★ unit - III

Processing data with Hadoop

- A large file to be processed is divided into chunks and is processed parallelly
- using Map reduce
- 2 tasks

keys
(sorting & Shuffling)

i) Map task - process each every chunk of data & produce intermediate result

ii) reduce task - O/P of map task is I/P to reduce task. & combines the o/p of all map task

HDFS & map reduce is present in same node making data to be available increasing performanc

Processing using Map reduce

2 trackers

job tracker . task tracker

- Present in name node.

- present in every datanode

- gives task

- rescheduling

- check for failure

MapReduce phases & daemons

Phase

map

- does mapping & produce o/p as key/value pair

reduce

- combine o/p from different maps & gives result

daemons

jobtracker

- master
- schedule task

task tracker

- slave
- exe: task

Daemons

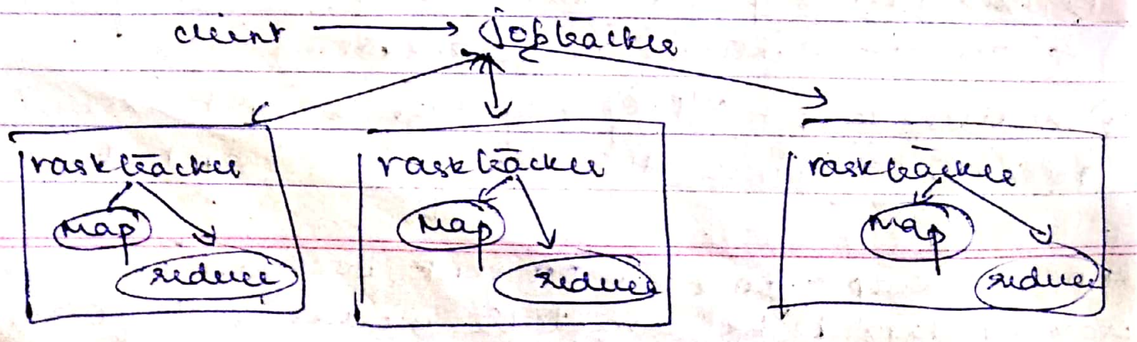
i) Job tracker: - provide connectivity b/w application & Hadoop

- figures out how to assign task to task tracker
- in case of failure it tries again & again & task is given to other node

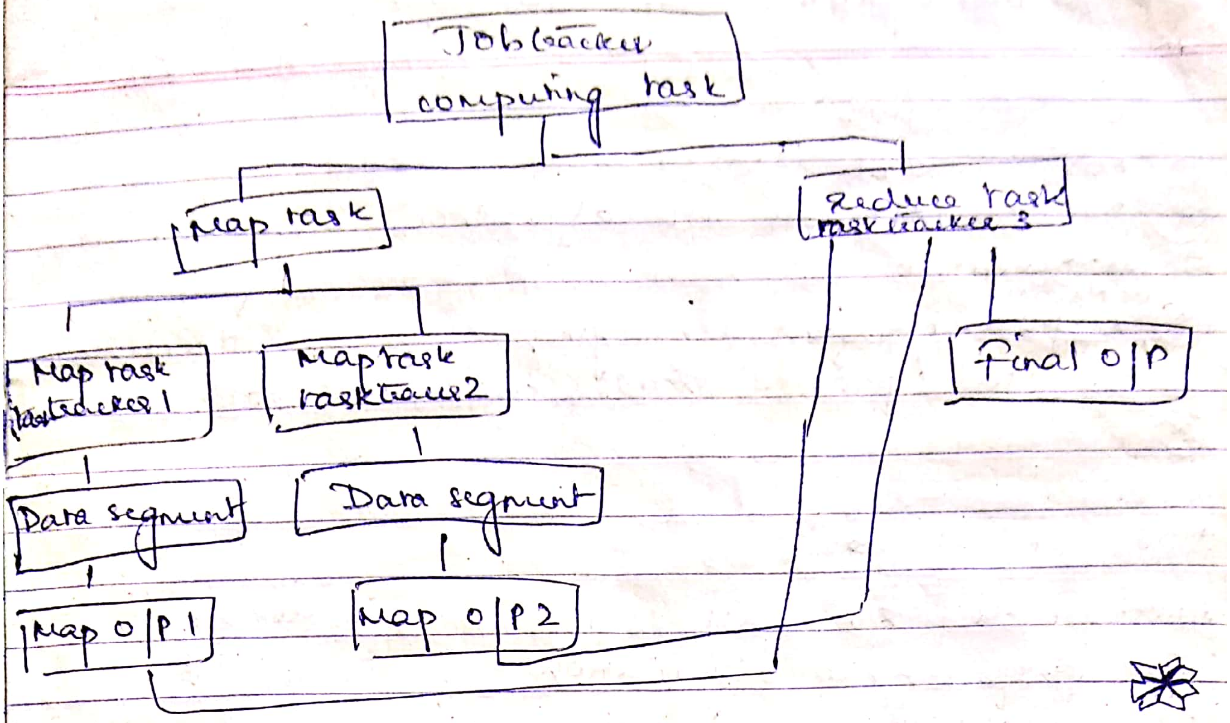
ii) Task tracker - execute task assigned by job tracker

- each node has one task tracker & multiple VMs
- Sends heartbeat message to job tracker to show existe
- & incase of failure the task is rescheduled to another node.

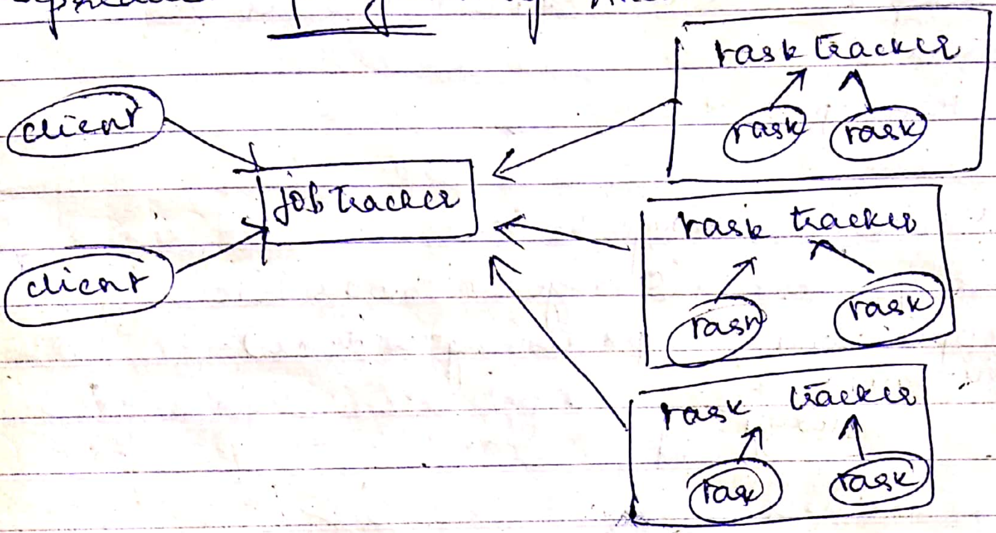
Jobtracker & tasktracker interaction



mapreduce programming workflow



Mapreduce programming Architecture.

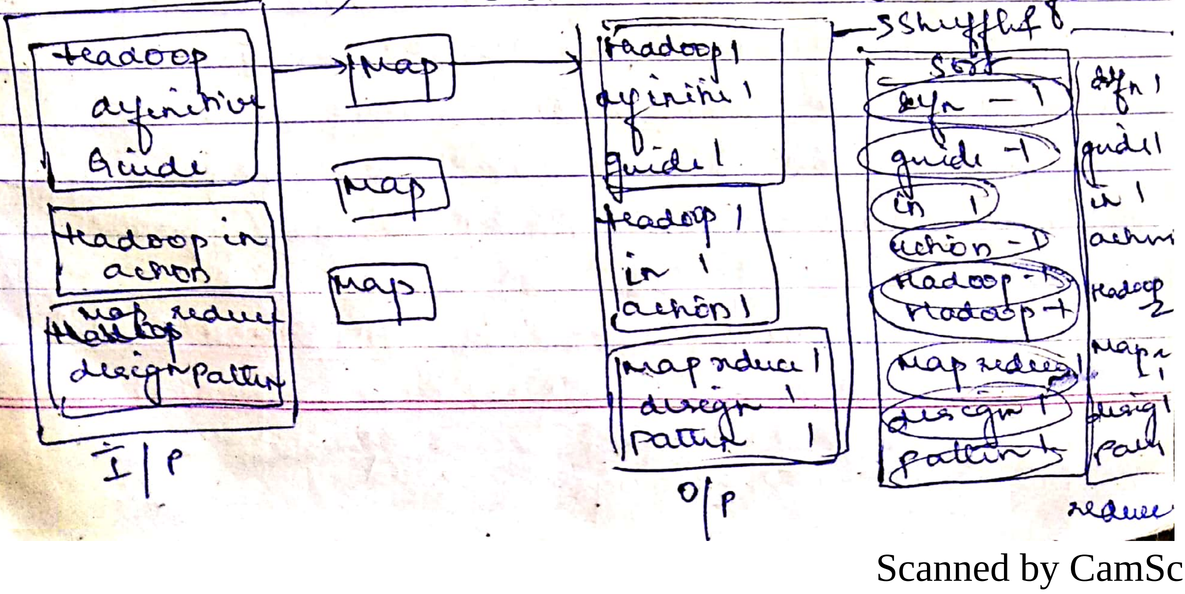


Eg: word count

support class i) driver class - info of job config

ii) mapper class - overwrite map func

iii) reducer class - overwrite reduce func



Limitations of Hadoop 1.0

① single Namenode - all the responsibility.

② Processing is restricted that is carried - batch oriented jobs

③ not suitable for interactive ~~processing~~ analysis of data

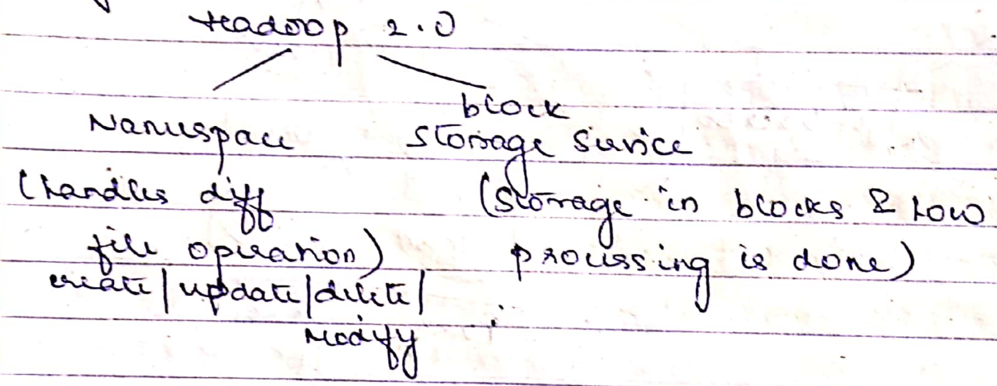
④ not suitable for ML algorithms & graph, memo application

⑤ mapreduce responsible for managing resources & processing of data (produce resource utilization problems)

⑥

HDFS limitations

① Namenode stores all data in main memory leading to performance degradation
This is overcome using Hadoop 2.0
On Hadoop 2.0 we use YARN - another resource negotiator

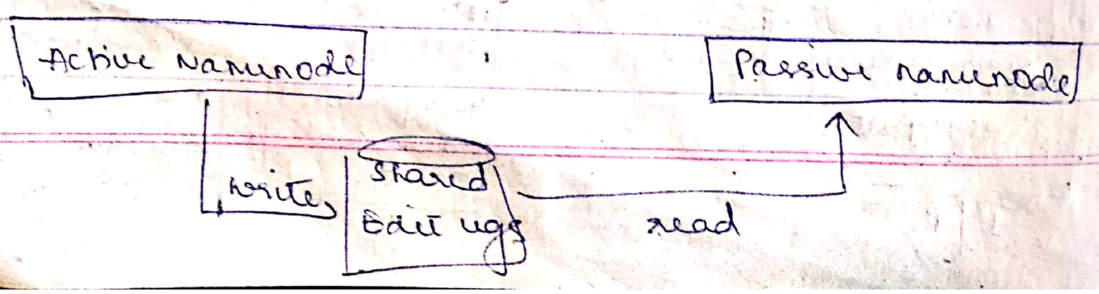

Hadoop 2.0
Namespace
(handles diff file operation)
create | update | delete | modify
block Storage service
(storage in blocks & low processing is done)

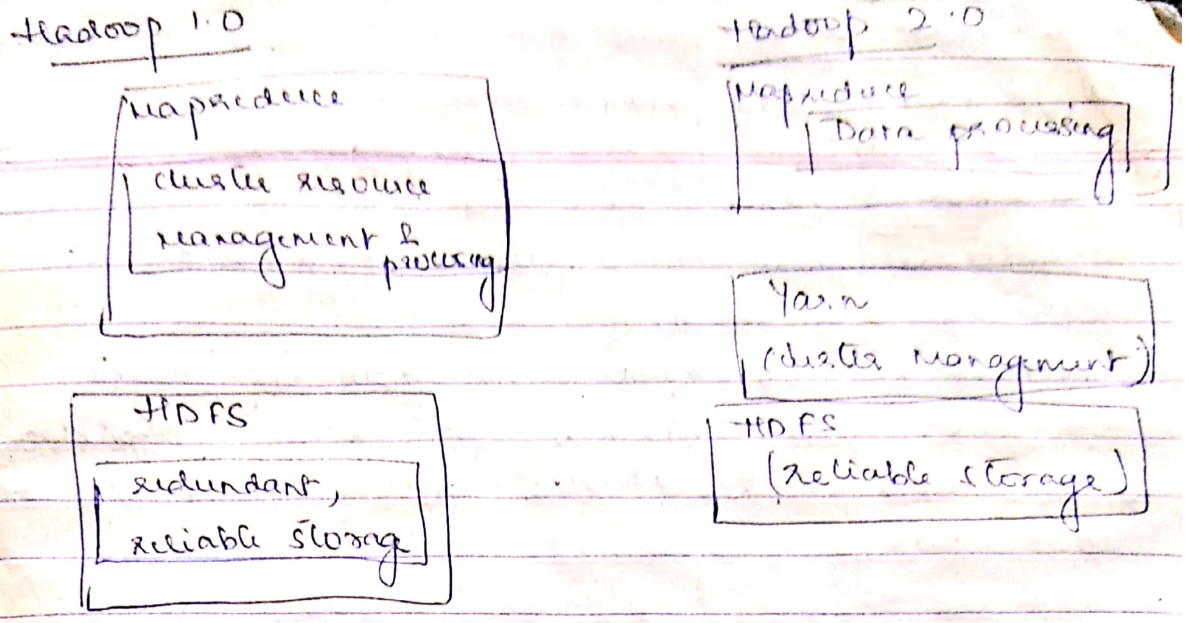
HDFS-2 features

① Horizontal ~~stability~~ scalability

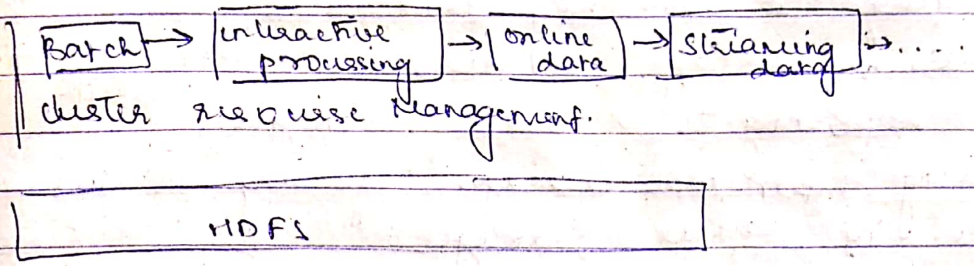
② Highly availability.

① multiple name nodes with data node & register itself with namenode. No direct interaction within name node: one namenode is active & other namenodes will be passive, in case of failure, passive becomes active namenode





Hadoop Yarn



Major Components of yarn

Resource Manager	Node Manager	Application Master
- resides in Master node	- in slave node	- in slave node

- responsibility
i) Managing resources (optimal)
ii) Schedule & appln manager -
(co-ordination
> scheduling exe of job by RM
- allocating

Responsibility of RM
i) Managing resources
ii) Schedule & application manager
Scheduler:
- Schedules job execution as requested by RM
- allocating resource to application submited to cluster

- communicating with AM, keep track of resources of running application

iii) Application Manager

- co-ordinate with scheduler to keep track of running application
- accepting the job submission from the client
- negotiating first container for executing application specific task with suitable application master on slave node.

② Node Manager - responsibility

i) managing & executing containers

ii) monitoring usage of resources - Memory, CPU and reporting to resource manager

iii) Sending heart beat messages regarding status update to resource Manager

③ Application Master - responsibility

i) per application specific library which works with node manager to execute task

ii) if multiple jobs are submitted on cluster, more than one instances of application master on slave node

iii) negotiating resource containers

iv) working with one or more node managers

Imp

Differences b/w yarn & map reduce

yarn	Map reduce
i) support variety of processing engines & appln	i) support its own appln ~~Overlapping~~ of batch processing
ii) separates its duties across multiple component	ii) consolidate most of task by single component.
iii) dynamically allocate ports of ~~alloca~~ resources to appln	iii) static allocation of resources for designated task.

Needs of yarn

- opened up new users for Apache HBase & ApacheHive
- offers scacability - any amt of data
- resource utilization
- high availability
- performance is improved wrt to mapreduce
- suitable for real time processor since it separates HDFS from map reduce, real time processing & appln can't wait for batch jobs to finish
- Manages resources in cluster envt.
- in case of searcing of resources - comminutes with complex resources & assign resources.

Application of Map-reduce — Softcopy.