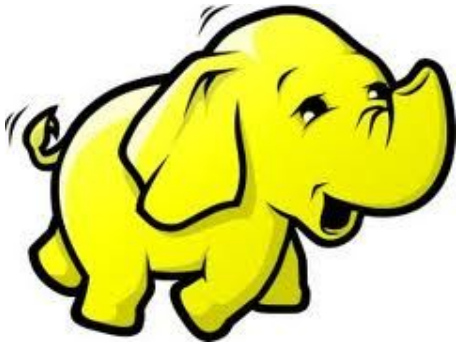
하둡소개

한기용

목차

- 하둡이란?
- HDFS
- MapReduce
- 부록: 데이터센터 설명



하둡이란?

하둡 소개

- 하둡은 대용량의 데이터를 분산처리해줄 수 있도록 해주는 아파치 톱 레벨 오픈소스 프로젝트
  - http://hadoop.apache.org/
  - 순전히 소프트웨어 프레임웍이며 자바로 작성.
  - Nutch/Lucene 프로젝트의 서브컴포넌트로 시작하여 2006년에 독립프로젝트로 분리.
  - 크게 분산파일시스템(HDFS)과 분산처리시스템(MapReduce)으로 구성
    - 이밖에도 하둡커몬(Common)이라고 위의 두 시스템에서 모두 필요한 라이브러리들도 하둡의 일부.

하둡의 특징 (1)

- 데이터가 있는 곳으로 코드를 이용
  - 대부분의 경우 데이터의 크기가 더 크다.
- 스케일 아웃 (vs. 스케일 업)
  - 소수의 비싼 서버 보다는 다수의 저렴한 서버 사용
- 단순한 데이터 모델
  - 반복적인 Key/Value pair의 트랜스포메이션.
  - 데이터의 locality를 최대한 이용한 프로그래밍 모델
- 오프라인 배치 프로세싱에 최적화

하둡의 특징(2)

- 기본적으로 하나의 마스터와 다수의 슬레이브로 구성된 마스터/슬레이브 아키텍쳐를 HDFS와 MapReduce모두에 적용.
- HDFS:
  - 하나의 Name Node(마스터)와 하나 이상의 Data Nodes (슬레이브)
  - Secondary NameNode가 존재하여 주기적으로 Name Node의 내용을 백업 (snapshot)
- MapReduce:
  - 하나의 Job Tracker(마스터)와 하나의 Task Trackers (슬레이브)
- 대부분의 경우 이 둘은 한 물리적인 클러스터에 공존.
  - Name Node/Job Tracker가 같이 살고 Data Node/Task Tracker가 같이 동거.

하둡의 발전

- 2005년 Doug Cutting이 Nutch 크롤/검색패키지에 구글페이퍼를 기반으로한 HDFS/MapReduce 프레임웍을 추가하면서 시작.
- 2006년 Doug Cutting 야후 검색팀 조인. 20노드 클러스터 셋업
- 2006년 Hadoop이 Nutch에서 떨어져나와 아파치 톱레벨 프로젝트로 변신.
- 2008년 야후에서 1000 노드 하둡클러스터를 프로덕션에서 사용시작.
- 2012년 현재 하둡 생태계가 활발히 커가고 있음.
  - 컨퍼런스: Hadoop Summit, Hadoop World
  - 많은 종류 하둡 기반 혹은 변방 소프트웨어들과 스타트업들

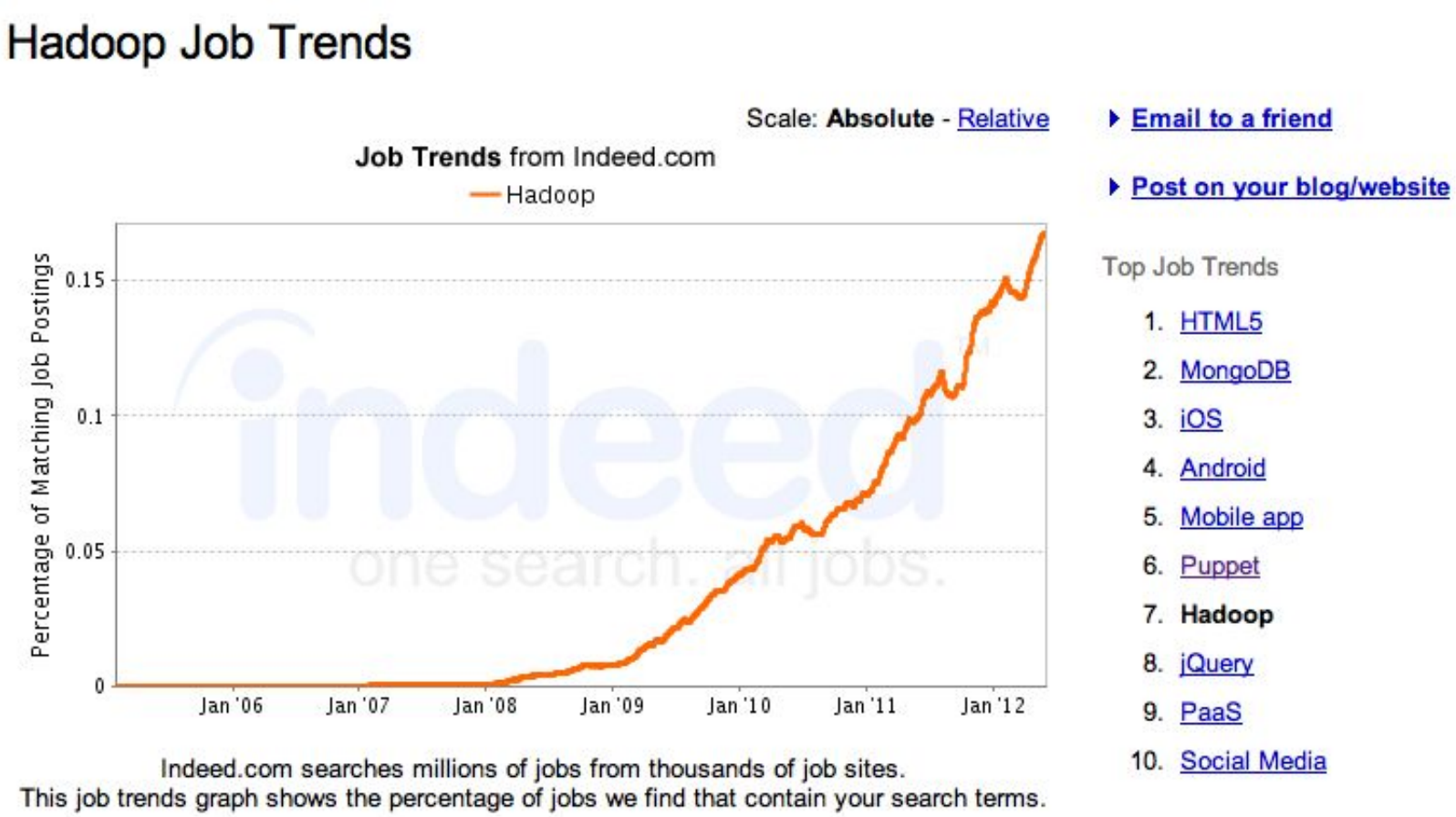
하둡 배포판

- 아파치재단 (Apache Foundation)이 제공하는 하둡은 0.10버전부터 시작해서 현재 0.23까지 나와있으면 이중 일부 버전은 1.0과 2.0의 메이저 버전으로 존재한다.
- 지금 현재 액티브하게 개발되고 있는 버전들은 다음과 같다.
  - 2.x.x: 현재 알파버전 (0.23.x 기반)
  - 1.1.x: 현재 베타버전
  - 1.0.x: 현재 안정버전 (0.22.x 기반)
  - 0.20.x: 현재 가장 많이 쓰이는 legacy 안정 버전.
- 뒤에서 랩세션에서는 0.20.205의 하둡을 사용할 것이다.

3rd 파티 하둡 배포판

- 대표적으로 클라우데라(Cloudera)나 홀튼웍스(HortonWorks), MapR등의 회사도 하둡 배포판을 만듬.
- 참고로 클라우데라의 배포판들은 http://www.cloudera.com/hadoop/에서 더 자세한 사항을 볼 수 있는데 실질적으로 가장 많이 사용되는 배포판이며 흔히 CDH라 부름.
- 홀튼웍스의 배포판은 홀튼웍스 데이터 플랫폼이라 불리우며 HDP라 부르기도 한다. http://hortonworks.com/products/hortonworksdataplatform/에서 다운로드 가능.
- 둘다 역시 오픈소스이며 개인이 사용할 경우 무료이다.
- MapR의 배포판은 AWS의 ElasticMapReduce에서 사용.
- VMWare가 가상화버전의 Hadoop을 발표 (2012 Hadoop Summit) – Project Serengeti

Job Market에서의 하둡 수요 증대



하둡 라이센스

- 아파치 라이센스를 준수
  - 무료 소프트웨어 라이센스
  - 누구든 코드를 갖고 내부적으로 사용하거나 재배포하거나 심지어 파는 것까지도 가능.
    - 몇가지 조건이 존재 (Attribution)

작업 모델

- 하둡 자체는 아파치재단의 소유물
  - 아파치재단 자체는 비영리조직
- 4가지 형태의 contribution이 가능
  - 사용자 (대부분)
  - 컨트리뷰터: 패치생성
  - 커미터: 패치적용. 패치생성
  - 프로젝트 관리 커미티티: 새 릴리스와 커미터 선정 투표

하둡 사용회사들

- http://wiki.apache.org/hadoop/PoweredBy/
- 대표적 회사들
  - 국외: Facebook, Twitter, EBay, Linkedin, Yahoo, ...
  - 국내: NHN, NCSoft, SDS
  - 점점 늘어나는 추세

하둡의 문제점

- 너무나도 많은 버전과 부실한 서포트
  - 3rd Party 배포판이 인기가 높은 이유.
- 셋업과 사용이 쉽지 않음
  - 비용/시간이 들며 맞는 스킬셋을 가진 사람의 고용도 쉽지 않음.
- 하둡에 맞지 않는 작업도 존재
  - 소규모이거나 대용량의 데이터처리가 필요하지 않다면 하둡으로 옮겨갈 이유가 없음.

HDFS 상세소개

HDFS 개요

- 2003년 구글랩에서 발표된 The Google Filesystem이란 논문을 바탕으로 작성된 파일시스템.
- 이 시스템의 특징
  - 파일을 여러개의 블록으로 나눠 저장 (기본 64MB)
  - 하드웨어 고장에 견고
    - 한 데이터블록을 보통 3군데 (Replication factor)에 저장하며 저장시 같은 rack에 있는 서버들에 두 개에 저장하고 다른 하나는 다른 rack에 있는 서버에 저장.
  - Write Once Read Many
    - Append 작업은 가능하지만 내용을 바꾸기 위해서는 파일 전체를 새로 써야한다.
  - 스트리밍 데이터 액세스
    - 배치잡에 최적화
  - MapReduce나 HBase와 같은 시스템의 기본구성블록으로 사용
  - 계층구조의 파일시스템을 제공

NameNode (1)

- HDFS마다 단 하나만 존재 (Hadoop 1.X나 이전 버전)
- HDFS의 마스터 노드로 저장되는 각종파일들의 메타정보를 관리하고 실제 데이터는 다수의 Data Node 에 분산 저장.
  - 파일이 블록단위(기본 64MB)로 나뉘어 저장되고 설정에 따라 보통 세 군데((replication factor)의 Data Node에 중복 저장
    - 블록크기는 hdfs-site.xml의 dfs.block.size 파라미터로 조절가능.
  - Rack awareness: 중복저장시 Rack위치를 유념하여 한 rack에 모든 복제블록이 놓이지 않도록 함.
- Data Node들의 계속적으로 통신 (Heartbeat)
  - 각 DataNode들로부터 현재 상태와 보유 데이터블록 리스트(블록리포트)를 체크.
  - 문제 Data Node가 감지되면 그 노드의 블록들을 다른 노드들에 복제. (replication factor를 유지하려 시도).
  - 기본적으로 둘간의 통신은 3초마다 일어나는데 이는 hdfs-site.xml의 dfs.heartbeat.interval 파라미터로 조절가능.

NameNode (2)

- A single point of failure.
    - NameNode는 메타정보(HDFS namespace와 파일블록맵핑)를 메모리에 유지하며 또한 모든 HDFS 클라이언트와의 트랜잭션을 EditLog라는 파일에 수록함(문제시 복구목적).
    - Checkpoint: 이 작업이 시작되면 NameNode의 메타정보를 FsImage라는 디스크파일로 쓰고 EditLog 파일을 리셋함.
        - Secondary Name Node는 주기적으로 이 checkpoint를 요청하고 FsImage를 백업. 하지만 NameNode에 문제가 생길 경우 이를 바로 대체할 수 있는 것은 아님.
    - Hadoop HA (High Availability)의 주요 개선 포인트 (Introduction of standby NameNode) - Hadoop 0.23 or Hadoop 2.0
- 세이프 모드 (Safe Mode)
    - 처음 스타트업시 NameNode는 마지막으로 저장된 FsImage를 읽어들이고 다음으로 EditLog의 내용을 리플레이한다.
    - 그 다음으로 클러스터내의 DataNode들로부터 상태와 보유블럭리스트를 받아서 자신이 갖고 있는 정보와 맞춰본 다음에 replication factor들이 보장되고 있는지 확인하는데 이 과정 중에는 외부 요청에 반응하지 않음. 이 모드를 세이프 모드라고 함.

Data 읽기

1. 클라이언트는 먼저 NameNode와 통신하여 해당 파일의 데이터블록 위치 리스트(DataNode와 블록ID)를 얻음.
   - 파일의 크기가 데이터블록크기(기본 64MB)보다 작다면 한쌍의 DataNode/블록ID로 충분
2. 클라이언트는 DataNode들과 직접 통신하여 블록데이터들을 차례대로 읽어들임.

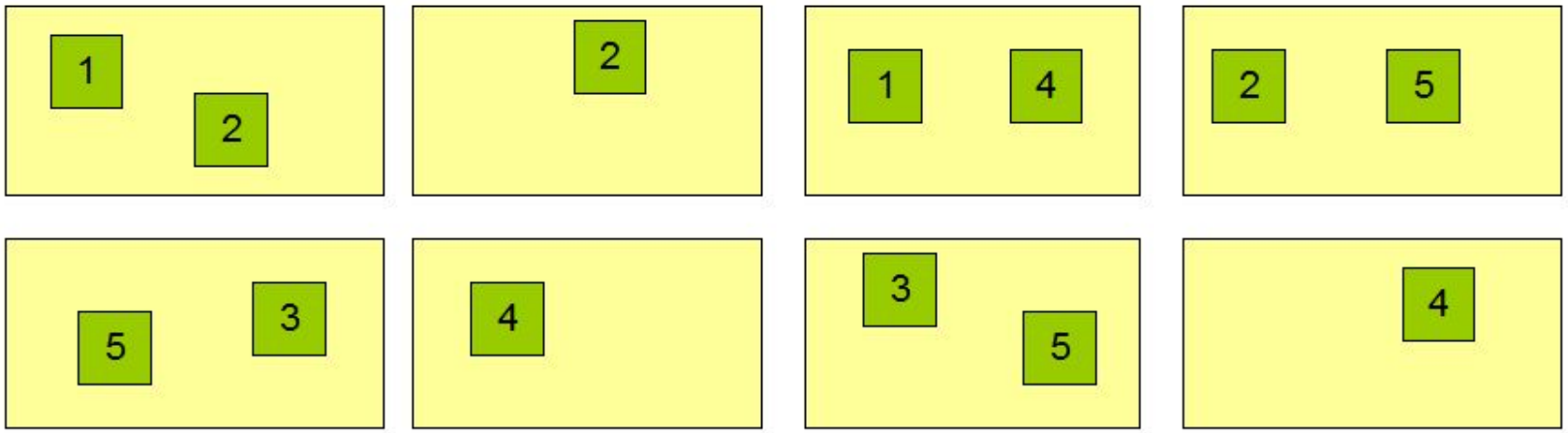
Data 쓰기

1. 클라이언트는 HDFS 파일을 생성하고자 하면 먼저 로컬파일시스템에 파일을 생성
2. 파일 생성이 끝나거나 크기가 데이터블록의 크기보다 커지면 이때 NameNode를 컨택. NameNode는 파일생성요청을 메모리메타정보와 EditLog에 저장.
3. NameNode는 Replication factor만큼의 DataNode와 블럭ID를 클라이언트에게 전송.
4. 클라이언트는 이중 첫번째 DataNode에 데이터를 쓰면서 replication이 벌어져야하는 나머지 DataNode들의 리스트를 같이 넘긴다.
5. 첫번째 DataNode는 데이터를 복제받으면서 두번째 DataNode로 복제를 시작한다.
6. 마지막 DataNode에서 블록의 복제가 완료되면 이 시점에서 해당 데이터블록의 생성은 완료된 것으로 간주됨. 이 프로세스를 Replication pipelining이라 함.
7. 클라이언트에서 파일에 써야할 데이터(데이터의 크기가 블록크기가 되거나 파일생성이 끝날때까지 기다림)가 더 있으면 다시 3으로 가서 반복.

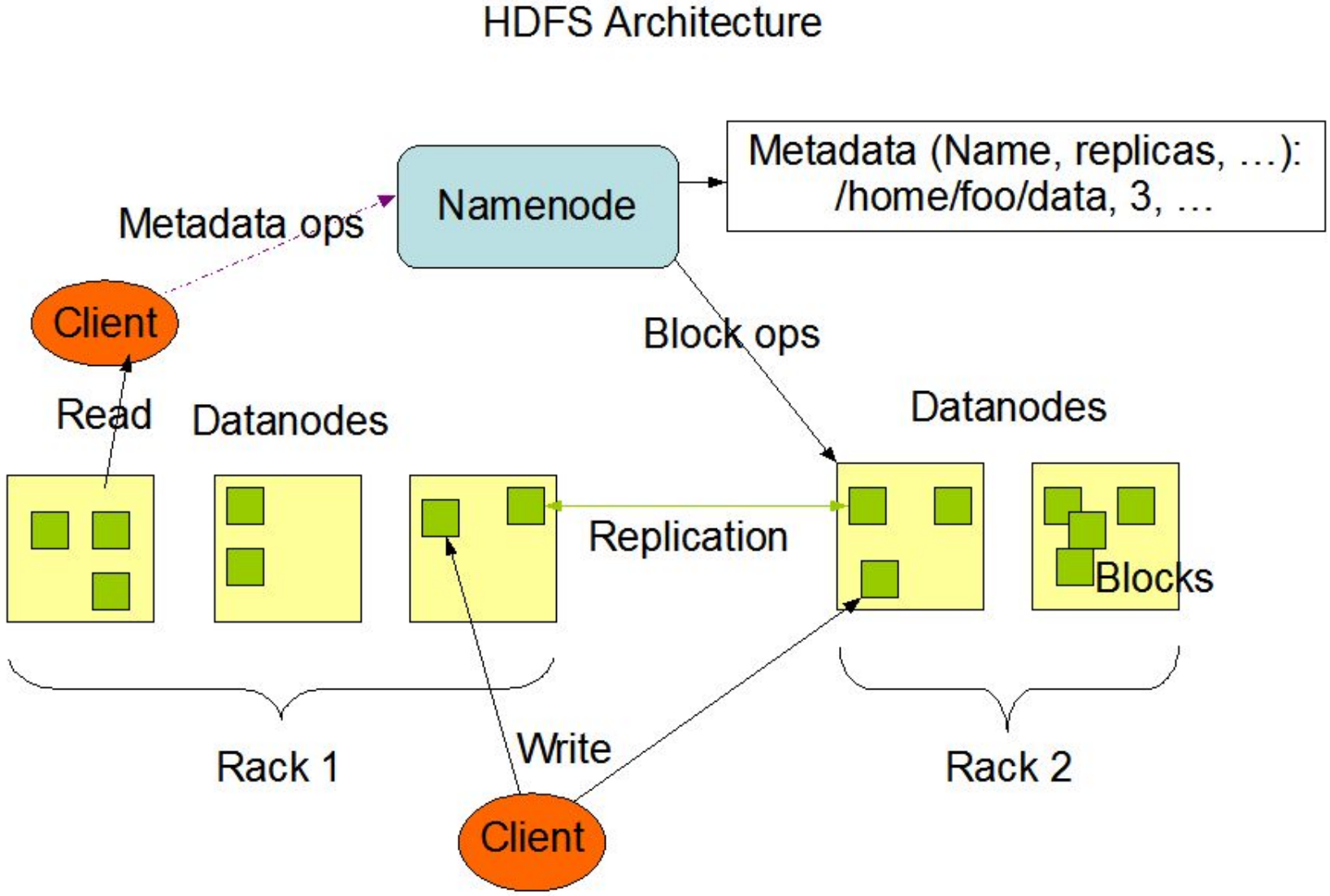
Block Replication

Namenode (Filename, numReplicas, block-ids, …)
/users/sameerp/data/part-0, r:2, {1,3}, …
/users/sameerp/data/part-1, r:3, {2,4,5}, …

Datanodes



Apache Hadoop Homepage에서



Apache Hadoop Homepage에서

HDFS 액세스

- HDFS 라이브러리를 이용하거나 하둡 셀 커맨드를 이용하여 액세스 가능
  - HDFS 라이브러리: 자바 라이브러리 실습 예정
  - Hadoop 셀커맨드
    - 예) hadoop fs -mkdir 디렉토리이름
  - Hadoop DFS 어드민
    - 예) hadoop dfsadmin -report

MAPREDUCE 상세소개

MapReduce 프레임웍

- 2004년 구글랩에서 발표한 MapReduce: Simplified Data Processing on Large Cluster란 논문을 바탕으로 작성된 분산처리시스템.
- MapReduce 프레임웍은 일종의 대규모 분산 Merge-Sorting 프레임웍.
- 특징
  - 데이터가 있는 서버로 코드를 전송.
  - 데이터프로세싱을 키/밸류 데이터셋의 변환으로 진행 (mapper와 reducer)
  - Share Nothing 아키덱쳐.
    - MapReduce 프레임웍에서 동작하는 mapper들끼리 그리고 reducer들끼리는 서로에 대한 의존성없이 동작.
    - 프레임웍이 mapper와 reducer의 중간에서 셔플링/소팅을 해주기에 가능.
  - Data Locality를 최대한 활용:
    - Mapper를 실행한 서버를 찾을때 입력파일블럭을 이미 갖고 있는 서버나 그 서버와 같은 Rack에 있는 서버를 찾으려고 시도..

Job Tracker (1)

- MapReduce 프레임웍의 마스터로 한 클러스터에 하나만 존재.
- 프레임웍에서 실행되는 모든 Job들을 실행을 관리.
  - 사용자로부터 하둡 잡 실행 요청(하둡코드가 들어간 jar 파일, 입력데이터 위치, 출력데이터 위치 등등)을 받아 클러스터내의 Task Tracker들로 나눠서 Job을 실행.
    - 정확히 이야기하면 사용자의 하둡 잡 실행 요청은 Job 스케줄러로 들어가고 Job Tracker는 Scheduler로부터 다음 실행할 Job을 얻는다.
  - 태스크들이 종료될때까지 관리하며 만일 특정 태스크가 실패하면 다른 Task Tracker에 그 태스크를 다시 실행.
  - T보통 Job Tracker는 HDFS의 마스터의 NameNode와 같은 서버에 위치
  - Task Tracker 역시 HDFS의 DataNode들과 같이 공존.
  - 하둡 셀커맨드나 웹 인터페이스를 Job/Tasks들의 상태를 볼 수 있음.
- NameNode와 마찬가지로 A Single Point of Failure. 무슨 이유로건 Job Tracker가 재시작되면 모든 Job들도 재시작되어야 함. NameNode와 마찬가지로 이 문제는 Hadoop 0.23이나 Hadoop 2.0에서 개선됨.

Job Tracker (2)

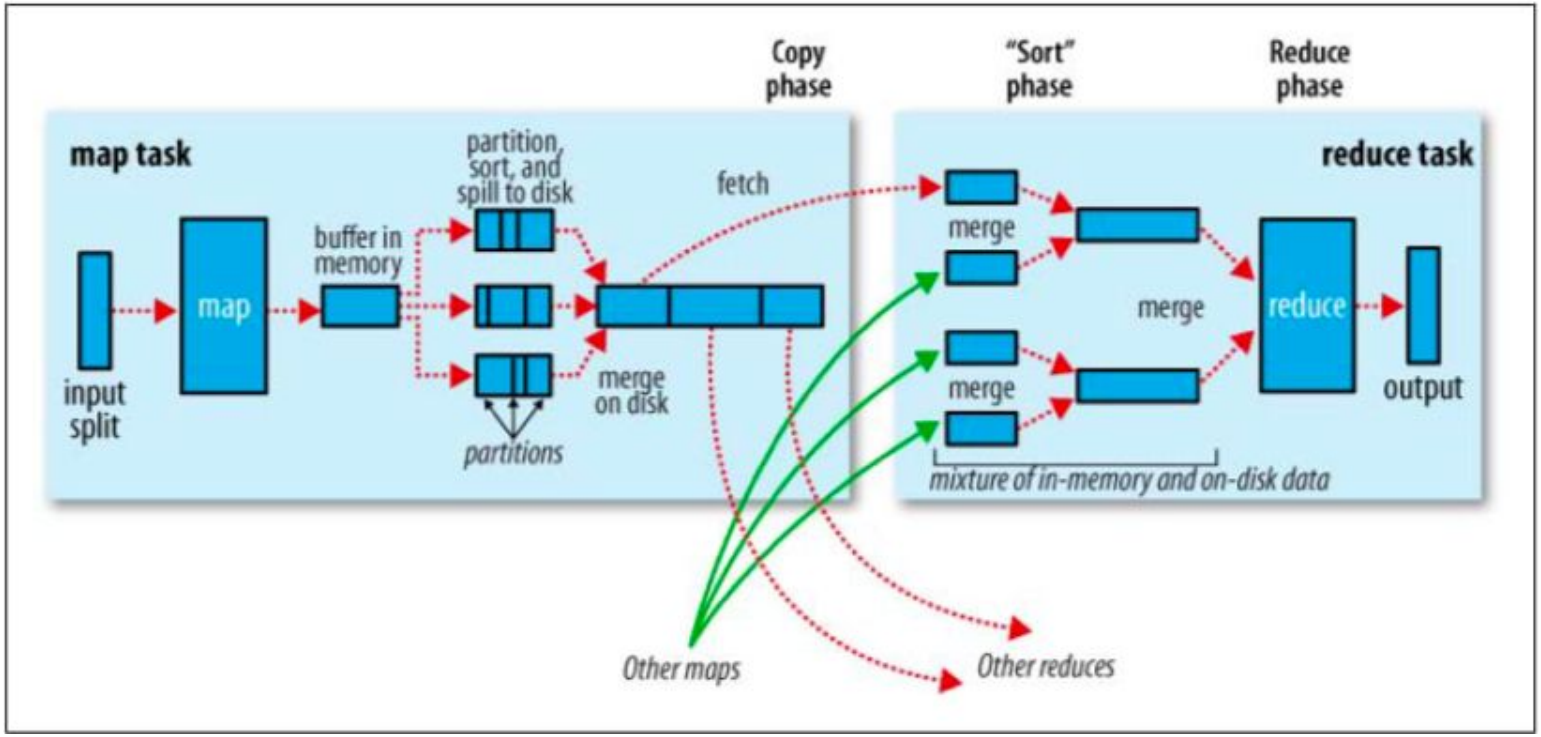
- Task Tracker는 주기적으로 Job Tracker에 상태보고 (heartbeat)를 함
  - "나 살아있소".
  - 실행중인 태스크의 상태 (카운터 정보 포함)
    - mapper의 경우 이는 입력 레코드들의 처리 퍼센트를 알림.
    - reducer의 경우 조금 더 복잡
      - 셔플링이 끝나면 33%
      - 소팅이 끝나면 66%
      - 그 이후부터는 reducer의 입력 레코드 처리 퍼센트 * 0.34 + 66%.
  - 놀고 있는 태스크 슬롯의 유무.
  - 만일 heartbeat이 지정된 시간동안 안 오면 해당 서버는 죽은 것으로 간주되고 그 서버에서 돌던 태스크들은 다른 Task Tracker에서 재수행됨.

Job & Tasks

- 보통 사용자가 실행하고자 하는 MapReduce 프로그램을 칭함. Job Tracker가 관리.
- 보통 Job은 하나이상의 mapper와 하나이상의 reducer로 구성되며 이 mapper들과 reducer들을 task라고 부름.
  - 각각의 task는 Task Tracker에 의해 관리되며 각 task는 별개의 JVM에서 실행.
  - 실패한 task는 Job Tracker에 의해 다른 노드에서 재시도됨.
  - Speculative Execution: JobTracker는 다른 태스크들이 실행이 현저하게 느린 태스크들을 proactively하게 다른 TaskTracker들에서 중복실행하게 할 수 있음. mapred-site.xml의 mapred.map.tasks.speculative.execution 파라미터로 조절.
- 하나 이상의 Job들이 엮어서 실제로 원하는 일을 수행하게 되는 경우가 대부분 (Hadoop Job Chaining). 이러한 웍플로우 관리가 굉장히 중요 (Cascading, Oozie 등등).

Scheduler

- 기본적으로 MapReduce 프레임웍은 FIFO 스케줄링을 지원.
  - Job 제출시 job의 우선순위를 지정해줄 수 있지만 실행중인 job의 pre-emption은 불가
  - Job 하나가 전체 클러스터의 리소스를 독점가능.
- Capacity 스케줄러
  - 야후에서 만든 스케줄러. 여러 큐를 만들어 큐별로 쿼터 제공.
- Fair 스케줄러
  - 페이스북에서 만든 스케줄러. 사용자별로 쿼터 제공.
- 스케줄러는 플러그인 형태라 커스텀 스케줄러의 개발이 용이

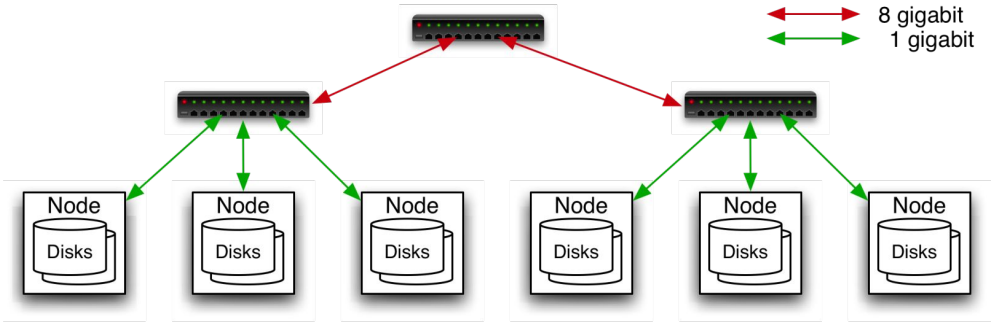


Tom White의 "Hadoop Definite Guide"

MapReduce 프로그래밍

- 기본적으로 자바
- Hive/Pig 등의 하이레벨 언어
  - UDF 등으로 확장가능.
    - 작업성격에 따라 프로그래밍이 훨씬 간편하지만 느림.
- Streaming (stdin/stdout)
  - Python, Perl, shell, ruby, ...
  - 자바에 비해 20% 정도 느림.
- Pipe (C++)
  - 소켓을 입출력으로 사용하는 모델.

하둡 클러스터 구성 (2009년)



- Commodity hardware
  - Linux PCs with local 4 disks
- Typically in 2 level architecture
  - 40 nodes/rack
  - Uplink from rack is 8 gigabit
  - Rack-internal is 1 gigabit all-to-all

EBay 하둡 클러스터 구성예

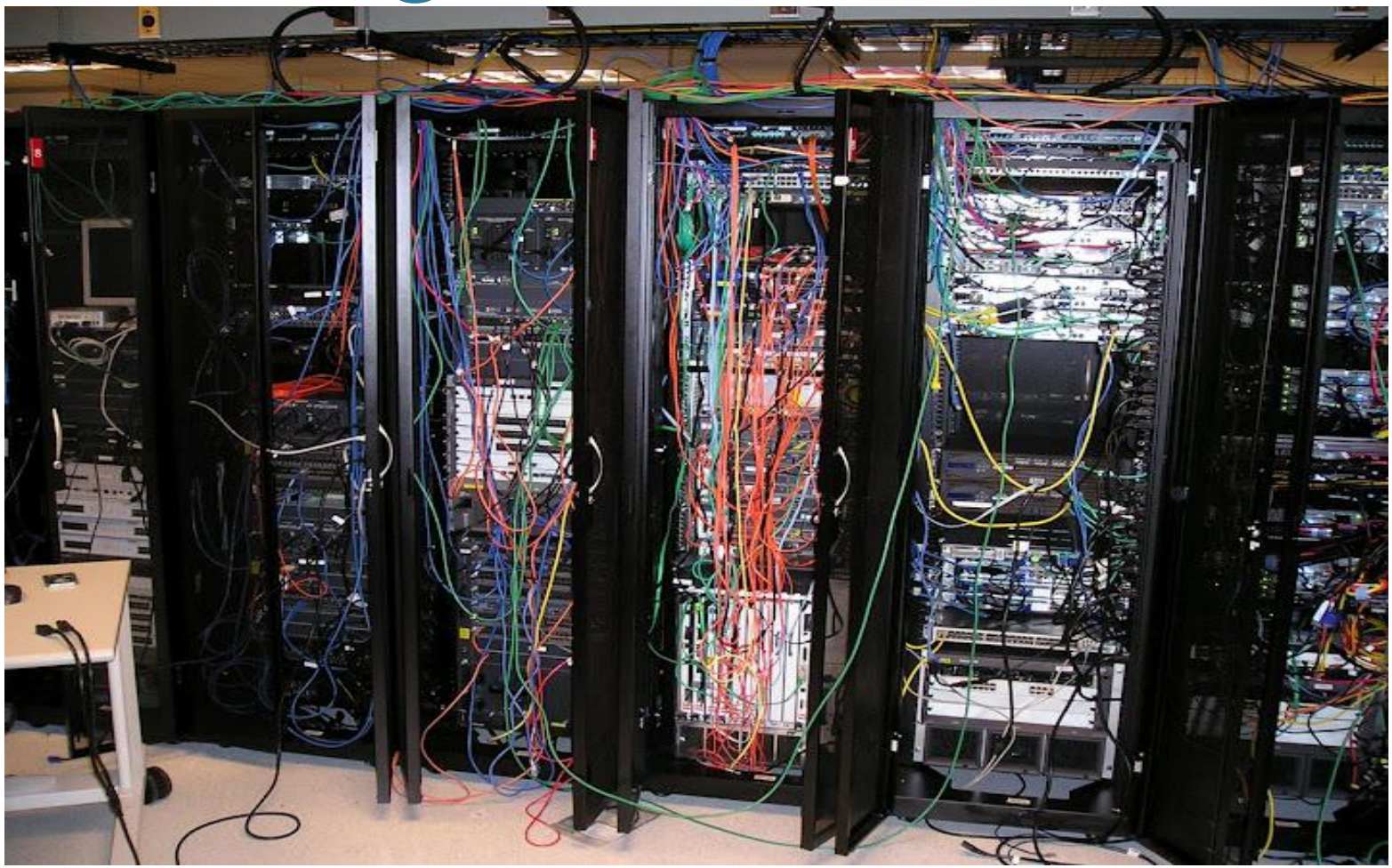
- 2012년 6월 Hadoop Summit에서 발표
- 노드수: 532대 - 1008대
- 총 스토리지크기: 5 - 18PB
- 네트웍 스위치: 1Gbps and uplink 40Gbps
- 서버 스펙:
  - Cent OS 4 64 bit
  - Intel Dual Hex Core Xeon 2.4GHz
  - 72GB RAM
  - 2*12TB HDD (24TB)
  - SSD for OS

Facebook 하둡 클러스터 구성예

- 2010년 5월 자료
- 노드수: 2000대
- 총 스토리지크기: 21PB
- 네트웍 스위치: 1Gbps and uplink 40Gbps
- 서버 스펙:
  - Cent OS 4 64 bit
  - 8 core 서버: 1200대, 16 core 서버: 800대.
  - 32GB RAM
  - 12TB HDD

부록: 데이터센터 내부구성 컴포넌트 설명

A typical datacenter during dot com bubble



Facebook datacenter in Prineville Oregon

What is in Datacenter?

- Rack or Cabinet.
- Servers (CPU, memory, ...)
- Storage (NAS, SAN, ...)
- Network Equipment.
- Power Supply (Input Power Unit, Power Distribution Unit, UPS, ...)
- Air Conditioner.
- Security.

Rack (or Cabinet)

- A standardized frame or enclosure for mounting multiple equipment modules.
- Dimension
  - 19 inches or 23 inches wide.
  - Height is measured in Rack Unit (“**U**”). 1U is 1.75 inches (4.45cm)
    - 42U or 44U high are common.

Server

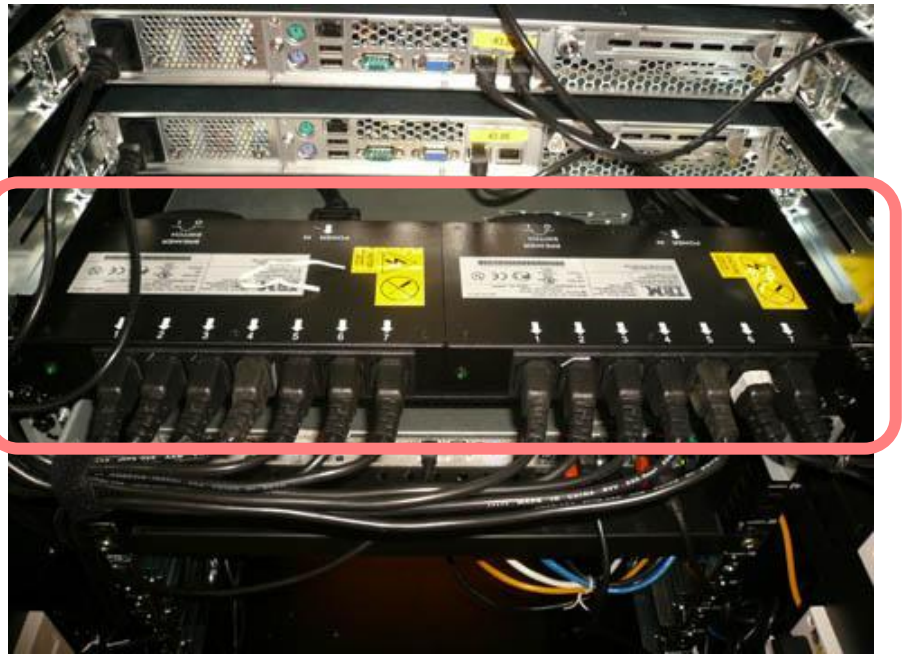
Facebook's 1.5U server

- Servers can be 1U to 4U high.
- Some companies design servers themselves
  - Facebook (open compute), Google, Amazon, …
- Most will just purchase Dell, HP and so on.

Network Equipment

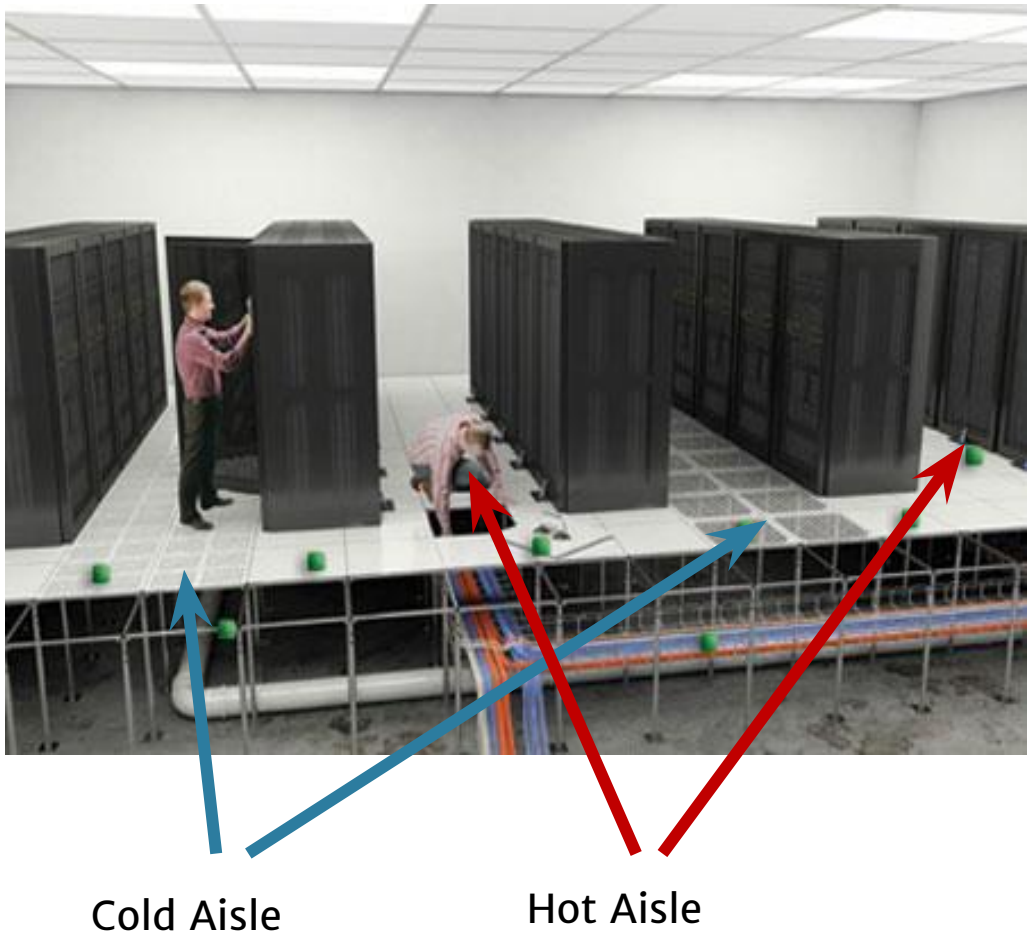
- Placed in a rack or on top of a rack.
- Handle different types of connections
  - Internal (topology is important)
  - External (Internet or other data centers)

Power Distribution Unit



- Distributes power to racks
- There are two types
  - Floor mounted
  - Rack mounted

Cooling – Raised Floor



- Goal is to keep 16 ~ 24°C
  - But 26 to 27°C is ok according to Google.
- Cold Aisle
  - Cool air is pushed out from floor (needs power).
- Hot Aisle
  - Hot air is flowing upward and sucked out.
- Usually cables and AC will be under floor.
- This is a traditional way

Cooling – Penthouse

Picture of upper floor ("penthouse").

- This is a new way used by Facebook and set up two floors within datacenter.
  - Lots of innovations in this area.
- Upper floor is used to direct air and lower floor contains racks.
- Cool air enters thru the upper floor and falls down.
- Hot air rises and goes out of exhaust air plenum.