Big Data Concepts & Map-Reduce & Hadoop

Ankara JUG - Aralık 2014

Serkan ÖZAL
serkan@hazelcast.com

Serkan ÖZAL

Lisans : 2005 - 2009, @ Hacettepe Bil. Müh.

Master : 2010 - 2013, @ ODTÜ Bil. Müh.

Doktora : 2013 - ..., @ ODTÜ Bil. Müh.

Technical Author @ Rebellabs

Open-Source Contributor @ Oracle

Coder @ Hazelcast

github.com/serkan-ozal

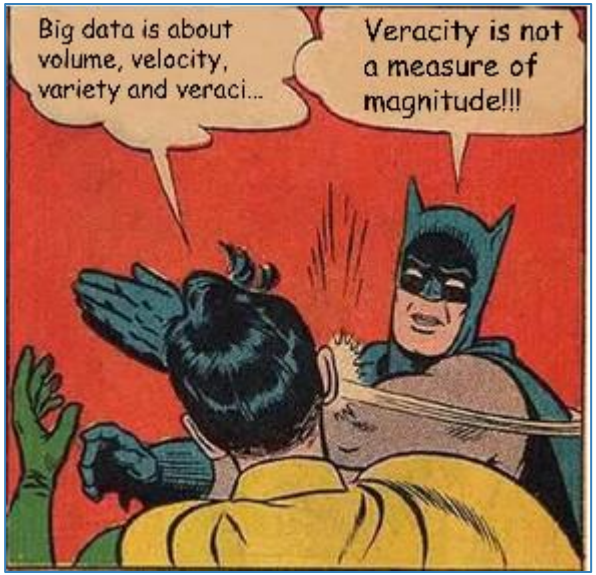
Outline

- Big Data
- Scalability
- NoSQL
- Batch Data Processing
- Stream Data Processing
- Map-Reduce
- Hadoop
- HDFS
- AWS
- Demo

What is Big Data ???

En temel ve basit anlamıyla uğraşması zor veri demektir.

- Big Data’nın Özellikleri
  - Volume (Hacmi)
  - Velocity (Hızı)
  - Variety (Çeşitliliği)
  - Veracity (Belirsizliği)



Scalability

- Çalışan bir sistemin işlem kapasitesinin gereksinimlere göre büyütülebilip küçültülebilmesidir.

- Scalability:
  - Vertical Scalability: Daha fazla/az CPU, RAM,
  - Horizontal Scalability: Daha fazla/az makina

- Auto Scalable

Big Data Concepts

- NoSQL
- Batch Data Processing
- Stream Data Processing
- Data Flow
- Cloud Computing

NoSQL

- **N**ot **O**nly **SQL**
- İlişkisel olmayan verileri tutmak için tasarlanmıştır.
- Verinin belirli bir formatta olması gerekmez.
- Scalable ve fail-safe sistemlerdir.
- Genelde “UNION”, “JOIN” gibi ilişkisel sorguları desteklemezler.

NoSQL Databases

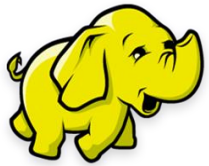
- Column Stores:
  - HBase
- Key-Value Stores:
  - Hazelcast, Infinispan, Redis, Terracotta, Coherence
  - Cassandra
  - Amazon DynamoDB
- Document Stores:
  - MongoDB
  - CouchDB
- Graph Databases:
  - Neo4J

Batch Data Processing

- İşlenecek büyük miktarda veri vardır (TB'larca yada PB'larca)
- İşlenecek veri offline'dır yani işlenecek veri miktarı baştan bilinir ve işlem süresince değişmez.
- Tek bir makinada işlenmesi günler, haftalar sürebilir (bu işlem süresince makinanın herhangi bir sebeple göçmediğini varsayıyoruz) Bu sebeple veri distributed olarak işlenmelidir.

Batch Data Processing Frameworks

- Hadoop
- Hive
- Pig
- Presto
- Impala
- Spark
- Mahout

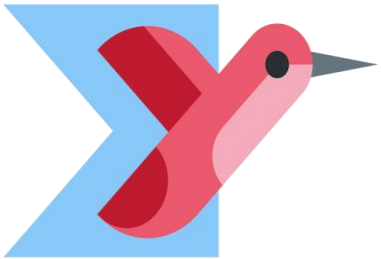


Stream Data Processing

- Batch data processing’in tersine veri sabit değildir
- Verinin miktarı önceden bilinmez
- Veri real-time yada near real-time olarak işlenmelidir
- Sistemin yükü önceden kestirilemeyebilir ve zaman içinde değişiklik gösterebilir

Stream Data Procesing Frameworks

- Storm
- AWS Kinesis
- Akka
- SummingBird
- Kafka
- Samza



Data Flow Frameworks

- Cascading
- Oozie
- AWS Data Pipeline

Map-Reduce

- 2004 yılında Google tarafından yayınlanan bir makale ile ortaya atılmış bir yazılım mimarisi pattern'idir.
- Yapılacak işin daha küçük alt işlere bölünüp tüm cluster'da paralel şekilde yapılması esasına dayanır
- Cluster'daki her node diğer node'lardan bağımsız bir şekilde kendisine atanan veri seti üstünde kendisine atanan işi yapar.
- Temelde 2 fazdan oluşur
  - Map
  - Reduce

Map

- Master node büyük veri setini daha küçük bloklar halinde mapper node'lara atar.
- Mapper node'lar kendilerine atanan veri seti üstünde çözümlerini yapıp sonuçlarını master node'a gönderirler.

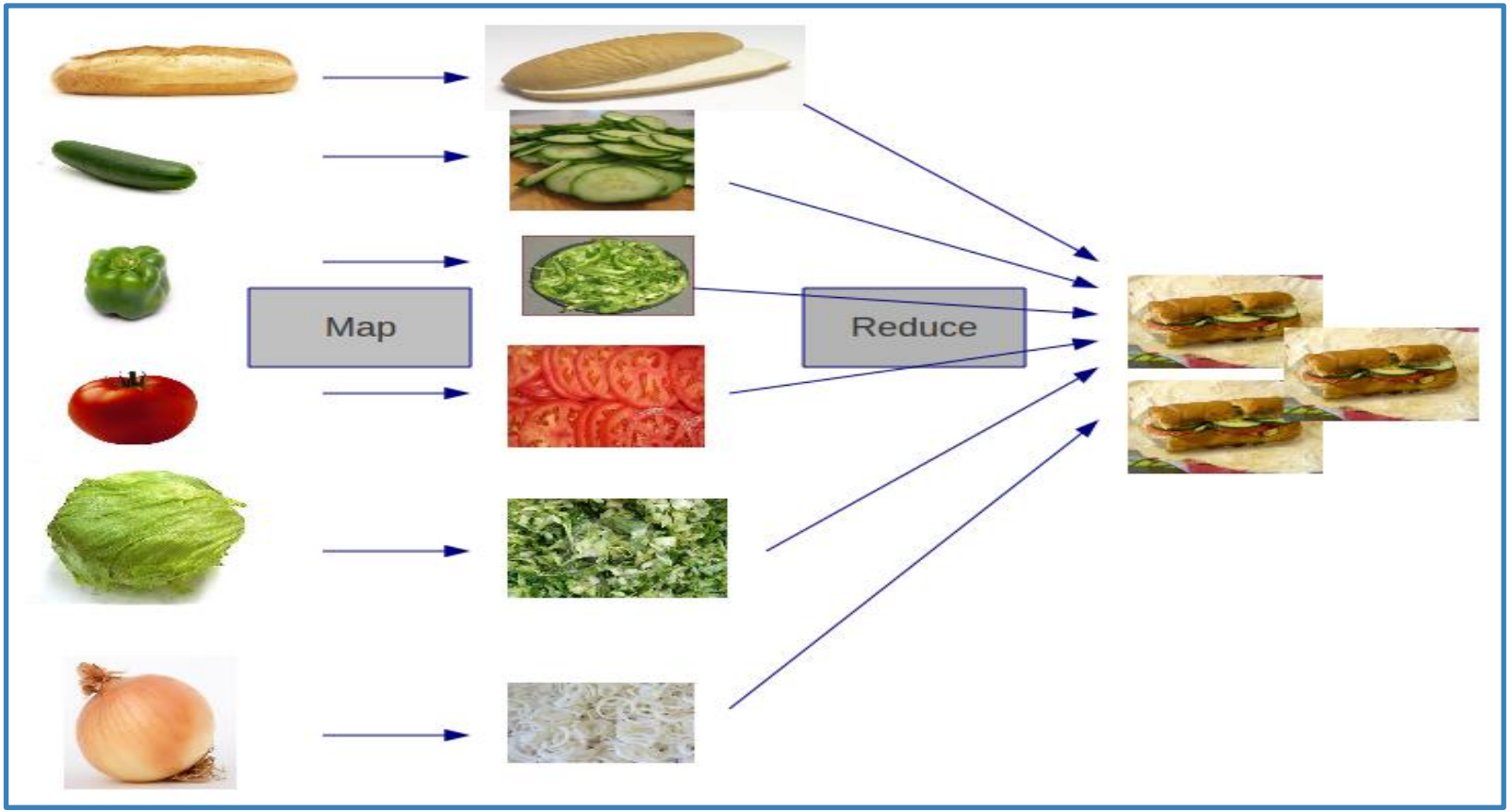
```
function map(String name, String document):
        // name: document name
        // document: document contents
        for each word w in document:
                emit (w, 1) // emit(key, value)
```

Reduce

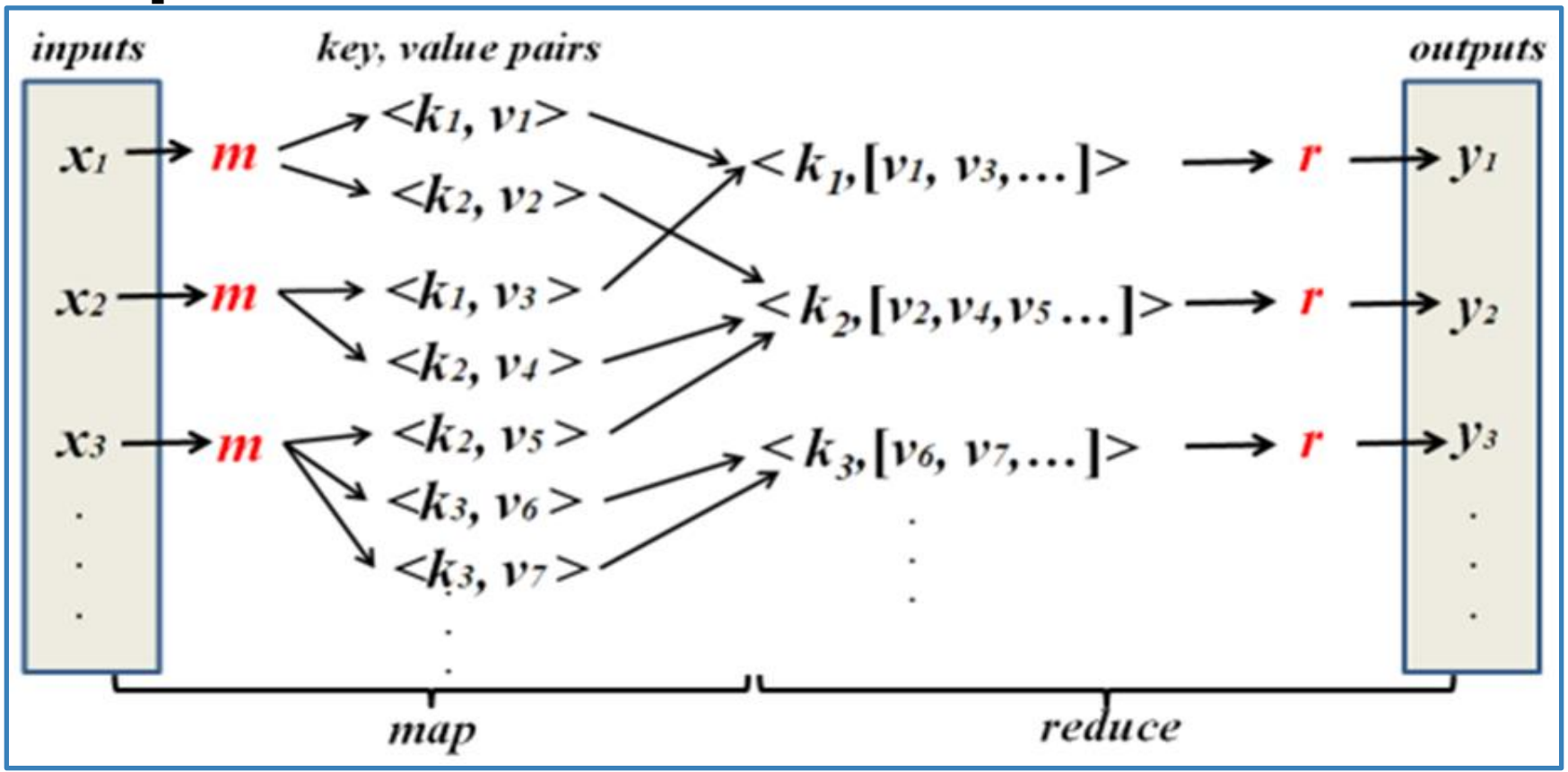
- Map aşamasından sonra master node mapper node'lardan aldığı alt problemlerin sonuçlarını key'lere göre gruplandırır.
- Key'lere göre gruplandırılmış sonuçları farklı reducer node'lara gönderir.
- Her reducer node aynı key'e ait alt sonuçları kullanarak o key'e ait genel sonucu üretir.

```
function reduce(String word, Iterator partialCounts):
        // word: a word
        // partialCounts: a list of aggregated partial counts
        sum = 0
        for each pc in partialCounts:
                sum += ParseInt(pc)
        emit (word, sum)
```

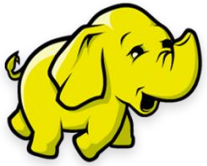
Map-Reduce for Babies



Map-Reduce for Academicians

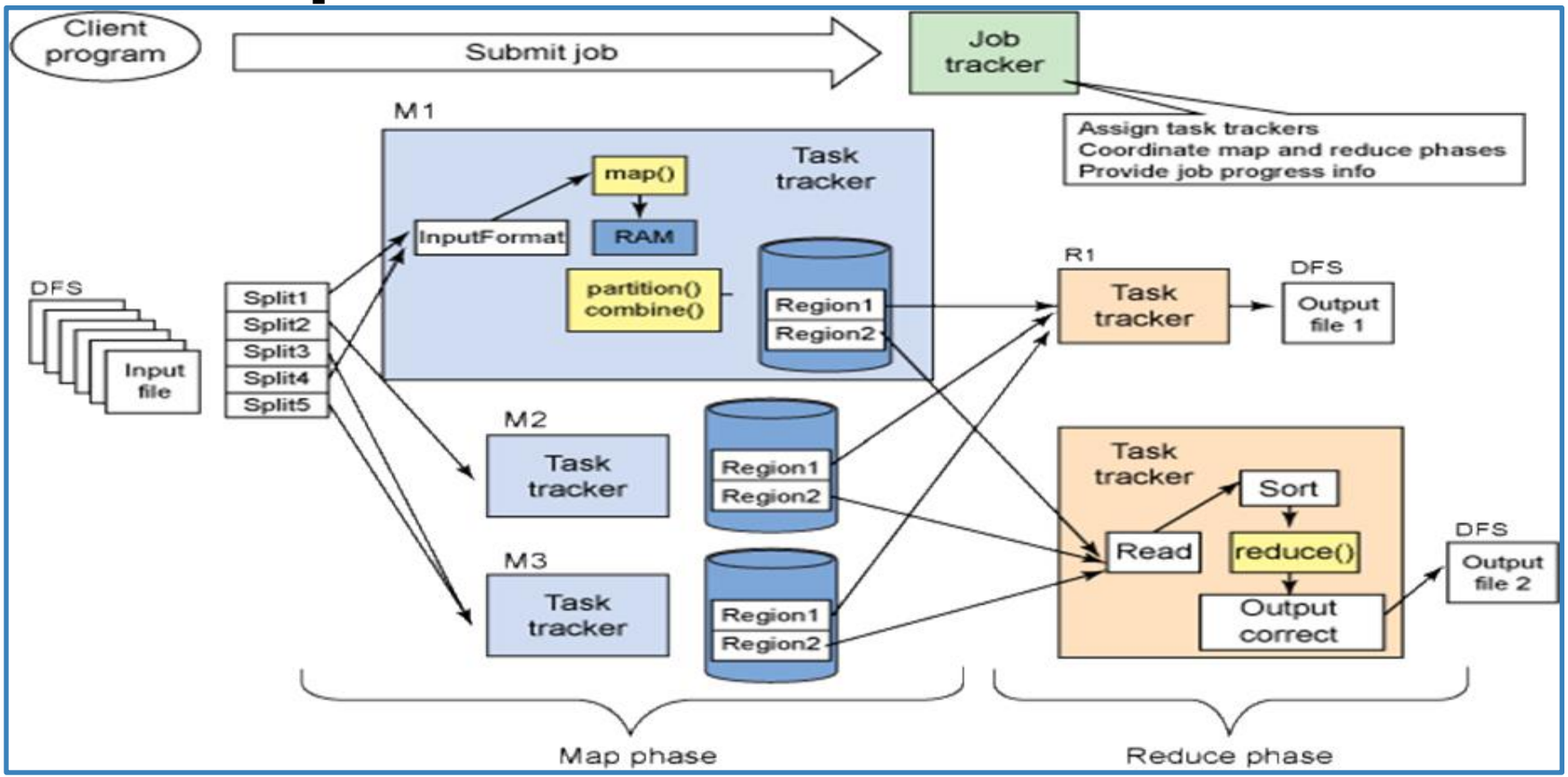


Hadoop

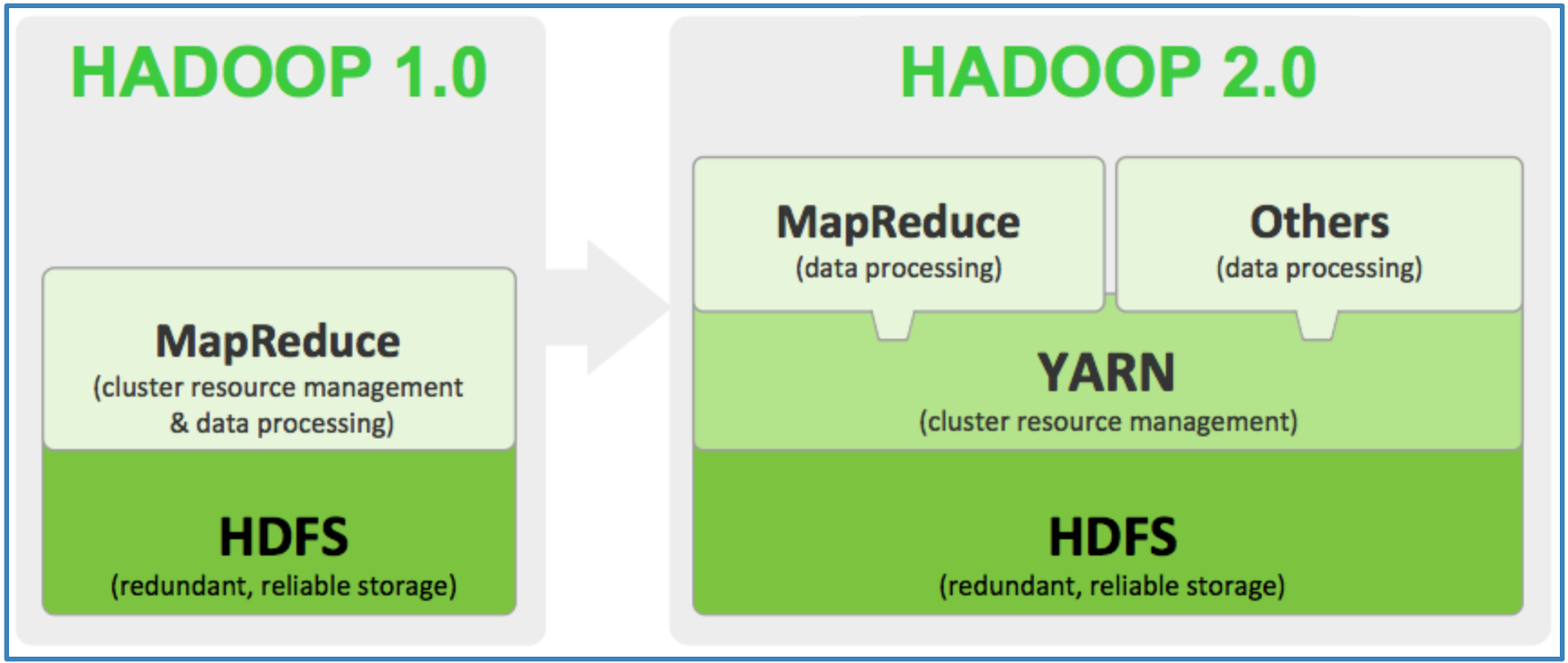


- Distributed ve Fail-Safe veri depolama ve işleme için kullanılır
- Open-Source under Apache License
- Temelde 2 bileşenden oluşur.
  - HDFS: Distributed ve reliable file system bileşenidir. Data replicate biçimde farklı node'larda tutulur ve gerektiğinde bozulan replike datalar otomatik olarak onarılır.
  - MapReduce: Distributed ve fault-tolerant veri işleme bileşenidir. Tanımlanan "Mapper" ve "Reducer" task'ların distributed olarak işletilmesini ve monitor edilmesini yönetir.

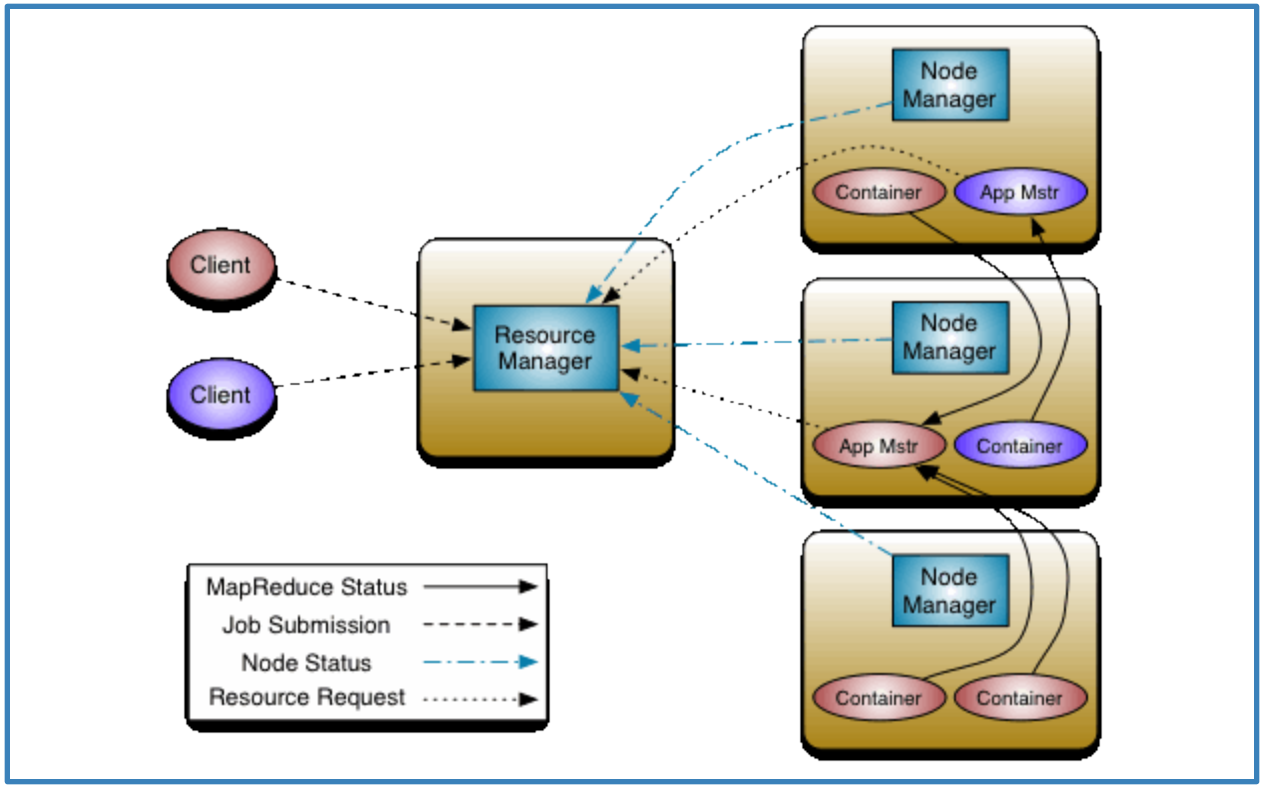
Hadoop 1 - Architecture



Hadoop 1 vs Hadoop 2 YARN



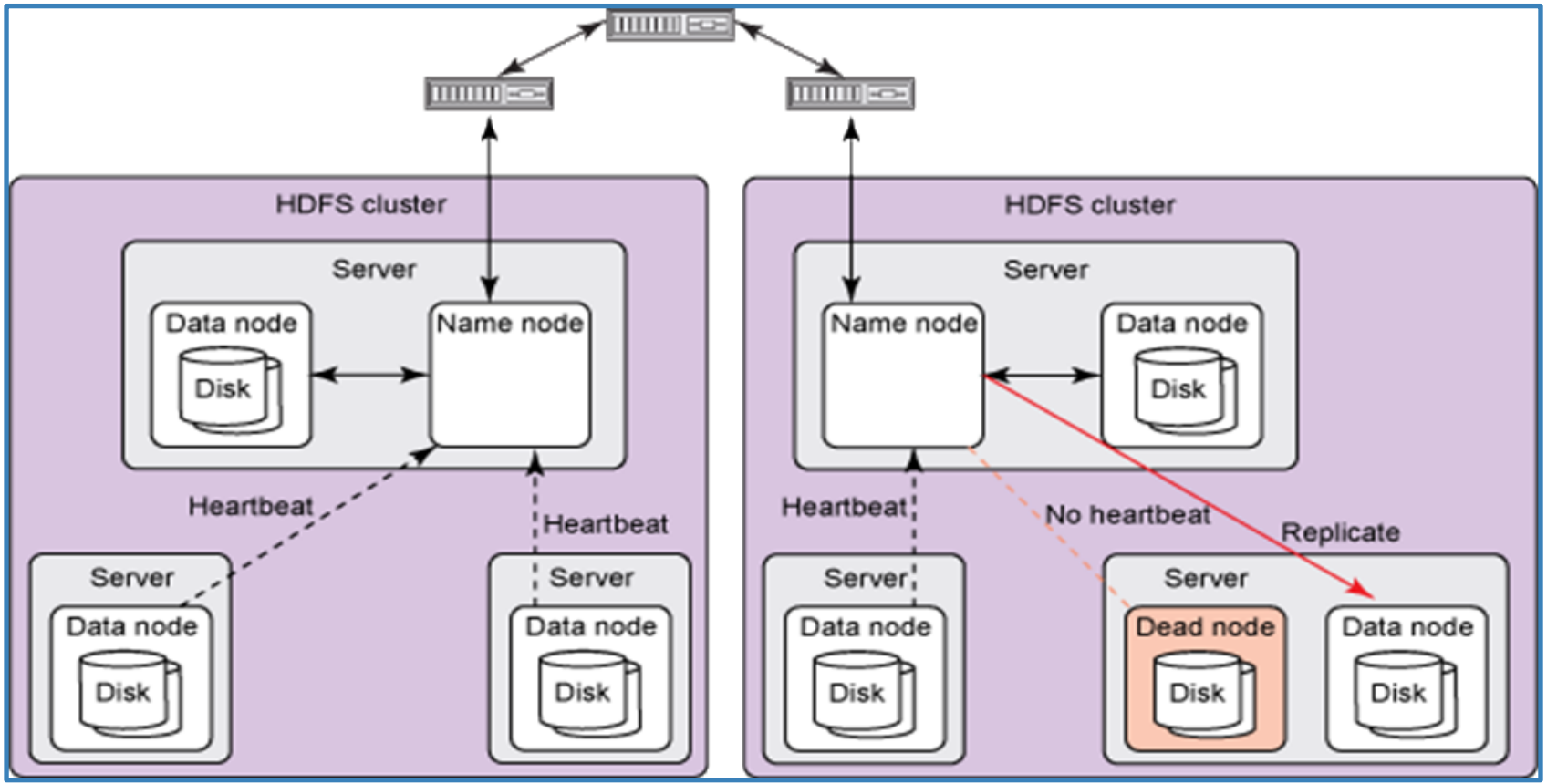
Hadoop 2 (YARN) - Architecture



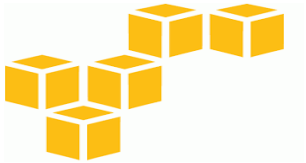
HDFS

- HDFS, Google File System'den esinlenilerek tasarlanmış distributed ve fault-tolerant bir file system'dir.
- Veri bloklara ayrılarak replike'ler halinde cluster üstünde distributed olarak tutulur.
- Temelde 2 tip node vardır:
  - Name Node: File system'in metadata'sını tutar
  - Data Node: Verinin kendisini tutar

HDFS Architecture



AWS



- AWS, Amazon’un sunduğu scalable, reliable, distributed cloud computing servisleridir.
  - Storage
  - Computing
  - Big Data (Map-Reduce, Streaming, Data Pipe Line, ...)
  - Network
  - ...
- Big Data için temel AWS bileşenleri:
  - AWS S3
  - AWS EC2
  - AWS EMR

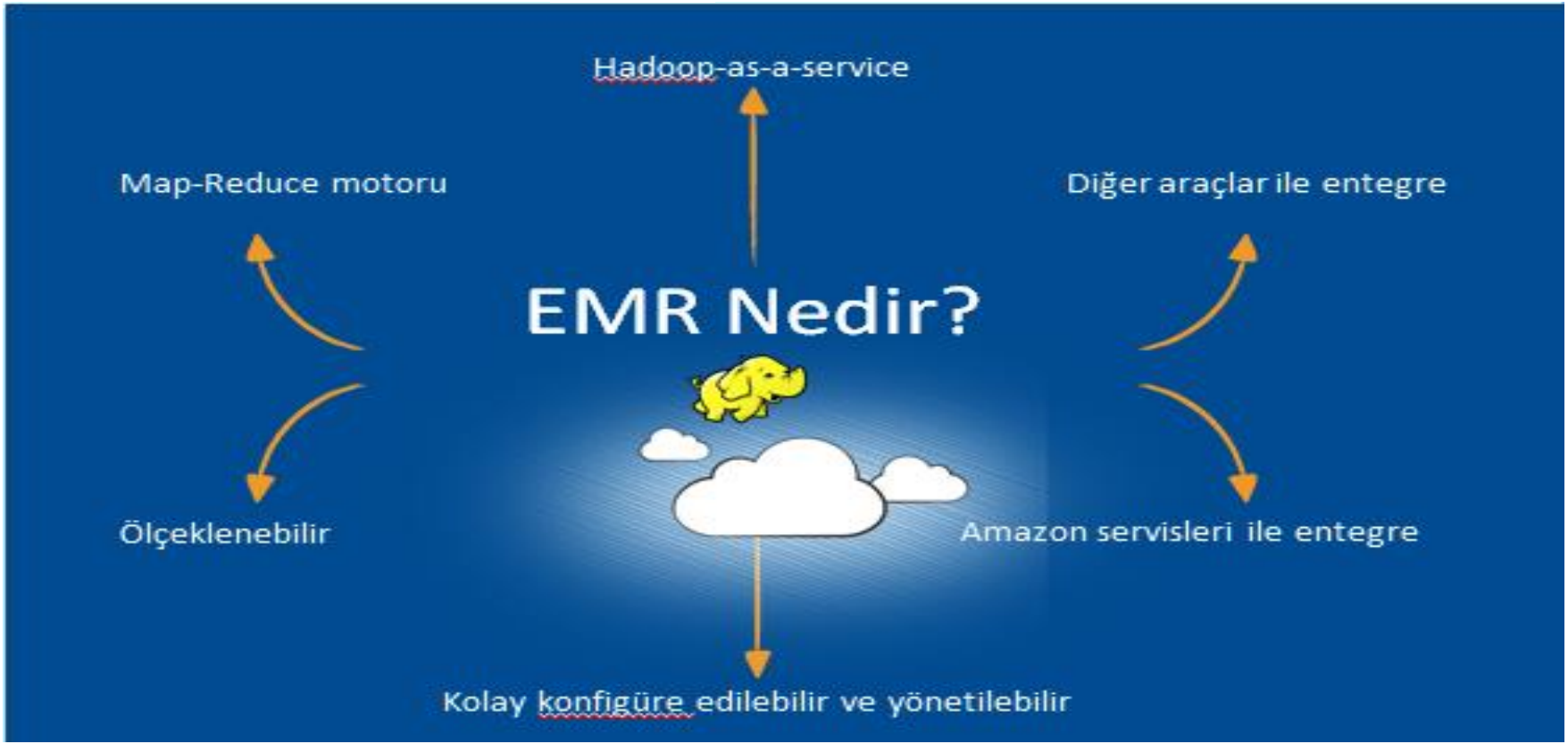
AWS - S3

- Amazon’un Distributed File System servisidir.
- 1 byte’dan 5 terabyte’a kadar veri tutan object yazılıp okunabilir.
- Object’ler bucket’lar içinde tutulur.
- Tutulan veriler için 99.999999999% dayanıklılık ve 99.99% devamlılık için tasarlanmıştır.
- Veriler şifrelenerek tutulabilir ve veri upload/download işlemleri de güvenli bağlantı üstünden yapılabilir.

AWS EC2

- Amazon’un scalable CAAS (computing as a service) servisidir.
- Esnek: Kapasite kolay bir şekilde arttırılıp azaltılabilir.
- Biçimlendirilebilir: Birçok instance tipi (CPU, Memory, Storage), işletim sistemi ve yazılım paketi destekliyor
- Güvenilir: Her Amazon EC2 Region’da 99.95% oranında kullanılabilirlik
- Düşük Maliyet: Reserved Instance ve Spot Instance

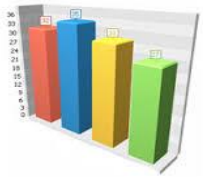
AWS - EMR


Hadoop-as-a-service
Map-Reduce motoru
Diğer araçlar ile entegre
EMR Nedir?
Ölçeklenebilir
Amazon servisleri ile entegre
Kolay konfigüre edilebilir ve yönetilebilir

Demo

- Hadoop üstünde çalışan 2 Map-Reduce demosu gösterilecektir:
  - Number Frequency
  - Log Data Processing
- Yazılan Hadoop uygulamaları build edilip “jar” şeklinde paketlenir.
- Daha sonra bu paketlenen uygulama AWS S3 üstüne upload edilir.
- AWS S3’e upload edilen uygulama burada AWS EMR servisi ile çalıştırılır.
- AWS EMR servisi üstünde çalışan Hadoop uygulaması kendisine AWS S3 üstünde belirtilen dizindeki veriler üstünde işlem yapar.
- Sonuçları yine kendisine belirtilen AWS S3 üstündeki dizine yazar.

Demo 1 - Number Frequency



- 128’er MB’lık 9 dosyadan oluşan bir dizindeki tüm dosyalarda geçen sayıların ve bu sayıların kaç defa geçtiğini bulan bir Map-Reduce uygulamasıdır.

- Soru açıklaması: https://github.com/serkan-ozal/ankarajug-bigdata-demo/blob/master/README.md#demo-1

- Map-Reduce uygulaması kodları: https://github.com/serkan-ozal/ankarajug-bigdata-demo/tree/master/src/main/java/tr/com/jug/ankara/bigdata/demo/mapreduce/numberfreq

Demo 2 - Log Data Processing

- 128’er MB’lık 9 dosyadan oluşan bir dizindeki tüm dosyalarda geçen log kayıtlarını ünceleyip belirtilen başlangıç ve bitiş tarihleri arasındaki log kayıt sayısını bulan bir Map-Reduce uygulamasıdır.

- Soru açıklaması: https://github.com/serkan-ozal/ankarajug-bigdata-demo/blob/master/README.md#demo-2

- Map-Reduce uygulaması kodları: https://github.com/serkan-ozal/ankarajug-bigdata-demo/tree/master/src/main/java/tr/com/jug/ankara/bigdata/demo/mapreduce/logdata

Teşekkürler