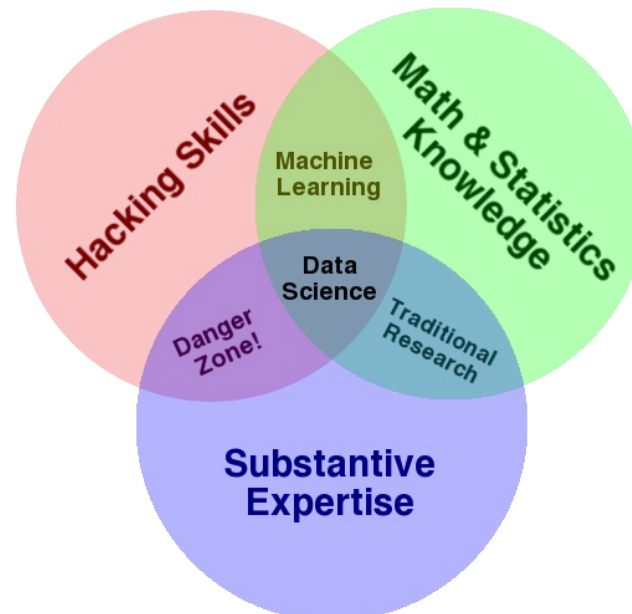


R+Hadoop = Rhadoop*



Des logiciels libres complémentaires,
une implémentation, une réponse au nouveau
paradigme du bigdata !



Big Data =
Nosql =
Yarn =
Cloud =
Hadoop =
Mapreduce =
Pregel =
Dremel =
Buzz ?

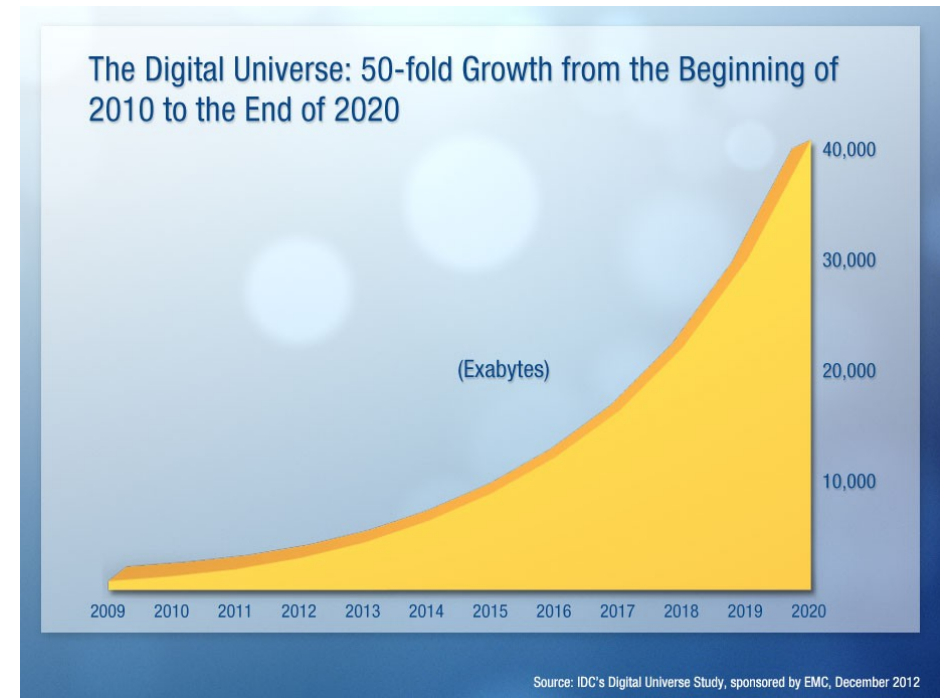
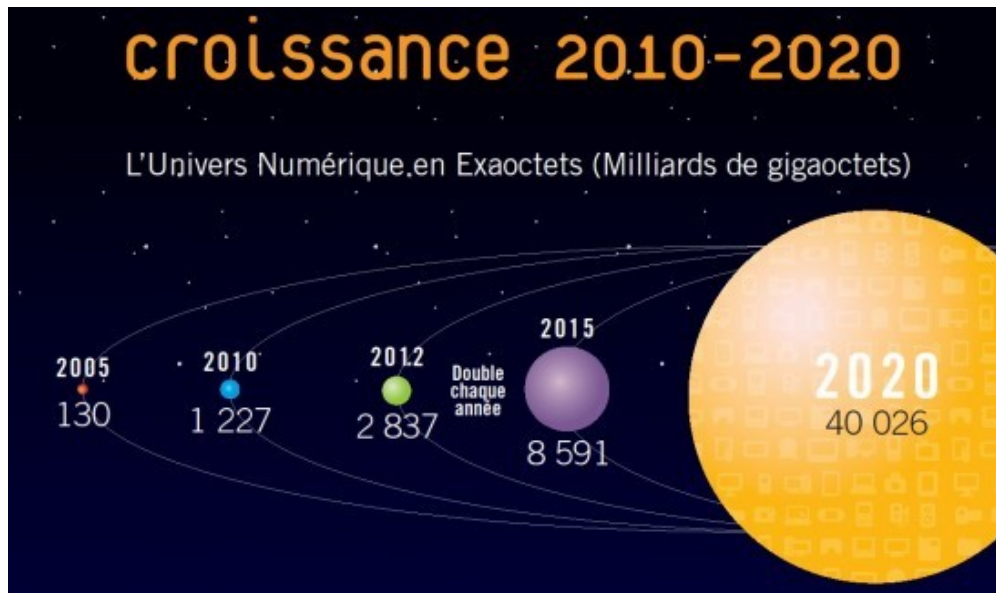


Rappel historique, quelques chiffres :

Evolution de la production mondiale de données depuis 2005

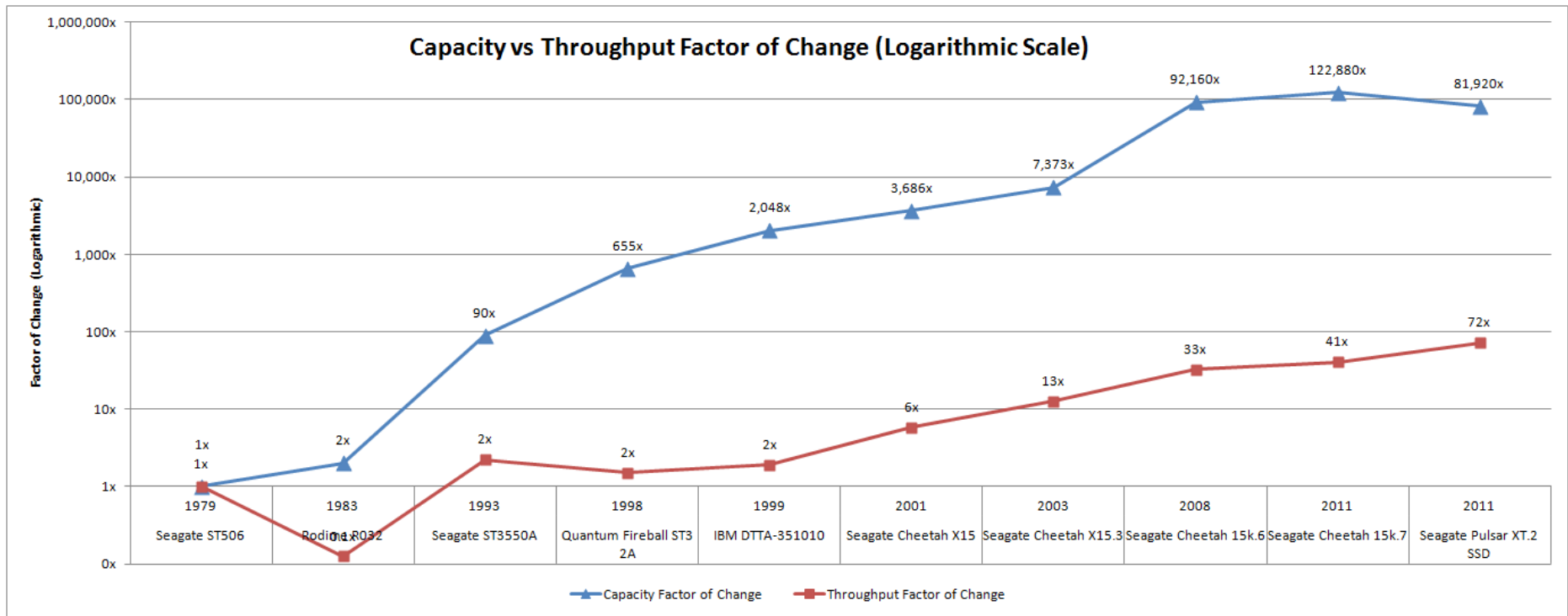
-2005 : 130 Exaocets

-2011 : 1800 Exaocets/1.8 Zettaoctets (1.8×10^{21})



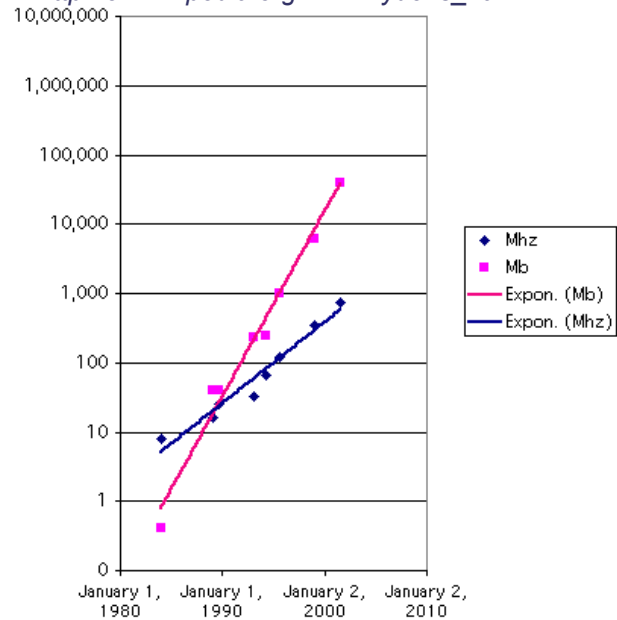
173 million	107 trillion	566 billion	340 million	50 billion	82 petabytes	60 hours	845 million
blogs online	emails sent in 2010	objects stored on Amazon's S3 cloud service by the end of 2011	tweets posted to Twitter every day by its 140 million active users	pages indexed by Google (December 2011)	stored on the largest Yahoo! Hadoop cluster	of video is uploaded to YouTube every minute. That's 1 hour every second	monthly active Facebook users resulting in an average of 15TB of data collected each day

Et l'evolution des disques durs (taille & débit) !



Loi de Kryder Vs Loi de Moore

http://en.wikipedia.org/wiki/Kryder's_Law

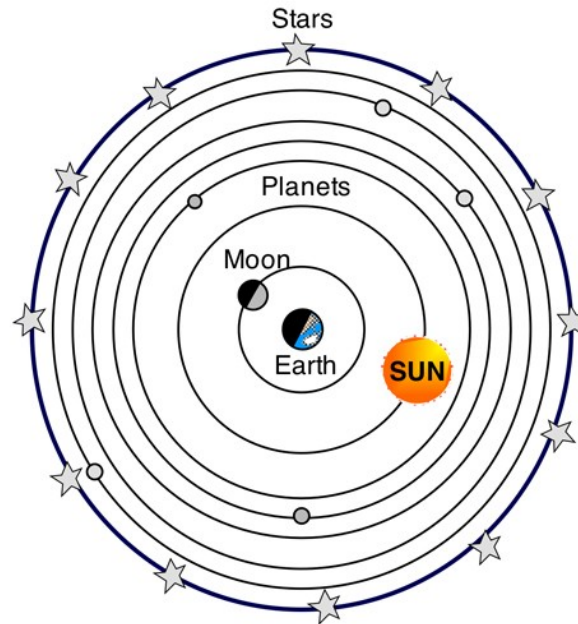


Changement de paradigme ...



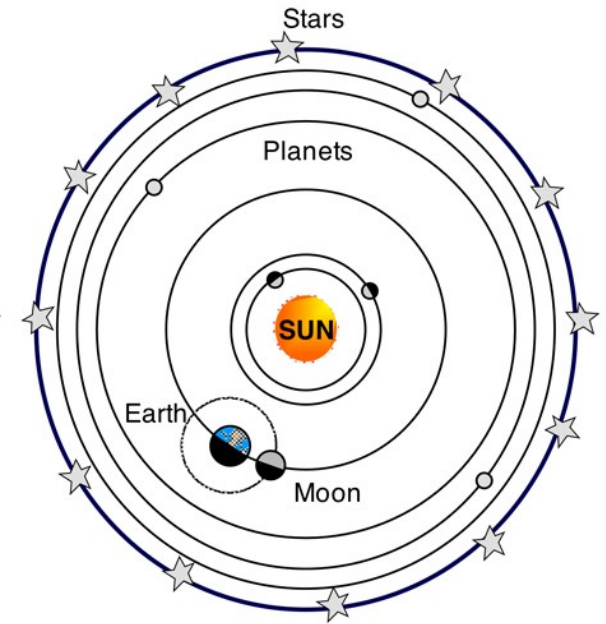
L'astronome - Vermeer - 1668

Geocentric



Geocentric Theory

Heliocentric



Heliocentric Theory

Changement de paradigme :

Déluge de donnée ou « datafication »
Changement quantitatif qui implique une rupture

Big Data, un double concept :
les 3 V : Volumétrie, Vélocité, Variabilité
(Véracité, Valeur ...)

+

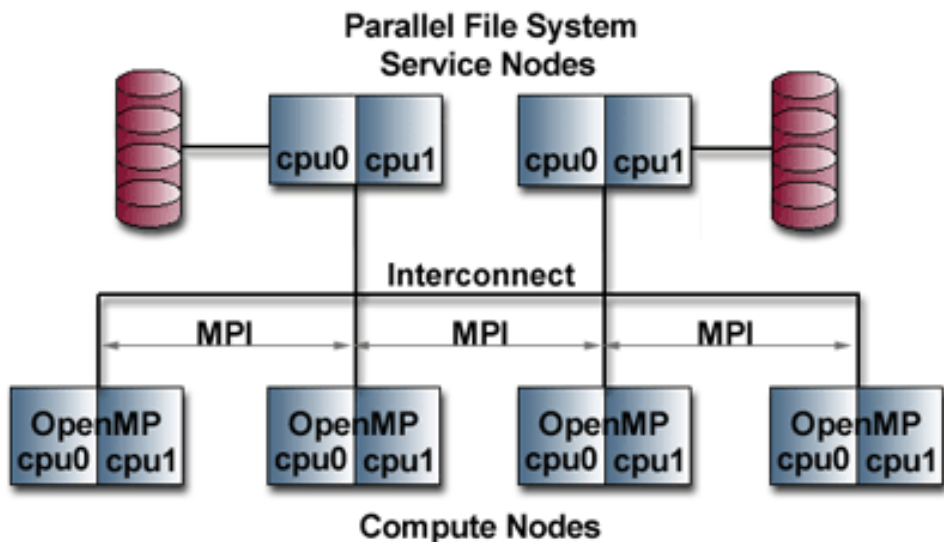
l'Algorithmique : Traitement des données



The Great Wave of Kanagawa - K. Hosukai (1760-1849)

CPU Centric

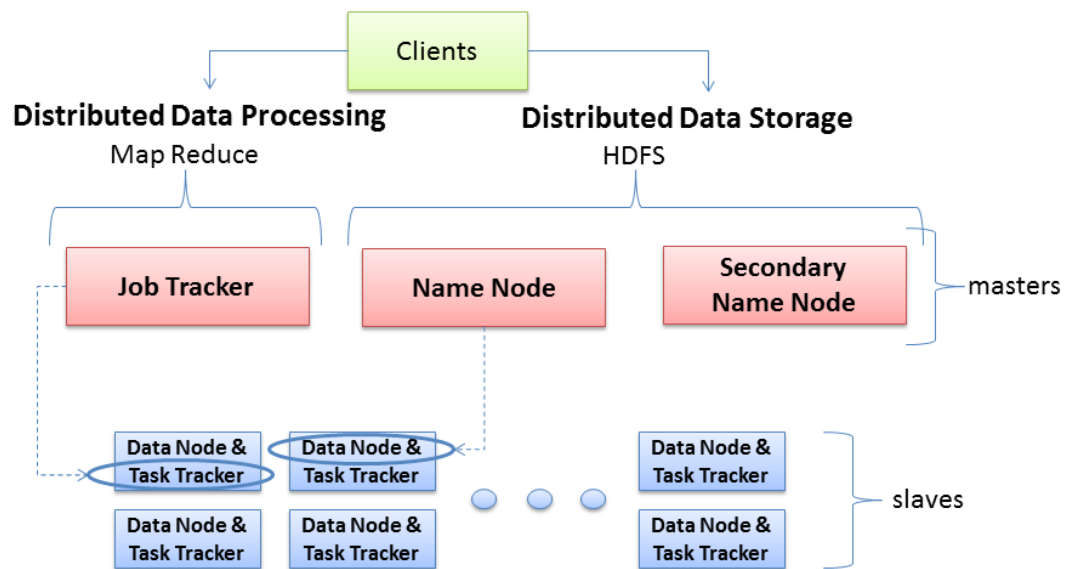
Cluster de nœuds calculs avec à coté un espace de stockage



HPC - MPI (Message Passing Interface)

Data Centric

Cluster de nœuds [calculs + stockage]



Hadoop - Mapreduce - HDFS

Bigdata:

"data centric"

"move code to data"

"distributed I/O"

"distributed compute"

"data locality"



Big Data, un bref historique :

Google
September 1998
Google founded



WIKIPEDIA
The Free Encyclopedia
January 2001
Jimmy Wales and Larry Sanger launch Wikipedia

Gmail
April 2004
Google launches Gmail

YouTube
February 2005
Youtube is launched

facebook
September 2006
Facebook is launched

Android OS
November 2007
Android OS beta released

twitter
February 2006
Twitter is launched

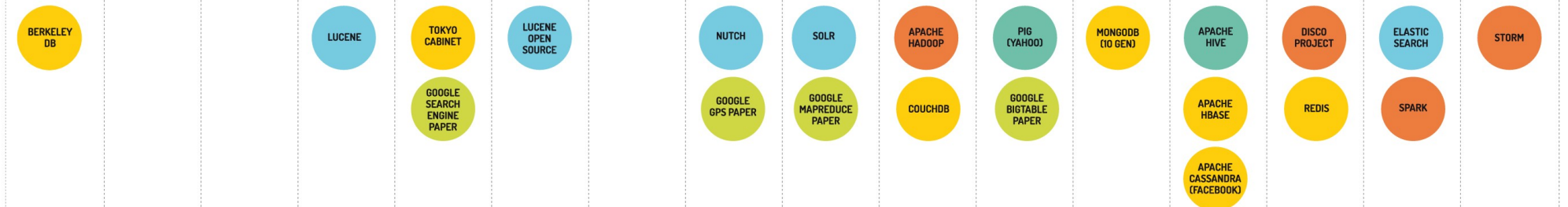
Apple iPhone
June 2007
First Generation iPhone

foursquare
March 2009
Foursquare is launched

WEB DATA

SOCIAL DATA

MOBILE DATA



January 1996
Larry Page and Sergey Brin begin their research project @ Stanford University.

1999
Doug Cutting writes Lucene, a key component in Apache Nutch, Apache Solr and ElasticSearch.

2000
Larry Page and Sergey Brin present Google Design in the paper "The Anatomy of a Large-Scale Hypertextual Web Search Engine".

2003
Doug Cutting and Mike Cafarella write and demo Nutch, an open source web search engine, successfully indexing 100-million pages.

2004
Yonik Seeley writes Solr at CNET Networks, a project to add search capabilities to websites. Google releases a paper about applying MapReduce on Large Data Clusters.

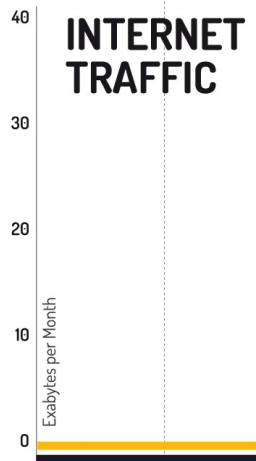
2005
Doug Cutting and Mike Cafarella implement Hadoop, a framework derived from Google MapReduce and GFS papers.

2006
Google presents a paper about BigTable, an scalable, distributed storage system for managing structured data.

2009
Nokia Research Center develop the Disco Project, a lightweight implementation of MapReduce. Jobs are typically written in Python, thus freeing MapReduce from Java jail.

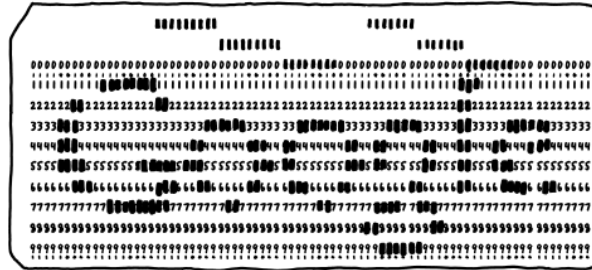
February 2010
Shay Banon releases the first version of ElasticSearch, an easily scalable search solution, based on Lucene.

September 2011
Storm, and event processor and distributed computation framework is released. Backtype, the original company who developed it, was acquired by Twitter in July, 2011.



● HIGH-LEVEL LANGUAGES
● DISTRIBUTED COMPUTATION
● RESEARCH PAPERS
● SEARCH TECHNOLOGIES
● DATABASE TECHNOLOGIES

Big Data, les précurseurs :



2003 : Publication de Google FS

2004 : Publication de Google Mapreduce

2006 : Publication de Google BigTable

2006 : 1ere version Hadoop (Doug Cutting)

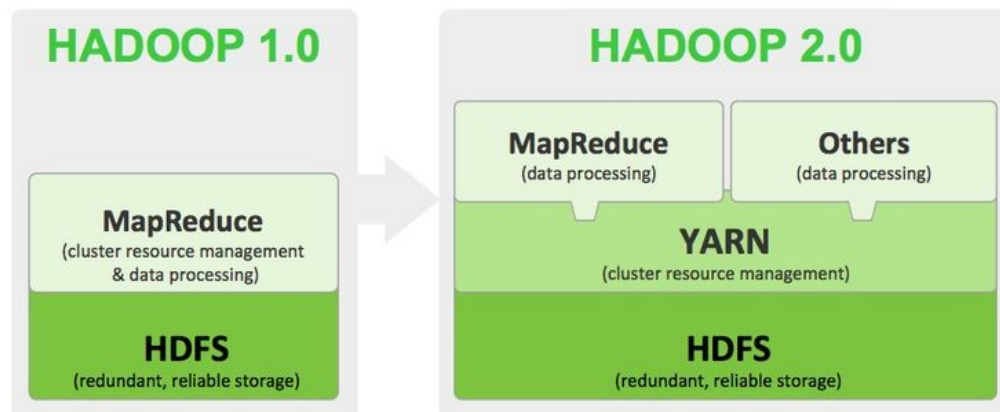
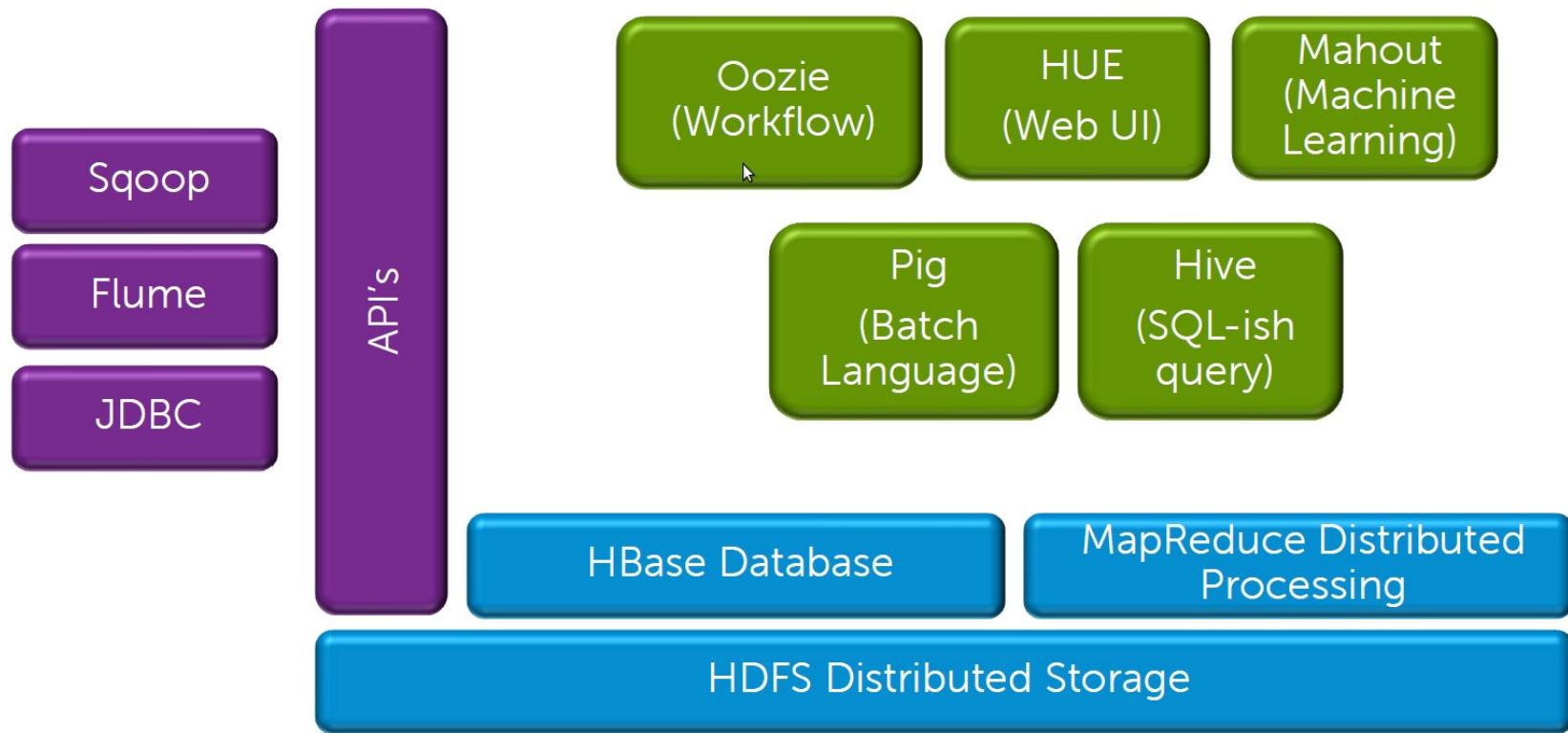
Hadoop = mapreduce + hdfs + hbase

research.google.com/archive/gfs-sosp2003.pdf

research.google.com/archive/mapreduce-osdi04.pdf

research.google.com/archive/bigtable-osdi06.pdf

Hadoop : un écosystème



RHadoop : 3 packages R



<https://github.com/RevolutionAnalytics/RHadoop/wiki/Downloads>

Rmr2 : librairie pour utiliser Hadoop Mapreduce

Rhdfs : librairie pour utiliser Hadoop HDFS

Rhbase : librairie pour utiliser Hadoop Hbase

```
library(rhdfs)
library(rhbase)
library(rmr2)

hdfs.init()

# Generate a 1:100 vector and dump it explicitly in the output defined
ints = to.dfs(1:100, output="/tmp/ints.out/")

# Define MapReduce and execute it
calc = mapreduce(input = ints, map = function(k, v) cbind(v, 2*v))

# Read data back from HDFS
results <- from.dfs(calc)

# Dump the results
results
```

Autre alternative: Rhipe
<http://www.datadr.org/getpack.html>

Mapreduce :

« Divide and conquer » « programmation fonctionnelle »

1960 - Karatsuba_algorithm, Divide & Conquer

http://en.wikipedia.org/wiki/Karatsuba_algorithm

1960 - Lisp par John McCarthy (MIT) :

« Fonctions Récursives d'expressions symboliques et leur évaluation par une Machine, partie I »

<http://fr.wikipedia.org/wiki/Lisp>

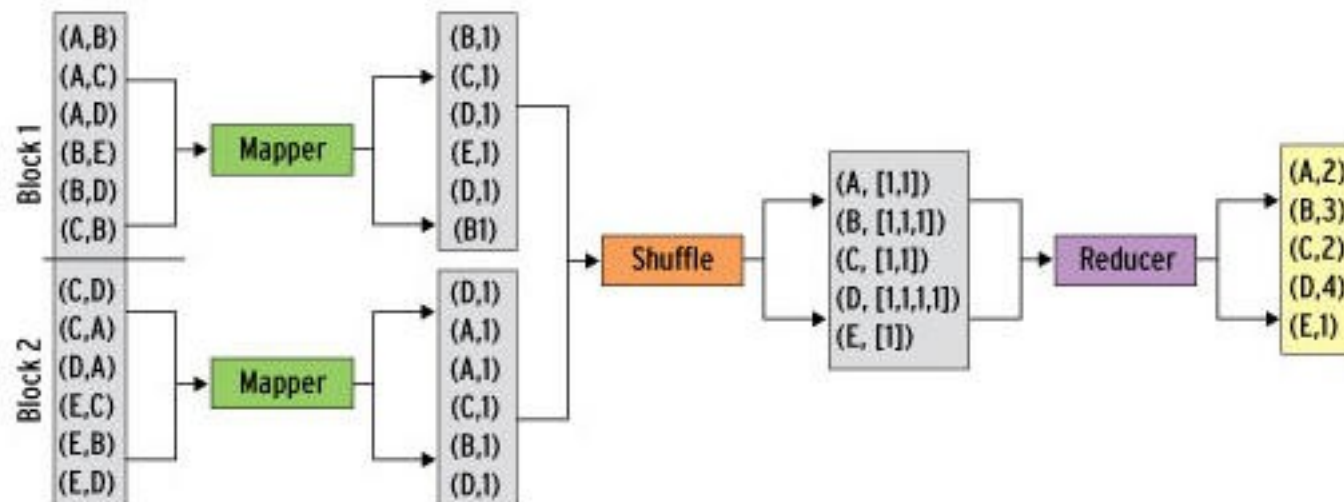
Mapreduce consiste en deux fonctions map() et reduce().

map: $(K1, V1) \rightarrow list(K2, V2)$

reduce: $(K2, list(V2)) \rightarrow list(K3, V3)$

Exemple : Moteur de recherche & PageRank

ilpubs.stanford.edu/422/1/1999-66.pdf



Mapreduce : exemple de traitement de données météo

Trouver la température maximale par année

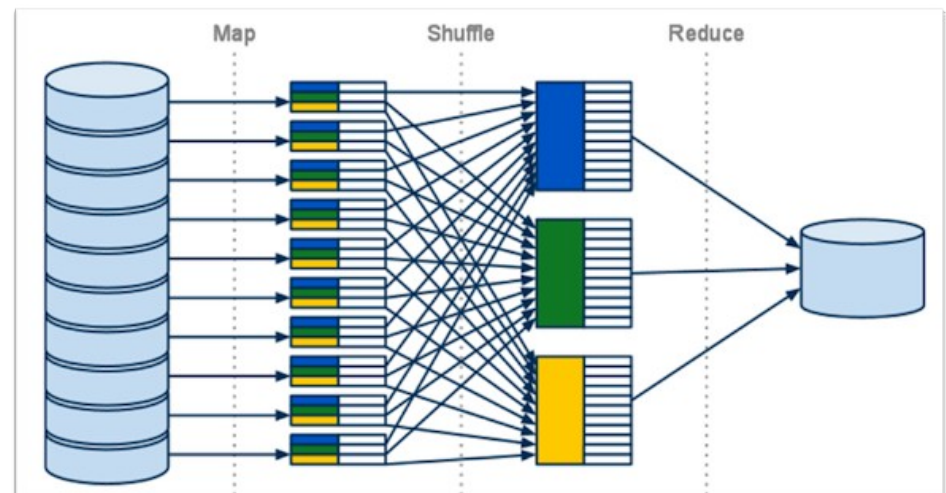
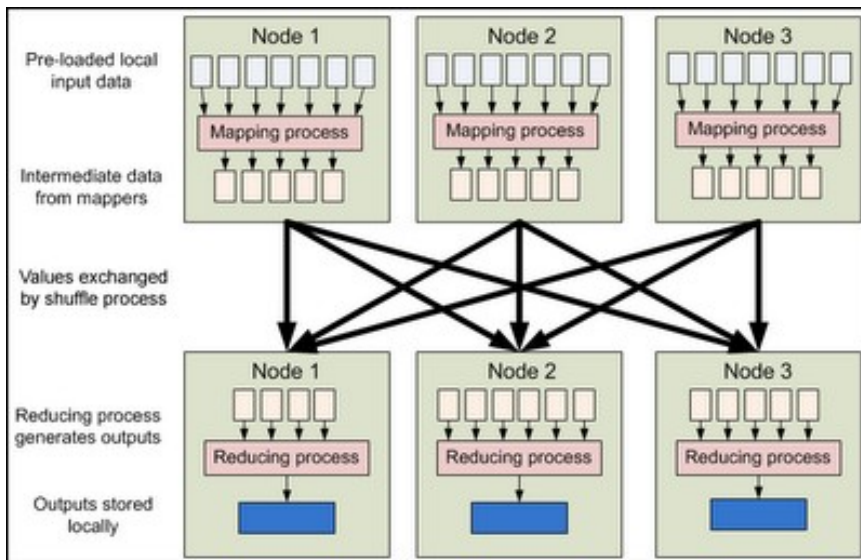
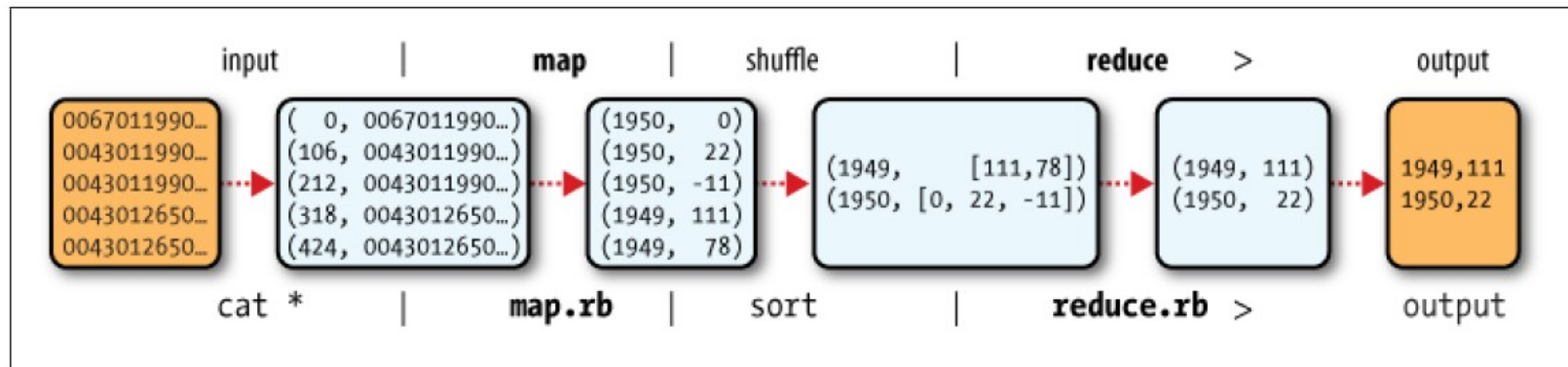
Données météorologique USA depuis 1900, <http://www.ncdc.noaa.gov/>

YYYY = Year

TTTTT = Temperature in units of 0.1 C; 9999 means missing

Q = Quality code

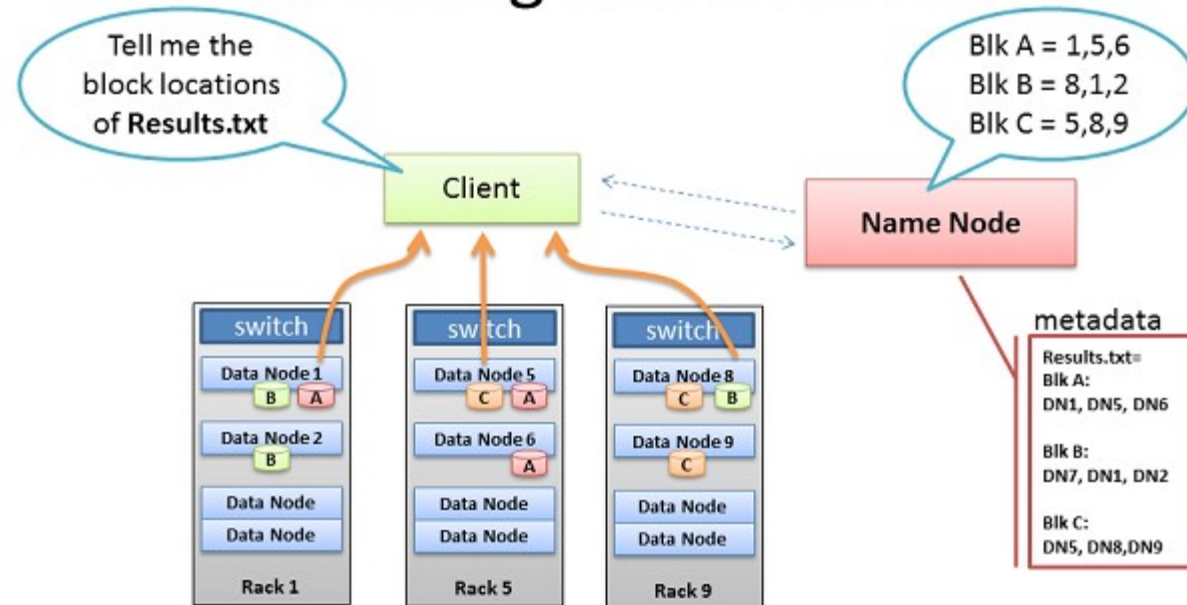
YYYY
TTTTTQ
 0029029070999991901010106004+64333+023450FM-12+000599999V0202701N01591999999N0000001N9-00781+99999102001ADDGF108991999999999999999999
 0029029070999991901010113004+64333+023450FM-12+000599999V0202901N00821999999N0000001N9-00721+99999102001ADDGF104991999999999999999999
 0029029070999991901010120004+64333+023450FM-12+000599999V0209991C00001999999N0000001N9-00941+99999102001ADDGF108991999999999999999999
 0029029070999991901010206004+64333+023450FM-12+000599999V0201801N00821999999N0000001N9-00611+99999101831ADDGF108991999999999999999999



HDFS

HDFS est un système de fichiers **distribué, extensible et portable** développé par Hadoop à partir du **GoogleFS**. Il a été conçu pour stocker de très gros volumes de données sur un grand nombre de machines équipées de **disques durs banalisés**. Il permet l'abstraction de l'architecture physique de stockage, afin de manipuler un système de fichiers distribué comme s'il s'agissait d'un disque dur unique.

Client reading files from HDFS



- Client receives Data Node list for each block
- Client picks first Data Node for each block
- Client reads blocks sequentially

BRAD HEDLUND .com

http://hadoop.apache.org/docs/stable1/hdfs_design.html

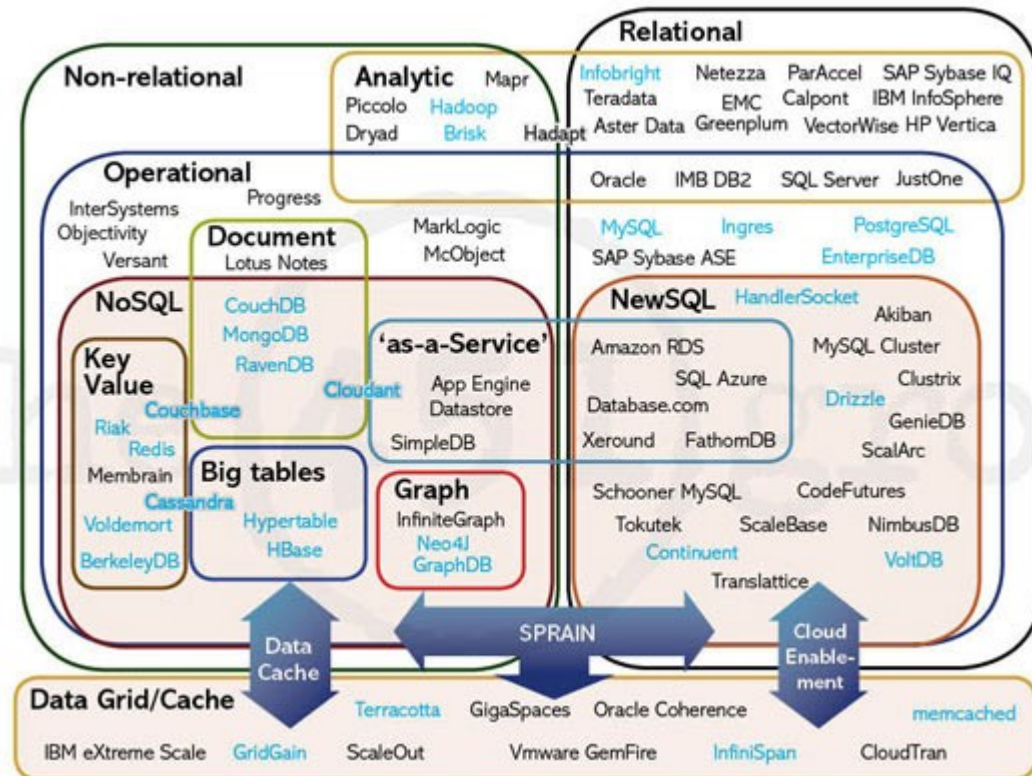
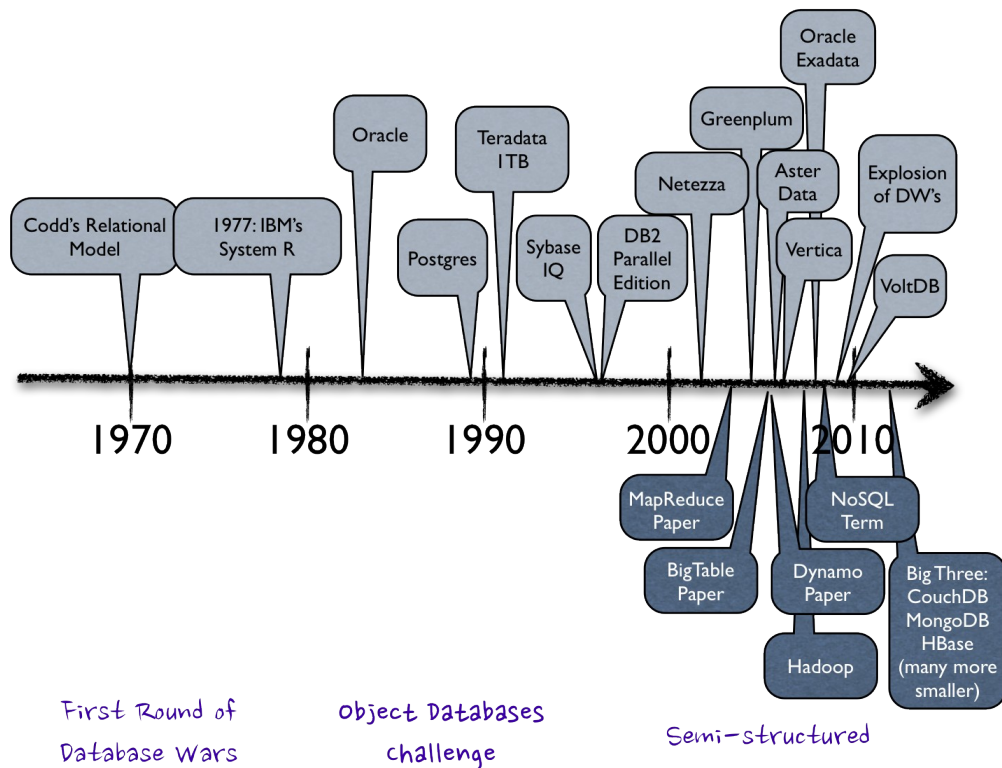
<http://bradhedlund.com/2011/09/10/understanding-hadoop-clusters-and-the-network/>

Hbase,

Nosql, CouchDB, MongoDB, Newsql, Giraph, Neo4j ,GraphLab ...

HBase est un système de gestion de base de données non-relationnelles distribué, écrit en Java, disposant d'un stockage structuré pour les grandes tables.

HBase est inspirée des publications de Google sur BigTable. Comme BigTable, elle est une base de données orientée colonnes.



Hadoop , Rhadoop: Comment installer, tester, utiliser ? Quelques pistes :

-Piste 1 : Université Paul Sabatier / DTSI



Cluster Hadoop 4 Noeuds (8xCPU/32GoRam/250Go HDD) + R + Rhadoop + Rstudio
Installation sur infrastructure virtualisée Vmware/Netapp, via Cloudera Manager CDH4
(solution sous optimale , limitée à de petits tests)

The screenshot displays the Cloudera Manager web interface. At the top, there is a navigation bar with 'cloudera manager' and various menu items like 'Accueil', 'Clusters', 'Hôtes', 'Diagnostics', 'Audits', 'Tableaux', 'Backup', and 'Administration'. A search bar and user profile 'admin' are also visible. Below the navigation bar, there are several tabs for monitoring: 'Etat', 'Tous les problèmes d'état d'intégrité', 'Tous les problèmes de configuration' (with a '4' notification), and 'Toutes les commande récentes'. The main content area is divided into several sections:

- Etat:** A sidebar showing the status of 'Cluster 1 (CDH4)' and 'Cloudera Management Services'. Under 'Cluster 1', services like hbase, hdfs, hive, hue, mapreduce, oozie, and zookeeper are listed with their respective status icons and configuration problem counts (e.g., hdfs has 1 problem, zookeeper has 1). Under 'Cloudera Management Services', 'mgmt' is listed with 2 configuration problems.
- Tableaux:** A central area with four performance charts for 'Cluster 1 (CDH4)'.
 - UC du cluster:** A line graph showing CPU usage in percent over time, peaking at 100% around 04:15 PM.
 - E/S disque du cluster:** A line graph showing disk I/O in bytes/second, with a significant spike reaching 5.7M/s at 04:15 PM.
 - E/S réseau du cluster:** A line graph showing network I/O in bytes/second, with a peak of 1.9M/s at 04:15 PM.
 - E/S HDFS:** A line graph showing HDFS I/O in bytes/second, with multiple peaks reaching up to 38.1M/s.
- Travaux MapReduce en cours d'ex...** and **Demandes HBase:** Partially visible sections at the bottom of the interface.

Hadoop : en pratique

Méthode 1 : appel natif d'un programme mapreduce écrit en java

Nous utilisons ici le programme « wordcount »

```
wget http://www.gutenberg.org/files/4300/old/ulyss12.txt
hadoop dfs -mkdir gutenbergs/books
hadoop dfs -put ulyss12.txt /data/gutenberg/books
hadoop jar $HADOOP_HOME/hadoop-examples.jar wordcount gutenbergs/books gutenbergs/out
hadoop dfs -cat gutenbergs/out/* |sort -rn -k2 |more
the 12621
of 7516
and 5929
```

...

Méthode 2 : utilisation de la fonctionnalité de « streaming » autorisant la réutilisation de programmes écrits en Python, R , Perl , Bash,

```
hadoop jar $HADOOP_HOME/hadoop-streaming.jar \  
-input REPIN \  
-output REPOUT \  
-mapper /bin/cat \  
-reducer /bin/wc
```

```
hadoop jar $HADOOP_HOME/hadoop-streaming.jar \  
-input REPIN \  
-output REPOUT \  
-mapper monscript1.py \  
-reducer monscript2.py \  
-file monscript1.py \  
-file monscript2.py
```

```
hadoop jar $HADOOP_HOME/hadoop-streaming.jar \  
-input REPIN \  
-output REPOUT \  
-mapper monscript1.R \  
-reducer monscript2.R \  
-file monscript1.R \  
-file monscript2.R
```


Exemple 1 / jeu de donnée : stat-computing.org - Challenge 2009: Horaires/Départs/Arrivées/Retards des compagnies aériennes USA (1987-2009) 120 millions d'entrées, 29 variables

```
dsrtadm@node1:~$ hadoop dfs -ls airline/data
```

Found 22 items

```
-rw-r--r--  3 dsrtadm hdfs  127162942 2014-01-22 22:32 airline/data/1987.csv  
-rw-r--r--  3 dsrtadm hdfs  501039472 2014-01-22 22:31 airline/data/1988.csv  
.....  
-rw-r--r--  3 dsrtadm hdfs  671027265 2014-01-22 22:39 airline/data/2005.csv  
-rw-r--r--  3 dsrtadm hdfs  672068096 2014-01-22 22:38 airline/data/2006.csv  
-rw-r--r--  3 dsrtadm hdfs  702878193 2014-01-22 22:39 airline/data/2007.csv  
-rw-r--r--  3 dsrtadm hdfs  689413344 2014-01-22 22:41 airline/data/2008.csv
```

Traitement :

Générer un fichier de sortie : YEAR MONTH COUNT-DEPTDELAY AIRLINE_CODE AVG_DELAY(Ms)

```
hadoop jar $HADOOP_HOME/contrib/streaming/hadoop-streaming-*.jar \  
-input airline/data \  
-output dept-delay-month \  
-mapper map.R \  
-reducer reduce.R \  
-file map.R -file reduce.R
```

Ou

```
R CMD BATCH detpdelay.R
```

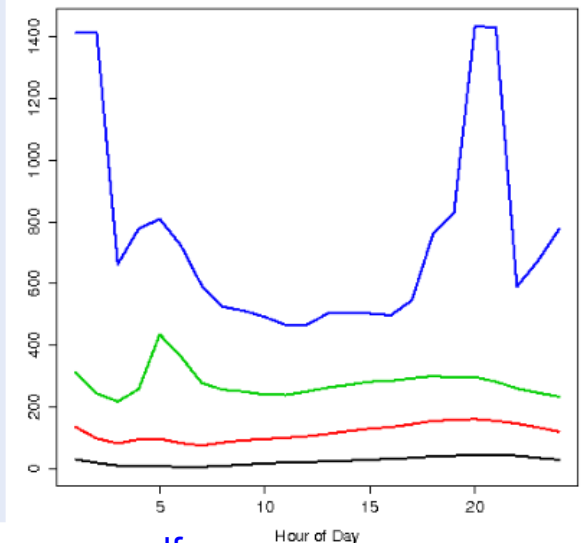
Résultats :

```
hadoop dfs -cat dept-delay-month/part* \  
| sort -rn -k3 | more  
2008      7    102986    WN    9.946634  
2008      5    102216    WN    9.428857  
2008      3    101423    WN   12.7478  
...
```

LESSONS LEARNED: TIPS FOR TRAVELERS



- Avoid flying during holidays and summer
- Fly in April, May, and September
- Watch the weather!
- Avoid airports (Newark, JFK, Chicago,...) with consistent delays
- Use carriers (Aloha, Hawaiian, Southwest,...) with superior on-time performance
- Fly early in the day
- Avoid flights that depart between 5 and 7 p.m.



<http://stat-computing.org/dataexpo/2009/posters/kane-emerson.pdf>
<http://stat-computing.org/dataexpo/2009/posters/wicklin-allison.pdf>

Amazon Web Services

Compute & Networking

- Direct Connect**
Dedicated Network Connection to AWS
- EC2**
Virtual Servers in the Cloud
- Route 53**
Scalable Domain Name System
- VPC**
Isolated Cloud Resources

Storage & Content Delivery

- CloudFront**
Global Content Delivery Network
- Glacier**
Archive Storage in the Cloud
- S3**
Scalable Storage in the Cloud
- Storage Gateway**
Integrates On-Premises IT Environments with Cloud Storage

Database

- DynamoDB**
Predictable and Scalable NoSQL Data Store
- ElastiCache**
In-Memory Cache
- RDS**
Managed Relational Database Service
- Redshift**
Managed Petabyte-Scale Data Warehouse Service

Deployment & Management

- CloudFormation**
Templated AWS Resource Creation
- CloudTrail**
User Activity and Change Tracking
- CloudWatch**
Resource and Application Monitoring
- Elastic Beanstalk**
AWS Application Container
- IAM**
Secure AWS Access Control
- OpsWorks**
DevOps Application Management Service

Analytics

- Data Pipeline**
Orchestration for Data-Driven Workflows
- Elastic MapReduce**
Managed Hadoop Framework
- Kinesis**
Real-time Processing of Streaming Big Data

App Services

- CloudSearch**
Managed Search Service
- Elastic Transcoder**
Easy-to-use Scalable Media Transcoding
- SES**
Email Sending Service
- SNS**
Push Notification Service
- SQS**
Message Queue Service
- SWF**
Workflow Service for Coordinating Application Components

Amazon EC2 instances pricing

VCPU	ECU	Mémoire (GiB)	Stockage des instances (Go)	Utilisation de Linux/UNIX
m1.small	1	1	1.7 1 x 160	\$0.060 par heure
m1.med	1	2	3.75 1 x 410	\$0.120 par heure
m1.large	2	4	7.5 2 x 420	\$0.240 par heure
m1.xlarge	4	8	15 4 x 420	\$0.480 par heure

Spot (ponctuelle) / région USA Est (Virginie) Instance Linux

- m1.small \$0.007 par heure**
- m1.med \$0.013 par heure**
- m1.large \$0.026 par heure**
- m1.xlarge \$0.052 par heure**

Méthode 1 : Amazon EMR (Elastic Map Reduce) :

Activer un cluster Hadoop 1.0 pré-configuré + Rhadoop de 10 Noeuds avec 10To stockage en ligne de commande (Activation en ~15 mn).
Utilisation de l'outil écrit en Ruby « elastic-mapreduce » d'Amazon



```
elastic-mapreduce --create --alive --master-instance-type=m1.large \  
--slave-instance-type=m1.medium --num-instances=10 --enable-debugging \  
--hadoop-version 1.0.3 --ami-version 2.4.2 --name 10-NODES-RHADOOP \  
--bootstrap-action s3://elasticmapreduce/bootstrap-actions/install-ganglia --bootstrap-action \  
s3://bigdataths/script/bootstrap-rhadoop3-deb6.sh \  
--jar s3://elasticmapreduce/libs/script-runner/script-runner.jar --step-name instpig --args \  
"s3://elasticmapreduce/libs/pig/pig-script,--base-path,s3://elasticmapreduce/libs/pig/,--install-pig,--pig-versions,latest" \  
--jar s3://elasticmapreduce/libs/script-runner/script-runner.jar --step-name insthive --args \  
"s3://elasticmapreduce/libs/hive/hive-script,--base-path,s3://elasticmapreduce/libs/hive/,--install-hive,--hive-versions,latest"  
  
elastic-mapreduce --list  
  
j-2GHO33PZ7U9N1      BOOTSTRAPPING  ec2-54-195-17-24.eu-west-1.compute.amazonaws.com  10-NODES-RHADOOP  
  PENDING           Setup Hadoop Debugging  
  PENDING           instpig  
  PENDING           insthive  
  
elastic-mapreduce  -j j-34TUDSWU0RK3J --ssh  
  
elastic-mapreduce --terminate j-34TUDSWU0RK3J
```

Méthode 2 : Amazon EC2 et l'outil Whirr d'apache.

Activer un cluster de 10 nœuds + Hadoop personnalisé + Rhadoop , 10 To en ligne de commande (Activation en ~15 mn)

Utilisation de « whirr » de la fondation apache.

```
1 compte Amazon AWS
1 machine locale linux (debian/ubuntu/centos)
Avoir installer whirr (et JDK)
1 fichier de conf « hadoop.properties »

whirr.service-name=hadoop
whirr.cluster-name=10-nodes-demo
whirr.instance-templates=1 jt+nn,9 dn+tt
whirr.provider=ec2
whirr.identity=[AWS ID]
whirr.credential=[AWS KEY]
whirr.private-key-file=${sys:user.home}/.ssh/id_rsa
whirr.public-key-file=${sys:user.home}/.ssh/id_rsa.pub
whirr.hadoop-install-runurl=cloudera/cdh/install
whirr.hadoop-configure-runurl=cloudera/cdh/post-configure

whirr launch-cluster --config hadoop.properties

whirr destroy-cluster --config hadoop.properties
```

<http://www.evanconkle.com/2011/11/run-hadoop-cluster-ec2-easy-apache-whirr/>

Méthode 3 : Amazon EC2 et Cloudera manager installer

Activer un cluster 10 nœuds + Hadoop personnalisé + Rhadoop , 10 To en ligne de commande (Activation en ~15 mn)

Utilisation de « cloudera manager installer » installation graphique depuis un premier nœud sous linux dans le cloud amazon ec2.

Lancer le binaire « cloudera-manager-installer.bin » qui va détecter l'écosystème Amazon EC2 et proposer une installation complète via un Wizard



cloudera

Merci !

Des questions ?

