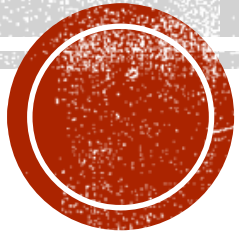
UVOD U ANALITIKU VELIKIH PODATAKA

MAPREDUCE PARADIGMA

Aleksandar Kartelj

Matematički Fakultet Univerziteta u Beogradu



UVODNI POJMOVI

Definicija, primeri upotrebe, resursi i tehnike

DEFINICIJA VELIKIH PODATAKA (ENGL. BIG DATA)

- Veliki podaci predstavljaju kolekcije podataka koje imaju:
  - veliku dimenziju (engl. high-**volume**);
  - brzo se stvaraju (engl. high-**velocity**);
  - i imaju veliku raznovrsnost (engl. high-**variety**).

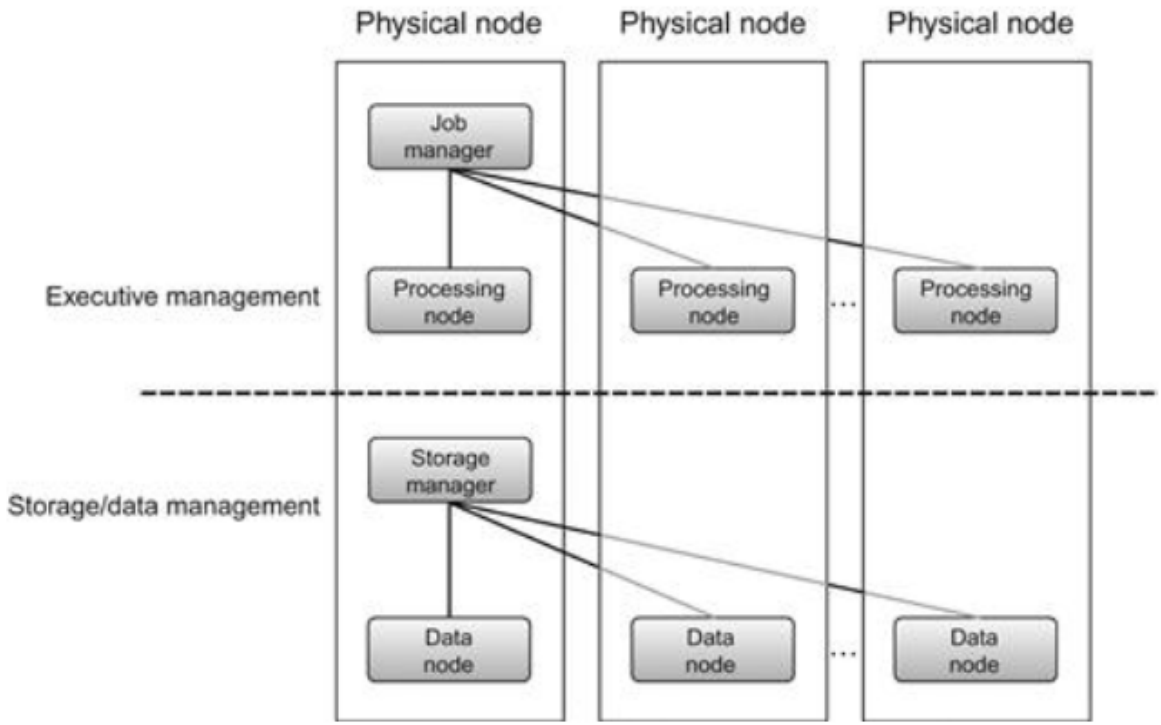
- Fokus je na osmišljavanju efikasnih tehnika:
  - reprezentacije i skladištenja;
  - procesiranja i donošenja zaključaka!

PRIMERI UPOTREBE VELIKIH PODATAKA

- Sistemi za predlaganje na Internetu (engl. recommendation systems)
- Nadgledanje socijalnih mreža
- Personalizovana pretraga Interneta
- Detekcija anomalija
- Nadgledanje rada mobilnih, senzorskih mreža i drugih mreža
- Navigacija u saobraćaju
- Semantička analiza silka i video snimaka
- ...

KLJUČNI RESURSI

- Procesor
- Radna memorija
- Memorija za skladištenje
- Mreža



"Big Data Analytics", David Loshin, 2013

TEHNIKE ZA SAVLADAVANJE VELIKIH PODATAKA

- Tradicionalne tehnike prilagođene novim izazovima:
  - Masivni paralelizam (pre svega paralelizam nad podacima);
  - Ogromna skladišta za podatke;
  - Distribucija podataka (+replikacija);
  - Mreže visokih performansi (protok, RTT);
  - Izračunavanja visokih performansi (klasteri {C|G}PU, globalna izračunavanja);
  - Upravljanje zadacima i nitima;
  - Tehnike pretraživanja podataka;
  - Tehnike mašinskog učenja;
  - Tehnike vizuelizacije podataka;
  - …

MALO ISTORIJE...

Herman Holerit, …, Google pretraga

ISTORIJA VELIKIH PODATAKA

- Neki autori počinju događajima od P.N.E, ali je to malo isforsirano…
- 1880 – Popis u Americi: sa 10 godina obrade na 3 meseca. (IBM)
- 1926 – Nikola Tesla:
  - Predviđa pojavu mobilnih telefona i bežičnog prenosa informacija;
  - „Cela Zemlja će biti poput jednog velikog mozga“.
- 1958 – IBM: uvođenje koncepta poslovne inteligencije (BI).
- 1991 – WWW
- 1997 – Michael Lesk: teoretska diskusija o količini informacija (12 eksabajta= 12 x 10^{18}?).

“A brief history of big data everyone should read”, Bernard Marr, 2015

KOLIKO INFORMACIJA?

- 2000: Google izveštaj o količini informacija:
  - ~1.5 milijardi gigabajta = 1.5 x 10^{18} B;
  - Ili ~ 250MB po svakoj osobi.
- Ovo nije baš bilo jednostavno oceniti, uzimali su se u obzir:
  - Duplikati;
  - Kompresija;
  - Arhivirani medijumi;
  - TV i Radio sadržaji (emitovani – neskladišteni sadržaj...);
  - Rast i pad određenih vidova komunikacije...
- Koliko je informacija u Univerzumu?

ISTORIJA VELIKIH PODATAKA (2)

- 2004: Jeffrey Dean i Sanjay Ghemawat,
  MapReduce: Simplified Data Processing on Large Clusters
  - **Obavezno pročitati!**
  - **Implementacija MapReduce modela za velike klastere mašina**
  - **Ključni aspekt Google pretrage**
  - **Paralelizacija**
  - **Otpornost na otkaze**
  - **Distribucija podataka**
  - **Raspoređivanje opterećenja**

MAPREDUCE PRIMER

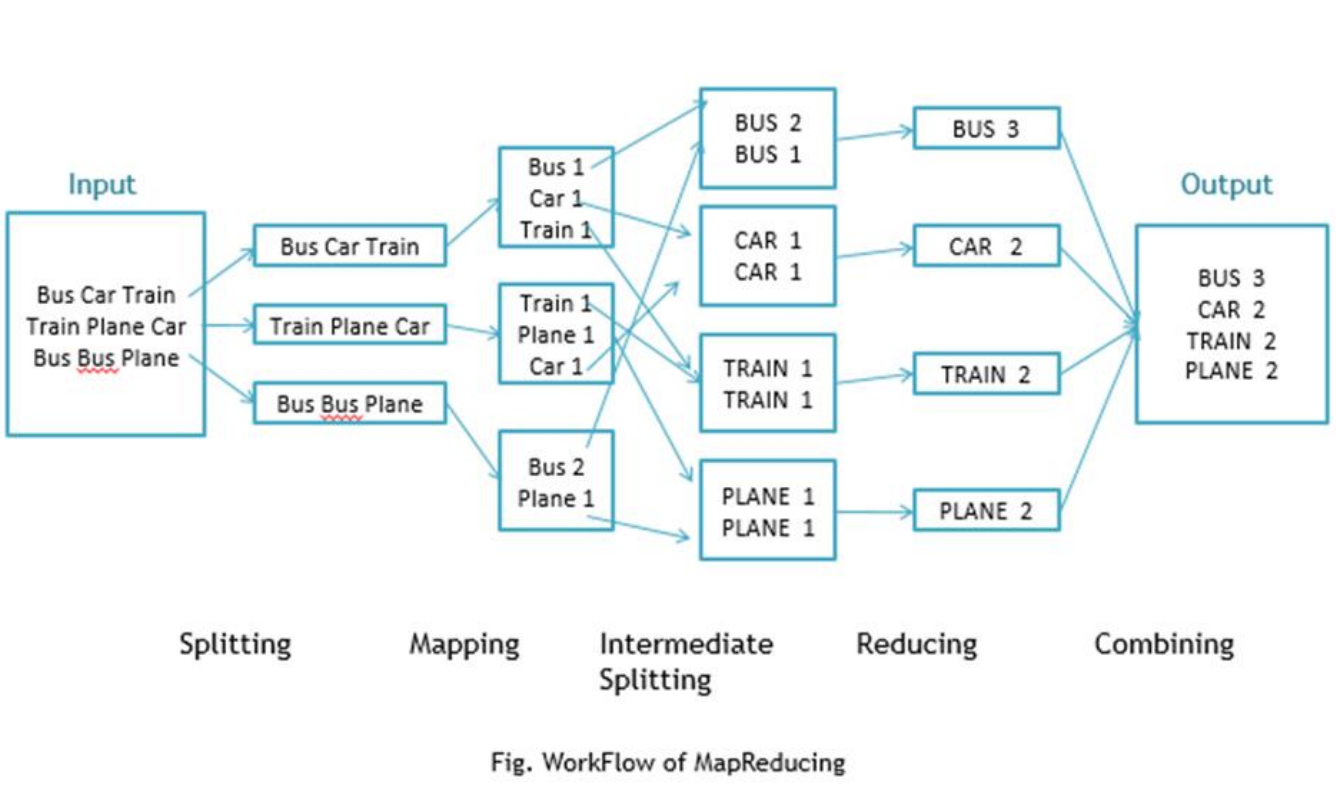


Fig. WorkFlow of MapReducing

https://dzone.com/articles/word-count-hello-word-program-in-mapreduce

ISTORIJA VELIKIH PODATAKA (3)

- 2008: serveri proizvode 9.57 x 10^{21}B ili oko 12GB po osobi.
- 2010: Dva dana informacija = Ukupno informacija do 2003. godine.
- Procena je da će do 2020. ovaj broj biti 5200 GB?
  - Zašto, koje su to informacije?

23

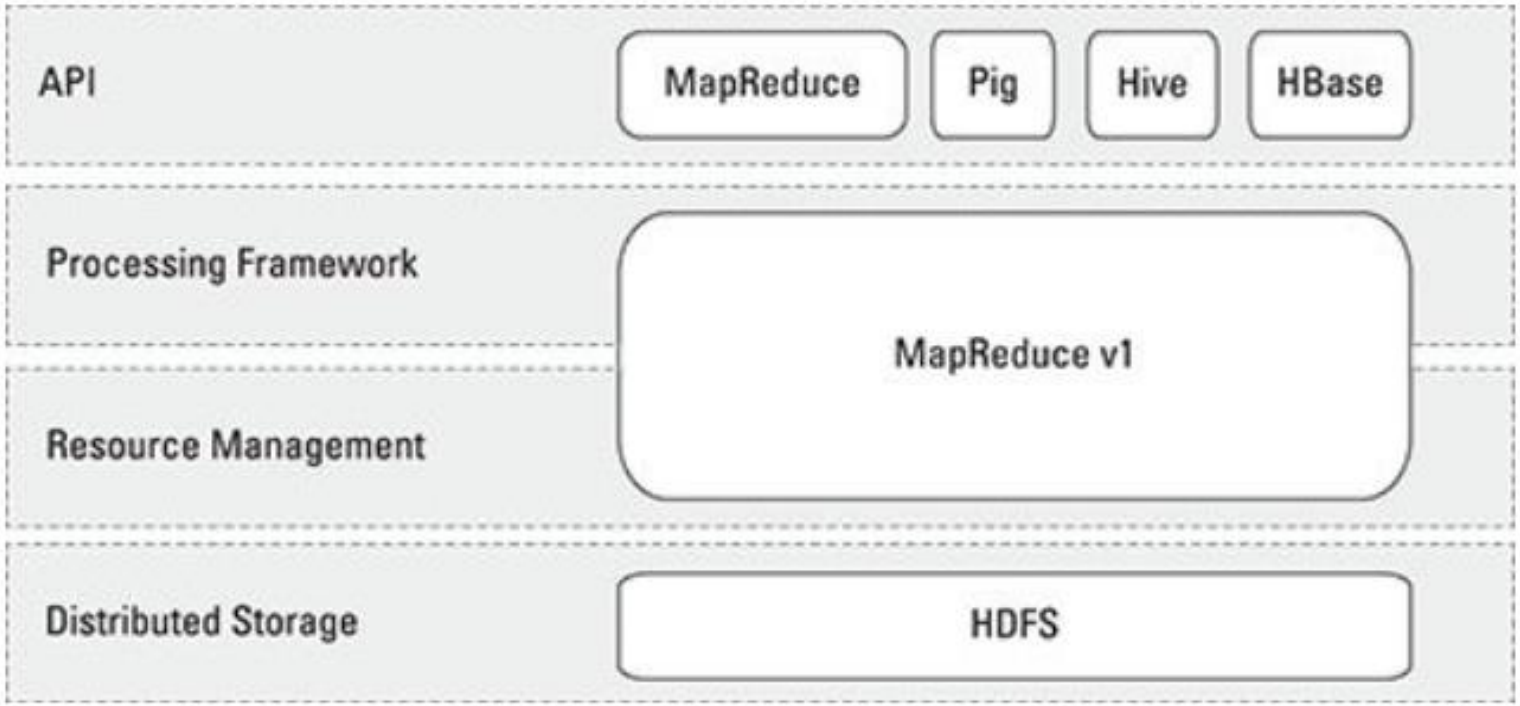
MAPREDUCE PLATFORME ZA VELIKE PODATKE

Hadoop i Spark

RAĐANJE HADOOP PLATFORME

- 2003: Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung, **The Google File System (GFS).**
- 2004: Jeffrey Dean i Sanjay Ghemawat , **MapReduce: Simplified Data Processing…**
- 2006: Doug Cutting , Mike Cafarella, započet projekat Hadoop.
- 2006: Spojeni NDFS (GFS)+ MapReduce osnovni elementi.
- 2006: Verzija Hadoop 0.1.0.
- 2006: Hadoop sortirao 1.8 TB podataka na 188 jezgara za 47.9 sati.
- 2007: Yahoo koristi Hadoop na 1000 mašina.
- 2008: Yahoo indeks pretrage na 10.000 Hadoop jezgara.
- 2009: Hadoop sortira 1TB za 62 sekunde.

KLJUČNI ELEMENTI HADOOP PLATFORME



HDFS MOTIVACIJA

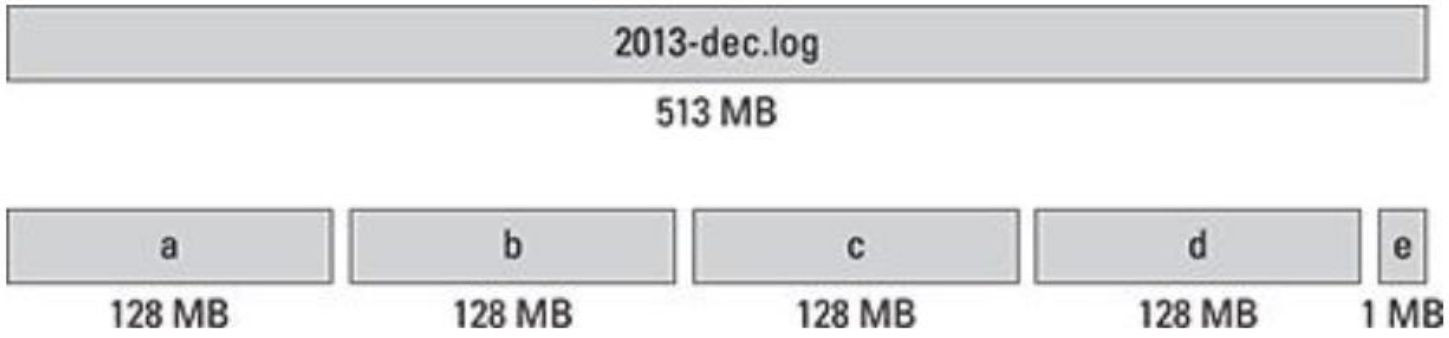
- Engl. Hadoop Distributed File System
- Zasnovan na radu iz 2003. godine:
  Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung,
  The Google File System (GFS)
- Model izvršavanja u kojem se preferiraju obrada velikih podataka.
  - Pretpostavlja se da su datoteke jako velike.
  - Da se čitaju linearno najčešće.
  - Da nema puno izmena nad njima.
- Ne implementira keširanje zbog ovakvih pretpostavki.
  - Iako postoji keširanje na nivou lokalnih sistema datoteka (pod Linux čvorovima)

HDFS ORGANIZACIJA

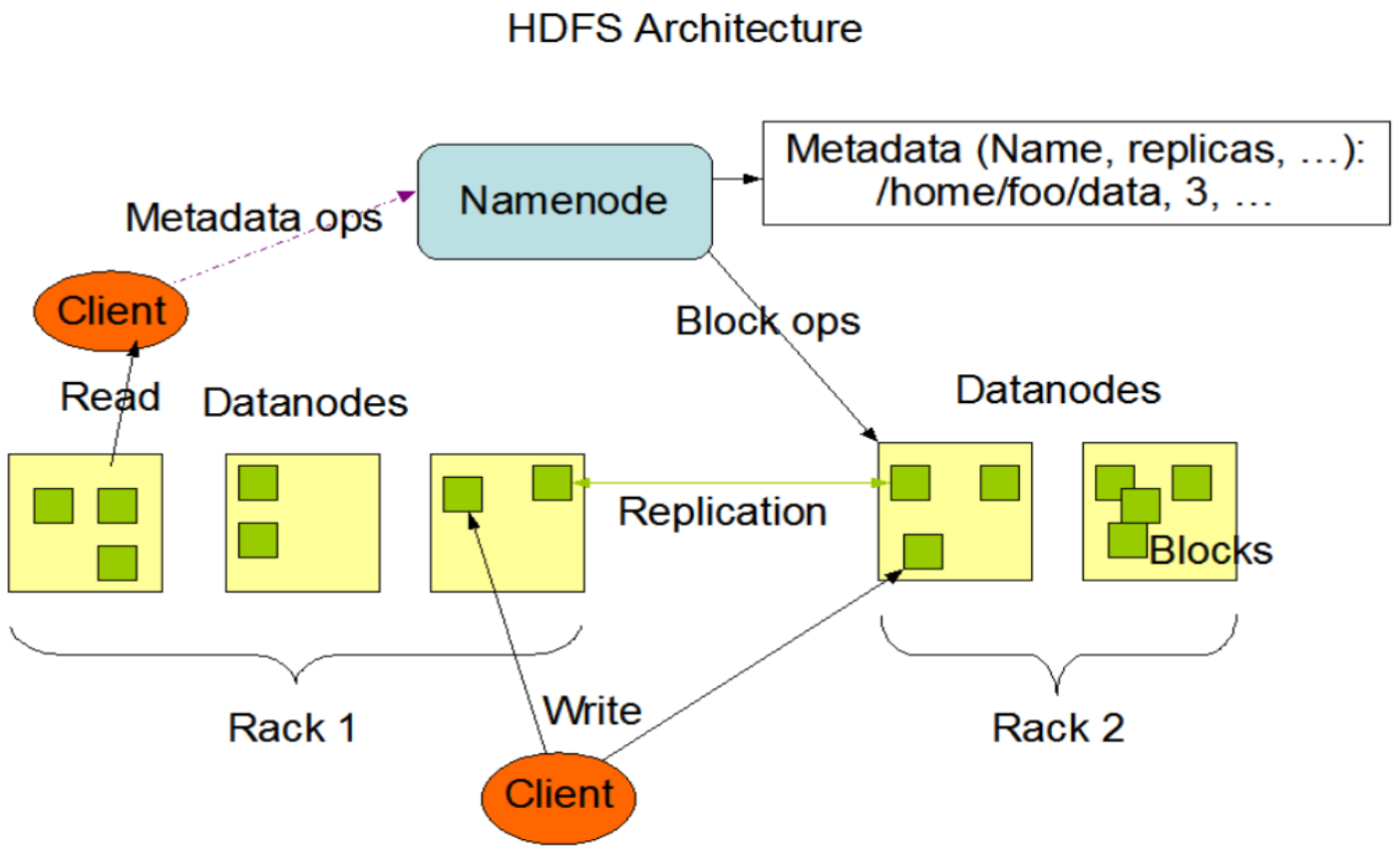
- Hadoop najbolje radi sa srednje velikim brojem ekstremno velikih datoteka >500MB.
- Poželjan je model upotrebe: „piši jednom, čitaj često“.
- Dozvoljene operacije:
  - kreiranje nove datoteke;
  - dodavanje na kraj datoteke;
  - brisanje;
  - preimenovanje.
- Omogućava konkurentno nadovezivanje.

HDFS BLOKOVI

- Datoteka se deli na blokove (npr. 64MB)
  i kopira na višestrukim čvorovima (podrazumevano 3).
- Ovo liči na straničenje u OS samo što je dimenzija veća
  i postoji redudantnost.
- Svi blokovi imaju pridružene metapodatke
  o kojima brine čvor(ovi) za imenovanje.
- Dizajnirano tako da radi na skali petabajta 10^{15}B.



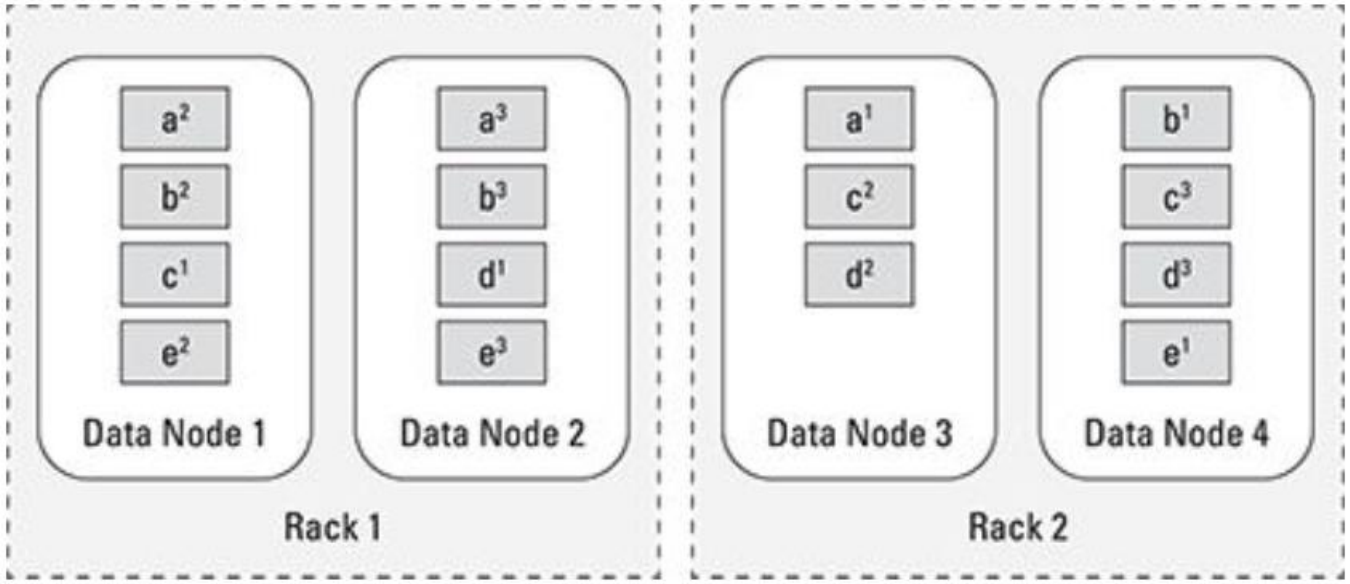
HDFS ARHITEKTURA



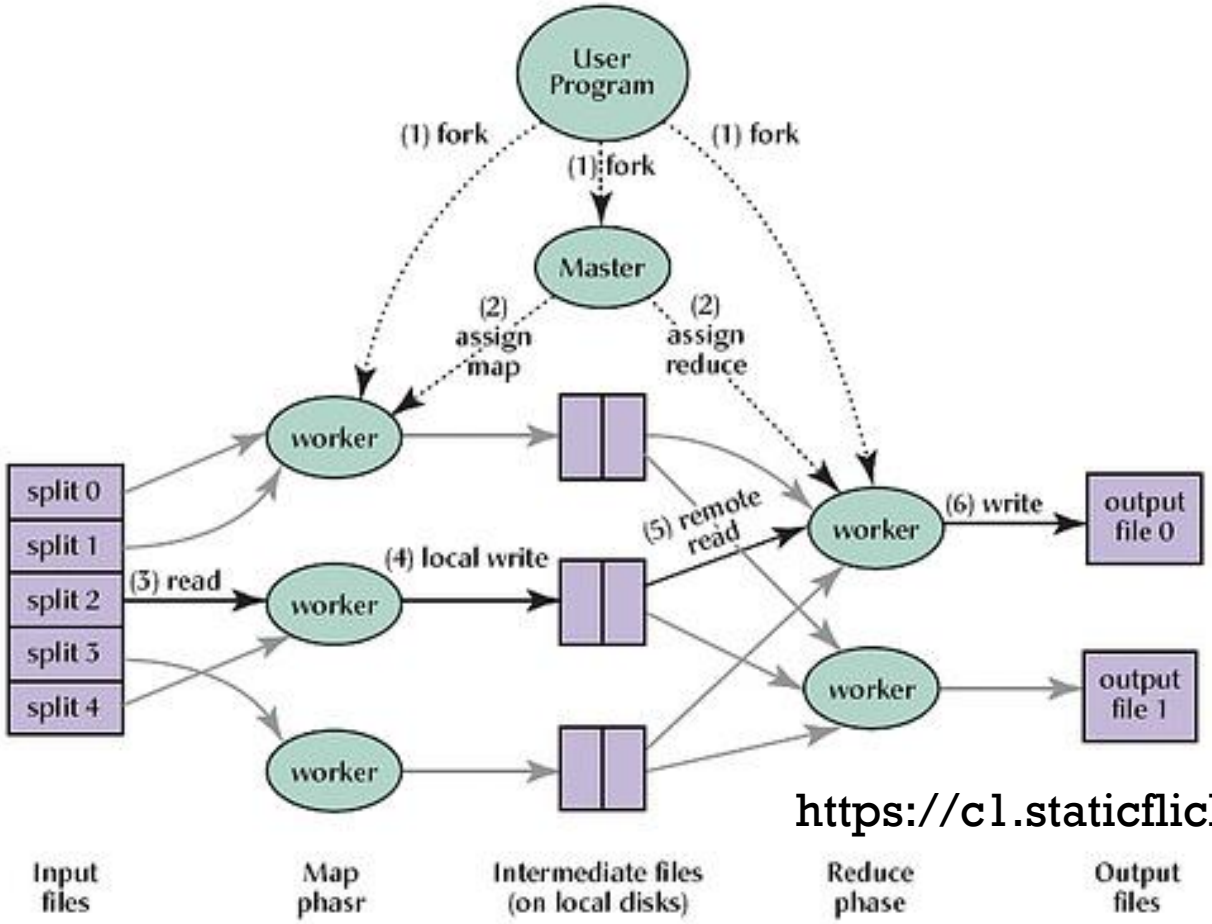
http://hortonworks.com/hadoop/hdfs/

HDFS REPLIKACIJA

- HDFS pokušava da postavi replike
  tako da smanji mogućnost gubljenja podataka.
- Poželjno da se replike nalaze na različitim jedinicama otkaza
  tj. nezavisnim skupovima čvorova.



TOK IZVRŠAVANJA



https://c1.staticflickr.com/3/2133/2179187226_e2e107e0cd.jpg

YARN

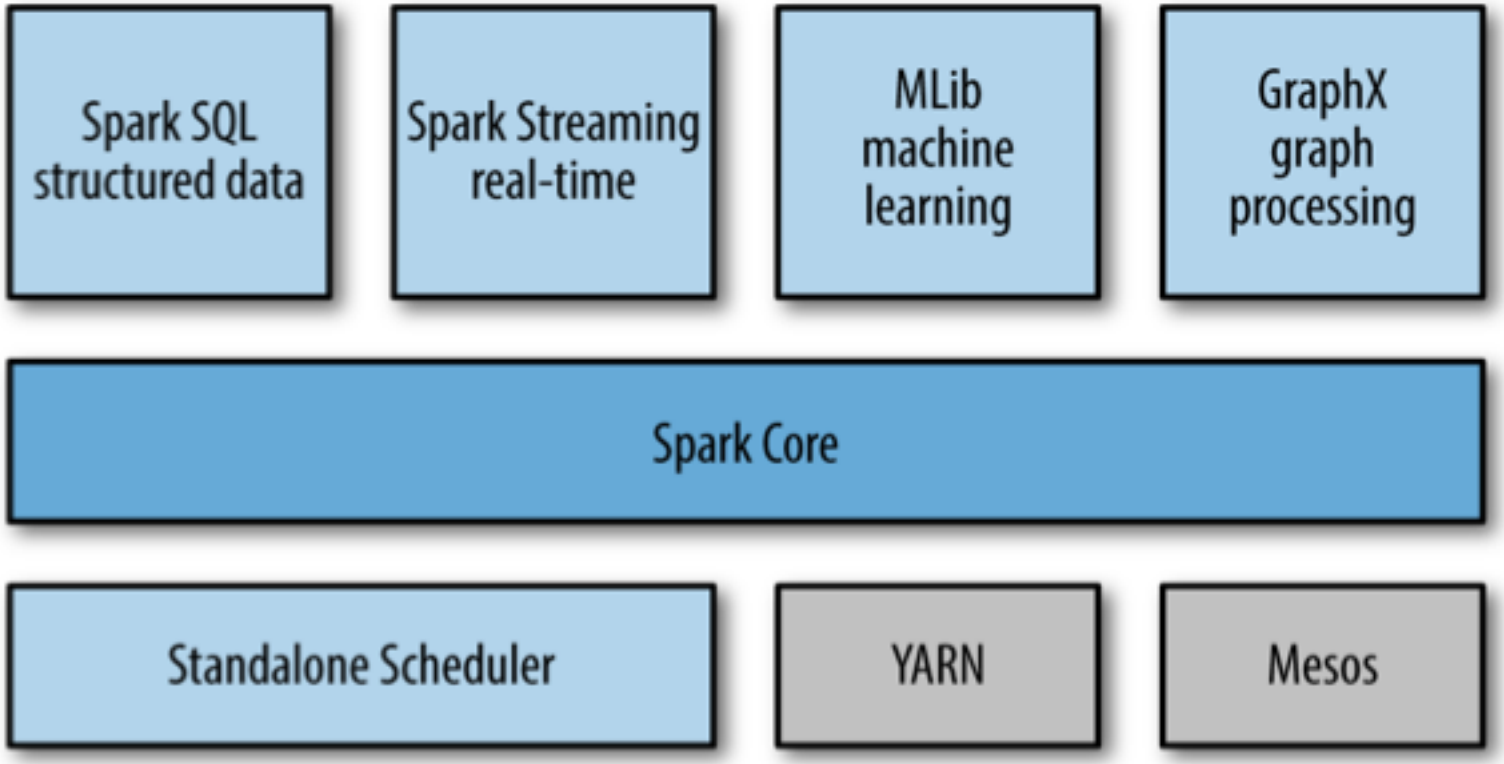
- Enlg. Yet Another Resource Negotiator
- Obezebeđuje planiranje resursa na Hadoop-u:
  - dodeljivanje CPU vremena,
  - memorije,
  - mrežnog protoka,
  - itd.
- Alat koji obezbeđuje i drugim programskim okvirima da pozivaju Hadoop.

SPARK

- Implementacija MapReduce paradigme novijeg datuma (2014).
- Većina izračunavanja u radnoj memoriji.
  - U nekim primerima i 100x brži od Hadoop-a, npr. obradi tokova podataka.
- Lakši za razvoj aplikacija: Python, Java, Scala, R.
- Ne poseduje distribuirani sistem datoteka poput HDFS.
  - Ima neki vid sistema datoteka,
    ali bez replikacije na disku pa nije toliko otporan na otkaze.
- Ipak, način organizovanja podataka u memoriji (RDD)
  obezbeđuje određenu otpornost na otkaze.
- Može se koristiti u kombinaciji Spark+Yarn+Hadoop.

https://www.kdnuggets.com/2015/08/big-data-question-hadoop-spark.html

SPARK RAZVOJNI OKVIR



SPARK CORE

- Osnovne funkcionalnosti:
  - upravljanje zadacima;
  - upravljanje memorijom;
  - oporavak od otkaza;
  - interakcija sa sekundarnom memorijom;
- Glavnu programsku apstrakciju čine tzv. **RDD**
  „otporni distribuirani skupovi podataka “
  (engl. resilent distributed datasets)
- 2012: Matei Zaharia i ostali,
  Resilient Distributed Datasets:
  A Fault-Tolerant Abstraction for In-Memory Cluster Computing

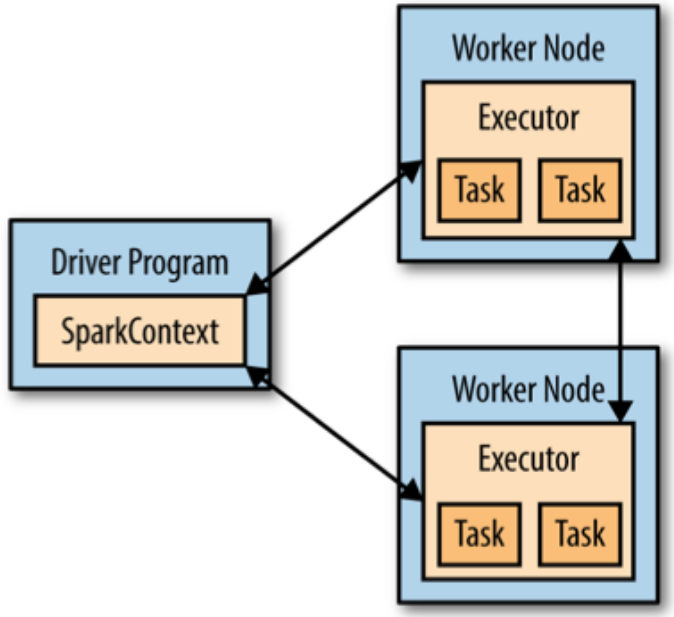
RDD

- Formalno, RDD je nepromenljiva izdeljena kolekcija podataka.
  RDD se može kreirati samo upotrebom neke determinističke operacije nad:
  1. podatkom u stabilnom skladištu (npr. učitavanje sa datoteke na disku);
  2. drugim RDD podatkom.
- Umesto ažuriranja podataka u sistemu datoteka poput HDFS,
  RDD pamti spisak transformacija.
- Kako je u pitanju paralelizam nad podacima,
  sva jezgra izvršavaju istu operaciju nad različitim ulaznim podacima.
- U slučaju otkaza samo se izvrše sve potrebne transformacije nad odgovarajućim parčetom podataka.
- RDD radi u glavnoj memoriji što ga čini pogodnim za:
  - izvršavanje iterativnih algoritam
  - i interaktivnih algoritama za istraživanje podataka.

TOK IZVRŠAVANJA

- Gledano iz perspektive programera,
  Spark program pokreće dva tipa programa:

1. Glavni program
   (engl. Driver program)
   koji pokreće različite paralelne operacije nad klasterom i vrši sažimanje parcijalnih rezultata;
2. Izvršni programi (engl. wokers/executors) koji izvršavaju iste operacije nad različitim particijama podataka.



PRIMER – BROJANJE REČI

```
import sys
from operator import add

from pyspark import SparkContext


if __name__ == "__main__":
    if len(sys.argv) != 2:
        print >> sys.stderr, "Usage: wordcount <file>"
        exit(-1)
    sc = SparkContext(appName="PythonWordCount")
    lines = sc.textFile(sys.argv[1], 1)
    counts = lines.flatMap(lambda x: x.split(' ')) \
                  .map(lambda x: (x, 1)) \
                  .reduceByKey(add)
    output = counts.collect()
    for (word, count) in output:
        print "%s: %i" % (word, count)

    sc.stop()
```

PRETPOSTAVKE ZA PRIMENU MAPREDUCE PARADIGME

- MapReduce paradigma je praktična kada je reč o paralelizmu nad podacima (iste operacije nad različitim podacima).
- Nisu svi problemi pogodni za rešavanje, glavni preduslovi je da je problem moguće dekomponovati na delove.
- Takođe je pogodno da su:
  - Ili čvorovi dovoljno bliski zbog malih troškova sinhronizacije tj. komunikacije;
  - Ili da su operacije dovoljno dugotrajne da kompenzuju trošak komunikacije.

PRIMERI (NE)UPOTREBE MAPREDUCE

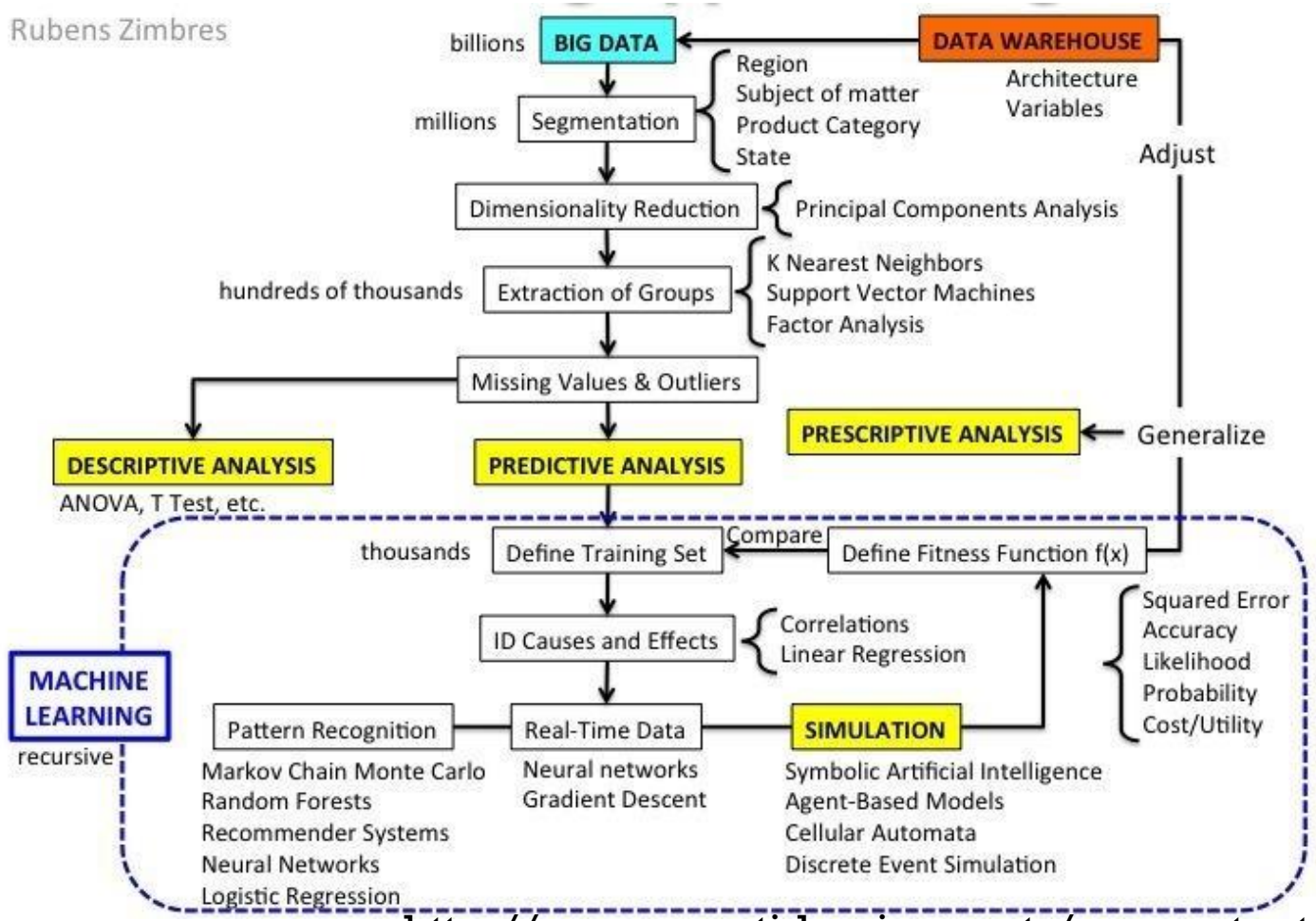
Google search, ML primene, distribuirana optimizacija?

GOOGLE INTERNET PRETRAGA – KAKO RADI?

1. Na osnovu neke početne liste Internet adresa, tzv. Internet pauci (eng. spiders) započinju rekurzivni obilazak Web-a.
   - Prilikom obilaska, oni otvaraju stranice simulirajući Internet pregledače (kao što čovek to radi).
2. Informacije o pretraženim Web adresama i njihovim pridruženim ključnim rečima, kvalitetu, svežini sadržaja itd. čuvaju na distribuiranoj mreži računara.
   Ova distribuirana mreža čini tzv. Indeks pretrage i on u slučaju Google pretraživača:
   - sadrži informacije o nekoliko stotina milijardi stranica;
   - zauzima proctor od preko 100 miliona gigabajta;
   - i nagađa se oko 650 hiljada jezgara (poslovna tajna).
3. Kada korisnik unese ključne reči u polje za pretragu i potvrdi:
   - Hiljade jezgara započinju simultano pretraživanje dela indeksa i vraćaju parcijalne rezultate koji se potom usklađuju (sortiraju) prema PageRank metrici.

http://www.zdnet.com/article/googles-650000-core-warehouse-size-computer/)

PRIMENE U MAŠINSKOM UČENJU



Rubens Zimbres, http://www.necatidemir.com.tr/wp-content/uploads/2016/09/ml-applied-to-big-data.jpeg

PRIMENE U MAŠINSKOM UČENJU (2)

- U mašinskom učenju postoje problemi koji se uklapaju u koncept paralelizma nad podacima… …Ili se bar njihovi delovi uklapaju, npr.:
  - distribuirana unakrsna-validacija
  - Ili optimizacija hiperparametara.
- Spark MLlib –distribuiranim realizacije popularnih ML algoritama
  - Regresija, klasifikacija, dimenzionu redukciju, …, i neke tehnike optimizacije.
- Spark GraphX –paralelna izračunavanja nad grafolikim podacima
- Google Cloud Natural language API
  - Prepoznavanje sentimenta, entiteta, korisničkog zadovoljstva, …

DISTRIBUIRANA OPTIMIZACIJA?

- Ima smisla distribuirati samo probleme koji su razdvojivi:
  mogu se razbiti na delove, rešiti pa uklopiti u konačno rešenje.
- Ovo naravno ne važi u kontekstu kombinatorne optimizacije
  i NP-teških problema.
- Nema smisla „napadati“ NP-težak problem MapReduce algoritmom:
  - struktura optimalnog rešenja drastično odstupa
    od strukture rešenja formiranog od optimalnih rešenja delova problema;
  - P=NP?
- Ima smisla za neke probleme iz domena konveksne
  i globalne optimizacije.

ZAKLJUČAK

- Očekuje se dramatičan porast količine podataka u budućnosti:
  - ljudi i mašine generišu sve više podataka:
    slike, video snimci, senzorski podaci u pametnim kućama, fabrikama...
  - podaci iz svemira;
  - podaci koji nastaju u istraživanju elementarnih čestica.
- Posledično, privreda i nauka se okreću induktivnom zaključivanju.
  - Naučni metod se menja, emprijski rezultati ukazuju na teorijske.
  - Privrednici imaju mogućnost da zaključuju i testiraju svoje zaključke vrlo brzo.
- Tehnologije se razvijaju, ali nedovoljno brzo za rast količine podataka.

DISKUSIJA.

kartelj@matf.bg.ac.rs

aleksandar.kartelj@gmail.com