République **A**lgérienne **D**émocratique et **P**opulaire
Ministère de l'**E**nseignement **S**upérieur et de la **R**echerche **S**cientifique

UNIVERSITE **M**OULOUD **M**AMMERI DE **T**IZI-**O**UZOU

FACULTE DU **G**ENIE **E**LECTRIQUE ET D'INFORMATIQUE
DEPARTEMENT D'INFORMATIQUE

Mémoire de Fin d'Etudes de Master ACADEMIQUE

Domaine : **Mathématiques et Informatique**
Filière : **Informatique**
Spécialité : **Conduite de projet informatique**

Présenté par
Kara Madouda
Belabdet Fraoudja

Thème

Etude comparative des framework de gestion de calcul intensif sur les cloud (cas d'étude Hadoop vs Spark).

Mémoire soutenu publiquement le 27/09/ 2016 devant le jury composé de :

Président : M^r^ Younes YACINE

Encadreur : M^r^ Ahmed DIB

Examinateur : M Mhand KARBICHE

Liste Des Figure

Chapitre I

Figure 01 : Processeur

Figure 02 : Processeur multi-cœur

Figure 03 : Threading

Figure 04: Multi-threading.

Figure 05 : Message passing interface

Figure 06: High Performance Computer

Figure 07: Grilles de calcul

Figure 08 : Définitions et contours du Cloud Computing

Figure 09 : Les SaaS (Saftwore as a Service)

Figure 10 : Les PaaS (Plateform as a Service)

Figure 11: IaaS (Infrastructure as a Service)

Figure 12 : Les différents modèles de déploiement d'un Cloud computing

Figure 13: Les 4vs de Big data

Figure 14 : Fonctionnement de map reduce

Figure15 : Schéma du Hadoop Distributed File System

Figure 16 : Couche de chargement des données dans le BigData.

Figure 17: Architecture de Big Data.

Figure 18 : Lien entre Big Data et DW

Chapitre II

Figure 01: Plate-forme HPCC et architecture.

Figure02: Architecture du storm

Figure03: Ecosysteme d'hadoop

Figure 04: L'architecture d'HDFS

Figure 05: Lecture d'un fichier HDFS

Figure 06: Processus d'écriture dans un volume ou fichier HDFS.

Figure07: Le traitement de MapReduce.

Figure 08: Exemple d'un programme MapReduce (WordCount)

Figure 09: Architecture maître-esclave du MapReduce.

Figure 10: Spark Framework Libraries.

Chapitre III

Figure01: Segmentation de la vidéo

Figure02 : Segmentation de vidéo basée sur les images clés**.**

Figure03: Processus de détection des shots avec Hadoop

Figure04: Acer Aspire 5733Z-P624G50Mn-performance

Figure05: Ubuntu 14.04

Figure06: Virtuel Box

Figure07: Comparaison du nombre de shots détecté entre l'approche basée sur MapReduce et des existantes approches.

LISTE DES TABLEAU

Tableau I.1: Avantages et inconvénients des services.

Tableau II.2: Hadoop vs Spark.

Tableau III.1: Caractéristiques du contenu actuel OVD

Tableau III.2: Quelque commende de hadoop.

Tableau III.3: Quelque commende de FFMPEG.

Tableau III.4: Résultat d'exécution.

Liste des abréviations

MPI : Message Passing Interface

HPC : High performance computing

SaaS: saftwore as a Service

PaaS: Plateform as a Service

IaaS : Infrastructure as a Service

DW: Data warehouse

HPCC : Cluster de calcul haut performance

THOR: the refinery cluster data

Roxie : Rapid en ligne XML Inquiry Engine

ECL: Enterprise Control Language

HDFS: Hadoop Distributed File System

RDD: Resilient Distributed Datasets

SVD: Singular Value Decomposition

OVDL: Open Video Digital Library

API: Application Programming Interface

UDF: User Defined Function

UDAF: User Defined Aggregate Function

IO: Input Output

IT: INFORMATIQUE TECHNOLOGY

GPL: General Public License

Sommaire

Introduction général ……

Chapite1 l'évolution d l'unité du calcul

1 Introduction : 1

2 Les bases de l'unité de calcul : 1

2.1 Les processeurs : 2

2.2 Definition d'un cœur: ..... **Erreur ! Signet non défini.**

2.3 Processeur multi-cœur : ..... **Erreur ! Signet non défini.**

2.3.1 Le Dual Core : ..... **Erreur ! Signet non défini.**

2.3.2 Le Quad Core: ..... **Erreur ! Signet non défini.**

2.3.3 Présentation des processeurs Intel Core: ..... **Erreur ! Signet non défini.**

2.4 Les clusters : ..... **Erreur ! Signet non défini.**

2.4.1 Objectifs: ..... **Erreur ! Signet non défini.**

2.4.2 Architecture: ..... **Erreur ! Signet non défini.**

2.5 Comparisons entre multiproceseurs et les clusters : ..... **Erreur ! Signet non défini.**

3 La gestion parallèle : ..... **Erreur ! Signet non défini.**

3.1 Définition d'un thread : ..... **Erreur ! Signet non défini.**

3.1.1 Les avantages d'un thread : ..... **Erreur ! Signet non défini.**

3.2 Définition d'un multi- threading: ..... **Erreur ! Signet non défini.**

3.2.1 Les avantages de l'hyper-threading sont les suivants : ..... **Erreur ! Signet non défini.**

3.3 MPI (Message Passing Interface) : ..... **Erreur ! Signet non défini.**

3.3.1 Définition de Message Passing Interface : ..... **Erreur ! Signet non défini.**

3.3.2 Principales catégories de MPI : ..... **Erreur ! Signet non défini.**

3.3.3 Fonctionnalité MPI : ..... **Erreur ! Signet non défini.**

3.4 HPC (High performance computing): ..... **Erreur ! Signet non défini.**

3.4.1 Définition d'un HPC : ..... **Erreur ! Signet non défini.**

Sommaire

3.4.2 Utilisation : ..... **Erreur ! Signet non défini.**

4 Les grille de calcule : ..... **Erreur ! Signet non défini.**

4.1 Définition d'une grille informatique : ..... **Erreur ! Signet non défini.**

4.1.1 Propriété d'une grille informatique: ..... **Erreur ! Signet non défini.**

4.2 La Grille de calcul : ..... **Erreur ! Signet non défini.**

4.3 L'ingénierie concourante : ..... **Erreur ! Signet non défini.**

5 Cloud computing : ..... **Erreur ! Signet non défini.**

5.1 Définition : ..... **Erreur ! Signet non défini.**

5.2 L'historique: ..... **Erreur ! Signet non défini.**

5.3 Les modèles de Cloud Computing : ..... **Erreur ! Signet non défini.**

5.3.1 Les SaaS (saftwore as a Service): ..... **Erreur ! Signet non défini.**

5.3.2 Les PaaS (Plateform as a Service) : ..... **Erreur ! Signet non défini.**

5.3.3 Les IaaS (Infrastructure as a Service) : ..... **Erreur ! Signet non défini.**

5.3.4 Avantages et inconvénients des services : ..... **Erreur ! Signet non défini.**

5.4 Les catégories de cloud computing : ..... **Erreur ! Signet non défini.**

5.4.1 Les Clouds privés : ..... **Erreur ! Signet non défini.**

5.4.2 Les Clouds publics: ..... **Erreur ! Signet non défini.**

5.4.3 Les Cloud hybrides : ..... **Erreur ! Signet non défini.**

5.4.4 Les Cloud communautaires : ..... **Erreur ! Signet non défini.**

5.5 Caractéristique de Cloud: ..... **Erreur ! Signet non défini.**

5.6 Avantages et inconvénients du Cloud Computing : ..... **Erreur ! Signet non défini.**

6 Big Data: ..... **Erreur ! Signet non défini.**

6.1 Définition: ..... **Erreur ! Signet non défini.**

6.2 Historique: ..... **Erreur ! Signet non défini.**

6.3 Les principes de base de Big Data : ..... **Erreur ! Signet non défini.**

6.4 Technologie du big data: ..... **Erreur ! Signet non défini.**

6.4.1 Map Reduce: ..... **Erreur ! Signet non défini.**

6.5 Hadoop: ..... **Erreur ! Signet non défini.**

6.5.1 Hadoop et les analyses massives: ..... **Erreur ! Signet non défini.**

6.6 Processus de chargement et de collecte de données dans Big Data : ..... **Erreur ! Signet non défini.**

6.7 Architecture Big Data: ..... **Erreur ! Signet non défini.**

6.8 Avantages de l'architecture Big Data : ..... **Erreur ! Signet non défini.**

6.9 Big Data et Datawarehouse : ..... **Erreur ! Signet non défini.**

6.10 Calcul parallèle avec R (HPC, Hadoop, BigData): ..... **Erreur ! Signet non défini.**

6.11 High Performance Computing et Big Data: 29

6.12 Quelques domaines d'utilisation du Big Data : 29

7 Conclusion: 29

Chapitre2 les frameworks de big data

1 Introduction: ..... **Erreur ! Signet non défini.**

2 Framework ..... **Erreur ! Signet non défini.**

2.1 Type des Frameworks: ..... **Erreur ! Signet non défini.**

3 FrameWork pour les cloud computing ..... **Erreur ! Signet non défini.**

3.1 HPCC System: ..... **Erreur ! Signet non défini.**

3.1.1 Définition: ..... **Erreur ! Signet non défini.**

3.1.2 Evolution de HPCC: ..... **Erreur ! Signet non défini.**

3.1.3 Types de cluster HPCC: ..... **Erreur ! Signet non défini.**

3.1.4 Les principaux avantages d'ECL: ..... **Erreur ! Signet non défini.**

3.1.5 Système HPCC Serveurs Middleware ..... **Erreur ! Signet non défini.**

3.2 Apache Storm: ..... **Erreur ! Signet non défini.**

3.2.1 Architecture du Storm: ..... **Erreur ! Signet non défini.**

Sommaire

3.2.2 Composants d'un cluster de Storm: ..... **Erreur ! Signet non défini.**

3.2.3 Modes de fonctionnement: ..... **Erreur ! Signet non défini.**

3.2.4 Les caractéristique de storm ..... **Erreur ! Signet non défini.**

4 Présentation et comparaison entre les frameWorks Hadoop et Spark ..... **Erreur ! Signet non défini.**

4.1 HADOOP: ..... **Erreur ! Signet non défini.**

4.1.1 Historique: ..... **Erreur ! Signet non défini.**

4.1.2 Définition: ..... **Erreur ! Signet non défini.**

4.1.3 Caractéristique d'Hadoop: ..... **Erreur ! Signet non défini.**

4.1.4 Écosystème d'Hadoop: ..... **Erreur ! Signet non défini.**

4.1.5 Les composants d'écosystème d'hadoop: ..... **Erreur ! Signet non défini.**

4.1.6 Les concepts fondamentaux d' Hadoop: ..... **Erreur ! Signet non défini.**

4.1.7 HDFS (Hadoop Distributed File System): ..... **Erreur ! Signet non défini.**

4.1.8 MapReduce: ..... **Erreur ! Signet non défini.**

4.1.9 MapReduce dans hadoop: ..... **Erreur ! Signet non défini.**

4.2 SPARK: ..... **Erreur ! Signet non défini.**

4.2.1 Historique: ..... **Erreur ! Signet non défini.**

4.2.2 Définition: ..... **Erreur ! Signet non défini.**

4.2.3 Les caractéristiques de Spark: ..... **Erreur ! Signet non défini.**

4.2.4 L'écosystème de Spark: ..... **Erreur ! Signet non défini.**

4.2.5 Les concepts fondamentaux de Spark : ..... **Erreur ! Signet non défini.**

4.2.6 MapReduce dans Spark : ..... **Erreur ! Signet non défini.**

4.2.7 RDD dans Spark : ..... **Erreur ! Signet non défini.**

4.2.8 Mode de cluster Présentation: ..... **Erreur ! Signet non défini.**

4.3 Hadoop vs Spark: ..... **Erreur ! Signet non défini.**

4.4 Résumé d'Hdoop vs Spark: ..... **Erreur ! Signet non défini.**

5 CONCLUSION: ..... **Erreur ! Signet non défini.**

Chapitre3 Segmentations de vidéos basées sur MapReduce

1 Introduction: ..... **Erreur ! Signet non défini.**

2 La segmentation de vidéo (par image clé ou keyframe en englais):..... **Erreur ! Signet non défini.**

2.1 Définition de la segmentation de la vidéo: ..... **Erreur ! Signet non défini.**

2.2 La segmentation des vidéos basée sur les key frames: ..... **Erreur ! Signet non défini.**

3 L'algorithme de keyframe avec Hadoop..... **Erreur ! Signet non défini.**

4 L'environnement de travail (caractéristique de la machine utilisée et version de chaque outil):
…… **Erreur ! Signet non défini.**

4.1 Outils utilisées: ..... **Erreur ! Signet non défini.**

4.1.1 Machine physique :..... **Erreur ! Signet non défini.**

4.1.2 Ubuntu: ..... **Erreur ! Signet non défini.**

4.1.3 Virtual machine: ..... **Erreur ! Signet non défini.**

4.1.4 VirtualBox: ..... **Erreur ! Signet non défini.**

4.1.5 Java: ..... **Erreur ! Signet non défini.**

4.1.6 XML: ..... **Erreur ! Signet non défini.**

4.2 Open image : ..... **Erreur ! Signet non défini.**

4.3 HADOOP 2.1.7: ..... **Erreur ! Signet non défini.**

5 Les données utilisées : ..... **Erreur ! Signet non défini.**

6 Résultat de l'exécution : ..... **Erreur ! Signet non défini.**

7 Conclusion:.....79

Conclusion général……

Annexe……

Sommaire

Introduction général

Nous sommes confrontés depuis quelques années à de nouvelles technologies qui envahissent le monde de l'informatique et l'internet ; cette situation nous oblige à prendre les défis, de connaitre et maitriser ces nouvelles sciences afin de nous permettre de s'adapter aux changements forcés par cette révolution.

Le volume de données exploité par les entreprise à considérablement augmenté émanant de sources diverses (transaction, systèmes d'information automatises, réseaux,…), elles sont souvent susceptibles de croitre très rapidement.

Lorsqu'on parle de manipulation de gros volume de données, on pense généralement à des problématiques sur le volume des données et sur la rapidité de traitement que on trouve généralement dans les vidéos.

Beaucoup de concepts inséparables dominent actuellement le monde de l'IT(**Information technology)**. On entend souvent de Cloud Computing, la technologie actuelle, hébergeant un big data sous forme **Nosql** et traité par un simple programme **MapReduce** dans des « clusters » distribués partout dans le monde.

Le terme Cloud Computing, ou « informatique dans les nuages », est un nouveau modèle informatique qui consiste à proposer les services informatiques sous forme de services à la demande, accessibles de n'importe où, n'importe quand et par n'importe qui. Cette nouvelle technologie permet à des entreprises d'externaliser le stockage de leurs données et de leur fournir une puissance de calcul supplémentaire pour le traitement de grosse quantité de données.

Cette dernier à besoin des plates-formes et des framework de Cloud Computing qu'il est étonnant, comme Hadoop qui est un framework libre, géré par la fondation Apache, conçu pour analyser de très grande quantités de données. Il supporte le passage à l'échelle est très performent en termes de tolérance aux pannes, il est composé d'HDFS (Hadoop Distributed File System), son propre système de fichiers, et de MapReduce son « moteur » de calcul distribué, et Spark qui est un grand cadre de traitement de données open source qui peut traiter des quantités massives de données à grande vitesse à l'aide informatique en grappe, et d'autre framework qui existe(HPCC, Storm…).

Avec l'existence de dizaine de framework de gestion des Cloud, il est devenu crucial de connaitre le framework qui offre la meilleure scalabilité et le passage à l'échelle l'un part

rapport à l'autre. Le présent mémoire fait l'objet d'une étude comparative qui porte sur les framework les plus connus, par la suite, on met l'accent sur Hadoop qui est la référence de l'implémentation du MapReduce, et Spark qui présente des performances élevées. Nous concluons notre travail par une mise en place d'une application de segmentation de vidéo basée sur les keyframes avec MapReduce d'Hadoop.

Plan de travaille:

Pour organiser notre travail, nous l'avons structuré en trois chapitres principaux qui sont :

Chapitre I : « L'évolution de l'unité de calcul »

Notre objectif dans se chapitre est de donné un aperçu sur l'évolution de l'unité de calcul et quelque notions de base sur les Cloud Compting et les BigData.

Chapitre II : « Les framework de big data »

Dans ce chapitre en a présenté quelques framework de BigData telleque HPCC, Storm, Hadoop et Spark, l'objectif de se chapitre est de faire une comparaison entre deux framewok concourant Hadoop et Spark.

Chapitre III : « Segmentation des vidéos basée sur MapReduce »

Ici nous avons décrit la segmentation vidéo par la méthode keyframe pour le traitement d'image et l'environnement de développement avec tous les outils utilisés ainsi que le fonctionnement de notre application.

L'objectif de notre travail est justement d'approfondir et d'expérimenter nos connaissances sur ce thème.

Chapitre 1

Evolution de l'unité de calcul

1 Introduction :

Depuis les débuts de l'ère informatique la puissance de calcul des processeurs n'a cessé de croître. Cette augmentation de puissance, qui semble suivre une évolution linéaire, est souvent confondue à tort avec la célèbre loi de **Moore**[1]. L'augmentation de la fréquence, couplée à des changements architecturaux au sein des processeurs ont permis de maintenir ce gain de puissance. Cependant, ces changements architecturaux, comme par exemple les mécanismes d'exécution dans le désordre, compliquent grandement les processeurs. De plus l'augmentation de la fréquence implique une augmentation de la consommation d'énergie et du dégagement de chaleur, et les processeurs actuels se rapprochent dangereusement de la limite de chaleur que supportent leurs composants. Partant de ce constat, les concepteurs de processeurs ont adopté une autre approche. Ainsi, au lieu de complexifier le cœur d'un processeur, leur idée est de multiplier le nombre de cœurs au sein d'un processeur. Cela permet de continuer à augmenter la puissance de calcul globale d'un processeur, tout en contournant les difficultés liées à l'augmentation de fréquence et à la dissipation thermique.

Avec le courant chapitre, on va donner une vue détaillée sur les étapes avec lesquelles s'est déroulé l'évolution de l'unité de calcul et les points essentielles marquées, en passant par le simple processeur jusqu'aux grappes et centre de calcul.

2 Les bases de l'unité de calcul :

Une unité de calcul comme son nom l'indique, est la partie qui va s'occuper d'effectuer les calculs, ces derniers se font soit d'une manière séquentielle au par bloc, il y a plusieurs types d'unité de calcul en fonction des opérations que doit effectuer le processeur.

2.1 Les processeurs :

Le **processeur** aussi appelé « micro-processeur » ou « CPU » (Central Processing Unit ou en français: Unité Central de Traitement), est le cœur de l'ordinateur. Il permet de manipuler des informations numériques c'est-à-dire des informations codées sous forme binaire et d'exécuter des instructions codées dans la mémoire. [11]

1. La loi de Moore ne mentionne en rien les performances des processeurs, elle s'énonce comme ceci "la densité des transistors sur une puce double tous les 18 mois".

Figure 01: Le processeur

2.2 Definition d'un cœur:

Un processeur standard possède un cœur (on dit qu'il est single-core). Un processeur avec un seul cœur ne peut traiter qu'une seule instruction à la fois (des instructions peuvent être traitées ensemble mais toujours une par une). Une instruction est une tâche qu'on demande au processeur d'exécuter : convertir une vidéo, compresser des fichiers volumineux, etc. [8]

2.3 Processeur multi-cœur :

Un processeur multi-cœur est composé de deux ou plusieurs cœurs indépendants, chacun étant capable de traiter des instructions individuellement. Un processeur dual-core contient deux cœurs, un processeur quad-core quatre cœurs, un processeur hexa-core six cœurs. [8]

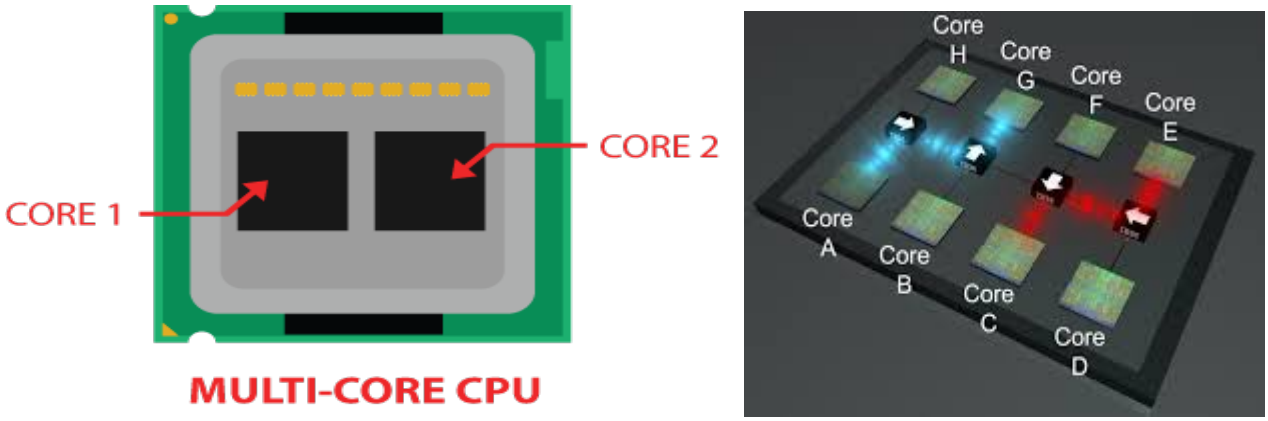

Figure 02 : **Processeur multi-cœur**

2.3.1 Le Dual Core :

Il ressemble à un processeur classique sauf qu'il est surmonté de deux parties centrales de core au lieu d'un seul, appelé également « die ». Pour utiliser un Dual Core sur une carte mère, il faut que le chipset[1] de celle-ci soit adapté à la gestion d'un Dual Core. [11]

2.3.2 Le Quad Core:

Le Quad Core dispose de quatre cœurs de traitement (core) sur une seule puce. Il ressemble à un Dual Core mais contient quatre core lui permettant de traiter de multiples instructions en même temps. Avec un Quad Core, les fabricants de puces peuvent générer de meilleures performances sans augmenter la vitesse dite d'horloge. Toutefois, le gain de performance sensé être réalisé ne peut l'être **que si le logiciel de l'ordinateur prend en charge le multitraitement**.

Ainsi, le logiciel répartit la charge de traitement d'informations entre plusieurs processeurs au lieu d'utiliser un seul processeur à la fois : les systèmes d'exploitation les plus modernes et de nombreux programmes fournissent désormais un soutien pour le multitraitement. [11]

L'exemple actuel avec les smartphones SamsunG Galaxy SIII et HTC One X prouvent l'efficacité d'une telle performance avec un Quad Core.

2.3.3 Présentation des processeurs Intel Core:

- **Core de 1ère génération « Nehalem » :**

Les puces Intel Core i3, i5 et i7 de première génération sont sorties de 2008 à 2010 et sont gravés en 45nm (nanomètres). Cela fait référence à la finesse de gravure et les différentes générations de processeurs n'ont pas forcément la même finesse de gravure (mesurée en micromètres (µm) ou nanomètres (nm). [11]

- **Core de 2nd génération « Sandy Bridge » :**

En 2011, Intel lance la seconde génération des processeurs Core gravés en 32nm .Leur nom de code est Sandy Bridge. Les processeurs Core de deuxième génération sont très différents de ceux de la génération précédente puisqu'ils intègrent le processeur et la carte graphique au sein d'une même puce, ces derniers étant habituellement séparés sur une carte mère. Cette intégration permet d'avoir des échanges beaucoup plus rapides entre le CPU et le contrôleur graphique. Les améliorations sont notamment visibles lors de la lecture de vidéos HD et dans les jeux vidéos. Globalement, les benchmarks ont montré que les processeurs de 2nd génération étaient 42% plus rapides que ceux de 1ère génération. [11]

- **Core de 3ème génération « Ivy Bridge » :**

Sortis en 2012, les processeurs INTL Core de troisième génération sont gravés en 22nm. Leur nom de code est Ivy Bridge. Combinés avec les nouveaux chipsets Intel Série **7**, cette association permet de bénéficier du support du PCI 3.0 (une version plus rapide du connecteur qui sert à brancher des cartes d'extension sur sa carte mère) et de l'USB 3.0, et permettre à la mémoire vive de monter à une vitesse de 1600 MHZ nativement. Processeurs Ivy Bridge fonctionnent avec beaucoup de cartes mères au socker 1155 de génération précédente (même si une mise à jour du BIOS est généralement nécessaire). De plus, les processeurs Sandy Bridge de génération précédente sont également utilisables sur la plupart des nouveaux chipsets Série 7, ce qui permet à beaucoup d'utilisateurs de mettre à jour leur configuration facilement, que ce soit au niveau de la carte mère ou du processeur. [11]

Figure 03: Les processeurs Intel Core

2.4 Les clusters :

Méthode permettant d'accélérer l'exécution d'un programme informatique en divisant celui-ci en multiples segments exécutés simultanément sur différentes machines. En autre terme c'est un regroupement d'ordinateurs permettant de dépasser les limitations d'une machines en termes de puissance de calcul.

La théorie est apparue dans les années 70 mais le premier cluster fonctionnel est apparu dans le milieu des années 80 par Digital Equipment Corporation, sous le nom de VAXCluster.

2.4.1 Objectifs:

- faciliter la montée en charge
- augmenter la disponibilité
- permettre une répartition de la charge
- diminuer les temps de calculs
- réaliser opérations non réalisables sur un serveur
- réduire les coûts : prix / performance d'une grappe de PC est de 3 à 10 fois inférieur à celui des supercalculateurs traditionnels

2.4.2 Architecture:

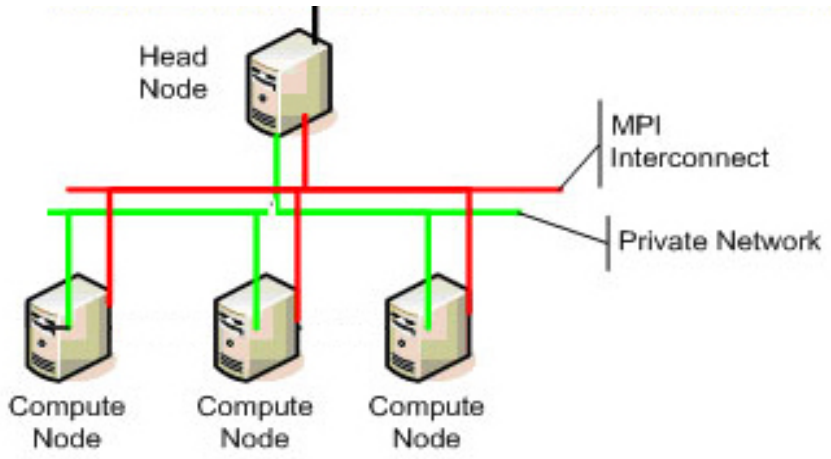

Figure 04: Architecture du cluster.

- Node : Poste client connecté au réseau et capable d'exécuter du code informatique sous l'ordre d'un serveur.
- Head Node : Serveur du cluster d'ordinateurs. Il est responsable de la répartition du travail entre les différents postes clients ainsi que de leur synchronisation. Il récupère également les erreurs et les résultats des calculs. On lui soumet les différents travaux qu'il gère à l'aide d'une file d'attente.

2.5 Comparisons entre multiproceseurs et les clusters :

Architecture multiprocesseurs	Architecture en clusters
➢ Coût important ➢ Maximum de 16 processeurs (systèmes Linux) ➢ Calculs rapides ➢ Accès facile à l'application et à la mémoire	➢ Utilisation de machines du réseau ➢ Coût de mise en œuvre relativement abordable ➢ Architecture client serveur difficile à mettre en place nécessite une bande passante importante.

Tableau 1: Comparaisons entre les clusters et les processeurs

3 La gestion parallèle :

L'exécution en parallèle **« Parallel Processing »** se rapporte à l'idée d'accélérer l'exécution d'un programme en le divisant en plusieurs fragments pouvant être exécutés simultanément, chacun sur son propre processeur. Un programme exécuté sur *N* processeurs pourrait alors fonctionner *N* fois plus vite qu'il ne le ferait en utilisant un seul processeur.

3.1 Définition d'un thread :

Un thread est un fil d'exécution. En fait, à tout programme est associé un processus (ou plusieurs) et tout processus comprend au moins un thread. Et c'est ce thread que suit un processeur lors de l'exécution d'un programme. Un thread ne peut être suivi que par un cœur à la fois. Autrement dit, un cœur ne peut exécuter qu'une seule tâche à la fois. Donc, si un processeur a deux cœurs, deux threads peuvent être exécutés simultanément et ainsi de suite.

3.1.1 Les avantages d'un thread :

Comparativement Aux processus Un à un flot de contrôle unique, un thread ou processus léger avec plusieurs flots De contrôle présente plusieurs avantages, notamment:

- **Réactivité** : Le processus léger continus 'exécuter même si certaines de ses parties sont bloquées.

- **Partage de ressources**.
- **économie d'espace mémoire et de temps**. Par Exemple sous Solaris, La créations d'un processus est 30 Fois plus lente que celle d'un processus et 30 fois plus lente que celle d'un processus-thread.

3.2 Définition d'un multi- threading:

Un processeur est dit **multithread** s'il est capable d'exécuter efficacement plusieurs threads simultanément. Contrairement aux systèmes multiprocesseurs (tels les systèmes multi-core), les threads doivent partager les ressources d'un unique cœur : les unités de traitement, le cache processeurs et la translation lookaside buffer ; certaines parties sont néanmoins dupliquées : chaque thread dispose de ses propres registres et de son propre pointeur d'instruction. Là où les systèmes multiprocesseurs incluent plusieurs unités de traitement complètes, le multithreading a pour but d'augmenter l'utilisation d'un seul cœur en tirant profit des propriétés des threads et des parallélismes au niveau des instructions. Comme les deux techniques sont complémentaires, elles sont parfois combinées dans des systèmes comprenant de multiples processeurs multithreads ou des processeurs avec de multiples cœurs multithreads.

3.2.1 Les avantages de l'hyper-threading sont les suivants :

- amélioration du support de code multi-thread.
- gestion de plusieurs threads en même temps.
- meilleur temps de réaction, meilleur temps de réponse.

Dans le cas d'un serveur informatique, augmentation du nombre d'utilisateurs possible

3.3 MPI (Message Passing Interface) :

La parallélisassions en MPI permet de diviser au maximum le temps de calcul par le nombre de cœurs de processeurs présents sur la machine. Ce but est légèrement revu à la baisse a cause des échanges de messages qui s'opèrent entre les cœurs. De plus, ces échanges peuvent être dans certains cas plus importants que le temps propre aux calculs. [1]

3.3.1 Définition de Message Passing Interface :

Une application MPI est un ensemble de processus autonomes exécutant chacun leur propre code et communiquant via des appels a des sous-programmes de la bibliothèque MPI. Elle est

devenue de factor un standard de communication pour des nœuds exécutant des programmes parallèles sur des systèmes à mémoire distribuée. [1]

Le principe de MPI (*Message Passing Interface*) s'appuie sur des processus qui possèdent une mémoire et une identité propres. L'échange de données se fait par envoi de messages entre ces processus.

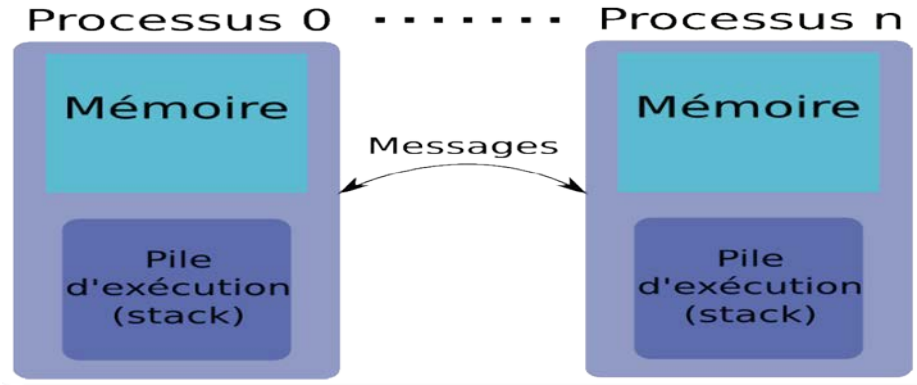

Figure 05: Message Passing Interface

3.3.2 Principales catégories de MPI :

A. Gestion de l'environnement :

- ✓ activation / désactivation de l'environnement (MPI INIT, MPI FINALIZE),
- ✓ identification des processus (MPI COMM SIZE, MPI COMM RANK).

B. Communications (envoie et réceptions de messages) :

- ✓ Point à point ;
- ✓ Collectives.

C. Gestion des groupes et communicateurs :

Communicateur : ensemble de processus sur lesquels portent les opérations MPI, groupe de processeurs et contexte de communication (géré par MPI).

Création d'un communicateur à partir :

- ✓ d'un groupe de processus préalablement crée par le programmeur,
- ✓ d'un autre communicateur (MPI COMM WORLD = l'ensemble des processus).

3.3.3 Fonctionnalité MPI :

Fonctions d'envoi et réception de messages standard : MPI_Send/ MPI_Recv

- utile pour les envoie de message de base

- fonction d’envoi bloquante (le retour de l’appel n’est fait que lorsque l’espace mémoire des données est disponible).

Fonction non bloquante

- initialisent l’envoi et la réception des données et retourne une valeur ’status’ immédiatement, on peut ensuite interroger ou attendre la réalisation des opérations
- l‘espace mémoire des données n’est pas accessible tant que les opérations ne sont pas finies.
- MPI_Probe or MPI_Wait() renvoie les infos sur l’etat de l’échange on bloquant. [1]

3.4 HPC (High performance computing):

Le calcul à haute performance (HPC – High Performance Computing) permet d'obtenir aujourd'hui des résultats qui ne seront atteignables sur de simples PCs que dans plus de dix ans.

3.4.1 Définition d’un HPC :

HPC (High Performance Computer) est un superordinateur conçu pour atteindre les plus hautes performances possibles avec les techniques connues lors de sa conception, en particulier en ce qui concerne la vitesse de calcul.[8]

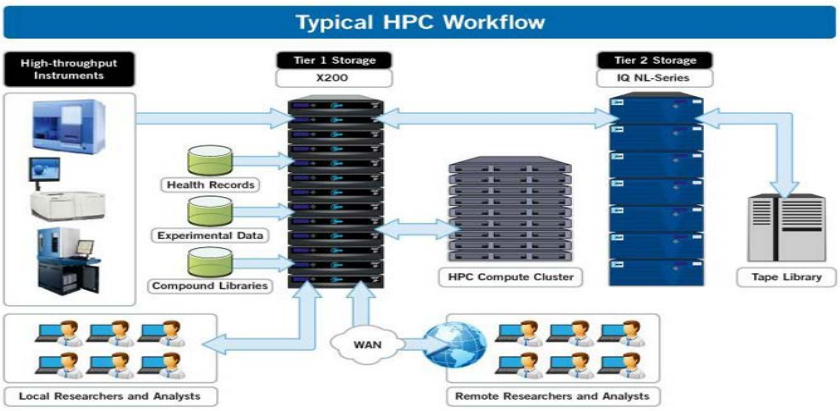

Figure 06: High Performance Computer.

3.4.2 Utilisation :

Les superordinateurs sont utilisés pour toutes les tâches qui nécessitent une très forte puissance de calcul, comme :

- Les prévisions météorologiques, l’étude du climat.
- La modélisation moléculaire.

- les simulations physiques (simulations aérodynamiques, calculs de résistance des matériaux, simulation d'explosion d'arme nucléaire, étude de la fusion nucléaire, etc.).
- La cryptanalyse.
- Les simulations en finance et en assurance (Calcul stochastique), etc. [8]

4 Les grille de calcule :

Le calcul sur une Grille un des éléments importants dans l'évolution des technologies de l'information, lié à la convergence des télécommunications, de l'informatique et de l'audiovisuel, est que l'ensemble de l'information dont nous disposons ou que nous générons *(*données, textes, images, vidéo, etc.*)* est maintenant numérique.

4.1 Définition d'une grille informatique :

Une grille informatique (en anglais, *grid*) est une infrastructure virtuelle constituée d'un ensemble de ressources informatiques potentiellement partagées, distribuées, hétérogènes, délocalisées et autonomes.

Une grille est en effet une infrastructure, c'est-à-dire des équipements techniques d'ordres matériel et logiciel. Cette infrastructure est qualifiée de virtuelle car les relations entre les entités qui la composent n'existent pas sur le plan matériel mais d'un point de vue logique. [2]

4.1.1 Propriété d'une grille informatique:

Ces ressource sont potentiellement qualifiées de:

Partagées : elles sont mises à la disposition des différents consommateurs de la grille et éventuellement pour différents usages applicatifs

Distribuées : elles sont situées dans des lieux géographiques différents

Hétérogènes : elles sont toutes de nature différente par exemple par le système d'exploitation ou le système de gestion des fichiers ;

Coordonnées : les ressources sont organisées, connectées et gérées en fonction de besoins (objectifs) et contraintes (environnements). Ces dispositions sont souvent assurées par un ou plusieurs agents, qu'ils soient centralisés ou répartis ;

Non-contrôlées (ou **autonomes**) : les ressources ne sont pas contrôlées par une unité commune.

Délocalisées : les ressources peuvent appartenir à plusieurs sites, organisations, réseaux et se situer à différents endroits géographiques. [11]

4.2 La Grille de calcul :

Une grille de calcul est un dispositif logiciel qui offre aux utilisateurs des puissances quasi illimitées de calcul ou de stockage de données, grâce à un accès transparent et facile (une simple connexion à un réseau à très haut débit de type Internet) à un vaste ensemble de ressources informatiques distribuées sur une grande échelle. [8]

4.3 L'ingénierie concourante :

L'**ingénierie concourante** ou **ingénierie simultanée** (*Concurrent Engineering* ou *CE* en anglais) est une méthode d'ingénierie qui consiste à engager simultanément tous les acteurs d'un projet, dès le début de celui-ci, dans la compréhension des objectifs recherchés et de l'ensemble des activités qui devront être réalisées. [2]

5 Cloud computing :

Aujourd'hui quasiment toutes les entreprises, de la petite à la grande multinationale, ont recours à l'informatique pour gérer leurs activités. Jusqu'à maintenant la solution qui prédominait était de disposer de ressources locales avec un grand nombre de paramètres à gérer, maintenance, mise à jour, sécurité, sauvegardes, interface logicielle etc.

Désormais un nouveau moyen est mis à disposition : le cloud computing, qui permet de délocaliser ces ressources afin de les confiés à des fournisseurs professionnels. [9]

5.1 Définition :

Le Cloud computing est un concept récent aux frontières encore un peu floues. Rien de plus normal pour un nuage. En fait le cloud computing correspond peu ou prou à la mise à disposition d'une capacité de calcul et de stockage évolutifs depuis des ressources accessibles par internet. [9]

5.2 L'historique:

Le cloud computing n'est pas nouveau, il est exploité depuis les années 2000, les changements qui ont permis l'apparition du cloud computing sont nombreux. Ainsi on peut citer l'apparition du SaaS (Software as a Service), le produit délivré par le cloud.

Puis il y à le concept de virtualisation qui permet une mutualisation des serveurs et offre donc une mise en production simplifiée et un meilleur ratio d'utilisation des ressources.

Le cloud computing est donc la juxtaposition de ces technologies pour passer à la vitesse supérieure sur l'exploitation de données à travers Internet.

Le concept du Cloud Computing a été mis en œuvre en 2002 par Amazon, un leader du e-business, pour absorber la charge importante des commandes faites sur leur site au moment des fêtes de Noël.

Récemment, d'autres acteurs comme Google et Microsoft proposent à leur tour des services similaires.

5.3 Les modèles de Cloud Computing :

Le but principal du Cloud est d'offrir des services à des utilisateurs suivant différents modèles. Nous présentons dans cette section les modèles de services principaux du Cloud.
La première typologie concerne la couche, comme le montre le schéma ci-dessous :

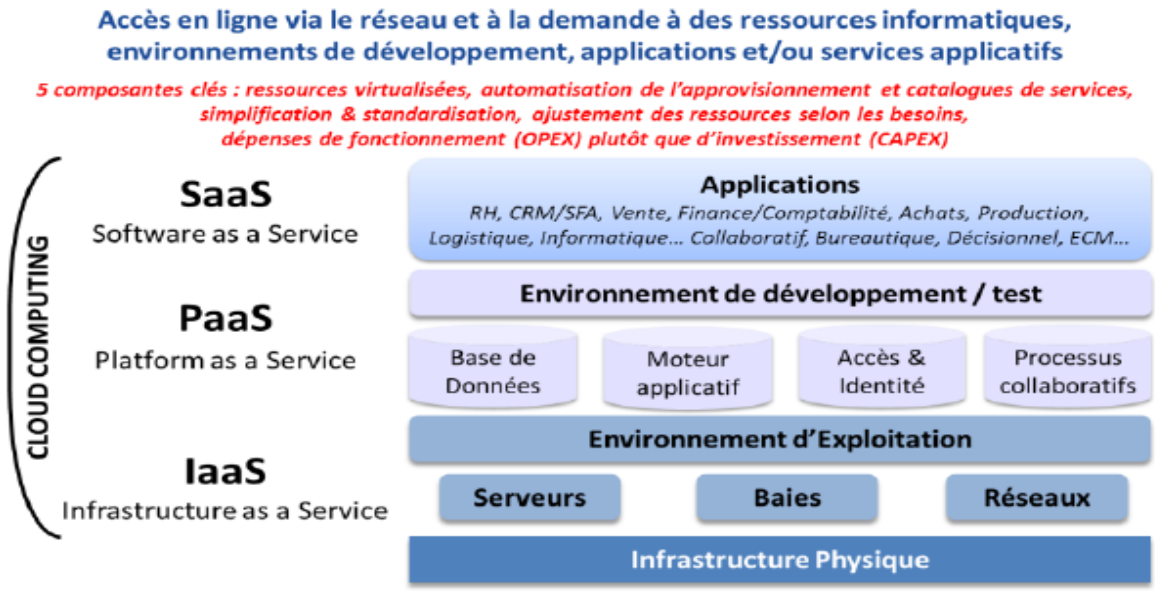

Figure 07 : Définitions et contours du Cloud Computing

On distingue 3 couches :

5.3.1 Les SaaS (saftwore as a Service):

Le Software as a Service (SaaS) est accessible à toutes les entreprises et il est facturé au nombre d'utilisateurs. [3]

L'entreprise loue les applications du fournisseur de services, plus besoin d'acheter un logiciel ces applications sont accessibles via différentes interfaces.

De nombreux logiciels sont disponibles dans le Cloud, en mode SaaS :

- Messagerie : L'e-mail est certainement l'application la plus utilisée en mode Cloud
- CRM : La gestion de la relation client est l'autre application phare du Cloud
- ERP : Certains progiciels de gestion intégrée sont proposés en mode Saas.
- Collaboratif : Les outils de collaboration (partage de documents, réseaux sociaux…

Figure08: Les SaaS (saftwore as a Service)

5.3.2 Les PaaS (Plateform as a Service) :

Facturée à la consommation, est un environnement qui permet à l'entreprise de déployer ses propres applications en dehors de sa salle informatique.

L'utilisateur gère, mais ne contrôle pas l'infrastructure Cloud (réseaux, serveurs, systèmes d'exploitation, stockage). Il a ainsi le contrôle sur les applications déployées et la possibilité de configurer l'environnement d'hébergement applicatif. [3]

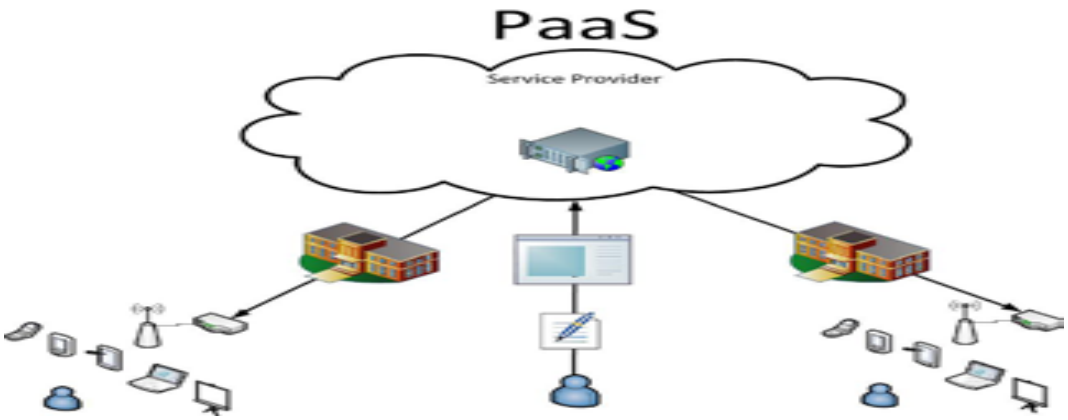

Figure 09: Les PaaS (Plateform as a Service)

5.3.3 Les IaaS (Infrastructure as a Service) :

L'Infrastructure as a Service (IaaS) c'est la mise à disposition par Internet de machines virtuelles aux ressources « facilement » modifiables (à la hausse ou baisse) et hautement disponible.

L'entreprise loue ainsi des capacités de traitement, de stockage et autres ressources qu'elle peut structurer et gérer de façon autonome côté logiciel dès le système d'exploitation Pour des calculs complexes ou ponctuels (simulations budgétaires, calculs prédictifs…) ou encore du stockage (notamment dans le cadre de sauvegardes), les administrateurs de ressources informatiques trouvent dans le Cloud des environnements aux capacités quasi illimitées et dont la mise en œuvre est quasi instantanée. [3]

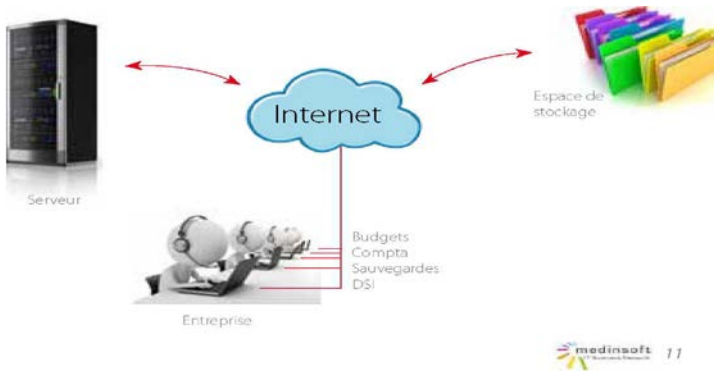

Figure 10: Les IaaS (Infrastructure as a Service)

5.3.4 Avantages et inconvénients des services :

	Avantages	Inconvénients
SaaS	• Pas d'installation • Plus de licence • Migration • Accessible via un abonnement	• Logiciel limité • Sécurité • Dépendance des prestataires
PaaS	• Pas d'infrastructure nécessaire • Pas d'installation • Environnement hétérogène	• Limitation des langages • Pas de personnalisation dans la configuration des machines virtuelles

IaaS	• Administration • Personnalisation • Flexibilité d'utilisation • Capacité de stockage infini	• Sécurité • Besoin d'un administrateur système • Demande pour les acteurs du Clouddes investissements très élevés

Tableau I.1 : Avantages et inconvénients des services

5.4 Les catégories de cloud computing :

On peut distinguer trois types principaux de modèles de déploiement pour le cloud computing:

5.4.1 Les Clouds privés :

Le cloud privé est une maniere d'émuler un cloud sur réseau interne en exploitant les fonctions de virtualisation des serveurs. [6]

5.4.2 Les Clouds publics:

Les utilisateurs ont accès à des services Cloud via l'Internet public sans savoir précisément où sont hébergées leurs données ni où sont exécutés leurs traitements. [6]

5.4.3 Les Cloud hybrides :

Ils associent à la fois des infrastructures et des Cloud privés et publics. Une partie des données ou des infrastructures est gérée en interne par l'entreprise, dans ses locaux ou chez un prestataire et communique avec des ressources Cloud. Il permet de différencier le lieu de traitement des données selon qu'elles soient stratégiques ou pas. [6]

5.4.4 Les Cloud communautaires :

Permettent à plusieurs entreprises ou organisations de partager des ressources en mode Cloud. Le Cloud communautaire peut être géré par les organisations membres ou par un prestataire externe. [10]

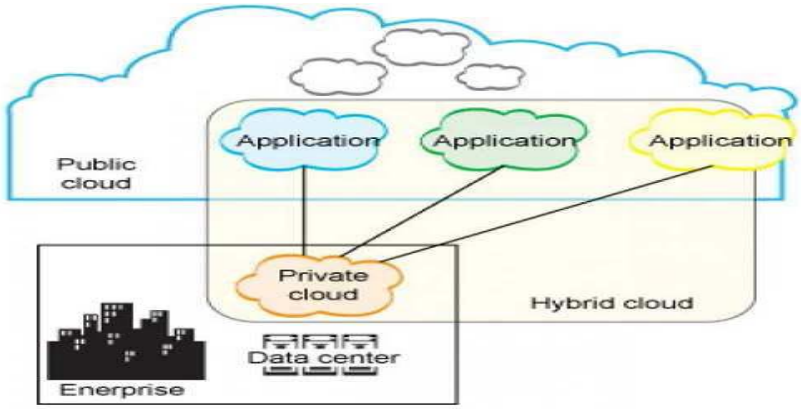

Figure11: Les différents modèles de déploiement d'un Cloud computing.

5.5 Caractéristique de Cloud:

Les caractéristiques essentielles d'un nuage sont :

- ✓ **Ressources en libre-service** : adaptation automatique à la demande. La capacité de stockage et la puissance de calcul sont adaptées automatiquement au besoin d'un consommateur.
- ✓ **Ouverture** : les services de cloud computing sont mis à disposition sur l'Internet, et utilisent des techniques standardisées qui permettent de s'en servir aussi bien avec un ordinateur qu'un téléphone ou une tablette.
- ✓ **Mutualisation** : elle permet de combiner des ressources hétérogènes (matériel, logiciel…) en vue de servir plusieurs consommateurs à qui les ressources sont automatiquement attribuées.
- ✓ **Paiement à l'usage** : la quantité de service consommée dans le cloud est mesurée, à des fins de contrôle, d'adaptation des moyens techniques et de facturation. [9]

5.6 Avantages et inconvénients du Cloud Computing :

A. Avantages :

✓ **Un démarrage rapide**

Le Cloud Computing permet de tester le business plan rapidement, à coûts réduits et avec facilité.

✓ **L'agilité pour l'entreprise**

Résolution des problèmes de gestion informatique simplement sans avoir à s'engager à long terme.

✓ **Un développement plus rapide des produits**

Réduction du temps de recherche pour les développeurs sur le paramétrage des applications.

- ✓ **Pas de dépenses de capital**

Plus besoin des locaux pour élargir les infrastructures informatiques. [10]

B. Inconvénients

- ✓ **La bande passante peut faire exploser le budget**

La bande passante qui serait nécessaire pour mettre cela dans le Cloudest gigantesque, et les coûts seraient tellement importants qu'il est plus avantageux d'acheter le stockage nous-mêmes plutôt que de payer quelqu'un d'autre pour s'en charger.

- ✓ **Les performances des applications peuvent être amoindries**

Un Cloud public n'améliorera définitivement pas les performances des applications.

- ✓ **La fiabilité du Cloud**

Un grand risque lorsqu'on met une application qui donne des avantages compétitifs ou qui contient des informations clients dans le Cloud.

- ✓ **Taille de l'entreprise**

Si l'entreprise est grande alors ses ressources sont grandes, ce qui inclut une grande consommation du Cloud. Du coup il serait peut-être plus intéressant à mettre au point son propre Cloud plutôt que d'en utiliser un externalisé. Les gains sont bien plus importants quand on passe d'une petite consommation de ressources à une consommation plus importante. [10]

6 Big Data:

Pour faire face à l'explosion du volume des données, on parle actuellement de pétaoctet (billiard d'octets) voir de zettaoctet (trilliard d'octets) et aussi face à la grande variété des données (image, texte, web, etc.) un nouveau domaine technologique a vu le jour : le Big Data inventé par les géants du web, au premier rang comme Yahoo, Google et Facebook, qui ont été les tous premiers à déployer ce type de technologie. [12]

6.1 Définition:

Le **big data**, littéralement les « grosses données », ou **méga données**, parfois appelées **données Massives,** désignent des ensembles de données qui deviennent tellement volumineux qu'ils

en deviennent difficiles à travailler avec des outils classiques de gestion de base de données ou de gestion de l'information. [12]

6.2 Historique:

Entre les années 60 et les années 80, vient l'éclosion de travaux théoriques sur les lois exponentielles de croissance de l'information scientifique, sur les recensements de flux d'information et sur les découvertes en ingénierie permettant la compression des données de stockage informatique

En octobre 1997, il semble que l'on enregistre **pour la première fois les termes Big Data**, dans un article de Michael Cox et David Ellsworth, deux chercheurs de la NASA, qui explique clairement que de telles masses de données nécessitent des méthodes d'analyse adaptées, et en particulier un recours à des techniques de visualisation spécifiques pour permettre de se les approprier.

En octobre 1999, une conférence d'IEEE s'est déroulée entièrement sur le thème de la visualisation des données.

En novembre 2000, Francis Diebold, professeur d'économie d'University *of Pennsylvania* a donné une conférence sur le recours aux Big Data par des modèles dynamiques de prévisions macroéconomiques.

Il faut attendre février 2001 pour voir apparaître la fameuse définition des Big Data par les trois V (pour Volume, Vélocité, et Variété),

On saute rapidement ensuite à 2007 où sont publiées par Gantz J, Reinsel D et coll des prévisions (jusqu'en 2010) des volumes générés par les Big Data, avec la notion que l'information produite double désormais tous les dix-huit mois. La même équipe publiera des mises-à-jour en 2010 et 2012 montrant que les prévisions ont été largement dépassées.

En septembre 2008, les Big Data acquièrent leurs lettres de noblesse, puisqu'elles font la Une d'un numéro spécial qui leur est dédié dans la revue britannique Nature . L'emballement qui suit est à la mesure des trois V des Big Data, considérés par certains analystes comme « *la plus grande innovation en informatique de la décennie* » dans les domaines du commerce, de la science et de la société, par Bryant R et coll., décembre 2008.

En février 2010, c'est le monde des affaires qui s'empare du sujet, avec la Une d'un numéro spécial de la revue (britannique) the economist, *titrant « the data déluge »* .

Un an plus tard, en février 2011, la recherche dans la télécommunication, publie à son tour, dans la revue américaine science, un article de Hilbert M et Lopez P retraçant l'inversion des modalités de stockage de données devenues en majorité digitales dès l'année 2002 (elles étaient analogiques jusque là). Puis le monde du conseil aux entreprises s'intéresse au sujet, avec la publication par Manyika J et coll. du *McKinsey Global Institute* d'un article portant sur les Big Data : nouvelle frontière de l'innovation, la compétition et la productivité.

En mai 2012, les sciences humaines et sociales (deux sociologues chercheurs chez Microsoft, à Cambridge, Massachussets, USA, extrait d'article seul disponible, en anglais) se penchent à leur tour sur l'objet. Danah Boyd et Kate Crawford définissent les Big Data comme une interaction culturelle, technologique et scientifique. [12]

6.3 **Les principes de base de Big Data :**

Le Big Data couvre quatre dimensions : volume, vélocité, variété et véracité.

Volume : les entreprises sont submergées de volumes de données croissants de tous types, qui se comptent en téraoctets, voire en pétaoctets.

- Transformer les 12 téraoctets de Tweets créés quotidiennement en analyse poussée des opinions sur un produit
- Convertir les 350 milliards de relevés annuels de compteurs afin de mieux prédire la consommation d'énergie

Vélocité : parfois, 2 minutes c'est trop. Pour les processus chronosensibles tels que la détection de fraudes, le Big Data doit être utilisé au fil de l'eau, à mesure que les données sont collectées par votre entreprise afin d'en tirer le maximum de valeur. [15]

- Scruter 5 millions d'événements commerciaux par jour afin d'identifier les fraudes potentielles
- Analyser en temps réel 500 millions d'enregistrements détaillés d'appels quotidiens.

Variété : le Big Data se présente sous la forme de données structurées ou non structurées (texte, données de capteurs, son, vidéo, données sur le parcours, fichiers journaux, etc.). De nouvelles connaissances sont issues de l'analyse collective de ces données.

- Utiliser les centaines de flux vidéo des caméras de surveillance pour contrôler les points d'intérêt
- Tirer parti de la croissance de 80 % du volume de données image, vidéo et documentaires pour améliorer la satisfaction client

Véracité : 1 décideur sur 3 ne fait pas confiance aux données sur lesquelles il se base pour prendre ses décisions. Comment pouvez-vous vous appuyer sur l'information si vous n'avez pas confiance en elle? Etablir la confiance dans les Big Data représente un défi d'autant plus important que la variété et le nombre de sources augmentent

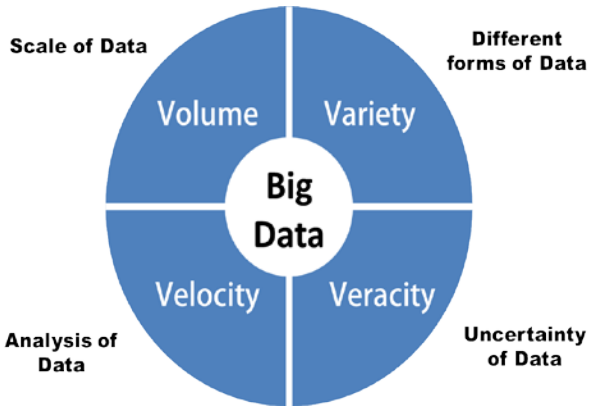

Figure 12: 4vs de Big data

6.4 Technologie du big data:

6.4.1 Map Reduce:

Map-reduce est un modèle de programmation massivement parallèle adapté au traitement de très grandes quantités de données. MapReduce est un produit Google Corp. Les programmes adoptant ce modèle sont automatiquement parallélisés et exécutés sur des clusters (grappes) d'ordinateurs. [12]

6.4.1.1 Principe de Map-Reduce:

Le système de traitement temps réel assure le partitionnement et le plan d'exécution des programmes tout en gérant les inhérentes pannes informatiques et indisponibilités. Ainsi, même les programmeurs inexpérimentés en programmation parallèle et distribuée sont à même d'utiliser ces

ressources.[18]

Une application typique MapReduce traite plusieurs téra-octets de données et exploite plusieurs milliers de machines.Map Reduce est ecrit en c++. L'index de Google est généré avec MapReduce.

6.4.1.2 Le fonctionnement du map reduce :

Un cluster MapReduce utilise une architecture de type maitre-esclave où un nœud maître dirige tous les nœuds esclaves.

MapReduce possède quelques caractéristiques :

- Le modèle de programmation du MapReduce est simple mais très expressif. Bien qu'il ne possède que deux fonctions, map() et reduce(), elles peuvent être utilisées pour de nombreux types de traitement des données, les fouilles de données, les graphes… Il est indépendant du système de stockage et peut manipuler de nombreux types de variable.
- Le système découpe automatiquement les données en entrée en bloc de données de même taille. Puis, il planifie l'exécution des tâches sur les nœuds disponibles.
- Il fournit une tolérance aux fautes à grain fin grâce à laquelle il peut redémarrer les nœuds ayant rencontré une erreur ou affecter la tâche à un autre nœud. [19]
- La parallélisation est invisible à l'utilisateur afin de lui permettre de se concentrer sur le traitement des données. [18]

Une fois qu'un nœud a terminé une tâche, on lui affecte un nouveau bloc de données. Grâce à cela, un nœud rapide fera beaucoup plus de calculs qu'un nœud plus lent. Le nombre de tâches Map ne dépend pas du nombre de nœuds, mais du nombre de blocs de données en entrée. Chaque bloc se fait assigner une seule tâche Map. De plus, toutes les tâches Map n'ont pas besoin d'être exécutées en même temps en parallèle, les tâches Reduce suivent la même logique.

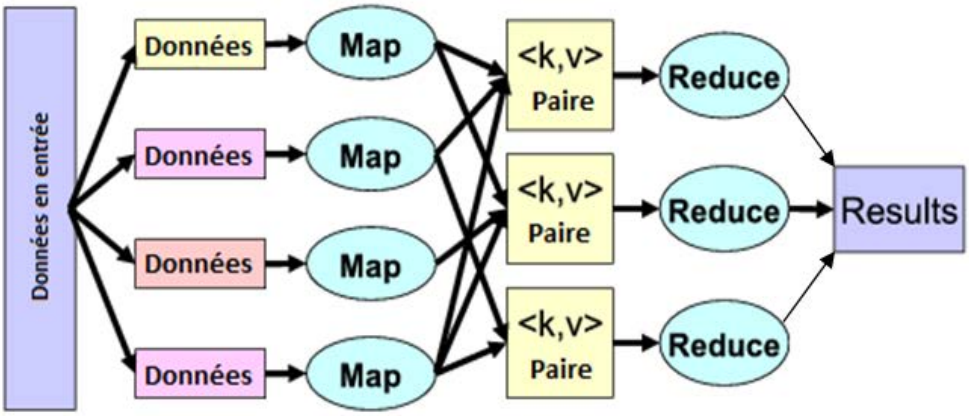

Figure 13: fonctionnement de map reduce

6.5 Hadoop:

Hadoop est un projet Open Source géré par Apache Software Fundation basé sur le principe Map Reduce et de Google File System, deux produits Google Corp. Le produit est écrit en langage Java.

Hadoop peut être considéré comme un système de traitement de données évolutif pour le stockage et le traitement par lot de très grande quantité de données. Il est tout à fait adapté aux stockages de grande taille et aux analyses de type "ad hoc" sur de très grandes quantités de données. Le besoin en analyse de grandes masses de données devient toujours plus pressant. Les analyses des données collectées sur le web, les traces laissées par les clients et prospects sont les applications les plus souvent citées. [19]

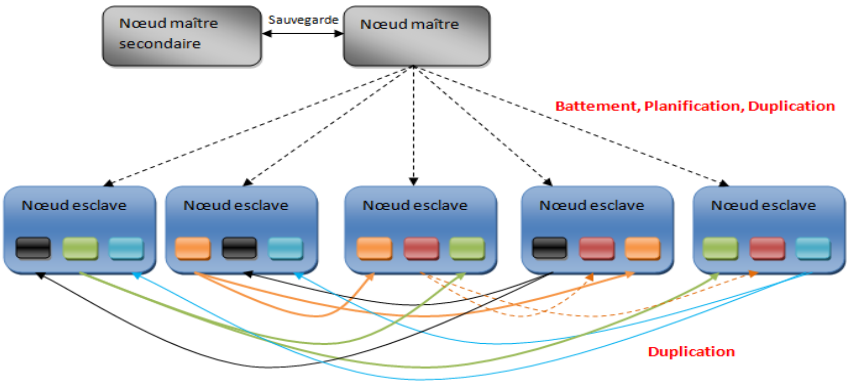

Figure14: Schéma du Hadoop Distributed File System

6.5.1 Hadoop et les analyses massives:

Le web n'est pas le seul à générer de grandes masses d'informations. Une gestion de suivi produit moderne, logistique ou traçabilité par exemple, exploitant l'identification généralisée des objets et des parcours de type RFID génère aussi des quantités incommensurables de précieuses données.

Les analyses massives autorisent alors des optimisations bien plus fines. Les suivis de trajet GPS que ce soient pour une maîtrise au plus juste des frais des itinérants ou pour un nouveau modèle économique d'assurance auto seront détaillés, recoupés et consolidés à la ligne près, avec ces nouveaux outils. [16]

6.6 Processus de chargement et de collecte de données dans Big Data :

La couche responsable du chargement de données dans Big Data, devrait être capable de gérer d'énorme volume de données, avec une haute vitesse, et une grande variété de données. Cette couche devrait avoir la capacité de valider, nettoyer, transformer, réduire (compression), et d'intégrer les données dans la grande pile de données en vue de son traitement. La Figure illustre le processus et les composants qui doivent être présent dans la couche de chargement de données. [17]

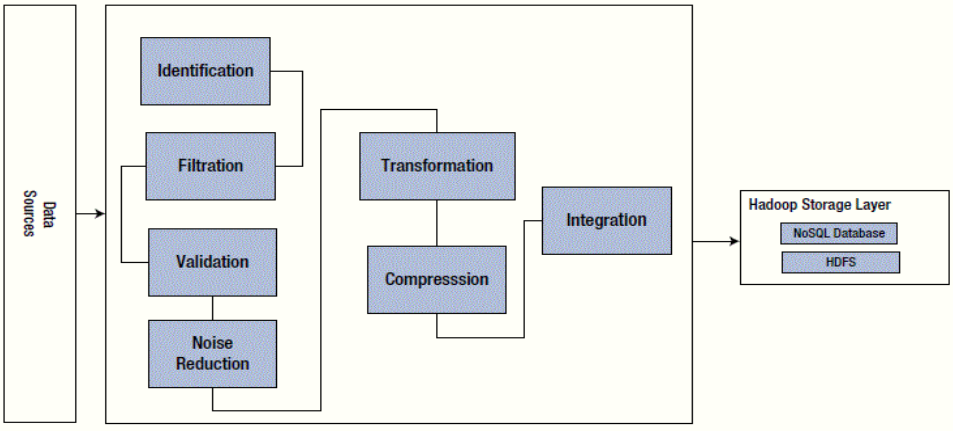

Figure14: Couche de chargement des données dans le Big Data.

La couche de chargement de données de Big Data collecte les informations pertinentes finales, sans bruit, et les charge dans la couche de stockage de Big Data (HDFS ou NoSQL base).Elle doit inclure les composants suivants :

- **Identification** des différents formats de données connues, par défaut Big Data cible les données non structurées.

- **Filtration et sélection** de l'information entrante pertinente pour l'entreprise.
- **Validation** et analyse des données en permanence.
- **Réduction** de bruit implique le nettoyage des données en supprimant le bruit.
- **La transformation** peut entrainer le découpage, la convergence, la normalisation ou la synthèse des données.
- **Compression** consiste à réduire la taille des données, mais sans perdre de la pertinence des données.
- **Intégration** consiste à intégrer l'ensemble des données dans le stockage de données de Big Data (HDFS ou NoSQL base).

6.7 Architecture Big Data:

On distingue principalement les couches suivantes :

- **Couche matériel** (infrastructure Layer) :peut-être des serveurs virtuels VMware, ou des serveurs lame blade ;
- **Couche stockage** (Storage layer) : les données seront stockées soit dans une base NoSQL, ou bien directement dans le système de fichier distribué ou les Datawarehouse ;
- **Couche management et traitement** : on trouve dans cette couche les outils de traitement et analyse des données comme MapReduce ou Pig ;
- **Couche visualisation** : pour la visualisation du résultat du traitement.[15]

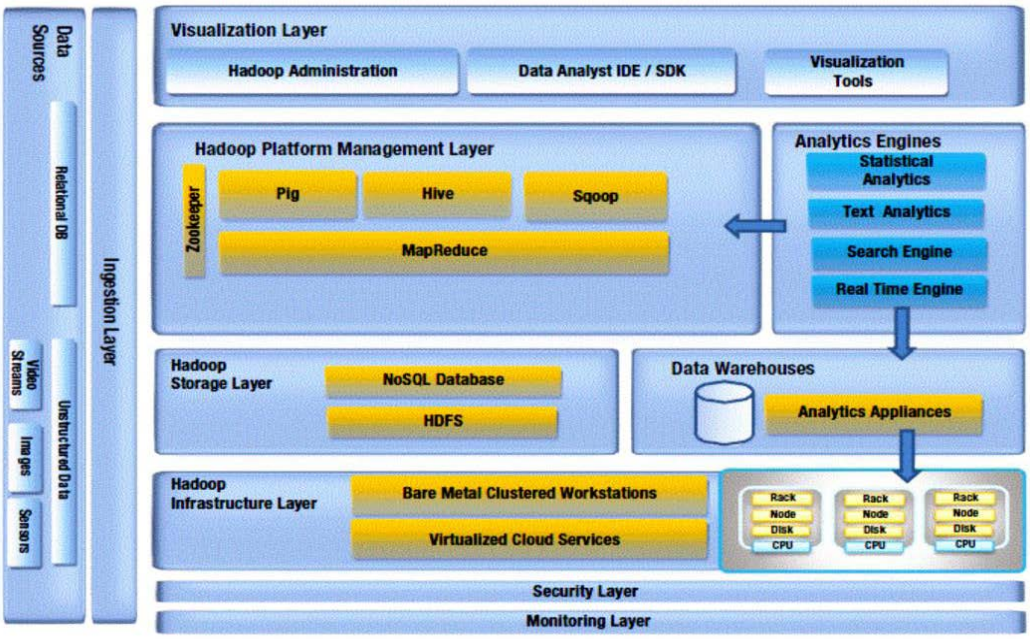

Figure15: Architecture de Big Data.

6.8 Avantages de l'architecture Big Data :

- **Evolutivité (scalabilité)** : Quelle est la taille que devra avoir votre infrastructure ? Combien d'espace disque est nécessaire aujourd'hui et à l'avenir ? le concept Big Data nous permet de s'affranchir de ces questions, car il apporte une architecture scalable.

- **Performance** : Grâce au traitement parallèle des données et à son système de fichiers distribué, le concept Big Data est hautement performant en diminuant la latence des requêtes.

- **Coût faible** : Le principal outil Big Data à savoir Hadoop est en Open Source, en plus on n'aura plus besoin de centraliser les données dans des baies de stockage souvent excessivement chère[12], avec le Big Data et grâce au système de fichiers distribués les disques internes des serveurs suffiront.

- **Disponibilité** : On a plus besoin des RAID disques, souvent coûteux. L'architecture Big Data apporte ses propres mécanismes de haute disponibilité.[12]

6.9 Big Data et Datawarehouse :

Les entrepôts de données sont traditionnellement des supports des données structurées et ont été étroitement lié aux systèmes opérationnels et transactionnels de l'entreprise (SGBDR). Ces systèmes qui sont soigneusement construits sont maintenant au milieu d'importants changements après l'émergence de Big Data. Datawarehouse est une base de données (données structurées)

regroupant une partie ou l'ensemble des données fonctionnelles d'une entreprise. Il entre dans le cadre de l'informatique décisionnelle ; son but est de fournir un ensemble de données servant de référence unique, utilisées pour la prise de décisions dans l'entreprise par les baies de statistiques et de rapports réalisés via des outils de reporting. [15] D'un point de vue technique, il sert surtout à 'délester' les bases de données opérationnelles des requêtes pouvant nuire à leurs performances.

Les organisations continueront inévitablement à utiliser des entrepôts de données pour gérer le type de données structurées et opérationnelles qui caractérise les systèmes relationnels(SGBDR). Ces entrepôts seront toujours fournis aux analystes avec la capacité pour analyser les données clés, les tendances, et ainsi de suite.

Cependant, avec l'avènement du Big Data, le défi pour les entrepôts de données est de réfléchir à une approche complémentaire avec le Big Data, on pourrait concevoir un modèle hybride. Dans ce modèle les restes de données optimisées opérationnelles très structurées seront stockées et analysées dans l'entrepôt de données, tandis que les données qui sont fortement distribuées et non structurées seront contrôlées par Big Data (Hadoop ou NoSQL).

La tendance serait de stocker la grande masse de données non structurées dans une vaste gamme de serveurs Big Data (Hadoop/MapReduce) pour tirer profit de la scalabilité et la rapidité d'analyse de Big Data, ensuite à l'aide d'outils, ces données seront déplacées dans le modèle relationnel de sorte qu'elles peuvent être interrogées avec le langage SQL traditionnel (SGBDR et Datawarehouse).

On peut donc interfacer Big Data avec le Datawarehouse(DW), effectivement les données non structurées provenant de différentes sources peuvent être regroupées dans un HDFS avant d'être transformées et chargées à l'aide d'outils spécifiques dans le Datawarehouse et les outils traditionnels de BI.

Comme les DW traditionnels ne gèrent pas les données non structurées, Big Data peut servir comme un moyen de stockage et d'analyse des données non structurées qui seront chargées dans les DW. [12]

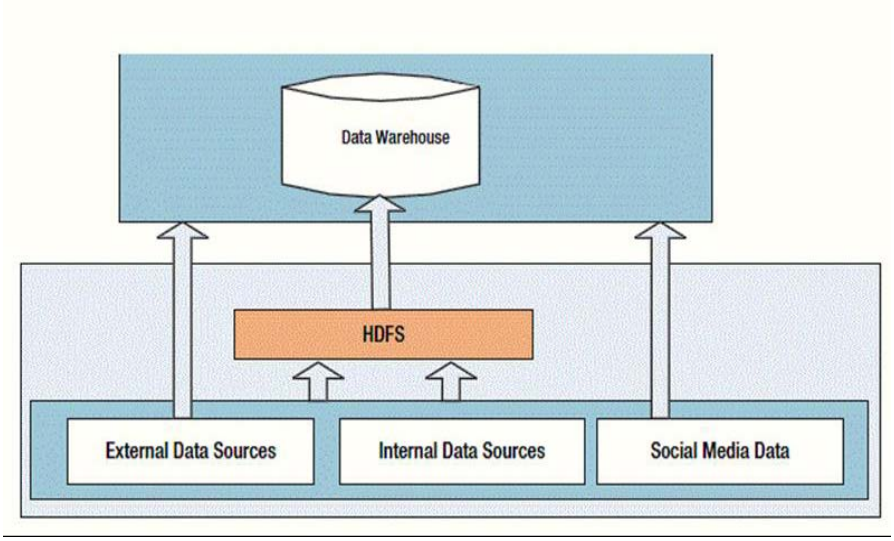

Figure 16:Lien entre Big Data et DW.

6.10 Calcul parallèle avec R (HPC, Hadoop, BigData):

L'explosion des quantités de données nécessite de réfléchir de plus en plus à des moyens efficaces de traitements de ces données. Les traitements demandent l'installation d'un cluster de calcul avec le logiciel R (et les technologies MPI, Hadoop).

Il n'est plus rare de devoir traiter des masses de données qui se mesurent en centaines de Gigaoctets (1To = 1000 Go = 1 millions de Mo = 10^12 octets). Par exemple, le nombre d'utilisateurs de Facebook atteint presque le milliard d'individus. Si on considère que chaque utilisateur a écrit une dizaine d'article de 1000 caractères, cela fait une masse de données de 10000 Go = 10 Terraoctets. Sur les marchés financiers, le nombre de transactions sur les bourses mondiales avoisinent les 10 milliards par jour. Vous pouvez multiplier par 100 pour avoir une idée de la quantité d'information si vous voulez analyser tout le carnet d'ordre (pour construire une stratégie de trading "High Frequency" par exemple). Dans le domaine biologique, l'ADN des 23 chromosomes du genre humain a une taille d'un peu moins d'1 Terraoctet. [15]

6.11 High Performance Computing et Big Data:

Il n'est pas possible de faire des calculs sur des masses de données de cette ampleur sur un simple PC. Pour analyser cette quantité d'information et trouver des stratégies hautes fréquence, il faut une architecture de calcul distribué adaptée. [12]

Au moins 3 solutions High Performance Computer (HPC) s'offrent à vous avec R.

- **RMPI:**

Elle est basée sur la technologie MPI (Message Passing Interface), qui est la solution standard de distribution de calculs.

- **RSERVE :**

Rserve est un serveur R qui peut s'installer sur de nombreuses machines, et ainsi former un cluster. Ce n'est pas directement une solution de calcul distribué. Vous devrez écrire le code pour distribuer les calculs vous-même.

- **RHADOOP :**

La technologie Hadoop est la plus récente et utilise la fonctionalité MapReduce pour distribuer les calculs. exemple: C'est probablement l'orientation future principale du HPC.

6.12 Quelques domaines d'utilisation du Big Data :

Avant de conclure, citons rapidement quelques domaines d'utilisation du Big Data. Le Big Data trouve sa place dans de nombreux domaines. [12]

Dans la première catégorie, on retrouve des secteurs qui manipulent quotidiennement des volumes de données très important, avec des problématiques de vitesse associées. On y trouve :

- Les **Banques** : la sanctuarisation de données anciennes dues à des contraintes réglementaires.
- La **Télécommunication** : l'analyse de l'état du réseau en temps réel ;
- Les **Médias Numériques** : le ciblage publicitaire et l'analyse de sites web ;
- Les **Marchés Financier** : l'analyse des transactions pour la gestion des risques et la gestion des fraudes, ainsi que pour l'analyse des clients.

La deuxième catégorie de secteur est plus hétérogène, les besoins, mais aussi l'utilisation qui est faite du Big Data, peuvent être très différents. On y trouve :

- **Les Services Publics** : l'analyse des compteurs (gaz, électricité, etc.) et la gestion des équipements ;
- **Le Marketing** : le ciblage publicitaire et l'analyse de tendance ;

- **La Santé:** l’analyse des dossiers médicaux et l’analyse génomique.

7 Conclusion:

Dans ce chapitre nous avons présenté les bases de l’unité de calcul de 1ère génération jusqu'à la 3ème génération, et l’introduction du parallélisme. Elles ont su se démocratiser grâce à leur large utilisation par les grandes entreprises du domaine informatique, qui ont adopté ces nouvelles solutions pour répondre à leurs besoins d'évolutivité.

Nous avons décrit les architectures, les Principes de fonctionnement, les caractéristiques techniques et les composants de cloud computing et les grilles de calcul, les Big data. Le chapitre a fait l’objet d’une présentation générale de l’environnement dans lequel notre étude comparative va être menée, à savoir les cloud compting qui vont être rapprochés ainsi que l’outil de comparaison des framworks qui va évaluer et comparer les performances des systèmes hadoop et spark.

Chapitre 2

Les frameworks de Big Data

1 Introduction:

Au cours de la dernière décennie, les systèmes répartis à large échelle ont connu une popularité importante en raison de leurs caractéristiques particulières, comme le passage à l'échelle et la tolérance aux fautes. Certains systèmes orientés données (BigData), comme les systèmes Pair-à-Pairs et MapReduce, ont atteint des millions d'utilisateurs et des pétaoctets de données traitées : Hadoop,Spark, Strom,HPCC etc.

Dans se chapitre nous allons présenter les Framework qui opèrent sur les cloud computing, leur objectifs, et leurs fonctionnalités, on finit forcément par parler sur les deux concurrents, Hadoop et Apache Spark. Si les deux outils sont parfois considérés comme des concurrents, il est souvent admis qu'ils fonctionnent encore mieux quand ils sont ensemble. On va faire un aperçu de leurs caractéristiques et de leurs différences.

2 Framework

Un framework est un ensemble d'outils et de compostant logiciels organisés conformément à un plan d'architecture et des patterns, l'ensemble formant ou promouvant un «squelette» de programme. Il est souvent fourni sous la forme d'une bibliothèque logicielle, et accompagné du plan de l'architecture cible du framework.

Le framework va permettre de mettre en place une application en effectuant des séries d'opérations de manière simplifiée en utilisant des méthodes qui auront été conçues au préalable au sein des librairies du framework

Nous pouvons trouver des frameworks gratuits ou payants, open-source ou non. Les communautés qui les entourent sont généralement très présentes et réactives. Ce qui permet de fournir des frameworks gratuits d'excellente qualité.

2.1 Type des Frameworks:

On trouve différents types de Framework:

- **Framework d'infrastructure système**:pour développer des systémes d'exploitation, des interfaces graphiques des outils de communication (exemple: Framework,Net, Struts).

- **Framework d'intégration intergicielle (*middleware):*** logiciel qui crée un réseau d'échange d'informations pour fédérer des applications hétérogènes. Pour mettre à disposition différentes technologies sous la forme d'une interface unique.
- **Framework d'entreprise**: pour développer des applications spécifiques au secteur d'activité de l'entreprise.
- **Framework de gestion de contenu:** sont les fondations d'un systéme de gestion d econtenu pour la création, la collecte, le classement, le stockage et la publication de «biens numérisés».[13]

3 FrameWork pour les cloud computing

3.1 HPCC System:

À la suite de l'explosion de l'information, de nombreuses organisations ont maintenant besoin de traiter et d'analyser des volumes massifs de données. Ces exigences en matière de calcul intensif peuvent être traitées par des systèmes évolutifs basés sur des clusters matériels de serveurs de produits couplés avec le logiciel de système pour fournir un système de stockage de fichiers distribué, demandes parallèles, et des outils de développement de programmation parallèles. [13]

3.1.1 Définition:

HPCC (Cluster de calcul haute performance) est une plate-forme cluster de calcul utilisé pour résoudre les problèmes Big Data. Son architecture unique et simple mais puissant langage de programmation de données (ECL) en fait une solution convaincante pour résoudre les besoins de calcul intensif. [8]

3.1.2 Evolution de HPCC:

Pour gérer, trier, lien, et d'analyser des milliards d'enregistrements en quelques secondes, Risk Solutions LexisNexis conçu un supercalculateur à forte intensité de données qui a été prouvé pour les 10 dernières années avec les clients qui ont besoin de traiter billons d'enregistrements en quelques secondes. Des clients tels que les grandes banques, compagnies d'assurance, les services publics, l'application de la loi et le gouvernement fédéral dépendent Risk Solutions LexisNexis. LexisNexis a offert cette plate-forme comme une solution open source sous Systèmes HPCC. LexisNexis Risk Solutions est une unité d'affaires de 1,5 milliard $ de LexisNexis, qui est une des solutions d'information entreprise de 6 milliards $. LexisNexis est détenue par Reed Elsevier, qui a eu des revenus de 9 milliards $ en 2013. [8]

3.1.3 Types de cluster HPCC:

THOR: Thor (the refinery cluster data) est responsable de consommer de grandes quantités de données, la transformation, la liaison et l'indexation de ces données, fonctionne comme un système de fichiers distribué avec la puissance de traitement parallèle répartis sur plusieurs noeuds.Un cluster peut évoluer à partir d'un seul nœud à des milliers de nœuds. [14]

ROXIE: Roxie (Rapid en ligne XML Inquiry Engine) est le moteur de la transmission des données utilisées dans les systèmes HPCC pour servir les données rapidement et peut prendre en charge plusieurs milliers de demandes par nœud par seconde.

ECL Langue*:* ECL(Enterprise Control Language) est décrite comme un langage centré sur les données déclaratives fortement optimisé, est un langage spécifiquement conçu pour permettre aux données opérations à préciser d'une manière qui est facile d'optimiser et de paralléliser.Il est également conçu pour permettre plusieurs programmeurs pour travailler sur un nombre toujours croissant de données et de maximiser l'effet de levier obtenu.

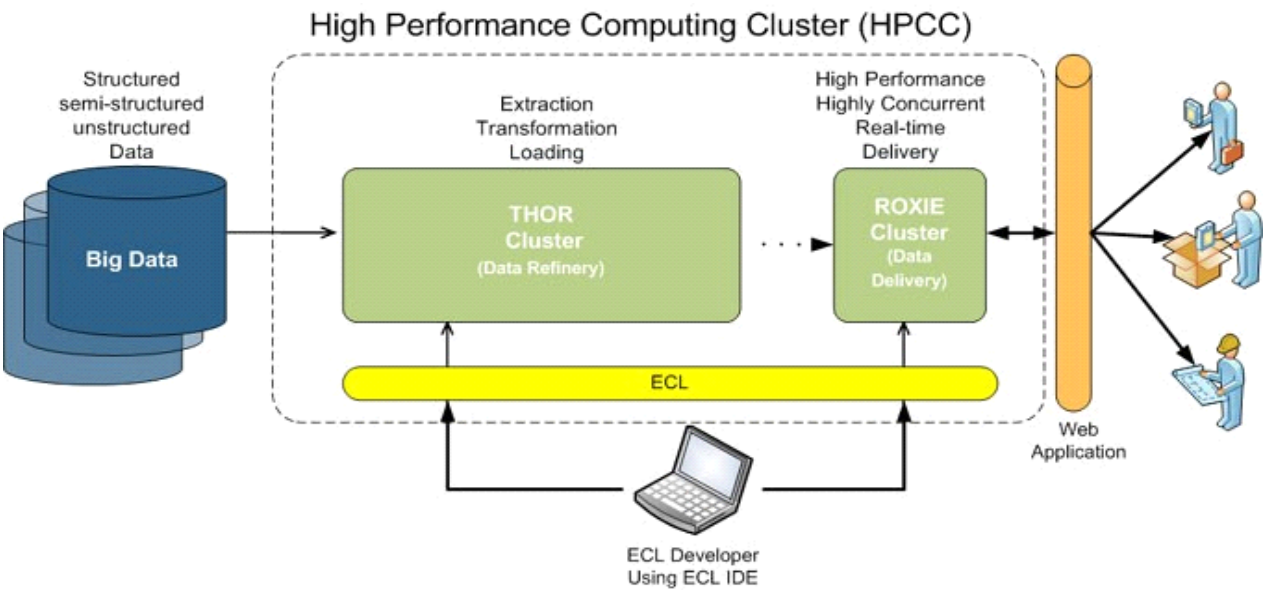

Figure 01: Plate-forme HPCC et architecture.

3.1.4 Les principaux avantages d'ECL:

• ECL fournir la fonctionnalité de données parallèles le traitement et réduit la complexité calcul grande taille de cluster de calcul.

• Avec ECL, nous pouvons mettre en œuvre grande quantité de données car elle est principalement conçue pour la manipulation de grande quantité de données et de recherche.

• ECL ont une amélioration de la productivité de 20 fois par rapport à d'autres langages de programmation tels que Java ou C ++.

• ECL est un haut niveau, la programmation parallèle langue qui est assez puissant pour faire, d'extraction d'information, de mémorisation d'informations, la recherche d'information, et la liaison des différents Enregistrements. [14]

• ECL est un langage très mature, mais il est encore fait progrès en soi tous les jours.

3.1.5 Système HPCC Serveurs Middleware

En HPCC nous avons un certain nombre de serveurs de système inclut dans la configuration du système, ces serveurs fournir une passerelle à partir de son groupe de deux données à le monde extérieur. Ces serveurs comprennent l'ESP serveur, serveur ECL, serveur Dali, serveur DFU et ils tous sont connus comme composants middleware parcequ'ils résident dans la couche de middleware du système HPCC configuration.

3.2 Apache Storm:

Apache Storm est un système utilisé pour récupérer les flux de données en temps réel à partir de sources multiples d'une manière distribuée, tolérante aux pannes et à haute disponibilité. Strom est principalement conçu pour travailler avec des données qui doivent être analysées en temps réel, telles que les données des capteurs émis avec une haute fréquence ou des données provenant de réseaux sociaux où il est parfois important de savoir ce qui est partagé en ce moment.

Il est principalement constitué de deux parties. La première est appelée **SPOUT** et est responsable de la collecte de l'entrée de flux de données. La seconde est appelée **BOLT** et est responsable du traitement ou de la transformation des données. [14]

3.2.1 Architecture du Storm:

Il est principalement constitué de deux parties :

La première est appelée **SPOUT** et est responsable de la collecte de l'entrée de flux de données.

La seconde est appelée **BOLT** et est responsable du traitement ou de la transformation des données.

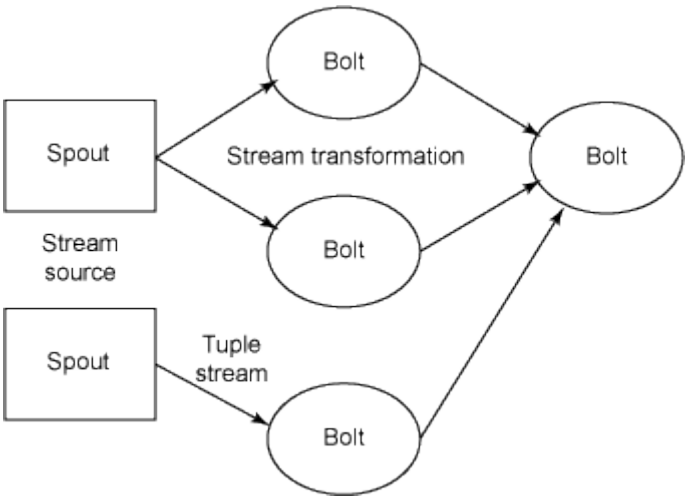

Figure02: Architecture du Storm

3.2.2 Composants d'un cluster de Storm:

Un groupe de Storm est très semblable à un cluster Hadoop. Le MapReduce équivalent dans Hadoop Job est l'idée de la topologie Strom. La principale différence est que, bien que la fin d'un emploi MapReduce lorsque la tâche est terminée, une topologie attend les données éternellement entrée, sans tuer le processus est clair.

L'architecture de Strom est assez simple, il est divisé en les composants suivants:

- Le **nœud maître** exécute le démon appelé Nimbus responsable de la distribution du code à travers le cluster (similaire à JobTracker Hadoop).effectue également les tâches d'allocation et de suivi dans les différentes machines du cluster.

- Le **travailleur nœuds** exécutant le démon superviseur en charge de la collecte et les affectations de traitement sur la machine où il fonctionne. Ces nœuds exécutent une partie de la topologie de sorte que le travail peut être distribué dans tout le cluster. Si vous avez raté un démon de noeud travailleur Nimbus remarquerait et rediriger le travail à un autre travailleur de noeud.

- **Zookeeper:** Bien que pas Strom composante en tant que telle, si elle est nécessaire pour monter un Zookeeper Apache car il sera responsable de la coordination entre les Nimbus et les superviseurs.Il est également responsable du maintien de l'Etat comme Nimbus et les superviseurs sont apatrides. [13]

3.2.3 Modes de fonctionnement:

Strom peut fonctionner en deux modes: Local et cluster. Le mode local est très utile pour tester le code développé dans la topologie de Strom parce qu'il fonctionne dans une seule JVM afin que nous puissions rendre notre système de test intégré, le code de débogage, etc.et pour ajuster les paramètres de configuration. Dans ce mode simule threads Storm différents nœuds du cluster.

Le cluster de Strom comme son nom l'indique, le fait d'exécuter la topologie du cluster, qui est distribué et exécuter notre code sur des machines différentes. Il est considéré comme «mode de production». [8]

3.2.4 Les caractéristique de storm

Cinq caractéristiques rendent Strom idéale pour les charges de travail en temps réel de traitement des données. Storm est:

- ✓ **Rapide** : traitement d'un million 100 messages d'octets par seconde par nœud
- ✓ **Scalable :** avec des calculs parallèles qui courent à travers un cluster de machines
- ✓ **Tolérant** aux **pannes** - lorsque les travailleurs meurent, Strom sera automatiquement redémarrer. Si un nœud meurt, le travailleur sera redémarré sur un autre noeud.
- ✓ **Fiable** : Strom garantit que chaque unité de données (tuple) sera traité au moins une fois ou une seule fois. Les messages ne sont rejoués quand il y a des échecs.
- ✓ **Facile à utiliser** : configurations standard sont adaptées à la production dès le premier jour. Une fois déployé, Storm est facile à utiliser. [14]

4 Présentation et comparaison entre les frameWorks Hadoop et Spark

4.1 HADOOP*:*

Le Big data regroupe plusieurs nouvelles technologies et d'outils pour répondre à une triple problématique : un **V**olume de données important à traiter, une grande **V**ariété d'informations (structurées ou non structurées), et un certain niveau de **V**élocité à atteindre. Pour répondre à ces besoins, il s'avère que l'écosystème Hadoop serait la solution open source par excellence.

4.1.1 Historique:

2003/2004: publication par Google de deux *whitepapers*, le premier sur GFS (un système de fichier distribué) et le second sur le paradigme Map/Reduce pour le calcul distribué.

2004: développement de la première version du *framework* qui deviendra Hadoop par Doug Cutting (archive.org).

2006: Doug Cutting (désormais chez Yahoo) développe une première version exploitable d'Apache Hadoop pour l'amélioration de l'indexation du moteur de recherche. Encore primitif (au maximum quelques machines, etc.).

2008: développement maintenant très abouti, Hadoop utilisé chez Yahoo dans plusieurs départements.

2011: Hadoop désormais utilisé par de nombreuses autres entreprises et des universités, et le *cluster* Yahoo comporte 42000 machines et des centaines de peta-octets d'espace de stockage.

Le nom lui-même n'est pas un acronyme: il s'agit du nom d'un éléphant en peluche du fils de l'auteur originel (Doug Cuttin) de Hadoop. [15]

4.1.2 Définition:

Hadoop est un Framework Java open source d'Apache pour réaliser des traitements sur des volumes de données massifs, de l'ordre de plusieurs pétaoctets (soit plusieurs milliers de To). [8]

4.1.3 Caractéristique d'Hadoop:

Robuste : si un nœud de calcul tombe, ses tâches sont automatiquement réparties sur d'autres noeuds. Les blocs de données sont également répliqués;

Coût : il optimise les coûts via une meilleure utilisation des ressources présentées;

Souple : car il répond à la caractéristique de variété des données en étant capable de traiter différents types de données;

Virtualisation : ne plus se reposer directement sur l'infrastructure physique (baie de stockage coûteuse), mais choisir la virtualisation de ses clusters Hadoop. [12]

4.1.4 Écosystème d'Hadoop:

Pour certains, l'avenir appartiendra à ceux qui seront capable d'analyser les vastes volumes de données qu'ils ont collectés.

En écologie, un écosystème est l'ensemble formé par une association ou communauté d'êtres vivants et son environnement biologique, géologique, édaphique, hydrologique, climatique, etc. l'écosystème Hadoop est l'ensemble des projets Apache ou non, liés à Hadoop et qui sont appelés à se cohabiter.

Hadoop désigne un framework Java libre ou un environnement d'exécution distribuée, performant et scalable, dont la vocation est de traiter des volumes des données considérables. Il est le socle d'un vaste écosystème constitué d'autres projets spécialisés dans un domaine particulier parmi lesquels on compte les entrepôts de données, le suivi applicatif (monitorring) ou la persistance de données.

Le schéma ci-après présente les différents éléments de l'écosystème Hadoop en fonction du type d'opération. [17]

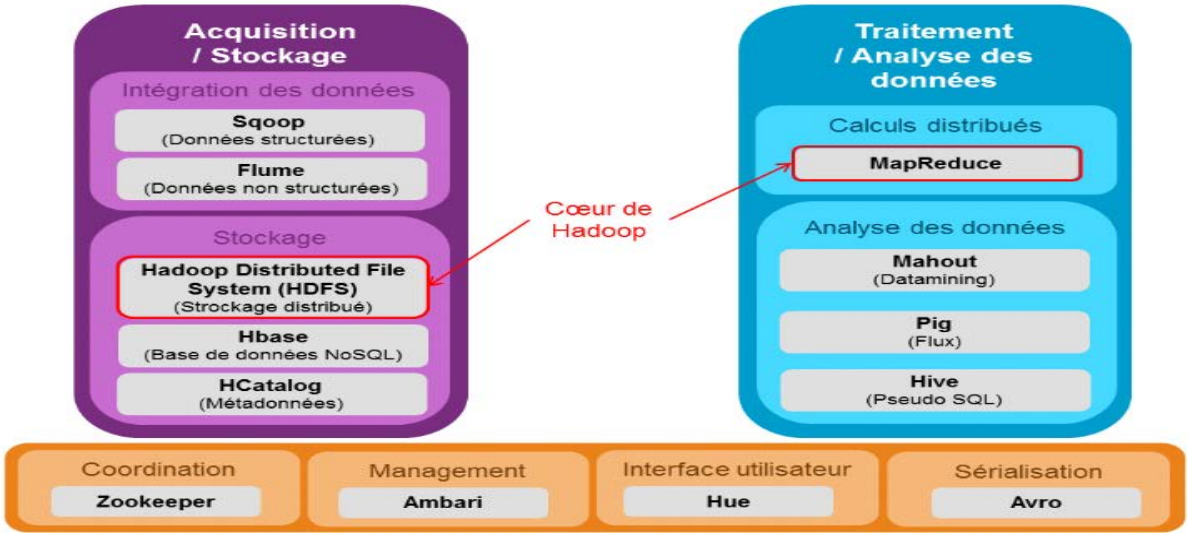

Figure03: Ecosysteme d'hadoop

4.1.5 Les composants d'écosystème d'hadoop:

✓ **Hbase**: Hbase est un système de gestion de bases de données non relationnelles distribuées, écrit en Java, disposant d'un stockage structuré pour les grandes tables. C'est une base

de données NoSQL, orientée colonnes. Utilisé conjointement avec HDFS, ce dernier facilite la distribution des données de Hbase sur plusieurs noeuds. Contrairement à HDFS, HBase permet de gérer les accès aléatoires read/write pour des applications de type temps réel. [17]

✓ **HaCatalog**: HCatalog permet l'interopérabilité d'un cluster de données Hadoop avec d'autres systèmes (Hive, Pig, ...). C'est un service de gestion de tables et de schéma Hadoop. Il permet :

- D'attaquer les données HDFS via des schémas de type tables de données en lecture/écriture.
- D'opérer sur des données issues de MapReduce, Pig ou Hive.

✓ **Hive**: Hive est un outil de requêtage des données, il permet l'exécution de requêtes SQL sur le cluster Hadoop en vue d'analyser et d'agréger les données. Le langage utilisé par Hive est nommé HiveQL. C'est un langage de visualisation uniquement, raison pour laquelle seules les instructions de type « Select » sont supportées pour la manipulation des données.

Hive proposent des fonctions prédéfinies (calcul de la somme, du maximum, de la moyenne), il permet également à l'utilisateur de définir ses propres fonctions qui peuvent être de 3 types :

UDF (User Defined Function) : qui prennent une ligne en entrée et retournent une ligne en sortie. Exemple : mettre une chaîne de caractère en minuscule et inversement

UDAF (User Defined Aggregate Function) : qui prennent plusieurs lignes en entrée et retournent une ligne en sortie. Exemple : somme, moyenne, max....

Hive utilise un connecteur jdbc/odbc, ce qui permet de le connecter à des outils de création de rapport comme QlikView. [17]

✓ **Pig**: Pig est une brique qui permet le requêtage des données Hadoop à partir d'un langage de script (langage qui interprète le code ligne par ligne au lieu de faire une compilation). Pig est basé sur un langage de haut niveau appelé PigLatin. Il transforme étape par étape des flux de données en exécutant des programmes MapReduce successivement ou en utilisant des méthodes prédéfinies du type calcul de la moyenne, de la valeur minimale, ou en permettant à l'utilisateur de définir ses propres méthode appelées User Defined Functions (UDF). [17]

- ✓ **Sqoop:** Sqoop est une brique pour l'intégration des données. Il permet le transfert des données entre un cluster et une base de données relationnelles.

- ✓ **Flume**: Flume permet la collecte et l'agrégation des fichiers logs, destinés à être stockés et traités par Hadoop. Il s'interface directement avec HDFS au moyen d'une API native.

- ✓ **Oozie**: Oozie est utilisé pour gérer et coordonner les tâches de traitement de données à destination de Hadoop. Il supporte des jobs Mapreduce, Pig, Hive, Sqoop, etc.

- ✓ **Zookeeper**: Zookeeper est une solution de gestion de cluster Hadoop. Il permet de coordonner les tâches des services d'un cluster Hadoop. Il fournit au composants Hadoop les fonctionnalités de distribution.

- ✓ **Ambari**: est une solution de supervision et d'administration de clusters Hadoop. Il propose un tableau de bord qui permet de visualiser rapidement l'état d'un cluster. Ambari inclut un système de gestion de configuration permettant de déployer des services d'Hadoop ou de son écosystème sur des clusters de machines. Il ne se limite pas à Hadoop mais permet de gérer également tous les outils de l'écosystème.

- ✓ **Mahout:** est un projet de la fondation Apache visant à créer des implémentations d'algorithmes d'apprentissage automatique et de datamining.

- ✓ **Avro:** est un format utilisé pour la sérialisation des données.Le caractère open source de Hadoop a permis à des entreprises de développer leur propre distribution en ajoutant des spécificités. [17]

4.1.6 Les concepts fondamentaux d' Hadoop:

Il est primordial de savoir qu'une architecture Hadoop est basée sur le principe maître/esclave, pour sela nous allons présenter deux concepts fondamentaux d'Hadoop : représentant les deux principaux rôles des machines. Les sous rôles relatifs au système de fichiers et à l'exécution des tâches distribuées sont associés à chaque machine de l'architecture. [21]

4.1.7 HDFS (Hadoop Distributed File System):

HDFS un système de fichier distribué, extensible et portable inspiré par le Google File System (GFS).

Il a été conçu pour stocker de très gros volumes de données sur un grand nombre de machine équipées de disques durs banalisés, il permet de l'abstraction de l'architecture physique de stockage, afin de manipuler un système de fichier distribué comme s'il s'agissait d'un disque dur unique. [19]

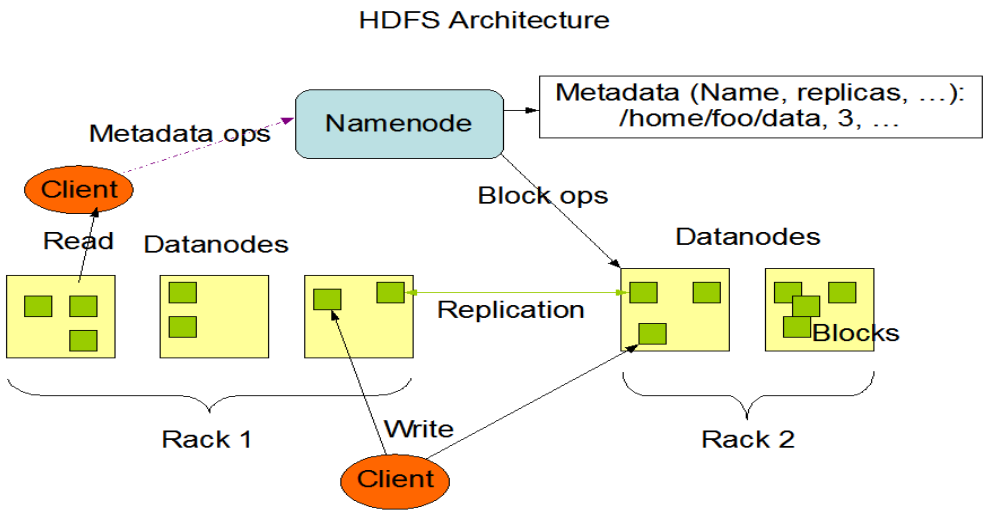

Figure 04: L'architecture d'HDFS

HDFS définit deux types de nœuds :

- **Le nœud principal ou NameNode**

Il se caractérise par:

- Responsable de la distribution et de la réplication des blocs.
- Serveur d'informations du HDFS pour le client HDFS.
- Stocke et gère les métadonnées.
- Comporte la liste des blocs pour chaque fichier (dans le cas de lecture).
- Contient la liste des DataNodes pour chaque bloc (dans le cas de l'écriture).
- Tenir les attributs des fichiers (ex : nom, date de création, facteur de réplication).
- Logs toute métadonnée et toute transaction sur un support persistant.
- Lectures/écritures.

- Créations/suppressions.

- **Le noeud de données ou DataNode**

Il se caractérise par:

- Stocke des blocs de données dans le système de fichier local
- Maintenir des métadonnées sur les blocs possédés (ex : CRC)
- Serveur de bloc de données et de métadonnées pour le client HDFS
- Heartbeat avec le Namenode : Heartbeat est système permettant sous Linux la mise en clusters de plusieurs serveurs pour effectuer entre eux un processus de tolérance de panne.

Le processus Heartbeat se chargera de passer un message-aller vers le NameNode indiquant : son identité, sa capacité totale, son espace utilisé, son espace restant. [19]

Remarque:

- **Secondary NameNode:** le NameNode dans l'architecture Hadoop est un point unique de défaillance, si ce service est arrêté, il n'y a pas moyen de pouvoir extraire les blocs d'un fichier donné, Pour résoudre ce problème, un NameNode secondaire appelé Secondary NameNode a été mis en place dans l'architecture Hadoop. Son rôle consiste à :
- Télécharger régulièrement les logs sur le NameNode.
- Crée une nouvelle image en fusionnant les logs avec l'image HDFS. [23]
- Renvoie la nouvelle image au NameNode.

4.1.7.1 Lecture d'un fichier HDFS

Pour lire un fichier au sein de HDFS, il faut suivre les étapes suivantes :

Etape 1 : Le client indique au NameNode qu'il souhaite lire le fichier data.txt

Etape 2 : Le NameNode lui indiquera la taille de fichier (nombre de blocs) ainsi que les différents Data Node hébergeant les n blocs.

Etape 3 : Le client récupère chacun des blocs à un des DataNodes.

Etape 4 : En cas d'erreur/non réponse d'un des DataNode, il passe au suivant dans la liste fournie par le NameNode. [17]

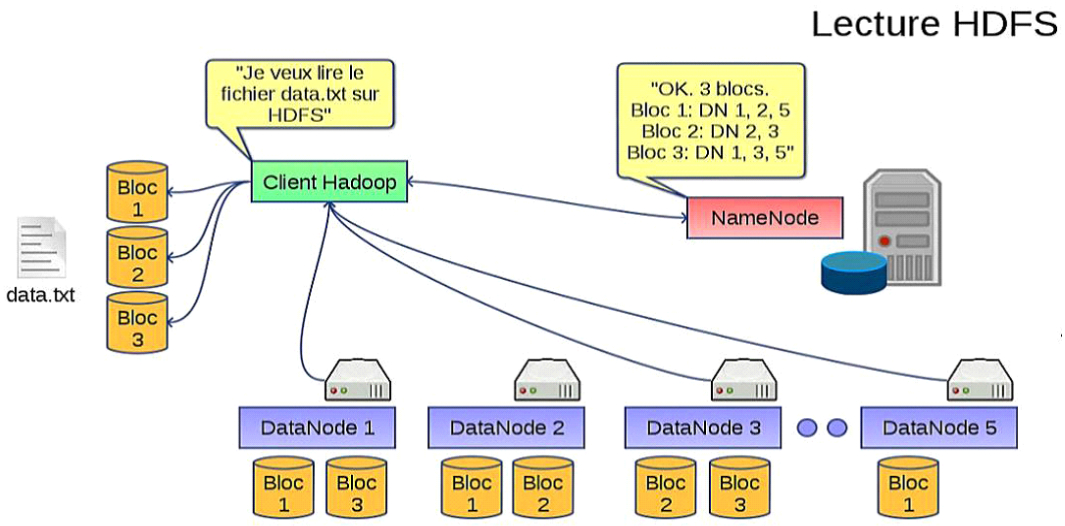

Figure 05: Lecture d'un fichier HDFS

4.1.7.2 Écriture dans un fichier ou volume HDFS :

Pour écrire un fichier au sein d'HDFS:

Étape 1 : On va utiliser la commande principale de gestion de Hadoop: Hadoop, avec l'option fs. Admettons qu'on souhaite stocker le fichier data.txt sur HDFS.

Étape 2 : Le programme va diviser le fichier en blocs de 64KB (ou autre, selon la configuration) – supposons qu'on ait ici 3 blocs.

Étape 3 : Le NameNode lui indique les DataNodes à contacter.

Étape 4 : Le client contacte directement le DataNode concerné et lui demande de stocker le bloc.

Etape 5 : les DataNodes s'occuperont – en informant le NameNode – de répliquer les données entre eux pour éviter toute perte de données.

Etape 6 : Le cycle se répète pour le bloc suivant. [17]

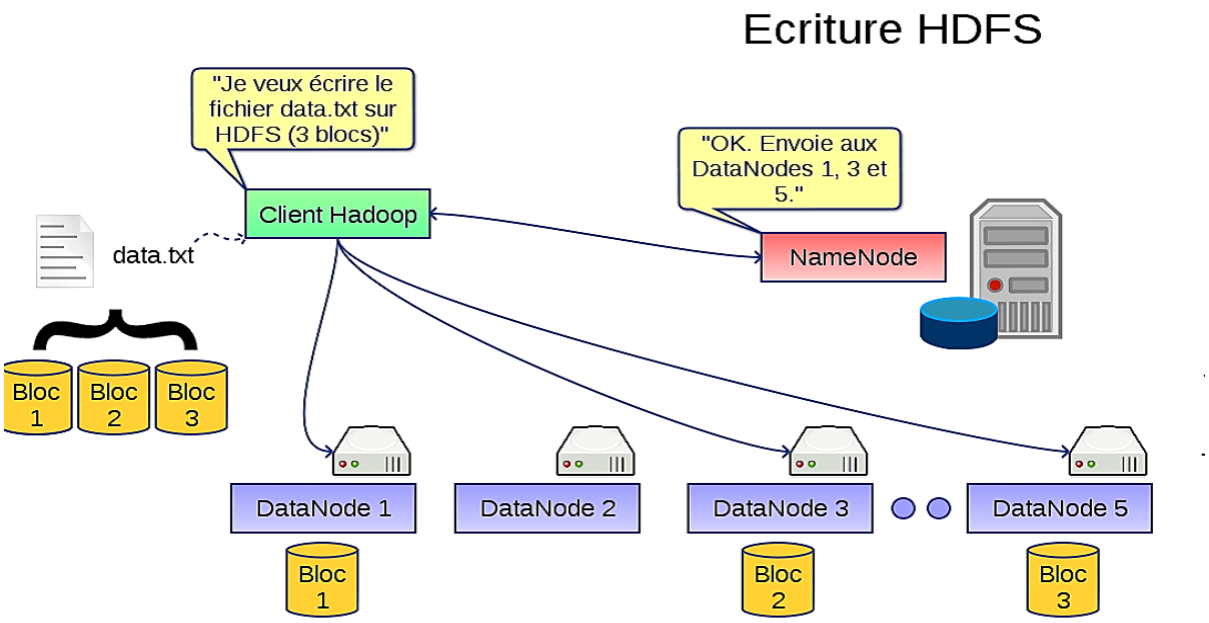

Figure 06: Processus d'écriture dans un volume ou fichier HDFS.

4.1.8 MapReduce:

MapReduce est un paradigme (modèle) de programmation parallèle proposé par Google. Il est principalement utilisé pour **le traitement** distribué sur de gros volumes de données aux seins d'un cluster de nœuds. Il est conçu pour la sociabilité et la tolérance aux pannes.

Le modèle de programmation fournit un cadre à un développeur afin d'écrire une fonction Map et une fonction Reduce. Tout l'intérêt de ce modèle de programmation est de simplifier la vie du développeur. Ainsi, ce développeur n'a pas à se soucier du travail de parallélisation et de distribution du travail. MapReduce permet au développeur de ne s'intéresser qu'à la partie **algorithmique.** [23]

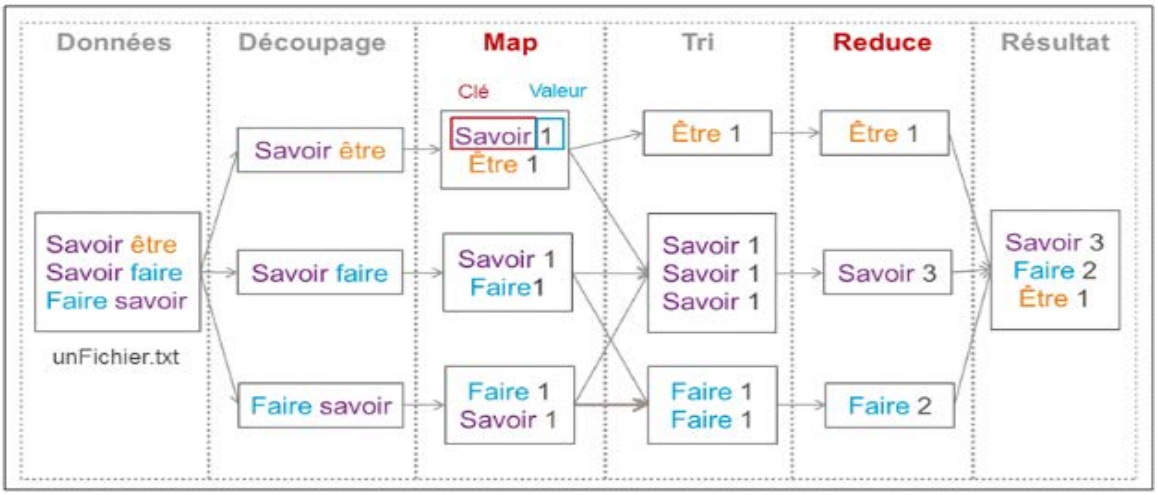

Figure07: le traitement de MapReduce.

Un programme MapReduce peut se résumer à deux fonctions Map () et Reduce ()

✓ La première **MAP** va transformer les données d'entrée en une série de couples clef /valeur. Elle va regrouper les données en les associant à des clefs, choisies de telle sorte que les couples clef/valeur aient un sens par rapport au problème à résoudre. Par ailleurs, cette opération doit être parallélisable: on doit pouvoir découper les données d'entrée en plusieurs fragments, et faire exécuter l'opération MAP à chaque machine du cluster sur un fragment distinct. La fonction Map s'écrit de la manière suivante : **Map (clé1, valeur1) → List (clé2, valeur2**). [23]

✓ La seconde **REDUCE** va appliquer un traitement à toutes les valeurs de chacune des clefs distinctes produite par l'opération MAP. Au terme de l'opération REDUCE, on aura un résultat pour chacune des clefs distinctes. Ici, on attribuera à chacune des machines du cluster une des clefs uniques produites par MAP, en lui donnant la liste des valeurs associées à la clef. Chacune des machines effectuera alors l'opération REDUCE pour cette clef. La fonction Reduce s'écrit de la manière suivante : **Reduce (clé2, List (valeur2)) → List (valeur2).**

Exemple :

L'exemple classique est celui du **WordCount** qui permet de compter le nombre d'occurrences d'un mot dans un fichier. En entrée l'algorithme reçoit un fichier texte qui contient les mots suivants **voiture la le elle de elle la se la maison voiture**

Dans notre exemple, la clé d'entrée correspond au numéro de ligne dans le fichier et tous les mots sont comptabilisés à l'exception du mot « se ». [17]

le résultat de la fonction map est donne ci-dessous :

(voiture, 1) /(la,1) /(le,1) / (elle,1) / (de,1) / (elle,1) / (la,1) / (la,1) / (maison,1) / (voiture,1)

(de, 1) / (elle, 2) / (la, 3) / (le, 1) / (maison, 1) / (voiture, 2)

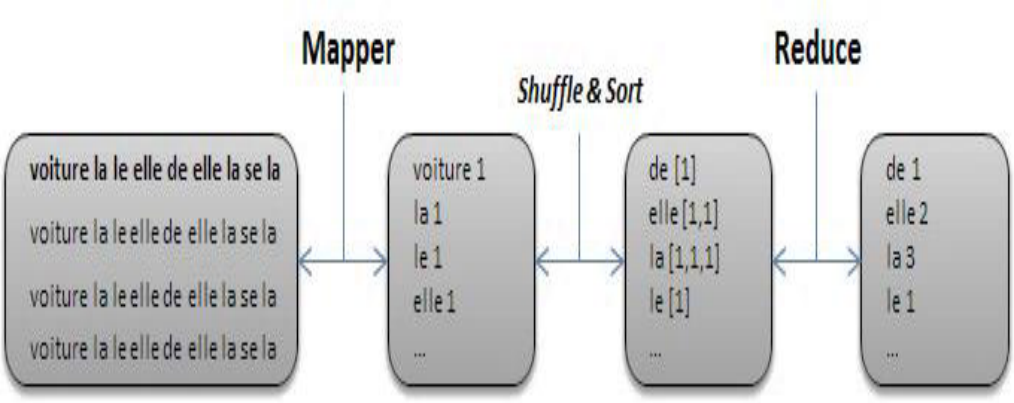

Figure 08: Exemple d'un programme MapReduce (WordCount)

4.1.8.1 Architecture de mapreduce

Le MapReduce possède une architecture maître-esclave

Le maître MapReduce : le JobTracker

Les esclaves MapReduce : les TaskTracker

- ✓ **Le JobTracker**

- Gère l'ensemble des ressources du système.
- Reçoit les jobs des clients.
- Ordonnance les différentes tâches des jobs soumis.
- Assigne les tâches aux TaskTrackers.
- Réaffecte les tâches défaillantes.
- Maintient des informations sur l'état d'avancement des jobs.

✓ **Un TaskTracker**

- Exécute les tâches données par le Jobtracker ;
- Exécution des tâches dans une autre JVM (Child) ;
- A une capacité en termes de nombres de tâches qu'il peut exécuter ;
- Heartbeat avec le JobTracker. [23]

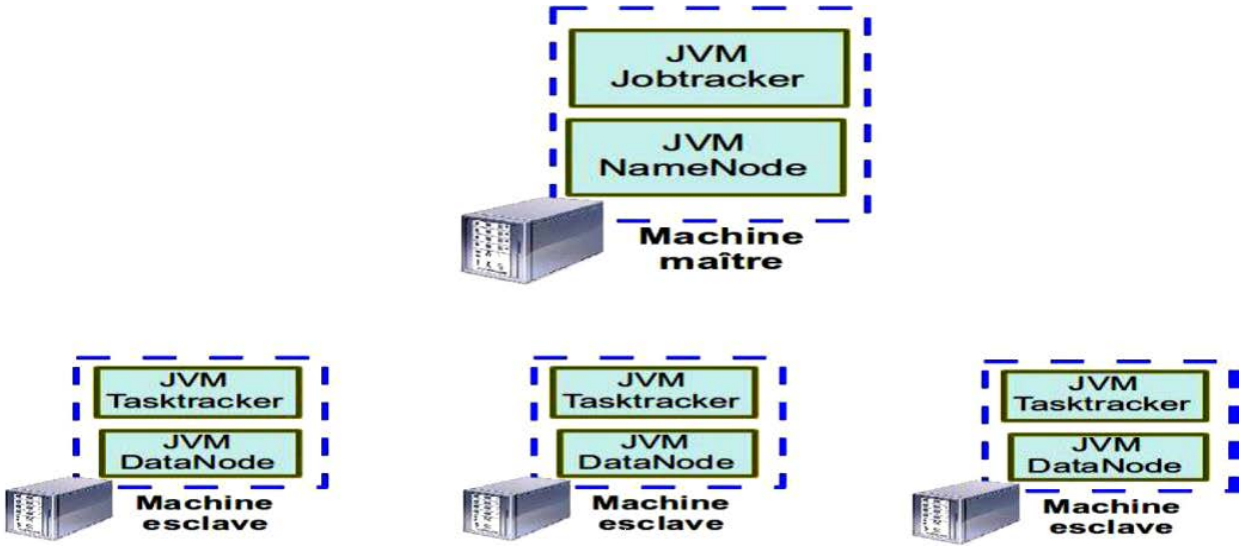

Figure 09: Architecture maître-esclave du MapReduce.

4.1.9 MapReduce dans hadoop:

Nous intéressons maintenant à voir comment le modèle de programmation MapReduce est exploité dans le framework Hadoop. La première étape concerne le découpage (split) des données. Cette étape est à la charge du framework qui se base sur le format des données. La taille de chaque découpage est fixe et généralement identique à la taille d'un bloc HDFS (64 Mo par défaut).

En s'appuyant sur l'exemple du compteur de mots fréquemment utilisés, le découpage consistera à réaliser des morceaux de 64 Mo qui contiendront chacun une liste de clé/valeur où la clé sera l'offset d'une ligne du fichier et la valeur le contenu de cet offset (voiture la le elle de elle la se la maison voiture). [18]

4.1.9.1 Étape Map

Hadoop crée pour chaque découpage de données une tâche de Map qui exécutera la fonction map développée en conséquence. Chaque découpage de données n'est traité que par une seule tâche de Map.

Précisons que ce n'est pas la donnée qui est transportée vers le programme, mais l'inverse. Cela permet de pouvoir profiter d'une optimisation appelée **optimisation de proximité de la donnée** (Data Locality Optimization en anglais). Hadoop va donc tenter de trouver le nœud le plus proche contenant la donnée pour y transférer la fonction Map.

Une fois la tâche Map terminée, le résultat de la fonction n'est pas stocké sur le système HDFS, mais sur le système de fichier local du nœud. En effet, il s'agit d'un résultat intermédiaire et qui n'a pas besoin d'être stocké de manière sécurisée via la réplication des blocs HDFS. Dans le cas d'une anomalie sur l'exécution de la tâche (nœud qui devient inaccessible par exemple), Hadoop est informé et pourra demander sa rééxécution. [19]

4.1.9.2 Étape Reduce

Hadoop ne lance les tâches de Reduce qu'une fois que toutes les tâches de Map sont terminées. Notez que le nombre de tâches de Reduce n'est pas fonction de la taille des données en entrée mais est spécifié en paramètre de configuration d'exécution du job. C'est donc un paramètre qui peut être modifié. Par défaut, le nombre de tâches de Reduce est 1.

L'entrée de la fonction reduce correspond à la sortie de l'ensemble des fonctions map. Comme la tâche de Reduce ne profite pas d'optimisation de proximité comme cela est fait pour la tâche de Map, les données de sortie de la fonction map sont transférées via le réseau vers le nœud où la tâche de Reduce est réalisée.

Chaque tâche de Reduce produit un fichier de sortie qui sera stocké, cette fois, dans le système de fichiers HDFS. Le format du nom du fichier produit est de la forme suivante : part-r-XXXX où XXXX est le numéro de la tâche de Reduce commençant par 0. Dans notre exemple, il y aura donc un seul fichier, car une seule tâche de Reduce, qui portera le nom part-r-0000. [19]

4.2 SPARK:

Industries utilisent Hadoop intensivement pour analyser leurs ensembles de données. La raison en est que le cadre Hadoop est basé sur un modèle de programmation simple (MapReduce) et permet une solution informatique qui est évolutive, flexible, tolérant aux pannes et rentable. Ici, la principale préoccupation est de maintenir la vitesse dans le traitement de grands ensembles de données en termes de temps d'attente entre les requêtes et de temps pour exécuter le programme d'attente. Spark à sa place dans le grand espace de données en raison de sa vitesse et sa facilité d'utilisation.

Dans Cette partie visant à faire découvrir Spark, son modèle de programmation, ainsi que son écosystème.

4.2.1 Historique:

Spark est né en 2009 dans le laboratoire AMPLAB de l'université de Berkeley en partant du principe que

- d'une part, la RAM coûte de moins en moins cher et les serveurs en ont donc de plus en plus à disposition
- d'autre part, beaucoup de jeux de données dits "Big Data" ont une taille de l'ordre de 10 Go et ils tiennent donc en RAM.

Le projet a intégré l'incubateur Apache en juin 2013 et est devenu un "Top-Level Project « en février 2014 ».

La version 1.0.0 de Spark a été réalisée en mai 2014 et le projet, aujourd'hui en version 1.1.0, poursuit une évolution rapide. L'écosystème Spark comporte ainsi aujourd'hui plusieurs outils:

- Spark pour les traitements "en batch"
- Spark Streaming pour le traitement en continu de flux de données
- MLlib pour le "machine learning"
- GraphX pour les calculs de graphes (encore en version alpha)
- Spark SQL, une implémentation SQL-like d'interrogation de données.

Par ailleurs, il s'intègre parfaitement avec l'écosystème Hadoop (notamment HDFS) et des intégrations avec Cassandra et ElasticSearch sont prévues.

Enfin, le framework est écrit en Scala et propose un binding Java qui permet de l'utiliser sans problème en Java. Java 8 est toutefois recommandé pour exploiter les expressions lambdas qui permettront d'écrire un code lisible.

4.2.2 Définition:

Apache spark est un framework de traitements Big Data open source construit pour effectuer des analyses sophistiquées et conçu pour la rapidité et la facilité d'utilisation. Celui-ci a originellement été développé par AMPLab, de l'Université UC Berkeley, et maintenant un projet de la fond ation apache. Contrairement a Hadoop qui utilise le patron d'architecture MapReduce sur des disques, Spark travaille en mémoire vive ce qui est potentiellement cent fois plus rapide. [8]

4.2.3 Les caractéristiques de Spark:

Permet de passer par quelques-unes des caractéristiques de Spark qui sont vraiment mettre en évidence dans le monde Big Data:

Vitesse : Spark permet aux applications en grappes Hadoop pour exécuter jusqu'à 100x plus rapide dans la mémoire, et 10 fois plus rapide, même lors de l'exécution sur le disque. Spark permet en réduisant nombre de lecture / écriture sur le disque. Il stocke ces données de traitement intermédiaire en mémoire. Il utilise le concept d'un Dataset Resilient (Distributed RDD), ce qui lui permet de stocker des données de manière transparente sur la mémoire et persistent sur disque seulement est nécessaire. Cela contribue à réduire la majeure partie du disque lire et écrire - le principal temps facteurs consommation - du traitement des données.

Facilité d'utilisation: Spark permet d'écrire rapidement des applications en Java, Scala ou Python. Cela permet aux développeurs de créer et exécuter leurs applications sur leurs langages de programmation familiers et faciles à construire des applications parallèles. Il est livré avec un ensemble de plus de 80 hauts niveaux intégrés operators

Combine sql, streaming, et des analyses complexes : En plus de simples "carte" et "réduire" les opérations, Spark prend en charge les requêtes SQL, des flux de données, et des analyses complexes

telles que l'apprentissage de la machine et des algorithmes de graphes out-of-the-box. Non seulement cela, les utilisateurs peuvent combiner toutes ces capacités de manière transparente dans un seul flux de travail

Exécute Partout : Spark fonctionne sur Hadoop, Mesos, autonome, ou dans le nuage. Il peut accéder à diverses sources de données, y compris HDFS, Cassandra, HBase, S3. [13]

4.2.4 L'écosystème de Spark:

Composants de Spark écosystème se développer et plusieurs contributions sont faites de temps en temps. Principalement, l'écosystème de la Spark comprend les composants suivants:

- ✓ Shark (SQL)
- ✓ Spark streaming (Streaming)
- ✓ MLlib (Machine Learning)
- ✓ GraphX (graphique de calcul)
- ✓ SparkR (R sur Spark)
- ✓ BlindDB SQL (approximative)

Ces composants sont construits sur le dessus de Spark Core Engine. Spark Core Engine permet d'écrire des programmes Spark premières et des programmes Scala et de les lancer; il permet également d'écrire des programmes Java avant de les lancer. Tous ceux-ci sont en cours d'exécution par Spark Core Engine. Pour couronner le tout, il y a plusieurs projets qui sont apparus très rapide et efficace.[25]

Requin : Shark est l'une des composantes de l'écosystème Spark. Il est utilisé pour effectuer une analyse des données structurées, en particulier si les données sont trop volumineux. Shark permet également l'exécution de requêtes Hive non modifiées sur le déploiement de Hadoop existant.

BlindDB : BlindDB ou aveugles Base de données est également connu comme une base de données SQL approximative. S'il y a une énorme quantité de barraging de données et vous n'êtes pas vraiment intéressé par l'exactitude, ou dans les résultats exacts, mais veulent juste avoir un rugueux ou d'image approximative, BlindDB obtient vous la même chose. Firing une requête, en faisant une sorte d'échantillonnage, et de donner quelques - uns de sortie est appelé SQL approximative.

Spark streaming : Spark Le streaming est une de ces caractéristiques uniques, qui ont habilitées Spark potentiellement prendre le rôle d'Apache Tempête. Spark Le streaming permet essentiellement de créer des applications analytiques et interactives pour les données en direct streaming. Vous pouvez faire la diffusion des données, puis, Spark pouvez exécuter ses opérations à partir des données en continu lui-même.

MLlib : MLlib est une bibliothèque d'apprentissage de la machine comme Mahout. Il est construit sur le dessus de Spark, et a la disposition pour soutenir de nombreux algorithmes d'apprentissage machine. Mais la différence de points avec Mahout est qu'il fonctionne presque 100 fois plus rapide que MapReduce. Il est pas encore aussi enrichi Mahout, mais il arrive assez bien, même si elle est encore dans la phase initiale de la croissance.

GraphX : GraphX est la nouvelle API (en version alpha) pour les traitements de graphes et de parallélisation de graphes. GraphX étend les RDD de Spark en introduisant le Resilient Distributed Dataset Graph, un multi-graphe orienté avec des propriétés attachées aux nœuds et aux arrêtes. Pour le support de ces traitements, GraphX expose un jeu d'opérateurs de base (par exemple subgraph, joinVertices, aggregateMessages),[25]ainsi qu'une variante optimisée de l'API Pregel. De plus, GraphX inclut une collection toujours plus importante d'algorithmes et de builders pour simplifier les tâches d'analyse de graphes.

SparkR : Il y a beaucoup de gens de la piste de la science des données, qui doivent être conscients que pour l'analyse statistique, R est parmi les meilleurs. Il existe déjà une intégration de R avec Hadoop. Maintenant, SparkR est un package pour le langage R pour permettre aux utilisateurs R pour tirer parti de la puissance de Spark de R shell. [25]

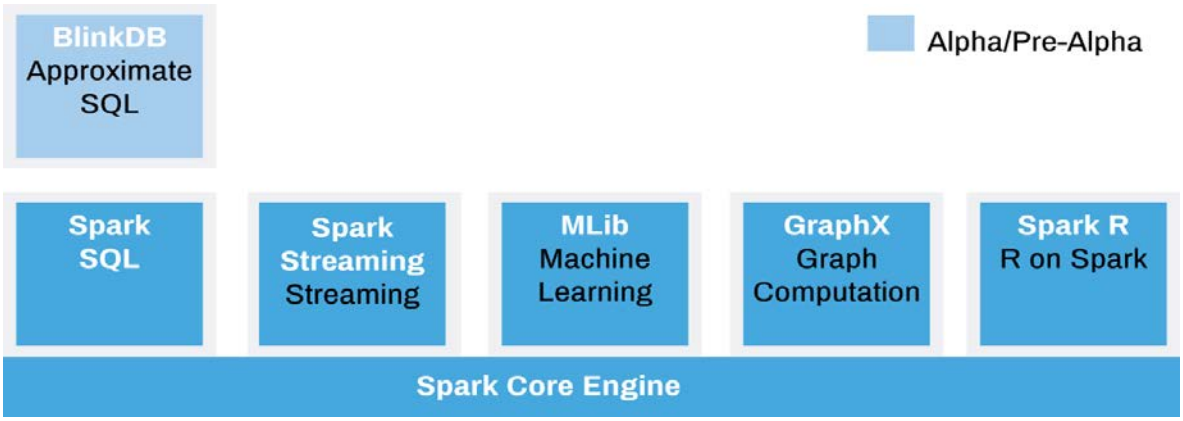

Figure 10: Spark Framework Libraries.

Il existe aussi des adaptateurs pour intégration à d'autres produits comme Cassandra (Spark Cassandra Connector) et R (SparkR). Avec le connecteur Cassandra, vous pouvez utiliser Spark pour accéder à des données stockées dans Cassandra et réaliser des analyses sur ces données.

4.2.5 Les concepts fondamentaux de Spark :

4.2.5.1 Les Resilient Distributed Datasets « RDD »

Les **Resilient Distributed Datasets** (basés sur la publication de recherche de Matei), ou RDD, sont un concept au cœur du framework Spark.[25] Vous pouvez voir un RDD comme une table dans une base de données. Celui-ci peut porter tout type de données et est stocké par Spark sur différentes partitions. Les RDD permettent de réarranger les calculs et d'optimiser le traitement. Ils sont aussi tolérants aux pannes car un RDD sait comment recréer et recalculer son ensemble de données. Les RDD sont immutables. Pour obtenir une modification d'un RDD, il faut y appliquer une transformation, qui retournera un nouveau RDD, l'original restera inchangé. Les RDD supportent deux types d'opérations:

- Les transformations
- Les actions

✓ **Les transformations:** Les transformations ne retournent pas de valeur seule, elles retournent un nouveau RDD. Rien n'est évaluée lorsque l'on fait appel à une fonction de transformation, cette fonction prend juste un RDD et retourne un nouveau RDD. Les fonctions de transformation sont par exemple map, filter, flatMap, groupByKey, reduceByKey, aggregateByKey, pipe et coalesce.

✓ **Les actions:** évaluent et retournent une nouvelle valeur. Au moment où une fonction d'action est appelée sur un objet RDD, toutes les requêtes de traitement des données sont calculées et le résultat est retourné. Les actions sont par exemple reduce, collect, count, first, take, countByKey et foreach.

Les principales transformations et actions disponibles dans Spark sont les suivantes :

Transformations	*map*(*f* : T ⇒ U)	:	RDD[T] ⇒ RDD[U]
	filter(*f* : T ⇒ Bool)	:	RDD[T] ⇒ RDD[T]
	flatMap(*f* : T ⇒ Seq[U])	:	RDD[T] ⇒ RDD[U]
	sample(*fraction* : Float)	:	RDD[T] ⇒ RDD[T] (Deterministic sampling)
	groupByKey()	:	RDD[(K, V)] ⇒ RDD[(K, Seq[V])]
	reduceByKey(*f* : (V, V) ⇒ V)	:	RDD[(K, V)] ⇒ RDD[(K, V)]
	union()	:	(RDD[T], RDD[T]) ⇒ RDD[T]
	join()	:	(RDD[(K, V)], RDD[(K, W)]) ⇒ RDD[(K, (V, W))]
	cogroup()	:	(RDD[(K, V)], RDD[(K, W)]) ⇒ RDD[(K, (Seq[V], Seq[W]))]
	crossProduct()	:	(RDD[T], RDD[U]) ⇒ RDD[(T, U)]
	mapValues(*f* : V ⇒ W)	:	RDD[(K, V)] ⇒ RDD[(K, W)] (Preserves partitioning)
	sort(*c* : Comparator[K])	:	RDD[(K, V)] ⇒ RDD[(K, V)]
	partitionBy(*p* : Partitioner[K])	:	RDD[(K, V)] ⇒ RDD[(K, V)]
Actions	*count*()	:	RDD[T] ⇒ Long
	collect()	:	RDD[T] ⇒ Seq[T]
	reduce(*f* : (T, T) ⇒ T)	:	RDD[T] ⇒ T
	lookup(*k* : K)	:	RDD[(K, V)] ⇒ Seq[V] (On hash/range partitioned RDDs)
	save(*path* : String)	:	Outputs RDD to a storage system, *e.g.*, HDFS

En fait, les **transformations** sont des opérations ***lazy*** qui permette de définir un nouveau RDD alors que les **actions** permettent d'exécuter un calcul et de retourner une valueur au programme ou de l'écrire sur un stockage externe. En plus de ces opérations, un RDD peut être persisté (méthode persist). En outre, il est possible d'obtenir l'ordre des partitions d'un RDD. Les opérations telles que groupByKey, reduceByKey et le sort fournissent un hash ou un interval de RDD partitionné.[26]

4.2.5.1.1 Les propriétés d'un RDD : Un RDD possède cinq propriétés principales*:*

- Une liste des partitions;
- Une fonction de calcul de chaque groupe;
- Une liste des dépendances sur d'autres RDD;
- En option, un Partitioner pour RDD clé-valeur (par exemple pour dire que le RDD est partitionné de hachage);
- En option, une liste de sites préférés pour calculer chaque division sur (par exemple l'emplacement des blocs d'un fichier HDFS);

4.2.6 MapReduce dans Spark :

MapReduce est largement adopté pour le traitement et la génération d'ensembles de données volumineux avec un algorithme parallèle et distribué sur un cluster. Il permet aux utilisateurs d'écrire des calculs parallèles, en utilisant un ensemble d'opérateurs de haut niveau, sans avoir à se soucier de la répartition du travail et de la tolérance aux pannes.

Malheureusement, dans la plupart des cadres actuels, la seule façon de réutiliser les données entre les calculs (Ex: entre deux emplois MapReduce) est d'écrire à un système de stockage stable externe (Ex: HDFS). Bien que ce cadre offre de nombreuses abstractions pour l'accès aux ressources informatiques d'une grappe, les utilisateurs veulent toujours plus.

Les deux applications **itératives** et **interactives** nécessitent le partage de données plus rapide entre les emplois parallèles. Le partage des données est lente en MapReduce due à la **réplication,** la **sérialisation,** et le **disque IO.** En ce qui concerne le système de stockage, la plupart des applications Hadoop, ils dépensent plus de 90% du temps à faire des opérations de lecture-écriture HDFS. [35]

4.2.7 RDD dans Spark :

Nous avons vu que le partage des données est lente en MapReduce due à la **réplication,** la **sérialisation,** et le **disque IO.**

Conscient de ce problème, les chercheurs ont développé un cadre spécialisé appelé Spark Apache. L'idée clé de l' étincelle est **R**esilient **D**istributed **D**atasets (RDD); il prend en charge en mémoire de traitement de calcul. Cela signifie, il stocke l'état de la mémoire comme un objet à travers les emplois et l'objet est partageable entre ces jobs. Le partage des données dans la mémoire est de 10 à 100 fois plus rapide que le réseau et le disque.[34]

*4.2.8*Mode de cluster Présentation:

Cette partie donne un bref aperçu de la façon dont Spark fonctionne sur les clusters, pour le rendre plus facile à comprendre. [26]

4.2.8.1 Topologie

Un cluster Spark se compose d'un **master** et d'un ou plusieurs **workers**. Le cluster doit être démarré et rester actif pour pouvoir exécuter des **applications**.

Le master a pour seul responsabilité la gestion du cluster et il n'exécute donc pas de code MapReduce. Les workers, en revanche, sont les exécuteurs. Ce sont eux qui apportent des *ressources*au cluster, à savoir de la mémoire et des cœurs de traitement.

Pour exécuter un traitement sur un cluster Spark, il faut soumettre une **application** dont le traitement sera piloté par un **driver**. Deux modes d'exécution sont possibles :

- **Mode *client* :** le driver est créé sur la machine qui soumet l'application
- **Mode *cluster*:** le driver est créé à l'intérieur du cluster.[26]

4.2.8.2 Communication au sein du cluster:

Les workers établissent une communication bidirectionnelle avec le master : le worker se connecte au master pour ouvrir un canal dans un sens, puis le master se connecte au worker pour ouvrir un canal dans le sens inverse. Il est donc nécessaire que les différents nœuds du cluster puissent se joindre correctement (résolution DNS…).

La communication entre les nœuds s'effectue avec le framework **Akka**. C'est utile à savoir pour identifier les lignes de logs traitant des échanges entre les nœuds.

Les nœuds du cluster (master comme workers) exposent par ailleurs une interface Web permettant de surveiller l'état du cluster ainsi que l'avancement des traitements. Chaque nœud ouvre donc deux ports :

- un port pour la communication interne : port 7077 par défaut pour le master, port aléatoire pour les workers
- un port pour l'interface Web : port 8080 par défaut pour le master, port 8081 par défaut pour les workers.[26]

4.3 Hadoop vs Spark:

Dans les premiers jours de gros volumes de données, Apache Hadoop est non seulement le «éléphant dans la pièce», comme certains l'ont appelé. Hadoop était la chambre. Mais cela est en train de changer comme Hadoop se déplace pour faire place à Spark Apache, un outil plus récent et plus avancé grande données de l'Apache Software Foundation.

Il ne fait aucun doute que Spark a déclenché une tempête de l'activité au sein de la communauté open source. Tant et si bien que les organisations qui cherchent à adopter une stratégie de données big sont maintenant interrogent la solution qui pourrait être un meilleur ajustement, Hadoop vs Spark, une comparaison directe est difficile, car ils font un grand nombre des mêmes choses, mais sont également non-chevauchement dans certaines régions. On à entamé les point essentielles dans se tableau:

	HADOOP	**SPARK**
Origine	**Hadoop**: a été inventé avec Yahoo pour indexer les données Web. Le projet a été lancé en 2006. En 2008, Yahoo a publié la source en open source. [8]	**Spark:** originellement été développé par AMPLab, de l'Université UC Berkeley, en 2009 et passé open source sous forme de projet Apache en 2010.[8]
Défini	**Hadoop** est un projet Apache.org qui est une bibliothèque logicielle et un cadre qui permet un traitement distribué de grands ensembles de données (grandes de données) à travers des grappes d'ordinateurs à l'aide des modèles de programmation simples.[18] Hadoop peut évoluer à partir des systèmes informatiques simples jusqu'à des milliers de systèmes de produits qui offrent le stockage local et calculer	**Les développeurs de Spark** Apache facturent comme «un moteur rapide et générale pour le traitement de données à grande échelle.» Par comparaison, et coller avec l'analogie, si le cadre Big Data de Hadoop est le gorille de 800 lb, alors Spark est le 130-lb grand guépard de données.[22]

	la puissance. Hadoop, en substance, est l'omniprésent 800 lb grand gorille de données dans le grand espace d'analyse des données.	
Performances	Hadoop MapReduce persiste vers le disque après une carte ou de réduire l'action, de sorte que Spark devrait surperformer Hadoop MapReduce.[23] MapReduce tue ses processus dès qu'un travail est fait, il peut facilement courir à côté d'autres services avec des différences mineures sur les performances. Hadoop MapReduce est conçu pour les données qui ne rentrent pas dans la mémoire et il peut bien fonctionner avec d'autres services.	**Spark** à besoin de beaucoup de mémoire pour les traitements de données il charge un processus en mémoire et y maintient jusqu'à nouvel ordre, pour des raisons de mise en cache. Si Spark fonctionne sur Hadoop YARN avec d'autres services de ressources exigeant, ou si les données est trop grande pour tenir entièrement dans la mémoire, alors il pourrait y avoir grandes dégradations de performance pour Spark. Spark fonctionne mieux lorsque toutes les données s'inscrit dans la mémoire, en particulier sur les clusters dédiés.[24]
Le Coût	**MapReduce** utilise des quantités standard de mémoire parce que son traitement est basé sur le disque, de sorte qu'une entreprise devra acheter des disques plus rapides et beaucoup d'espace disque pour exécuter MapReduce. MapReduce exige également plusieurs systèmes pour distribuer les E / S disque sur plusieurs	**Spark** nécessite beaucoup de mémoire, mais peut traiter avec une quantité standard de disque qui fonctionne à des vitesses standard. Certains utilisateurs se sont plaint de fichiers temporaires et leur nettoyage. Généralement, ces fichiers temporaires sont conservés pendant sept jours pour accélérer tout

	systèmes.[19]	traitement sur les mêmes ensembles de données. L'espace disque est un produit relativement peu coûteux et que Spark ne pas utiliser le disque I / O pour le traitement, l'espace disque utilisé peut être exploité SAN ou NAS.[26] Il est vrai, cependant que les systèmes d'allumage coûtent plus cher en raison des grandes quantités de RAM nécessaires pour faire fonctionner tout en mémoire. Mais ce qui est vrai aussi est la technologie que Spark réduit le nombre de systèmes requis. Donc, vous avez beaucoup moins de systèmes qui coûtent plus. Il y a probablement un point où **Spark réduit effectivement les coûts par unité de calcul**, même avec l'exigence de mémoire RAM supplémentaire.
Traitement de l'information	MapReduce est un moteur de traitement par lots. MapReduce fonctionne en étapes séquentielles par la lecture des données du cluster, effectuer son opération sur les données, écrire les résultats à la grappe, la lecture des données mises à jour à partir du cluster, effectuer l'opération suivante de données, l'écriture de ces résultats à la grappe et ainsi de suite.	Spark effectue des opérations similaires, mais il le fait en une seule étape et en mémoire. Il lit les données du cluster, effectue son opération sur les données, puis l'écrit au cluster. Spark comprend également sa propre bibliothèque graphique de calcul, GraphX. GraphX permet aux utilisateurs de visualiser les mêmes données sous forme de graphiques et

		de collections. Les utilisateurs peuvent également se transformer et à se joindre à des graphiques avec datasets résilients distribués (RDD), décrites dans la section de la tolérance aux pannes.[25]
Tolérance aux pannes	MapReduce utilise TaskTrackers qui fournissent des battements de coeur au JobTracker. Si un battement de coeur est manqué alors le JobTracker reprogramme tous les cours et les opérations en cours à un autre TaskTracker. Cette méthode est efficace pour fournir la tolérance aux pannes, mais il peut augmenter de manière significative les délais de réalisation des opérations qui ont même un seul échec.[16]	Spark utilise datasets résilients Distributed (RDD),[25] qui sont des collections d'éléments qui peuvent être exploités en parallèle à tolérance de pannes. RDD peut faire référence à un ensemble de données dans un système de stockage externe, comme un système de fichiers partagé, HDFS, HBase, ou toute autre source de données offrant une Hadoop InputFormat. Spark peut créer RDD provenant de toute source de stockage supporté par Hadoop, y compris les systèmes de fichiers locaux ou un de ceux énumérés précédemment. RDD peut être persistant dans le but de mettre en cache un ensemble de données dans la mémoire à travers les opérations. Cela permet à des actions futures à être beaucoup plus rapide, par autant que dix fois. le cache de Spark est tolérant aux pannes en ce que si une partition d'un RDD est perdue, il sera automatiquement

		recalculé en utilisant les transformations originales.
Evolutivité	MapReduce évolutive en utilisant le HDFS. Yahoo a censément un cluster Hadoop 42.000 noeud, donc peut-être le ciel est vraiment la limite.	Spark aussi évolutive en utilisant le HDFS. Le plus grand groupe de Spark connu est de 8.000 noeuds, mais aussi grand données augmente, il est prévu que la taille des grappes augmentera pour maintenir les attentes de débit.
La sécurité	Hadoop prend en charge l'authentification Kerberos, qui est un peu pénible à gérer. Toutefois, les fournisseurs tiers ont permis aux entreprises d'utiliser Kerberos Active Directory et LDAP pour l'authentification. Ces mêmes fournisseurs tiers offrent également crypter des données pour en vol et les données au repos. Le système de fichiers distribués de Hadoop prend en charge les listes de contrôle d'accès (ACL) et un modèle traditionnel des permissions de fichiers. Pour le contrôle de l'utilisateur dans la soumission des travaux, Hadoop fournit le niveau de service d'autorisation, ce qui garantit que les clients ont les bonnes permissions.	la sécurité de Spark est un peu clairsemée par actuellement seulement en charge l'authentification par secret partagé (authentification par mot de passe). Le bonus de sécurité que Spark peut profiter est que si vous exécutez Spark sur HDFS, il peut utiliser les ACLs HDFS et les autorisations de niveau fichier. En outre, Spark peut fonctionner sur YARN en lui donnant la possibilité d'utiliser l'authentification Kerberos

Topologie de traitements de de données	**Hadoop:** il est le mieux adapté pour le traitement par lots	**Spark:** il est conçu pour la haute performance, il travaille sur la totalité de données.
La rapidité	**Moins rapide** car : MapReduce fonctionne en étapes et la séquence de travail de MapReduce ressemble à ceci : « il lit les données au niveau du cluster, il exécute une opération, il écrit les résultats au niveau du cluster, il lit à nouveau les données mises à jour au niveau du cluster, il exécute l'opération suivante, il écrit les nouveaux résultats au niveau du cluster, etc. »	**Plus rapide** car : la méthode utilisée par Spark pour traiter les données fait qu'il est beaucoup plus rapide que MapReduce. Spark peut travailler sur la totalité des données en une seule fois, il exécute les opérations d'analyse de données en mémoire et en temps quasi réel : « Spark lit les données au niveau du cluster, effectue toutes les opérations d'analyses nécessaires, écrit les résultats au niveau du cluster, et c'est tout »
High level langages de script	Pig, Hive. Les deux langues convertissent les scripts de haut niveau à des emplois MapReduce.	Il propose une API **de haut-niveau** pour une meilleure productivité. Le Shell **Spark** est disponible pour **les langages** Scala et Python.
Difficulté d'installation	**Moins facile:** Hadoop MapReduce est écrit en Java et est tristement célèbre pour être très difficile à programmer.Pig rend plus facile, mais il faut un certain temps pour apprendre la syntaxe et Hive ajoute la compatibilité de SQL à la plaque.Certains outils Hadoop peuvent également exécuter des emplois	**Spark est plus facile** à programmer et comprend un mode interactif. il a des API confortables pour Java, Scala et Python, et comprend également Spark SQL (anciennement connu sous le nom de requin) pour le savvy SQL.il est facile d'écrire des fonctions définies par l'utilisateur. Il inclut même un mode interactif pour

	MapReduce sans aucune programmation.	l'exécution des commandes avec une rétroaction immédiate.
Possibilité d'utilisation de Spark indépendamment d'Hadoop et réciproquement	Hadoop comprend un composant de stockage, connu sous le nom de HDFS (Hadoop Distributed File System), et un outil de traitement appelé MapReduce. De fait, il n'est pas nécessaire de faire appel à Spark pour traiter ses données Hadoop.[17]	Il est possible d'utiliser Spark sans faire intervenir Hadoop. Spark n'a pas de système de gestion de fichiers propre, ce qui veut dire qu'il faut lui associer un système de fichiers - soit HDFS, soit celui d'une autre plate-forme de données dans le cloud.[34]
Les domaines d'utilisations	✓ Analyses de risques ✓ Suivi d'opérations bancaires (Ecarts d'utilisations / Utilisations frauduleuse de Carte bancaire) ✓ Analyse d'informations clients (ex : Suivi clients, prospections …) ✓ Moteur de recommandation de produits (ex : Amazon, Netflix) ✓ Récolte d'informations pour le Retargeting de publicités ✓ Données des réseaux sociaux (ex : Twitter) ✓ Suivi de comportement utilisateurs devant une page web (Netflix, Online learning).	✓ **Spark** permet de développer des pipelines de traitement de données complexes, à plusieurs étapes, en s'appuyant sur des graphes orientés acycliques(DAG). ✓ Des domaines couvrant l'administration, la sécurité, l'évolutivité et le streaming, « One Platform »[34]
Reprise après incident	**Hadoop**est résilient aux pannes ou aux défaillances du système, car les données sont écrites sur le disque après chaque opération.[18]	**Spark**offre la même résilience intégrée du fait que les objets de données sont stockés dans ce qu'on appelle des ensembles de données

		distribués résilients (RDD) répartis sur le cluster de données.[33]
Gestion facile	**Hadoop Spark**,il est possible d'effectuer Streaming, Traitement par lots et l'apprentissage automatique dans le même cluster.[34]	**Spark**,il est possible de contrôler différents types de charges de travail, donc s'il y a une interaction entre les différentes charges de travail dans le même processus,il est plus facile à gérer et sécuriser ces charges de travail qui viennent comme une limitation MapReduce.[34]
Les avantages	**-Le coût de stockage**: si vous avez besoin pour traiter vraiment Big Data, Hadoop sera certainement l'option moins cher depuisl'espacedisque dur est livré à un taux beaucoup plus faible quel'espacemémoire.[18] - Portable (clusters hétérogènes). - Indéfiniment scalable. - Très extensible et interconnectable avec d'autres solutions. - La flexibilité : un des grands intérêts d'Hadoop est de pouvoir stocker et analyser aussi bien des données structurées que des données non structurées, et ceci de la même manière ! Ainsi, il devient possible d'analyser et de croiser des bases de données et des données digitales. - La robustesse**:** map reduce d'Hadoop	- spark donne la capacité de traiter rapidement des données à grande échelle : les programmes sont exécutés jusqu'à cent fois plus rapidement que sur Hadoop MapReduce en mémoire ou dix fois plus rapidement sur disque.[34] - Des applications peuvent être écrites aisément en Java, Scala ou Python, rendant Spark facile d'utilisation. Également, les analyses complexes, en streaming et SQL peuvent être combinées ensemble. De plus, Spark peut lire tous types de données Hadoop, que ce soit HDFS, HBase ou Cassandra. - **Spark** rend la programmation des jobs plus aisée avec l'intégration dans le langage de programmation Scala. Au final, avec le chargement de datasets en mémoire et les requêtes

	n’est pas sensible à la panne d’un serveur du cluster. En d’autres termes, le framework Hadoop se comporte de la même façon même en cas de pannes de serveurs.[16] -ABORDABILITE*:* niveaux Hadoop le terrain de jeu en permettant même de petites organisations avec des budgets modestes pour récolter de grands bénéfices de données. - ÉVOLUTIVITE**:** Hadoop peut évoluer vers le haut ou vers le bas pour accueillir les petites et très grandes charges de travail. - architecture massivement distribuée permet un traitement (nettoyage, filtrage, etc.) rapide des données, et les coûts bas de stockage de la plate-forme Azure sur laquelle il s’appuie permettent d’y conserver les données brutes stockées plusieurs semaines voire plusieurs mois si nécessaires.	répétées, Spark facilite les analyses interactives de big data.[36] - Apache Spark est un outil adapté au Machine Learning puisque les algorithmes sont calculés 30 fois plus rapidement que sur Hadoop, tout en maintenant la scalabilité de MapReduce.[32]] - D’après YARN et Hadoop 2.0, Spark est devenu populaire parce que Spark peut fonctionner au-dessus de HDFS ainsi que d'autres composants Hadoop.[33]] - Traitement des données du capteur: Lorsque les données sont extraites et jointes provenant de sources multiples, en mémoire ensemble de données très utiles car ils sont faciles et rapides à traiter. - Mining Interactive Data et Data Processing
Les inconvenient	- Overhead supérieur à une solution propriétaire spécifique dédiée à un problème, encore expérimental et en développement.[23] - **Hadoop** ne fonctionne tout simplement pas très rapide lorsque l'on compare Spark vs Hadoop.En effet, la plupart des carte / réduire les emplois	Le mode de fonctionnement de MapReduce peut être suffisant si les besoins opérationnels et les besoins de reporting sont essentiellement statiques et s’il est possible d’attendre la fin du traitement des lots. Qui veut dire c’est pas tout le monde à besoin de la rapidité de spark.[33]

	sont des emplois de longue durée batch qui peut prendre quelques minutes ou heures ou plus pour terminer.En plus de cela, les grandes demandes de données et les aspirations sont en croissance, et la charge de travail par lots cèdent la place à des activités plus interactives que le framework Hadoop MapReduce est tout simplement pas fait pour.[20] - **Hadoop** fonctionne en mode batch et ne peut donc être confortablement utilisé comme source de données d'outils décisionnels pour effectuer des analyses ad-hoc interactives.[19] - les solutions Hadoop s'appuient généralement sur des clusters, qui sont difficiles à mettre en place et à administrer. Elles nécessitent aussi l'intégration de plusieurs outils pour les différents cas d'utilisation big data (comme Mahout pour le Machine Learning et Storm pour le traitement par flux).[21]	- **Spark** utilise un grand RAM il a besoin d'une machine haut de gamme dédié physique pour produire des résultats efficaces.[33] - Spark n'a pas la gestion des fichiers et à cet effet doit compter sur Distributed File System Hadoop (HDFS) ou une autre solution.[26]

Tableau II.1: Comparaison entre Hadoop et Spark

4.4 Résumé d'Hdoop vs Spark:

À première vue, il semble que Spark Apache est le nouveau jouet brillant sur le terrain de jeu Big Data, mais il y a encore des cas d'utilisation pour l'utilisation de Hadoop MapReduce.

Il n'y a pas un manque d'information sur l'Internet sur la façon dont Spark rapide est comparé à MapReduce. La raison pour laquelle Spark est si rapide est qu'il traite tout dans la mémoire. Oui, il peut également utiliser le disque pour les données qui ne tient pas entièrement dans la mémoire.

Le traitement en mémoire de Spark offre près de l'analyse en temps réel pour les données de campagnes de marketing, l'apprentissage machine, Internet des objets capteurs, ouvrez une session de surveillance, d'analyse de sécurité, et les sites de médias sociaux.

MapReduce utilise alternativement le traitement par lots et a été vraiment jamais construit pour la vitesse aveuglante. Il était à l'origine la configuration de recueillir en permanence des informations provenant de sites Web et il n'y avait pas d'exigences pour ces données ou en temps quasi réel.

Hadoop MapReduce est une plate-forme plus mature et il a été construit pour le traitement par lots. Il peut être plus rentable que Spark vraiment Big Data qui ne rentre pas dans la mémoire et aussi en raison de la plus grande disponibilité du personnel expérimenté. En outre, l'écosystème Hadoop MapReduce est actuellement plus gros grâce à de nombreux projets de soutien, des outils et des services cloud.

Mais même si Spark ressemble le grand gagnant, les chances sont que vous ne serez pas l'utiliser sur son propre-vous encore besoin HDFS pour stocker les données et vous pouvez utiliser HBase, Hive, Pig, Impala ou d'autres projets Hadoop. Cela signifie que vous aurez toujours besoin d'exécuter Hadoop et MapReduce aux côtés de Spark pour un forfait Big Data complète.

5 CONCLUSION:

Nous avons présenté dans ce chapitre les framework et leurs objectifs et quelques frameworks utilisés. On à vu les différents composants d'Hadoop, principalement, HDFS et le modèle de programmation MapReduce, nous asvons vu aussi comment le framework Apache Spark, avec son API standard, nous aide en matière de traitement et d’analyse de données. Nous avons aussi vu comment Spark se positionne par rapport aux implémentations MapReduce traditionnelles comme Apache Hadoop. Spark s’appuie sur le même système de stockage de fichiers qu’Hadoop.

Enfin nous avons terminez avec une étude comparatives entres les deux frameworks concourant Spark et Hadoop, ont put dire qu'ils ont une relation symbiotique avec l'autre. Hadoop fournit des fonctionnalités qui Spark ne possède pas, comme un système de fichiers distribué et Spark fournit en temps réel, le traitement en mémoire pour les ensembles de données qui en ont besoin. Le scénario de données grand parfait est exactement comme les designers destinés à des fins Hadoop et Spark à travailler ensemble dans la même équipe.

Chapitre 3

La segmentation des vidéos basée sur MapReduce

1 Introduction:

Dans le chapitre précédent on a présenté les framework hadoop et spark, ses points en communs, l'avantage de l'un par rapport à l'autre, et on à conclut que l'utilisation de spark est beaucoup plus performante que l'utilisation de hadoop pour les raison qu'on a déjà cité.

Dans le présent chapitre on va mettre en évidence la segmentation basé sur hadoop en s'appuyant sur des résultats bien réels. L'application sur laquelle porte ce chapitre est une application de traitement d'image, elle consiste en la segmentation des vidéos par l'utilisation de la technique d'image clé ou keyframe en anglais. Ce dernier algorithme sera présenté en utilisant le modèle de programmation adopté par Hadoop. De même, on va présenter l'architecture de l'application de segmentation des vidéos dans son cadre ou dans son environnement avec les résultats. A la fin du chapitre, nous définissons les différentes étapes d'installation et de configuration de Hadoop.

2 La segmentation de vidéo (par image clé ou keyframe en englais):

Segmentation de la vidéo et l'extraction d'images clés sont les bases de l'analyse vidéo et basée sur le contenu de récupération vidéo. Extraction d'image clé, est un élément essentiel dans la vidéo analyse et la gestion, en fournissant une vidéo appropriée récapitulation pour l'indexation vidéo, la navigation et récupération. L'utilisation d'images clés réduit la quantité des données requises pour l'indexation vidéo et fournit le cadre pour traiter le contenu vidéo. [27]

2.1 Définition de la segmentation de la vidéo:

La segmentation de la vidéo est un mécanisme pour produire un court résumé d'une vidéo pour donner à l'utilisateur un résumé visuel et utile de la séquence vidéo, il peut être une des images (**images clés)** ou des images **(écumes vidéo)** en mouvement. [8] Quand à regarder et navigation, un bon résumé de la vidéo permettra à l'utilisateur d'obtenir un maximum d'informations à propos de la séquence vidéo cible dans une contrainte de temps spécifiée ou des informations suffisantes dans le minimum de temps. Ces résumés générés peuvent aider les utilisateurs à naviguer sur de grandes archives vidéo et de prendre des décisions de manière plus efficace en ce qui concerne la sélection, la consommation, le partage ou la suppression de contenu. Les vidéos

résumées peuvent également être utilisés comme un produit final à partager, digéré, et apprécié par l'utilisateur. Retenir uniquement les informations essentielles d'une séquence vidéo améliore le stockage, la bande passante et le temps d'écoute. Les techniques développées dans la vidéo annotation touchent divers domaines, tels que les films, sports, vidéos, e-Learning, etc., Le processus d'abstraction de vidéo à généralement trois phases:

Analyse de l'information de la vidéo, la sélection du clip significative et la synthèse production. Pour analyser les informations vidéo, il est nécessaire de détecter les contours, les structures ou les motifs dans la composante visuelle, la composante audio et le composant textuel comme des sous-titres.

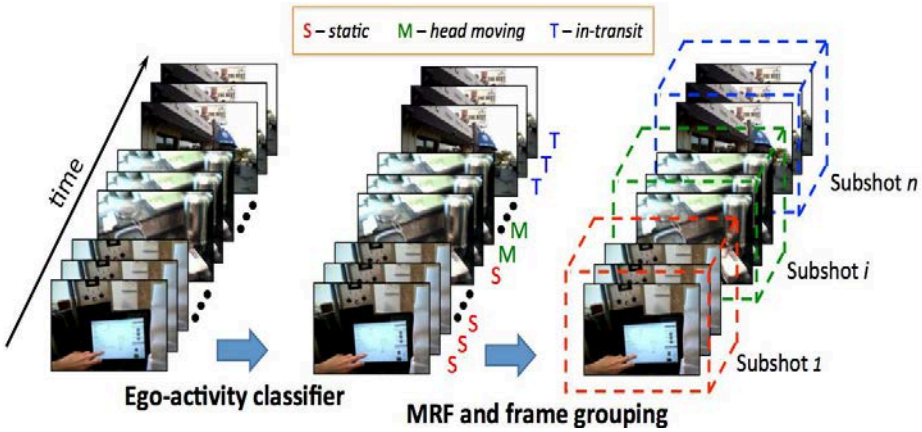

Figure01: segmentation de la vidéo

2.2 La segmentation des vidéos basée sur les key frames:

La segmentation de vidéo basée sur les images clés commence par extraire les frames de la séquence vidéo. Dans la deuxième étape les images extraites sont regroupées en se basant généralement sur la ressemblance entre les frames successives, si la différence dépasse un certain seuil on crée un nouveau groupe de frame sinon on ajout le nouveau frame au groupe en court de traitement. La sélection des images clés est effectuée par l'étape trois en sélectionnant en général les bords des groupes c'est-à-dire soit le premier ou le dernier frame. La procédure entière présentée sur la Figure qui suit.

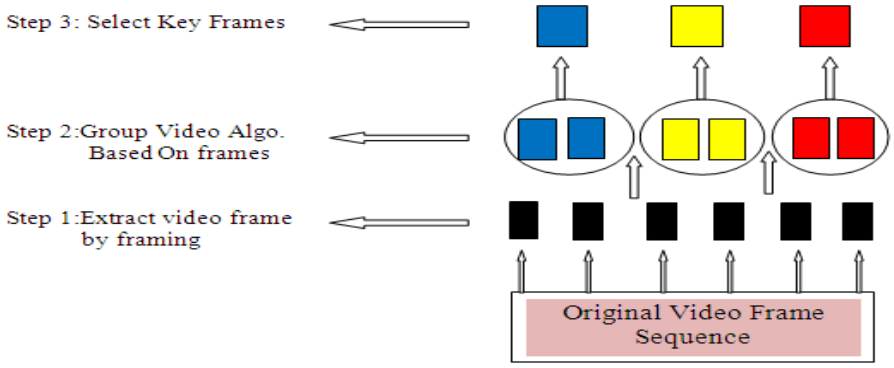

Figure02 : segmentation de vidéo basée sur les images clés.

En général, la segmentation de vidéo basée sur les keyframe peut être effectuée en trois façons différentes. Ceux-ci sont les suivantes.

- **Une segmentation basée sur l'échantillonnage:** Cette méthode choisit des images clés uniformément ou aléatoirement sous-échantillonnage, sans tenir compte du contenu vidéo. Le résumé produit par ces méthodes ne représente pas toutes les parties vidéo et peut provoquer une certaine redondance des images clés avec des contenus similaires.

- **Une segmentation basée sur une segmentation de scène:** Cette méthode extrait les images clés en utilisant la détection des scènes, la scène comprend toutes les pièces avec un lien sémantique dans la vidéo ou dans le même espace ou dans le même temps. L'inconvénient de ces techniques est la production d'un résumé, qui ne tient pas compte de la position temporelle des images.

- **Une segmentation basée sur la segmentation du shot**: Cette méthode extrait les images clés en utilisant la détection des shots, et les images clés choisis sont la première et la dernière image de chaque shot.

Ces méthodes sont efficaces pour un shot avec une petite variation du contenu, mais ils ne fournissent pas une représentation adéquate du shot avec des mouvements forts.

3 L'algorithme de keyframe avec Hadoop

//clé: nom de frame

// Valeur : contenu de frame

Map (Text key, BytesWritable value)

Image frame = collect_and_convert (value);

Vector v = gist(frame);

// v0: vecteur intermédiaire, ayant la même taille que le vecteur v

Vector v0 = new vector (v.size());

// s: similitude entre le cadre actuel et intermédiaire, chi_square est utilisé pour mesurer la distance entre les vecteurs.

Double s = 1 - CHI_SQUARE.compare(v, v0); calcul de facteur de similartité entre les vecteur

// Approximation de la valeur de similitude

S = around (s, _threshold)

// écrire la clé de valeur paire dans Hadoop contexte, la nouvelle clé est la valeur de similarité approximative, et l'élément de valeur est la valeur de la clé

emitIntermediate(s, key);

// clé: valeur de similarité approximative entre le cadre actuel et intermédiaire

// Valeur: nom du cadre

reduce (double key, Iterable<Text> value)

// Tri cadres nom en fonction de leur nombre (ex: f1, f2, f3 ..)

Collection c= Collections.sort(value);

// grappes d'extraction (plans) en utilisant l'algorithme de classification hiérarchique

List<cluster>

lc = Hierarchical_clustering(c);

// Extraire et émettre des plans limites dans Hadoop Contexte

For each cluster c in lc

Emit (c.first);

- **Description de l'algorithme :**

L'objectif principale de l'algorithme est de détecté le nombre de shots dans une vidéo après la segmentation à base de keyframe, et ces étapes sont:

Le mapper:

1. Récupérer le frame à traiter ;
2. Extraction de descripteur du frame en utilisant la méthode **GIST ;**
3. Calcule du facteur de similarité entre le frame lu et le frame intermédiaire (**f0** : frame de référence) en utilisant les vecteurs caractéristique. la similarité est calculée avec la fonction **S=1-distance(f, f0) ;**
4. Map retire les facteurs de la similarité s1, s2....sous forme de clé valeur (f1, s1), (f2, s2)....et il va l'écris dans un contexte ;
5. Avant de grouper les valeurs, le mapper va arrondir avec le **threshold** (seuil). On va calculer la valeur absolu S1-S2 si le résultat inférieur au seuil alors on va faire un traitement ;

exemple: {s1 s2 s3 } appartient à l'intervalle [v1,v2[(avec la valeur absolu de V1-V2= seuil) on les regroupe par exemple dans S11 et à partir de se principe on va arrondir. Le mapper dérige les tuples ayant la même clé vers le même reducer ;

Reducer :

6. Le reducer vas commencer là où le mapper est arrêté et il vas récupéré les (cle, valeur) et après il va les grouper selon le facteur de similarité, par exemple (f1, s1) (f3,s3), (f2, s2) et ainsi de suite. Connaissent les groupes alors on peut détecter le bord de chaque shot de la vidéo.

La figure03 résume le principe de l'algorithme keyframe avec MapReduce .

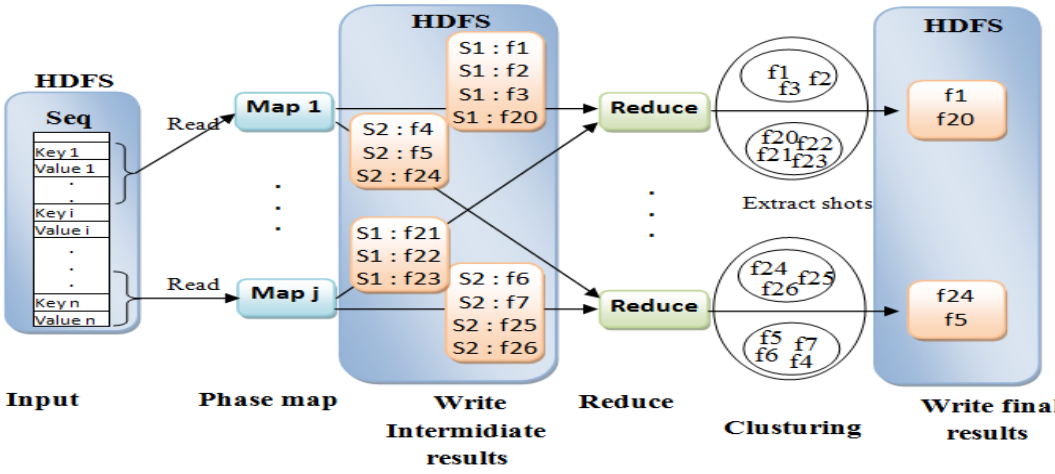

Figure03: Processus de détection des shots avec Hadoop

4 L'environnement de travail (caractéristique de la machine utilisée et version de chaque outil):

Dans la section suivante, on va décrire les outils et les systèmes composant notre environnement de travail.

4.1 Outils utilisées:

4.1.1 Machine physique :

La machine réel que on a utilisé dans notre pratique est de marque Acer Aspire 5733Z-P624G50MN avec un système d'exploitation windows7, 64 bits, qu'est un système d'exploitation de la société Microsoft, sorti le 22 octobre 2009 et un successeur de windows vista.

De plus, Windows 7 utilise la mémoire vive de façon bien plus intelligente, en utilisant beaucoup de mémoire s'il y en a, tout en sachant fonctionner correctement sur des configurations n'ayant que 512 Mo de mémoire vive. Windows 7 utilise également moins de données pour démarrer.

Notre ordinateur portable est caractérisé par :

- ✓ Ecran 15.6 puce
- ✓ Processeur Intel Pentium P6200 Dual core

- ✓ Une mémoire de 4 Go
- ✓ Un disque dur 500 Go

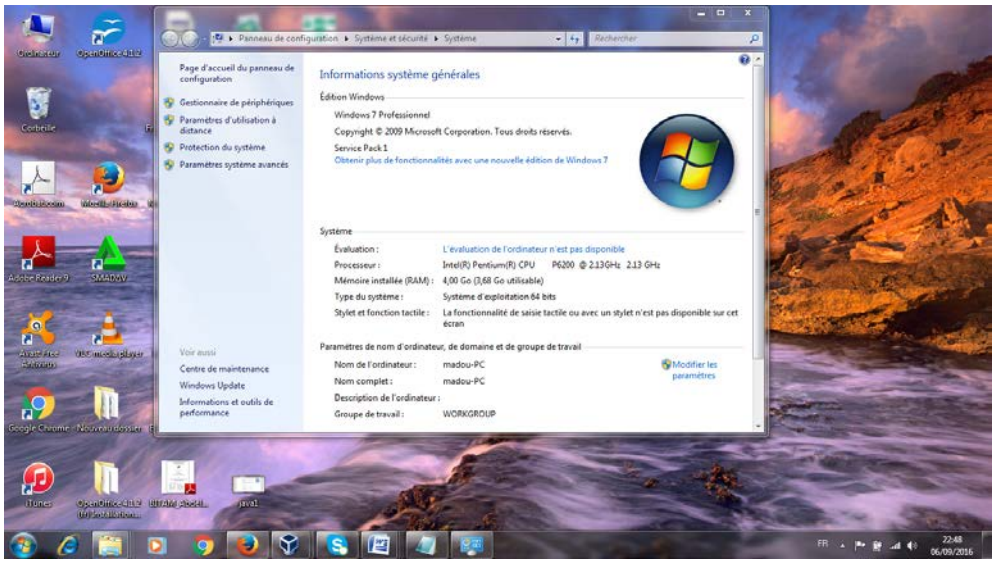

Figure04: Acer Aspire 5733Z-P624G50Mn-performance

4.1.2 Ubuntu:

Ubuntu est un système d'exploitation open source libre et gratuit basé sur la distribution Linux Debian. Son nom « Ubuntu » provient d'un ancien mot africain qui signifie « Humanité » et également qui signifie « je suis ce que je suis grâce à ce que nous sommes tous ». Dans le même ordre d'esprit, les utilisateurs sont encouragés à étudier son fonctionnement, le modifier, l'améliorer et enfin de le redistribuer.[17]

Ubuntu est construit sur l'architecture et l'infrastructure Debian, pour fournir des systèmes d'exploitation au serveur Linux, bureau, téléphone, tablette et d'exploitation de la télévision. Par défaut, Ubuntu est installé avec une pluralité de logiciels libres tels que LibreOffice, Filefox,…etc. Plusieurs autres logiciels libres peuvent être installés, notamment grâce à la logithèque Ubuntu (par exemple : GIMP, VLC), contrairement aux logithèques des autres systèmes d'exploitation les plus populaires, tel que Mac OS et Windows.

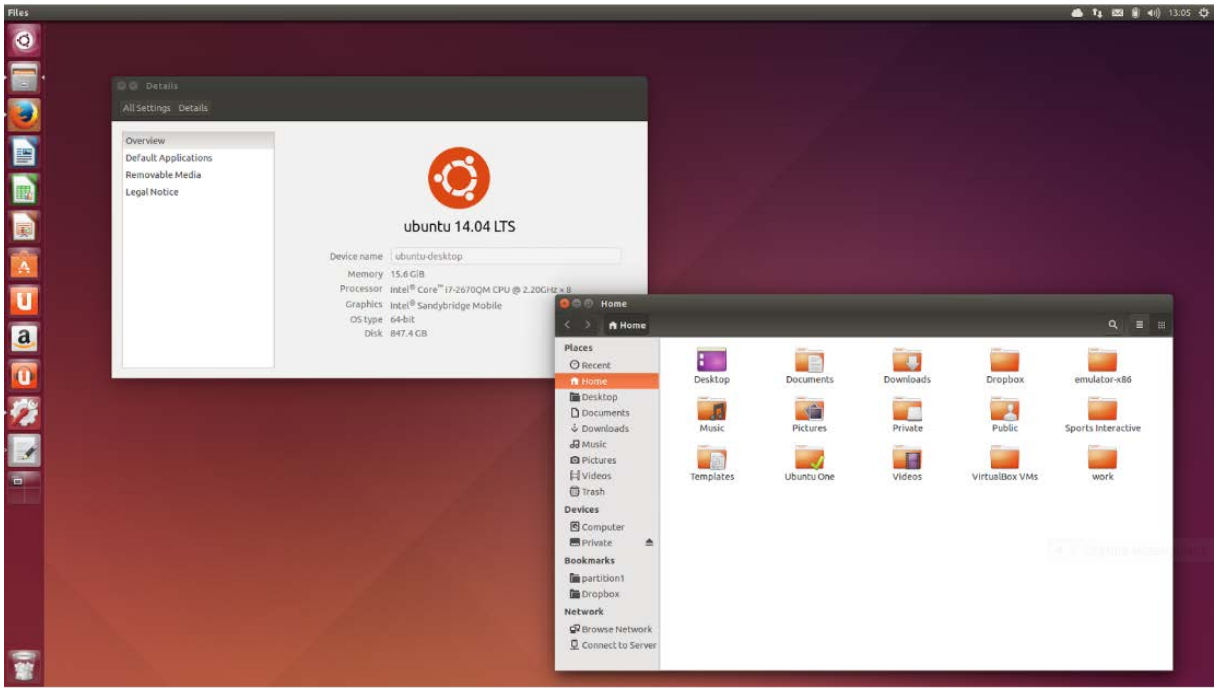

Figure05: ubuntu 14.04

4.1.3 Virtual machine:

Une machine virtuelle (VM) est une émulation d'un système informatique particulier. Les machines virtuelles fonctionnent sur les bases de l'architecture d'ordinateur et les fonctions réel ou hypothétique d'ordinateur, et leur mise en œuvre peut impliquer du matériel spécialisé, des logiciels, ou une combinaison des deux.

4.1.4 VirtualBox:

VirtualBox est un produit pour l'entreprise ainsi que l'utilisation à la maison. Non seulement VirtualBox extrêmement riche en fonctionnalités, c'est un produit de haute performance pour les clients de l'entreprise, il est également la seule solution professionnelle qui est librement disponible en tant que logiciel Open Source sous les termes de la GNU General Public License (GPL). [19]

]

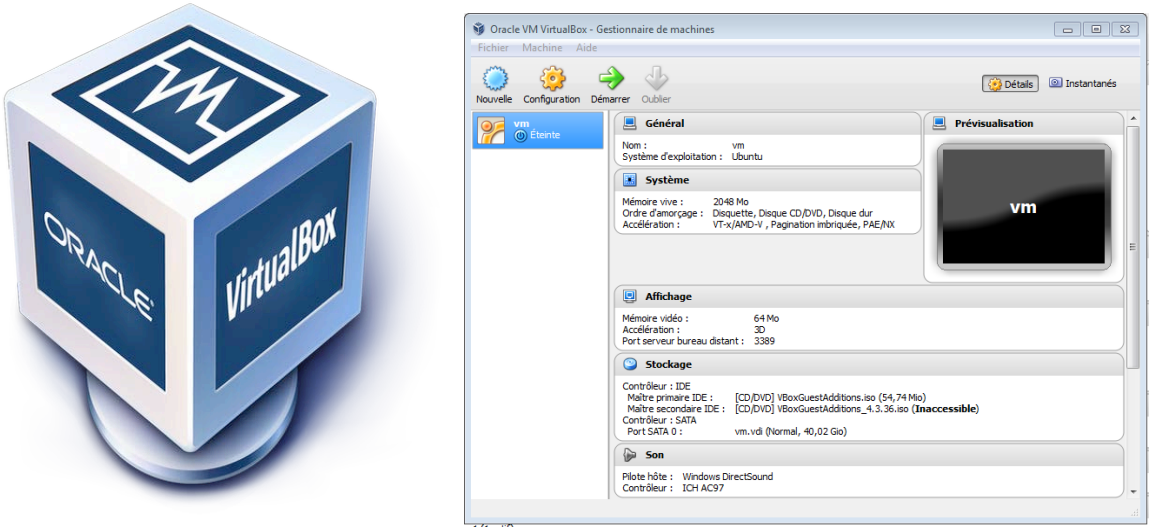

Figure06: Virtuel box

4.1.5 Java:

Java est un langage de programmation à usage général, évolué et orienté objet dont la syntaxe est proche du C. Il a été mis au point à partir de 1990 par la firme Sun Microsystems et officiellement présenté en 23 mai 1995. Ses caractéristiques ainsi que la richesse de son écosystème et de sa communauté lui ont permis d'être très largement utilisé pour le développement d'applications de types très disparates. Java est notamment largement utilisée pour le développement d'applications d'entreprises et mobiles. [23]

Java 8 est la dernière version de Java et offre de nouvelles fonctionnalités, des performances accrues et des corrections de bug pour améliorer l'efficacité de développement et d'exécution des programmes Java, le composant java est un outil indispensable pour l'exécution des jobs Hadoop.

Java 8 est à venir rapidement à différents clusters. Faire en sorte que Hadoop fonctionne de manière transparente avec Java 8 est important pour la communauté Apache**.**

4.1.6 XML:

XML (entendez extensible Markup Langage et traduisez Langage à balises étendu, ou Langage à balises extensible) est en quelque sorte un langage HTML amélioré permettant de définir de nouvelles balises. Il s'agit effectivement d'un langage permettant de mettre en forme des documents grâce à des balises (markup).

XML est un format fondé sur des balises comme HTML, mais il décrit le contenu plutôt que la présentation du contenu, les balises peuvent avoir aussi des attributs, et cela fait penser aux langages de programmation à objets.

La force de XML réside dans sa capacité à pouvoir décrire n'importe quel domaine de données grâce à son extensibilité. Il va permettre de structurer, poser le vocabulaire et la syntaxe des données qu'il va contenir. [8]

4.2 Open image :

OpenIMAJ est un ensemble de bibliothèques et d'outils pour l'analyse de contenu multimédia et la génération de contenu. OpenIMAJ est très large et contient tout de la vision state-of-the-art ordinateur et le regroupement de données avancées, par le biais d'un logiciel qui effectue une analyse sur le contenu, la présentation et la structure des pages Web.

OpenIMAJ est d'abord écrit en Java pur et en tant que tel est totalement indépendant de la plateforme. La capture vidéo et les matérielles bibliothèques contiennent du code natif, mais Linux (processeurs x86, x86_64 et ARM sont pris en charge), OSX et Windows sont pris en charge sur la boîte (sous 32 et 64 JVM bits). Il est possible d'écrire des programmes qui utilisent les bibliothèques dans toutes les langues JVM qui prend en charge l'interopérabilité de Java, tels que Groovy, Jython, JRuby ou Scala. OpenIMAJ peut même être exécuté sur les téléphones Android et tablettes.

Le logiciel OpenIMAJ est structuré en un certain nombre de modules. Les modules peuvent être utilisés indépendamment, donc si par exemple vous ont été le développement de logiciels de clustering de données en utilisant OpenIMAJ vous ne seriez pas besoin d'acquérir les modules liés à des images ou du texte. [30]

4.3 HADOOP 2.1.7:

Est une version stable elle est réalise le 06 juillet 2015, apache hadoop 2.7.1 est une version de point de la ligne 2.7.x de libération, et corrige quelques problèmes critiques en 2.7.0.

Hadoop est un framework écrit en Java pour exécuter des applications sur de grandes grappes de matériel de base et intègre des caractéristiques similaires à celles du système Google File (GFS) et du MapReduce paradigme informatique. De Hadoop HDFS est un système très

tolérant aux pannes distribué fichier et, comme Hadoop en général, conçue pour être déployée sur le matériel à faible coût. Il fournit un accès haut débit aux données d'application et est adapté pour les applications qui ont de grands ensembles de données. [20]

5 Les données utilisées :

Les vidéos sur lesquelles on à travail ont été ramenées de la bibliothèque gratuite ‘open vidéo project, on à travaillé sur quelques vidéos (anni03, USGS03_009, newport_4, hcli2001_01).

Vidéos	Duration	frames	shot
Anni003.mpg	142 s.	4267	24
UGS03_009.mpg	62 s.	1881	4
newport_4.mpeg	60 s.	1798	16
hcil2001_01.mpg	139 s.	4204	20

Tableau1.les données à traité

6 Résultat de l’exécution :

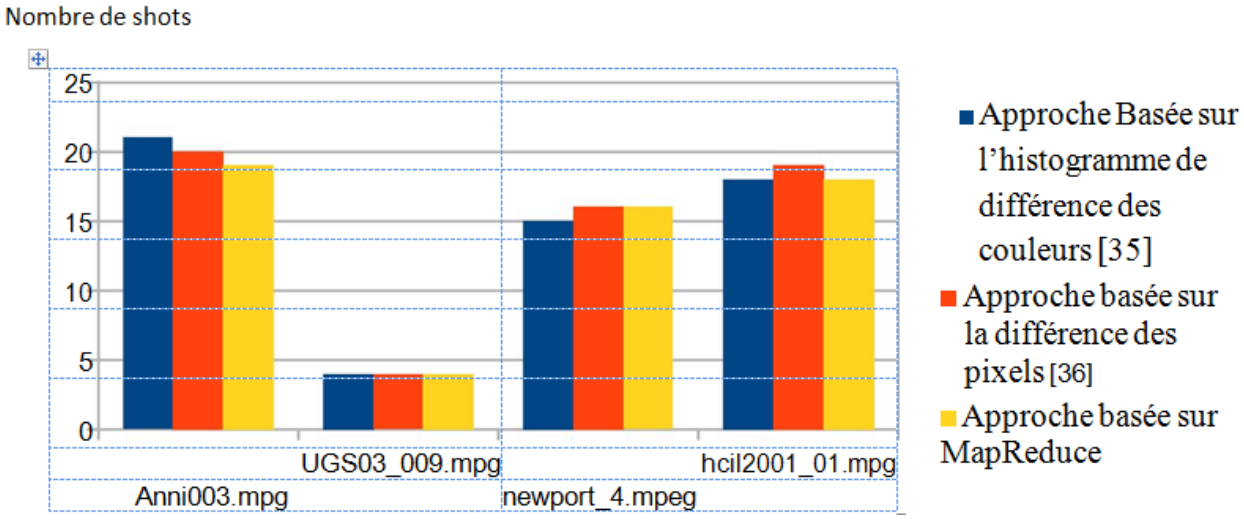

Figure07: Comparaison du nombre de shots détecté entre l’approche basée sur mapReduce et des existantes approches.

La méthode basée sur l'histogramme de différence des couleurs et **l'approche basé sur la différence des pixels** sont des méthodes qui existe déjà nous avons fais référence à ses méthodes la pour comparer avec **la méthode basée sur MapReduce** qu'on à utilisé dans notre application.

Les résultats obtenus et montré par **la figure07**, montre que la qualité de détection des shots est très acceptable en la comparant avec les autres approches citées plus haut. Plus que **la qualité de détection des shots**, le point fort de cette approche est qu'elle est basée sur le modèle MapReduce qui est caractérisé **par le point de passage à l'échelle** ce qui n'est pas offert par les autres méthodes.

7 Conclusion:

Dans ce chapitre, nous avons présenté au premier lieu la segmentation de la vidéo et ses différents types, puis on à décris la segmentation de la vidéo basé sur les keyframe en utilisent le framework hadoop, et d'autre part on présenté, l'environnement de travail et les outils utilisés pour implémenter notre projet, et ensuite nous avons présenté les vidéo que on à utilisé dans notre application. Enfin, nous avons conclus d'après l'exécution de l'algorithme des keyframe avec MapReduce est acceptable par rapport à d'autre méthodes, et elle est caractériser par le point de passage à l'échelle se que on ne trouve pas dans d'autre méthodes.

Conclusion général

L'objectif de ce travail était de comparer les calculs intensif des framework du cloud computing hadoop et spark deux système de traitement de grosse quantité de donnes, nous avons commencer, dans le premier chapitre, par définir les thèmes lies a ce travail, précisément, nous avons présenté les concepts de big data et de cloud computing.

Avant d'entamer la phase d'implémentation et expérimentation des solutions choisies pour l'évolution de calcul, nous avons jugé important de faire connaitre les concepts des framework de cloud computing, le deuxième chapitre était consacré a des notions préliminaires sur ces concepts, comme on à défini les caractéristiques d'hadoop et de spark. L'utilité d'hadoop était d'implémenter le système de fichier HDFS pour facilité l'exploitation de MapReduce.

Le troisième chapitre est consacré pour la segmentation de la vidéo basée sur MapReduce dans le cas de traitement d'image on représentant la segmentation par image clé (les keyframe), nous avons décret aussi l'environnement de travail et les versions des utiles utilisé pour développé notre application.

On travaillant avec une grande quantité de donnée ce qu'on appelle les big data le hadoop est le meilleur framwork qui répond a la multiplication de donnée et il offre une simplicité pour l'exécution de tache parallèle et pour garantir au futur une sclabilité avec notre application le MapReduce peut garantir ça, et moins chère à déployer et facile à réaliser avec le framework hadoop.

Nous estimons qu'avec la domination de l'internet dans presque tous les secteurs, industriels, medias, communication, et même familial, les technologies de big data sont aujourd'hui en plein essor. Dans les prochaines années, nous estimons que ces technologies seront de plus en plus utilisées pour répondre à de nouvelles problématiques pour la gestion de données.

1 Traitement d'image :

On désigne par " traitement d'images" toutes les techniques ayant pour but la modification des caractéristiques chromatiques des pixels des images bitmap.

Traitement d'images est souvent synonyme d'amélioration des images avec pour but l'obtention d'une plus grande lisibilité. Il n'y a pas création d'informations, mais mise en évidence de l'information pertinente déjà présente.

1.1 Définition de l'image numérique:

L'image se définirait comme une représentation ou une reproduction de quelque chose. Elle vient du latin « *imago* » qui désignait une sorte de masque moulé, à partir de cire d'abeille, sur le visage d'une personne morte afin d'en conserver les traits, comme d'un portrait, et d'en produire éventuellement un moulage.

Les images numériques sont le résultat de la conversion des données analogiques (par exemple une tension électrique) en données numériques (des 0 et des 1), moyennant un processus nommé « échantillonnage ». Une image numérique est une grille semblable à une mosaïque, composée d'éléments appelés « pixel » (picture element).

Chaque pixel est créé lorsqu'une couleur et une luminosité donnée sont capturées et enregistrées sous forme de nombre binaire (0 ou 1). Plus la fréquence d'échantillonage est élevée, plus grande sera la qualité.

Il existe aussi les images fixes et les images animées, une image fixe est l'ensemble de points lumineux appelés pixels. C'est le produit du nombre de points en largeur par celui en hauteur.

L'image animée est formée par une série d'image fixe défilée selon les préférences établies au préalable.

1.2 Types de l'image:

Il existe divers formats d'images qui se classent en deux grandes catégories, à savoir les images vectorielles et les images matricielles qu'on traite en bref dans cette section. Sur le web, seules les images matricielles existent sous les formats GIF, JPEG et PNG. Ils ont l'avantage d'être indépendants des plates formes matérielles et des logiciels (ils peuvent être échangés d'un système à un autre). De plus, ce sont des formats compressés permettant ainsi d'optimiser les temps de chargement, point fiable du web.

1.2.1 Images vectorielles:

Elle consiste en la représentation des données de l'image par des formules géométriques, autrement dit, c'est la mémorisation de la représentation des coordonnées des points caractéristiques des formes qui constituent l'image (carrés, rectangles, ellipses, cercles, courbes, etc).

1.2.2 Images matricielles (ou images bitmap)

La numérisation matricielle des images consiste a découper l'image en des milliers de points colorés, chaque point porte des informations de position et de couleur. Ce sont les pixels (contraction de l'anglais picture element). Ces points forment une matrice avec plusieurs dimensions ou chacune des dimensions représente une classe d'informations (spatiale, temporelle etc...). Exemple de format d'images bitmap: BMP, PCX, GIF, JPEG...

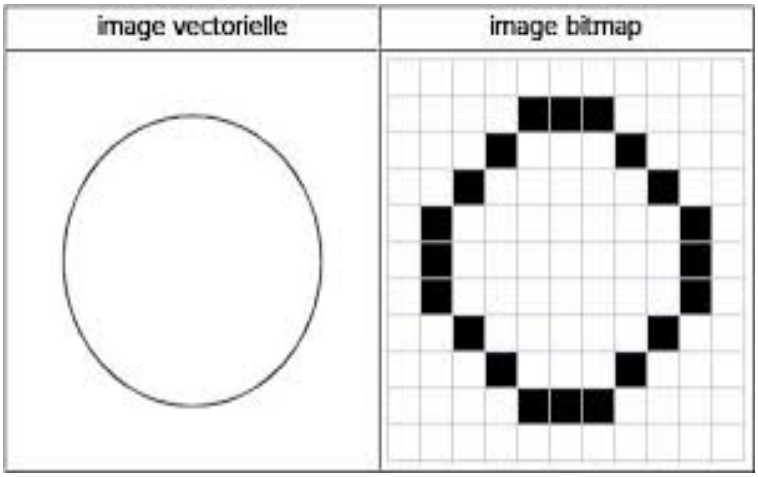

Figure1 : L'image vectorielle et l'image matricielle.

Les caractéristiques d'image numérique:

L'image est un ensemble structuré d'informations caractérisé par les paramètres suivants:

- **Volume :**

Le volume d'une image est tout simplement le nombre de bits de son code binaire. Le nombre de bits se calcule simplement en multipliant le nombre de pixels par le nombre de bits pour représenter un pixel

- **Dimension**

C'est la taille de l'image. Cette dernière se présente sous forme de matrice dont les éléments sont des valeurs numériques représentatives des intensités lumineuses (pixels). Le nombre de lignes de cette matrice multiplié par le nombre de colonnes nous donne le nombre total de pixels dans une image

- **Resolution**

C'est la clarté ou la finesse de détails atteinte par un moniteur ou une imprimante dans la production d'images. Sur les moniteurs d'ordinateurs, la résolution est exprimée en nombre de pixels par unité de mesure (pouce ou centimètre). On utilise aussi le mot résolution pour désigner le nombre total de pixels affichables horizontalement ou verticalement sur un moniteur; plus grand est ce nombre, meilleure est la résolution.

- **Bruit**

Un bruit (parasite) dans une image est considéré comme un phénomène de brusque variation de l'intensité d'un pixel par rapport à ses voisins, il provient de l'éclairage des dispositifs optiques et électroniques du capteur

- **Histogramme**

L'histogramme des niveaux de gris ou des couleurs d'une image est une fonction qui donne la fréquence d'apparition de chaque niveau de gris (couleur) dans l'image. Il permet de donner un grand nombre d'information sur la distribution des niveaux de gris (couleur)

et de voir entre quelles bornes est repartie la majorité des niveaux de gris (couleur) dans le cas d'une image trop claire ou d'une image trop foncée.

Il peut être utilisé pour améliorer la qualité d'une image (Rehaussement d'image) en introduisant quelques modifications, pour pouvoir extraire les informations utiles de celle-ci. Pour diminuer l'erreur de quantification, pour comparer deux images obtenues sous des éclairages différents, ou encore pour mesurer certaines propriétés sur une image, on modifie souvent l'histogramme correspondant

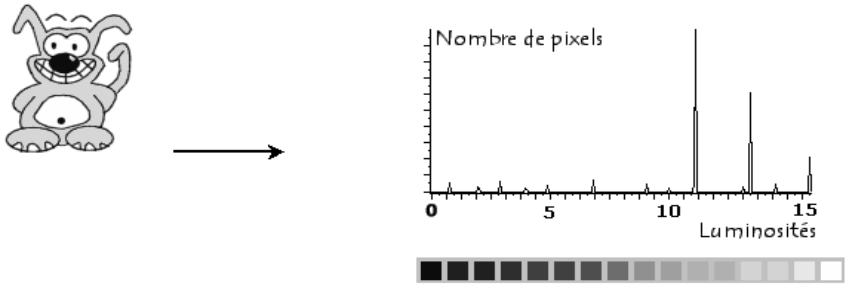

Figure2 :Histogramme et palette associés à une image

- **Luminance**

C'est le degré de luminosité des points de l'image. Elle est définie aussi comme étant le quotient de l'intensité lumineuse d'une surface par l'aire apparente de cette surface, pour un observateur lointain, le mot luminance est substitué au mot brillance, qui correspond à l'éclat d'un objet. Une bonne luminance se caracterise par:

- ✓ Des images lumineuses (brillantes);
- ✓ Un bon contraste : il faut éviter les images où la gamme de contraste tend vers le blanc ou le noir; ces images entraînent des pertes de détails dans les zones sombres ou lumineuses.
- ✓ L'absence de parasites.

- **Contraste**

C'est l'opposition marquée entre deux régions d'une image, plus précisément entre les régions sombres et les régions claires de cette image. Le contraste est défini en fonction des luminances de deux zones d'images. Si L1 et L2 sont les degrés de luminosité respectivement de deux zones voisines A1 et A2 d'une image, le contraste C

$$C = \frac{L1 - L2}{L1 + L2}$$

est défini par le rapport

1.3 Types de traitements d'image :

Les traitements généralement appliqués aux images sont deux types :

1.3.1 Amélioration ou retouche de l'image :

Le traitement de l'image informatique correspond à toutes les possibilités de travail, retouches ou modifications que l'on peut réaliser avec un ordinateur sur une image (enregistrée) quand elle est affichée à l'écran.

Les logiciels de traitement d'image en pixels permettent de réaliser des modifications sur ces images pour obtenir des effets précis : [11]

- ✓ recadrer une image
- ✓ modifier la taille (à l'écran ou à l'impression)
- ✓ retoucher des couleurs
- ✓ intégrer un texte
- ✓ changer un fond
- ✓ effacer une partie de l'image...

1.3.2 La Compression :

La qualité d'une image dépend de sa résolution (nombre de pixels) et du nombre de couleurs utilisées pour coder chaque pixel.

- ✓ En général IMAGE =GROS FICHIER.
- ✓ Le volume d'une image numérisée est égal à la quantité d'octets utilisés pour la coder.

Exemples

800 x 600 pixels représentent 480 000 points à mémoriser

C'est-à-dire 480 000 octets pour une image en noir et blanc avec 256 nuances de gris et 480 000 x 3 = 1,4 MO pour une image en couleur avec nuancier de 16 millions de couleurs. Pour diminuer le volume des fichiers, on peut soit:

- ✓ Modifier le nombre de couleurs
- ✓ Sauvegarder dans un format dit de compression.

Conséquences :

Lors du travail sur l'image, il ne faut pas faire de sauvegardes intermédiaires en format compressé car il y a alors une perte de la définition de l'image à chaque enregistrement. Attention avec certains logiciels (PSP), les résultats de la compression ne sont pas toujours visibles à l'écran (conservation de l'image de départ) mais seulement au redémarrage du logiciel ou lorsque l'image est réouverte.

1.3.2.1 Le but de la compression d'image :

Les images numériques sont des fichiers volumineux qui occupent beaucoup d'espace sur disque ou allongent considérablement les temps de transmission sur réseau.
La compression d'images permet de réduire énormément la taille des images.
Par exemple, pour une image 640 x 480 pixels en vraies couleurs :
Une seule image non compressée peut être enregistrée sur une disquette
1,4 Mo. Avec la compression d'images, une disquette peut contenir entre 20 et 50 images.

- ✓ Le temps de lecture non compressée est de 3 secondes sur un lecteur de cd-rom 8x. Avec compression d'image, cette durée peut être réduite à environ 1 seconde, y compris le temps de décompression.
- ✓ Le temps de transmission non compressée est de 3 minutes via un modem 28 Kbps. Après compression, la durée de transmission est réduite à une dizaine de secondes.

2 Définition d'une vidéo

La vidéo est une succession d'images animées défilant à une certaine cadence afin de créer une illusion de mouvement pour l'oeil humain. Elle peut être analogique (signal continu d'intensité de luminance) ou numérique (suite de trames ou images).

2.1 Types de vidéo:

On peut distinguer deux types de vidéo:

a- La vidéo analogique

La vidéo analogique est un flux continu de données analogiques destinées à être affichées sur un écran de télévision. Plusieurs normes : PAL, SECAM et NTSC.

b- **La vidéo numérique**

La vidéo numérique consiste à afficher une succession d'images numériques. Le débit nécessaire pour l'afficher correctement est : Débit (octets/seconde) = taille d'une image * nombre d'images par seconde

Exemple : Une image 640x480 à couleurs réelles (24 bits) (640x480x24/8) = 921600 Octets=900 Ko. Pour afficher correctement cette vidéo, il faut 30 images par seconde : 900 Ko * 30 = 27000 Mo/s

2.2 Caractéristique de la vidéo :

1. **Volume:** Les données vidéo sont trop volumineuses et nécessitent plus d'espace de stockage par rapport à l'image et au texte. En effet, une seconde de vidéo MPEG contient 25 ou 30 images.

2. **Hétérogénéité du contenu :** Les informations contenues dans un document vidéo proviennent de sources variées (audio, texte, image). Cette hétérogénéité cause souvent des problèmes au niveau segmentation et analyse du contenu. Pour le cas de documents images ou textuels le problème d'hétérogénéité ne se pose pas.

3. **La nature spatio-temporelle:** Les données textuelles sont ni spatiales ni temporelles. Dans les données images, la représentation spatiale est bien définie par contre il n'y a pas la notion du temps dans les images fixes. Dans un document vidéo l'aspect spatiotemporel est une de ses caractéristiques.

4. **L'expressivité sémantique:** Il est parfois facile d'interpréter le contenu d'un document en visualisant son contenu mais il est souvent encore mieux de disposer également d'une information sonore. Un document vidéo possède une expressivité sémantique beaucoup plus riche qu'un document textuel ou même qu'un document image.

5. **Durée:** Pour représenter le contenu d'une image fixe, la durée de présentation est variable. Dans un document vidéo, une image possède une durée de présentation bien définie qui dépend du type de vidéo (~1/30 seconde pour le cas de vidéo NTSC et 1/25 seconde pour le cas des vidéos PAL / SECAM).

6. **Variabilité de la qualité:** La qualité d'une vidéo dépend de la résolution dans laquelle elle est codée ainsi que du taux de compression utilisé pour son codage. Un même contenu peut être codé avec des résolutions très variables en fonction des capacités de stockage et/ou de transmission. Cela a un impact important sur la qualité des traitements qui peuvent être

effectués pour une indexation automatique. De ce point de vue, le cas de la vidéo est différent de celui du texte pour lequel le contenu et son interprétation ne sont pas ou sont peu affectés par le format dans le quel il est encodé.

7. **Son:** Un document video est caractiriser aussi par un son qu'on peux definir comme toute musique, parole, bruit que peut percevoir une oreille humaine, le son peut être enregistré, reproduit, modifie a l'aide de périphériques et de logiciels qui doivent accompagner un ordinateur multimédia pour faire un tel traitement.

2.3 Résolution temporelle d'une vidéo

La résolution temporelle d'une vidéo est définie par le nombre d'images défilant par seconde.

Afin d'éviter les désagréments dus aux papillotements, et prenant en considération les spécificités de l'œil humain qui garde une image environ 10 micro secondes au niveau de sa rétine ; les images doivent défiler à une certaine cadence:

- Environ 24 images/s pour un film de cinéma.
- 25 images/s pour la télévision européenne.
- Environ 30 images/s pour la télévision nord-américaine et japonaise.

2.4 Séquence d'image

Une séquence d'images est une succession d'images bidimensionnelles qui montre l'évolution temporelle d'une scène. La cadence est de 25 images par seconde, ce qui correspond au seuil à partir duquel l'œil humain perçoit la séquence comme un stimulus continu, grâce à la persistance rétinienne. Par la suite, nous appellerons « trame » ou « plan » chaque image bidimensionnelle correspondant à un instant donné de la séquence.

Figure 03 : Exemple d'une séquence d'images.

La considération de séquences d'images numériques ordonnées dans le temps introduit un champ spécifique d'investigation en analyse d'image, à savoir l'extraction d'informations de mouvement et plus généralement d'informations spatio-temporelles. Ce type d'études a émergé dans les années 70 à travers trois applications principales : le codage d'images de télévision et notamment la conception de codeurs avec compensation de mouvement [5-6], l'imagerie météorologique avec l'estimation de champs de vent via des traceurs nuageux, le suivi de cibles mobiles dans un contexte militaire.

2.4.1 Analyse du mouvement dans une séquence d'images:

L'analyse des séquences vidéo est l'un des sujets de recherche les plus tentants. Peut être que ceci découle de la richesse en informations que présente une scène ou encore de la variété des applications (médicales, sécuritaires, militaires, ...etc.). Les spécialistes du domaine ont tenté, et tentent encore d'imiter via le système caméra/calculateur le comportement humain en termes de vision. Pour y parvenir différents axes ont été développés. Ainsi apparaît le terme de " groupement spatio-temporel ".

C'est le terme qui englobe à la fois l'identification des objets et la détection du mouvement.

L'analyse de mouvement dans une séquence d'images doit être posée comme un problème d'estimation et de segmentation, puisqu'il s'agit d'appréhender des informations partielles observables et discontinues. C'est un problème particulièrement difficile, mais dont la résolution est cruciale pour la plupart des taches en analyse de scène dynamique. Ce problème peut se présenter en fait sous plusieurs variantes, suivant que l'objectif prioritaire se trouve être l'obtention d'une mesure dense ou paramétrique du mouvement, d'une partition de l'image en régions, ou l'extraction d'entités pertinentes

L'analyse du mouvement dans une séquence d'images repose soit sur la détection du mouvement soit sur la segmentation basée mouvement soit sur la poursuite de mouvement.

la bibliographie

[1] Introduction au calcul parallèle avec la bibliothèque MPI par Stéphaine DELAGE SANTACREU , novembre 2008.

[2] Mémoire conception et déploiement d'une grille de contrôle de processus physique présenté par SABATIER Fabrice, les 6 septembre 2004.

[3] Le cloud computing, septembre 2012.

[4] Introduction aux clusters et aux grilles de calcul, Sorina POP, 29 janvier 2009.

[5] Livre blanc : la révolution du big data…, par Thomas HILL, olivier Lebret.

[6] Ce qu'il faut savoir sur le cloud computing novembre 2012, réalise par DG Consultig communication.

[7] Introduction a MPI, christophe PERA , informatique scientifique pour le calcul école doctora le 2012/2013.

[8] Wikipédia.

[9] Cloud computing, licence professionnelle administration de système réseaux et application a bas de logiciels libres, IUT Nancy charlenagne, 2010.

[10] Livre blanc: Le « Cloud Computing » Édité par : T-Systems International GmbH, Hahnstraße 43d 60528 Frankfurt am Main Allemagne , avril 2009.

[11] le crabeinfo.net

[12] le grand manuel de big data .

[13] PAULI, Julien, PONçON, Guillaume, Zend Framework-bien développer en PHP. paris: eyrolles, 2009. 446p. (les cahiers du programmeur).

[14] POTENCIER, Fabien, hamon, hugo. symfony - mieux développer en php avec symfony 1.2 et doctrine. paris: eyrolles, 2009. 486p (les cahiers du programmeur).

[15] Big Data for Dummies, par Wiley Brand.

[16] Hadoop The Definitive Guide.3rd.Edition, May 2012.Edition OReilly.

[17] Mekideche Mounir, Conception et implémentation d'un moteur de recherche à base d'une architecture Hadoop (Big Data), Avril 2015.

[18]Benjamin Renaut, Hadoop/Big Data, Université de Nice Sophia-Antipolis, 114p, 2013-2014.

[19] Charley CLAIRMONT, Introduction à HDFS Hadoop Distributed File System, HUG France SL2013, 20p, Mai 2013.

[20]Jonathan Lejeune, Hadoop : une plate-forme d'exécution de programme Map-reduce, École des Mines de Nantes, 83p, Janvier 2015.

[21]M.CORINUS, T.Derey, J.Marguerie, W.Techer, N.Vic, Rapport d'étude sur le Big Data, SRS Day, 54p, 2012.

[22]http://blog.ippon.fr/2013/05/14/big-data-la-jungle-des-differentes-distributionsopen-source-hadoop/, consulté le15/04/2015.

[23] http://mbaron.developpez.com/tutoriels/bigdata/hadoop/introduction-hdfs-mapreduce/, consulté le 27/04/ 2015.

[24] http://fr.wikipedia.org/wiki/Hadoop, consulté le 10/04/2015.

[20]http://blog.ippon.fr/2013/05/14/big-data-la-jungle-des-differentes-distributionsopen-source-hadoop/, consulté le 10/04/2015.

[21]http://hadoop.apache.org/releases.html

[22]https://spark.apache.org/news/spark-1-6-1-released.html

[23]https://www.java.com/fr/download/faq/helpful_concepts.xml

[24]https://en.wikipedia.org/wiki/Scala_(programming_language)

[25]https://www.infoq.com/fr/articles/apache-spark-introduction

[26]https://translate.google.com/translate?hl=fr&sl=en&u=http://spark.apache.org/docs/latest/cluster-overview.html&prev=search

[27] Key Frame Extraction from MPEG Video Stream College of Information Science & Technology, Qingdao Univ. of Science & Technology,266061,P. R. China,

[28] https://open-video.org/index.php

[29] https://doc.ubuntu-fr.org/trusty

[30] http://clipartist.info/.

[31] https://trac.ffmpeg.org/wiki/Create a thumbnail image every X seconds of the video

[32] http://www.lemondeinformatique.fr/actualites/lire-hadoop-vs-spark-apache-5-choses-a-savoir-63271.html

[33] http://decrypt.ysance.com/2014/09/big-data-avec-hadoop-comparatif-hive-pig-impala-shark-spark/

[34] http://www.datamation.com/data-center/hadoop-vs.-spark-the-new-age-of-big-data.html

[35] http://www-nlpir.nist.gov/projects/tvpubs/tvpapers03/ramonlull.paper.pdf

[36] http://homepages.mcs.vuw.ac.nz/~mengjie/papers/cspsc97.pdf