UVOD

MapReduce

- Velik broj podataka
    - Distribuirani kroz velik broj računala
    - Unutar razumnog vremena
- Obrada za specijalne svrhe
    - Indeksirani dokumenti, dnevnik web zahtjeva

Nedostaci:

- Obradni paralelizam
- Distribucija podataka
- Tolerancija na pogrešku

Iz potrebe za rješavanjem ovih problema dolazi do novih software-a

Paralelizam:

- Velika nakupina hardvera
- Spojeni preko Etherneta ili jeftinih switch-eva

Novi software-i:

- Distribuirani datotečni sustav
- MapReduce

Primjer (brojač riječi):

- Ogromni tekstualni dokumenti
- Analiza serverskih dnevnika kako bi se pronašli popularni URL-ovi
- Brojanje koliko puta se svaka izričita riječ pojavila u datoteci

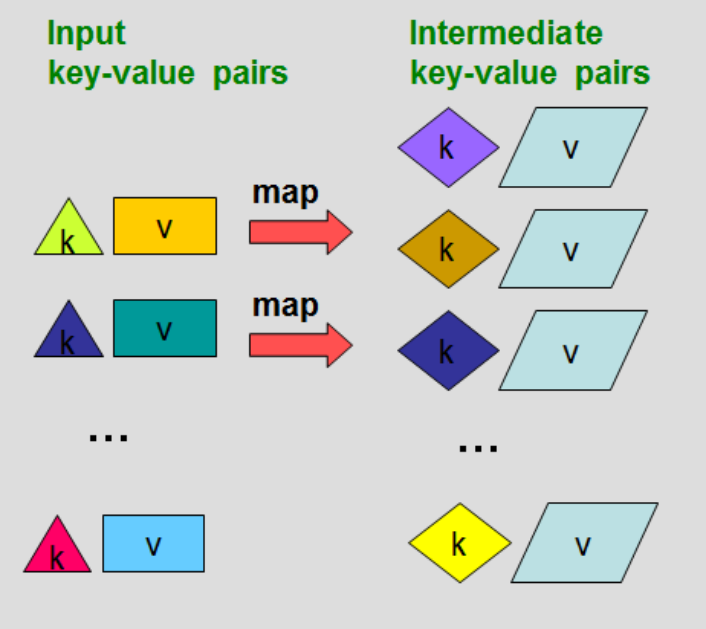
Rješenje:

- Datoteka prevelika za memoriju
- <riječ, brojač> par stane u memoriju
- words(doc.txt) | sort | uniq –c

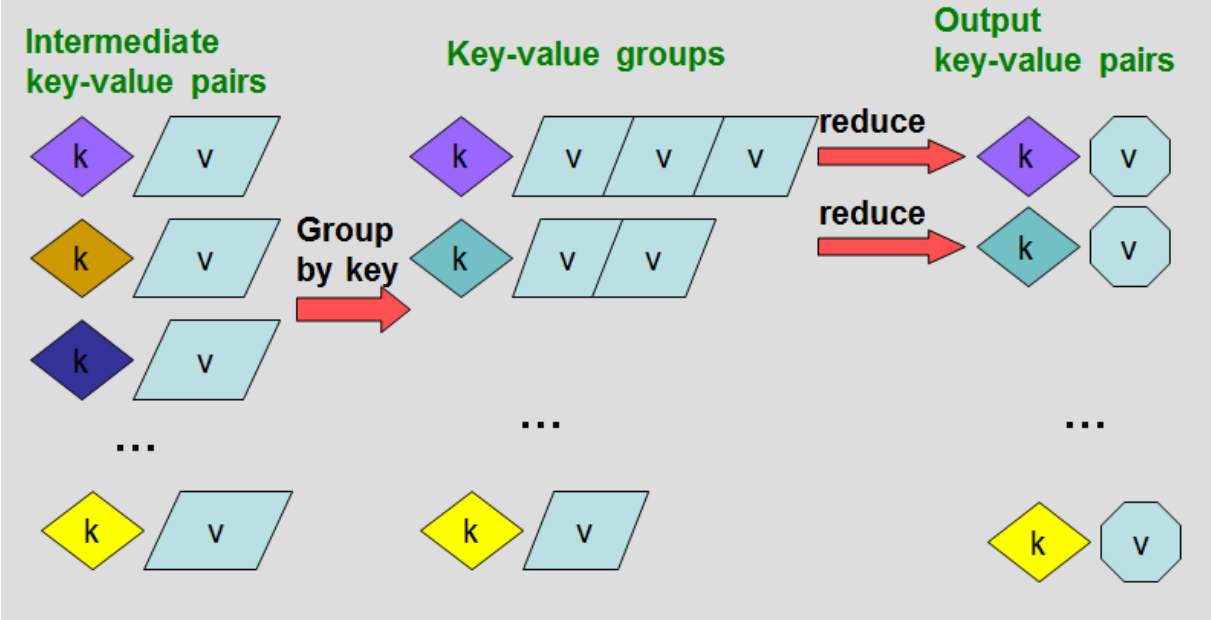
DEFINICIJA

MapReduce – programski model i povezana provedba za obradu i stvaranje velikog skupa podataka
Čitanje podataka – sekvencijalno
Map – izvlačenje vrijedne informacije (<ključ, vrijednost>)
Grupiranje i sortiranje po ključu
Reduciranje – spajanje podataka te generiranje rezultata

MAP



REDUCE



Ulaz je skup parova ključ-vrijednost

Map(k, v) → list<k' v'>

- uzima ključ-vrijednost par
- vraća skup parova ključ-vrijednost
- Map poziva za sve parove ključ-vrjednost

Reduce(k', list<v'>*)→<k", v">*

- Vrijednosti v' sa istim ključem k' se reduciraju zajedno i obrađuju se po v' slijedu
- Jedna Reduce funkcija je pozvana po jedinstvenom ključu k'

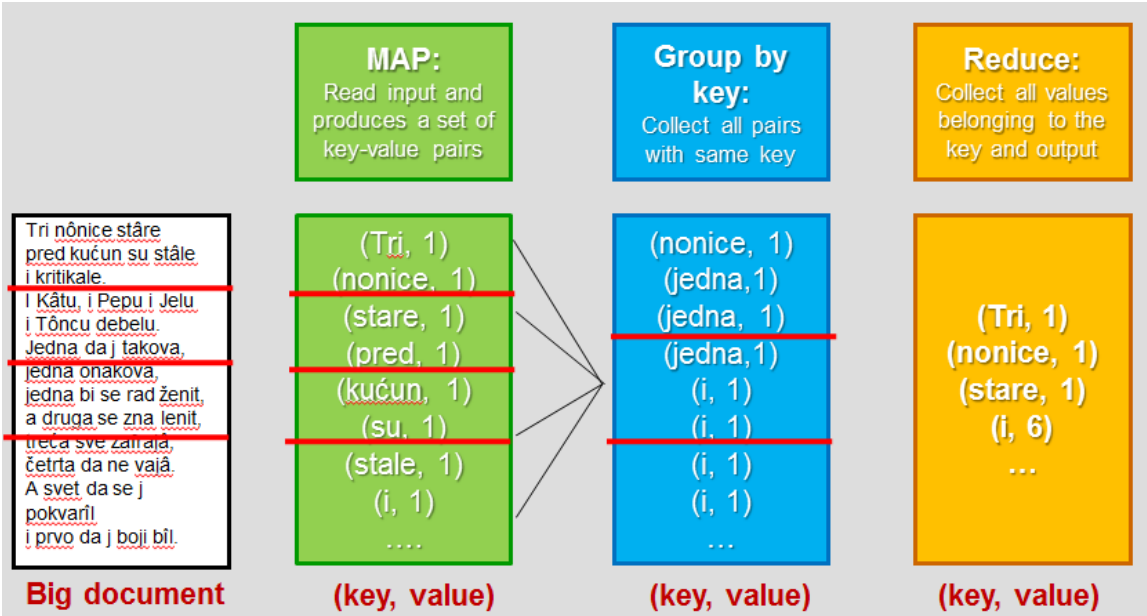
MapReduce – Broj riječi:

map(String ključ, String vrijednost)
//ključ: naziv dokumenta
//vrijednost: tekst dokumenta

for each word w in value:
    emit(w, „1“);

reduce(String ključ, Iterator vrijednosti)
//ključ: riječ;
//vrijednost: iterator tijekom brojanja

    int rezultat = 0;
    for each count v in vrijednosti:
        rezultat += Parseint(v);
    emit(ključ, rezultat);

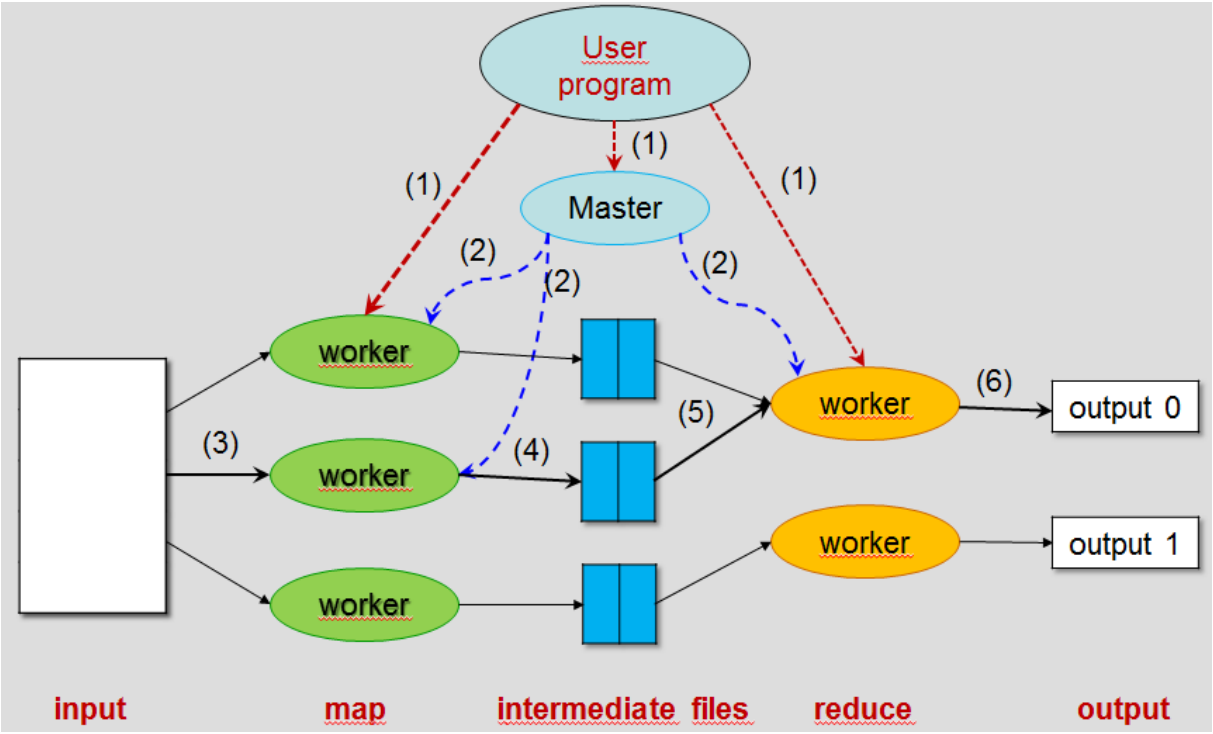


MapReduce – Implementacija

Map-Reduce okruženje se brine o:

- Podjeli ulaznih podataka
- Rasporedu izvršavanja u klasteru
- Grupiranju parova po ključu
- Toleranciji na pogrešku
- Komunikaciji između računala

Korisnik mora napisati funkciju za Map i funkciju za Reduce.



(1) **Račvanje**
- Podjela ulazne datoteke u M dijelova
- Pokrene više kopija programa na klasteru

(2) **Dodjeljivanje**
- Glavni program(Master) dodjeljuje M map i R reduce zadataka

(3) **Mapiranje**
- Map radnik(worker) čita ulazne komade
- Parsira ključ/vrijednost parove iz ulaza
- Proslijeđuje svaki par u definiranu Map funkciju
- Puni lokalnu memoriju izlazom Map funkcije

(4) **Lokalno zapisivanje**
- Izlaz mapiranja je zapisan na lokalni disk
- Podijeljeni u R regija od strane funkcije za podjelu
- Lokacije podijeljenih dijelova(particija) su proslijeđene glavnom programu(Master)

(5) **Grupiranje i sortiranje**
- Master obavještava reduce radnika o lokacijama particija
- Reduce radnik čita prijelazne podatke preko RPC (Remote Procedure Call)
- Sortira prijelazne podatke po prijelaznom ključu

(6) **Reduciranje**
- Prolaziti preko sortiranih prijelaznih podataka
- Reduce funkcija se poziva za svaki jedinstveni ključ prijelaznih podataka i skupa prijelaznih vrijednosti
- Izlaz reduciranja je dodan u izlaznu datoteku

MapReduce: Master node

Master – struktura podataka

- Stanje zadataka (idealno, u izvođenju, završeno)
- Identitet radnih strojeva
- Veličina i lokacija prijelaznih podataka
    - R regija prijelaznih podataka nastalih Mapiranjem
    - Poslani Master-u po završetku Mapiranja

Master – funkcije

- Šalje loakcije prijelaznih podataka zadatku Reduciranja
- Periodički proziva radnike

MapReduce: Fault tolerance

Pogreška radnika:

- Nema odgovora od radnika – greška radnika
- Zadaci Mapiranja dodijeljeni radniku s greškom resetiraju se na idle
    - Bili oni završeni ili u izvođenju
    - Preraspodijela po drugim radnicima
- A(map) → B(map)
    - Zadaci za Reduce će biti obaviješteni
    - Zadaci za Reduce će čitati podatke od radnika B
- Reduce greška – zadaci u izvođenju se resetiraju
    - Zadatak za Reduciranje se resetira
- Pogreška Mastera
    - Prekida se MapReduce operacija

MapReduce: Task granularity

M Map i R Reduce zadataka

M mnogo veći od broja radnika

- Poboljšano dinamičko balansiranje prilikom opterećenja
- Ubrzava obnovu kada dođe do greške radnika
- Jedan DFS dio po mapi (16 -64mb)

R – često ograničen od strane korisnika

- Svaki zadatak reduciranja u odvojenu datoteku
- Mali broj broja radnika

Fizičke granice

- O(M+R) odluka raspoređivanja, O(M*R) stanja

MapReduce: Backup Tasks

„Zaostali“ – spori radnik

- Uzima previše vremena za izvršavanje zadatka
- Produljuje vrijeme posla
    - Loš disk, obavljanje drugih poslova na mašini

Rješenje:

- Pred kraj MapReduce-a, Master preraspoređi rezervno izvršavanje zadataka u izvođenju
- Zadatak je gotov koji god zadatak prije završi (primarni ili rezervni)
- Značajno smanjuje vrijeme posla
- Ne povisuje resurse drastično

MapReduce: Refinements (preciziranja)

Funkcija podjele

- Funkcija podjele na prijelaznom ključu
- Defaultni – hash(key) mod R
- Zapisi sa istim ključem moraju biti kod istog radnika za Reduce
- Primjer – izlazni ključevi su URL-ovi
    - Svi ulazi za pojedinog hosta u istoj datoteci
    - Prilagođene funkcije podjele
    - Hash(Hostname(urlkey)) mod R

Jamstva redoslijeda

- Unutar dane particije, prijelazni parovi ključ/vrijednost se obrađuju u slijedu povećanja ključa
- Lagano za generiranje sortiranih izlaza po particiji

Preskakanje loših zapisa

- Determinisktički pad Map i Reduce
    - Bug u korisničkom kodu
    - Prihvatljivo ignoriranje nekih zapisa (na osnovi statističke analize)
- Detektiranje zapisa koji su prouzročili pad i preskakanje istih

Funkcija kombiniranja

- Značajno ponavljanje prijelaznih ključeva nastalih funkcijom Mapiranja (k,v1)(k,v2) za isti ključ k.
    - Reduciranje je komutativno i asocijativno
- Funkcija kombiniranja – spajanje podataka particija prije slanja preko mreže
    - Combine(k, list(v1)) →v2
    - Izvršeno na svakoj mašini koja izvodi Mapiranje
- Izlaz Reduca → izlazna datoteka
- Izlaz kombinacije → ulaz Reduce-a

Brojači

- Brojati pojave različitih događaja(Map)
- Master prikuplja vrijednosti brojača i vraća ih korisničkom kodu

Ulaz i izlaz

- Tekstualni način unosa – svaka linija je par ključ/vrijednost
- Svaka implementacija zna kako podijeliti podatke
- Korisnik mora implementirati jednostavno sučelje za čitanje
- Izlaz funkcionira na sličan način

Problemi pogodni za MapReduce

MapReduce nije rješenje za sve probleme

- Mnogo čvorova u paraleli
- Velike i rijetko ažurirane datoteke
- Primjer:
    - Online maloprodaja (Amazon) – nije prikladan
    - Računanje ranka stranice (Google) – prikladan

Brojanje učestalosti pristupa URL-u

- Funkcija mapiranja obrađuje zapise zahtjeva web stranice
    - Izlaz (URL, 1)
- Funkcija Reduca dodaje zajedno sve vrijednosti za isti URL
    - Emits (URL, total count)

Obrnuti Web-Link graf

- Map: (cilj, izvor) za svaku poveznicu(link) prema ciljanom URL-u, koji je pronađen na stranici izvora
- Reduce: konkatenacija svih izvora URL-a povezaih sa ciljanim URL-om, (targt, list(source))

Distribuirani grep

- Map: emitiraju liniju ako odgovara uzorku
- Reduce: kopira podržane podatke u izlaz

Distribuirano sortiranje

- Map: izvlačenje ključa iz svakog zapisa
    - Emits(key, record)
- Reduce: emitira sve nepromijenjene parove
- Garantira slijed i funkcija podijele

Pojam-Vector po domaćinu

- Sažima najvažnije riječi u dokumentu kao popis parova (riječ, učestalost)
- Map: emits (hostname, termvector) za svaki dokument
- Reduce: po dokumentu izrazvektor za određenog domaćina
    - Dodaje izrazvektore zajedno
    - Ignorira rijetke izraze
    - Emitira parove (domaćin, izrazvektor)

Matrix – Vector multiplication

- n*n matrica M
    - $m_{i,j}$ – element u i-to retktu i j-tom stupcu
- Vektor v duljine n sa j-tim elementom v_j
- (vektor) x = M (vektorski) v
- $x_i = \sum_{j=1}^{n} m_{ij} * v_j$
- M i v su spreljeni u DFS
    - n je velik, ali (vektor)v može biti spremljen u memoriju
    - poziciije $m_{i,j}$ i v_j su vidljivi

Funkcija za Map

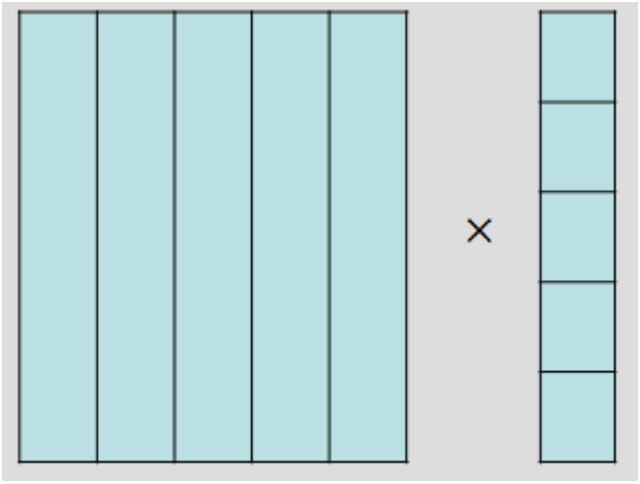
- primjeni se na jednom elementu od M
- (vektor)v se čita prvi, da bude dostupan svim Map radnicima
- Za svaki $m_{i,j}$ map emits(i, m_{ij}*v_j)

Funkcija za Reduce

- Zbroji sve vrijednosti povezane sa ključem i
- Rezultat je par (i, x_i)

Kada vektor v ne može stati u memoriju

- Podjela matrice M i vekora v na pruge
- Map: komad M pruge i cijelu v prugu



Upiti nad bazom podataka

- Upiti preveliki za zajedničke relacijske baze podataka
- SQL, relacija R, n-torke
- Relacija R može biti spremljena u DFS

Relacijsko-algebarske operacije

- Selekcija – stanje C, $\sigma_C(R)$
- Projekcija – podskup S, $\pi_S(R)$
- Unija, presjek, razlika
- Nature join $R \bowtie S$
- Grupiranje i agregacija

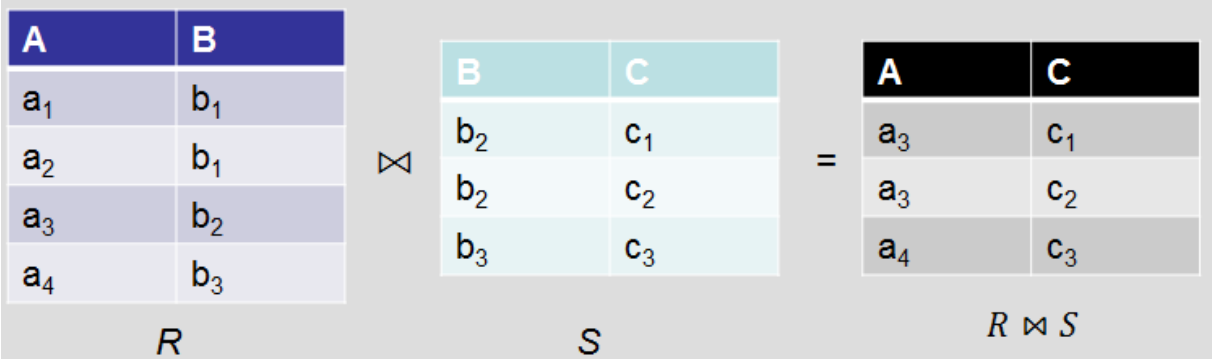
Selekcija

- Stanje C, $\sigma_C(R)$
- Map – ako n-torka $t \in R$ zadovoljava uvjet C, emit (t, t)
- Reduce – identitet

Projekcija

- Podskup S, $\pi_S(R)$
- Map
  - $\forall t \in R$ izgradi n-torku t' bez komponenti koje nisu u S
  - Emit (t', t')
- Reduce – eliminacija duplikata
  - Reduce $(\boldsymbol{t'}, [t', t', \ldots, t'])$ u (t', t')

Natural join $\boldsymbol{R(A,B)} \bowtie \boldsymbol{S(B,C)}$



- Map: za ulaz $\boldsymbol{R}(a, b)$ emit $(b, (\boldsymbol{a}, R))$
  - Za ulaz $S(b, c)$ emit $(\boldsymbol{b}, (c, S))$
- Reduce: spoji $(\boldsymbol{b}, (\boldsymbol{a}, \boldsymbol{R}))$ sa $(\boldsymbol{b}, (\boldsymbol{c}, \boldsymbol{S}))$
  - Emit (a, b, c)

Množenje matrica

- $P = M \times N$
- $p_{ik} = \sum_j \boldsymbol{m_{ij}} \cdot n_{jk}$
- Map
    - Za svaki $\boldsymbol{m_{ij}}$ emit $(j, (M, i, m_{ij}))$
    - Za svaki $\boldsymbol{n_{jk}}$emit $(j, (N, k, n_{jk}))$
- Reduce
    - Za svaki ključ jistraži listu vrijednosti
    - Za svaku vrijednost iz $(\boldsymbol{M}, \boldsymbol{i}, \boldsymbol{m_{ij}})$ i iz $(\boldsymbol{N}, \boldsymbol{k}, n_{\boldsymbol{jk}})$ napravi par ključ-vrijednost $((i, k), (m_{ij} \cdot n_{\boldsymbol{jk}})$
- Map2:
    - Identitet - $((i, k), v) \rightarrow ((i, k), v)$
- Reduce2:
    - Za svaki ključ $(\boldsymbol{i}, \boldsymbol{k})$ napravi sumu vrijednosti sa tim ključem
    - Rezultat: par $((\boldsymbol{i}, \boldsymbol{k}), v)$ gdje je $v = P_{ik}$

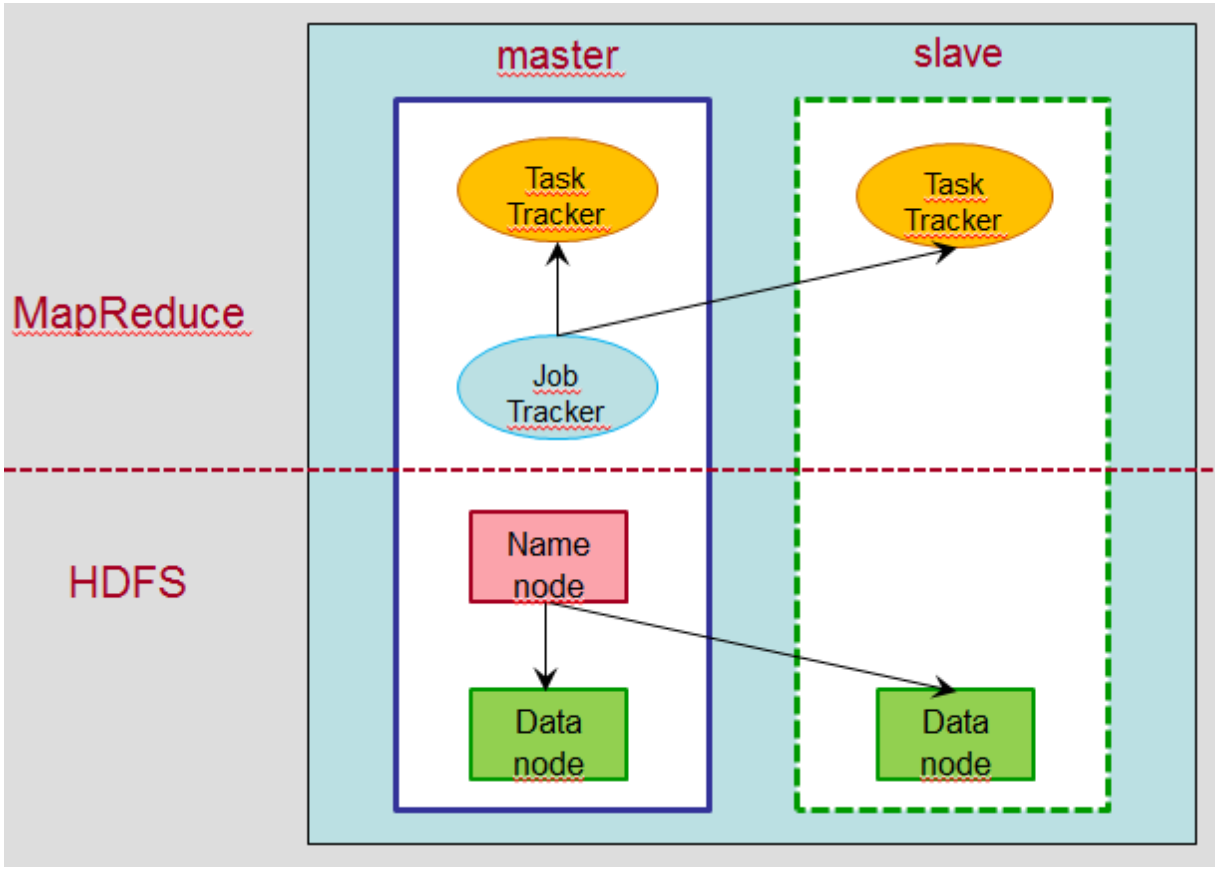
HADOOP – implementacija MapReduce-a otvorenog koda

- Napisan u Javi
- Koristi HDFS za spremanje
- Hbase – distribuirana baza podataka

HDFS

- Distribuirani, skalabilni i portabilni datotečni sustav
- 128 Mb blokovi
- Replikacija – 3 kopije svakog bloka
    - Dvije na istoj, jedna na različitoj polici
- Čvor naziva i čvor podataka

Konfiguracija klastera



Glavni čvor – JobTracker

- MapReduce program (job)
  - Job – JAR arhiva, konfiguracijska datoteka i particije
- Pruža JobClient s jedinstvenom identifikacijskom oznakom za posao
- Pridodjeli zadatke radnicima
  - Zadržati posao što je bliže moguće podacima
  - Svijest o policama – JobTracker zna koji čvor sadrži podatke i koje druge mašine su u blizini
- Izvršava zadatke konkurentno na radnicima
  - 4 utora (2 map, 2 reduce) na svakom radniku

JobClient

- Podijela i distribucija podataka preko HDFS-a

Job queue

- Poslovi dostupni u HDFS

Job scheduler – inicijalizacija zadaća (Map i Reduce)

Čvor radnika – TaskTracker

- Izvršava zadaće
- Periodički obavještava Mastera o svom stanju
    - Prazni Map i Reduce utore
- Stvara nove instance JVM- a i izvršava JAR zadaće
    - Spriječava rušenje TaskTracker-a ako posao sruši JVM

Tolerancija na pogreške

- Ako master čvor dobiva poruke o greška stanja od radnika
    - Master prerasporedi zadaće
    - Izbjegava preraspodjelu na istom čvoru
- Ako master ne dobiva poruke od radnika
    - Prerasporedi zadaće svih radnika
    - Ukloni radnika

Ograničenja

- Opterećenja sustava i dostupnosti dodijeljenog stroja
- Problem sporih radnika

Alternativna raspodijela

- Pravedan raspoređivač
- Raspoređivač po kapacitetu