4. *Map tasks produce intermediate results, which are shuffled and sorted.*
5. *Reduce tasks aggregate results and generate final output.*
6. *The results are stored back in HDFS or passed to other applications.*

Thanks to this distributed architecture, Hadoop can process petabytes of data efficiently.

**Advantages and Disadvantages of Hadoop**

**Advantages:**

- **Scalability:** Easily scale to thousands of machines.

- **Cost-effective:** Uses low-cost hardware to process big data.
- **Fault Tolerance:** Automatic recovery from node failures.
- **High Availability:** Data replication ensures no loss even if nodes fail.
- **Flexibility:** Can handle structured, semi-structured and unstructured data.
- **Open-source and Community-driven:** Constant updates and wide support.

**Disadvantages:**

- Not ideal for real-time processing (better suited for batch processing).
- Complexity in programming with MapReduce.
- High latency for certain types of queries.
- Requires skilled professionals to manage and develop.

**Applications**

Hadoop is used across a variety of industries:

1. **Banking:** Fraud detection, risk modeling.
2. **Retail:** Customer behavior analysis, inventory management.
3. **Healthcare:** Disease prediction, patient record analysis.
4. **Telecom:** Network performance monitoring.
5. **Social Media:** Trend analysis, user recommendation engines.