Introduction aux mégadonnées (Big Data)

Bernard ESPINASSE
Professeur à Aix-Marseille Université (AMU)
Ecole Polytechnique Universitaire de Marseille

Septembre 2021

1. **Introduction aux mégadonnées** (Big Data) **:**
   Modèle des 3V étendu au 5V ; Mégadonnées et informatique décisionnelle ; Usage des mégadonnées …
2. **Exploitation des Data Centers**
   Data Centers, Sharding & consistent hashing, Map-Reduce, MVCC & vector-clock, HADOOP …
3. **Stockage & gestion des mégadonnées** (Big Data Engineering)
   Limites des bases de données relationnelles et cloud computing ; Nouvelles solutions : BD NoSQL, Hadoop, Map Reduce, …
4. **Analyse des mégadonnées** (Big Data Analytics)
   Tâches et techniques principales
   Différents types d'analytiques (données, flots, textes, Web, …)

Plan (BIG DATA)

Bibliographie

Documents :
- C. Strauch, « NoSQL databases », Lecture Notes, Stuttgart Media University, 2011.
- S. K. Gajendran, « A Survey on NoSQL Databases ».
- S. Abiteboul, I. Manolescu, P. Rigaux, M-C Rousset, P. Senellart, « Web Data Management », Cambridge University Press 2011 (en ligne : http://webdam.inria.fr/Jorge/?action=chapters).
- J. Dean and S. Ghemawat, « MapReduce: Simplified Data Processing on Large Clusters », OSDI 2004.
- B. Espinasse, P. Bellot, « Introduction au Big-Data: opportunité, stockage et analyse des mégadonnées », in Dossiers Techniques de l'Ingénieur (DTI), Ref. H6040, 2017.

Présentations :
- F. Duchateau, « Les SGBD Non-relationnels », Univ. Lyon 1, 2014.
- P. Selmer, « NOSQL stores and Data analytics tools », Advances in Data Management, 2012.
- A.-C. Caron, « NoSQL », Université de Lille 1.
- M. Jaffré, P. Rauzy, « MapReduce », ENS, 2010.
- K. Tannir, « MapReduce : Algorithme de parallélisations des traitements », 2011,

Livres :
- P. Delort, « Le Big Data ». Presses Universitaires de France, 2015.
- P. Lemberger, M. Batty, M. Morel, J.L. Raffeëlli, « Big Data et Machine Learning », Dunod, 2015.
- P. Lacomme, S. Aridhi, R. Phan, « Bases de données NoSQL et Bog Data », Ellipses, 2014.
- R. Bruchez, « Bases de données NoSQL », Ellipses, 2013.
- …

1. Introduction au Big Data

- **Modèle des 3V étendu au 5V**
- **Mégadonnées et informatique décisionnelle**
- **Usage des mégadonnées**

Contexte

- Actuellement nous produisons **annuellement** une masse de données estimée à près de **3 trillions** (3 millions de millions) d'octets de données.
- On estime ainsi que **90% des données** dans le monde ont été créées au cours des **2 années précédentes** [1].
- La **masse totale des données** crées et copiées de par le monde pour 2011 était de **1,8 Zétabytes**, et s'accroit avec un **facteur de 9 tous les 5 ans** [2].
- Tous les secteurs sont touchés, tant **scientifiques qu'économiques**, ainsi que le développement des **applications Web** et les **réseaux sociaux** [3].
- Dans ce contexte, est apparu le terme « **Big Data** ».
- L'origine de ce terme anglo-saxon, littéralement « grosses données », est controversée, et sa traduction française officielle recommandée est "**mégadonnées**", même si parfois on parle de "données massives".

[1] BRASSEUR C. Enjeux et usages du big data. Technologies, méthodes et mises en œuvre, Paris, Lavoisier, p. 30. (2013).
[2] IDC-2011, GANTZ J., REINSEL D. Extracting Value from Chaos. IDC iView, pp. 1–12. (2011).
[2] HELBING D., POURNARAS, E. Build Digital Democracy : Open Sharing of Data that are Collected with Smart Devices would Empower Citizens and Create Jobs. Nature, Vol. 527, Nov. 2015, Macmillan Publishers. (2015).

Objectif de la présentation

- Cerner ce terme **Big Data** ou **mégadonnées**, et d'introduire différentes grandes ***méthodes*** et ***techniques*** qui s'y rattachent (notamment liées aux **Data Centers**), ainsi que leur opportunités
- On s'intéressera ici à 2 grandes problématiques :
  - **leur stockage et leur gestion / *Big Data Engineering***
    - les techniques traditionnelles de stockage de type bases de données relationnelles ne permettant pas de stocker de telles quantité de données
    - nouvelles solutions …
  - **leur analyse / *Big Data Analytics***
    - applications à visée analytique (analyses) traitant des données pour en tirer du sens.
    - généralement appelées « Big Analytics », ou « Analytique » ou encore « broyage de données », reposant généralement sur des méthodes *d'apprentissage* et de *calcul distribué*.

Caractérisation des mégadonnées

- Généralement faite selon 3 « V » : **Volume**, **Variété** et **Vélocité**
- Auxquels s'ajoutent 2 autres « V » complémentaires : **Valeur** et **Véracité/Validité**

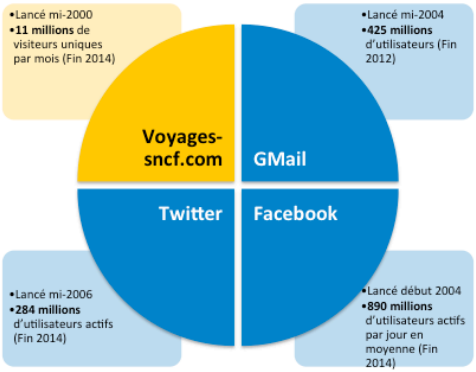
Caractérisation des mégadonnées : Volume (1)

- Fait référence à la **quantité d'informations**, trop volumineuses pour être acquises, stockées, traitées, analysées et diffusées par des outils standards,
- Peut s'interpréter comme le traitement d'objets informationnels de ***grande taille*** ou de ***grandes collections d'objets***,
- Le développement de l'**IoT** (Internet des objets) et la généralisation de la **géolocalisation** ou de l'analytique ont engendré une explosion du volume de données collectées,
- On estime qu'en 2020, **43 trillions de gigabytes** seront générés, soit 300 fois plus qu'en 2002.

(1 trillion = 10^{18} = un milliard de milliards de bytes)

Caractérisation des mégadonnées : Volume (2)

- Voyagiste **voyages-sncf.fr** :
  - En moyenne 360 000 par jour (11 millions par mois)
  - BD relationnelle respectant les propriétés ACID



- **Changement d'échelle** : Facebook, GMail et Twitter d'une quinzaine d'année d'existence, plusieurs centaines de millions de visiteurs par jour !

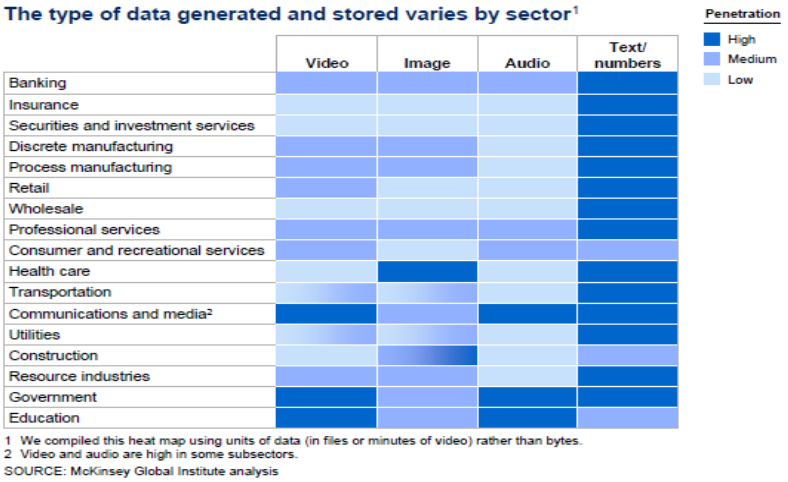
Caractérisation des mégadonnées : Variété

- Fait référence à **l'hétérogénéité** des *formats*, *types*, et *qualité* des informations,
- Est lié au fait que ces données peuvent présenter des ***formes complexes*** du fait qu'elles trouvent leurs origines dans :
  - des **capteurs** divers et variés (température, vitesse du vent, hygrométrie, tours/mn, luminosité...),
  - des **messages échangés** (e-mails, médias sociaux, échanges d'images, de vidéos, musique),
  - des **textes**, des **publications** en ligne (bibliothèques numériques, **sites** web, blogs, ...),
  - **enregistrements** de transactions d'achats, des **plans** numérisés, des annuaires, des informations issues des **téléphones mobiles**, etc.

⇨ Usage de technologies nouvelles pour **analyser** et **recouper** les **données non structurées** (mails, photos, conversations…) représentant au moins **80 % des informations collectées.**

Caractérisation des mégadonnées : Variété

Type de données générées et stockées par secteurs d'activité (Mc Kinsey) :



Caractérisation des mégadonnées : Vélocité

- Fait référence à l'aspect **dynamique** et/ou **temporel** des données, à leur **délai d'actualisation** et **d'analyse** ,
- les données ne sont plus traitées, analysées, en différé, mais en **temps réel** ou **quasi réel,**
- elles sont produites en **flots continus**, sur lesquels des ***décisions en temps réel*** peuvent être prises,
- ce sont les données notamment issues de **capteurs**, nécessitant un traitement rapide pour une **réaction en temps réel**,
- dans le cas de telles données de grande vélocité engendrant des volumes très importants, il n'est plus possible de les stocker en l'état, mais seulement de les **analyser en flux** (streaming) voire les **résumer**.

Caractérisation des mégadonnées : Valeur

- Associé à **l'usage** qui peut être fait de ces mégadonnées, de leur analyse, notamment d'un point de vue ***économique***.
- L'analyse de ces mégadonnées demande une certaine **expertise** tant liée à des méthodes et techniques en statistique, en analyse de données, que de domaine pour l'interprétation de ces analyses.

En 2013, McKinsey Global Institute :
- dans les seuls Etats Unis, il ***manquerait environ 150 000 personnes avec une expertise en analyse*** de Big Data,
- le système de **santé américain** pourrait créer **300 milliards de dollars de valeur par an** dont deux tiers correspondrait à des **réductions de coût d'environ 8%**.

⇨ Les termes de « **Data Scientist** » et de « **Data Science** » sont liés à cette expertise recherchée et à cette nouvelle **discipline émergente**.

Caractérisation des mégadonnées : Véracité ou validité

- Fait référence à la **qualité** des données et/ou aux **problèmes éthiques** liés à leur **utilisation**,
- comprend les problèmes de **valeurs aberrantes** ou **manquantes** (ces problèmes pouvant être résolus par le volume de données),
- fait référence aussi au niveau de la **confiance** que l'on peut avoir dans les données.

⇨ S'il existe des critères permettant de **qualifier la qualité** des données, dans le cas de Big Data, la **vérification de la qualité est rendue difficile** voire **impossible** du fait du *volume*, de la *variété* et de la *vélocité* spécifiques au Big Data.

Définition des mégadonnées

Mégadonnées (Big Data) =

- des données qui :
  - sont trop volumineuses
  - *ou* ayant une arrivée trop rapide
  - *ou* une variété trop grande
- pour :
  - permettre de les ranger directement dans des ***bases de données traditionnelles*** (Relationnelles)
  - ou de les traiter par les ***algorithmes actuels*** » [4].
  - **Masses de données très importantes : Informatique Décisionnelle** Versus **Mégadonnées** ?

[4] MOTHE J., PITARCH Y., GAUSSIER E. Big Data: Le cas des systèmes d'information, Revue Ingénierie des Systèmes d'Information, Hermès Editeur, Vol. 19/3, (2014).

Mégadonnées et Informatique Décisionnelle (1)

Informatique Décisionnelle et les **Mégadonnées** ont vocation à **stocker** et **analyser** des **masses de données très importantes**

L'informatique décisionnelle (ID) - Business Intelligence (BI) - est apparue dans les **années 1990** en management et en informatique :

- L'ID appréhende des données volumineuses principalement ***historisée*** et ***orientées sujet***, ***stockées*** dans des **entrepôts de données**.
- Caractéristiques des données traitées :
  - ***multidimensionnelles***, ***fortement structurées*** selon un modèle défini, de ***forte densité*** en information, et principalement ***numériques***,
  - stockées dans des ***entrepôts de données*** ou dans des ***cubes***.

[5] DAVENPORT, T. H. Competing on Analytics, Harvard Business Review (84:1), pp. 98-107. (2006).

Mégadonnées et Informatique Décisionnelle (2)

Business Analytics introduit fin des années 2000, **composante analytique** clé dans la ID [5] permettant des **analyses** :

- principalement réalisées par des opérateurs d'analyse en ligne OLAP (On Line Analysis Processing) sur de cubes extraits de ces entrepôts, ou par des techniques de **fouille de données** (**Data Mining**),
- reposant principalement sur des **agrégations** et grâce à des opérateurs OLAP spécifiques, ou méthodes de fouille de données (datamining),
- permettant de **mesurer des phénomènes**, notamment pour **détecter des tendances**,
- s'appuient principalement sur la ***statistique descriptive***

Statistique descriptive (ou exploratoire) :

- décrire des données à travers leur ***présentation*** (la plus synthétique possible), leur *représentation graphique*, et le calcul de *résumés numériques*.
- intéressante mais ***coûteuse*** car reposant sur des enquêtes portant sur un *nombre d'individus important*
- ***Ne fait PAS fait appel à des modèles probabilistes***

Mégadonnées et informatique décisionnelle (3)

Les mégadonnées :

- Concernent des données bien plus volumineuses que celles traitées par l'ID, **structurées ou non structurées**, et de **faible densité** en information.
- utilisent les ***statistiques inférentielles*** *(*ou *inductive)*, pour **inférer** des lois (régressions....)

Statistiques inférentielles :

- utilisent la théorie des **probabilités** pour restreindre le nombre d'individus en faisant des sondages sur des **échantillons**,
- précisent **un phénomène** sur une population globale, par observation sur une partie restreinte de cette population (échantillon),
- permettent **d'induire (inférer) du particulier au général** avec un objectif principalement *explicatif*, par des modèles et d'hypothèses probabilistes,
- donnent aux mégadonnées des **capacités prédictives** [5],
- permettent de faire intervenir une part de **hasard** (théorie des tests d'hypothèses).

[5] CATTELL R. (2011). Scalable SQL and NoSQL data stores. ACM SIGMOD Record, 39(4), pp. 12- 27.

Mégadonnées et Informatique Décisionnelle (4)

- Loin de les opposer, Informatique décisionnelle et mégadonnées peuvent s'enrichir l'un l'autre :
  - **l'Informatique Décisionnelle** apporte aux mégadonnées ses **méthodes** de **conception d'entrepôts** et **d'analyse**,
  - les **mégadonnées** apportent notamment ses **architectures de stockage distribuées** et ses **analyses à larges échelles** basées sur les *statistiques inférentielles*.
- Pour conclure [6] :
  - **l'Informatique Décisionelle** est basée sur un ***modèle défini*** du monde,
  - alors que les **mégadonnées** visent à ce que les mathématiques (statistique) pour ***trouver un modèle dans les données***.

[9] DELORT, P. (2015). Le Big Data. Presses Universitaires de France.

Usage des mégadonnées

- Les **Mégadonnées** ou Big Data sont dès à présent utilisées dans TOUS les secteurs d'activités, tant ***scientifiques***, ***techniques*** que ***socio-économiques***,

« ... depuis les données récupérées de l'exploitation de moteurs d'avion permettant de mieux maintenir ou concevoir ces derniers, …

… jusqu'aux données spécifiant nos relations sur les réseaux sociaux pouvant être utilisées par les banques pour estimer la qualité de notre crédit ... » [6].

[6] DELORT, P. (2015). Le Big Data. Presses Universitaires de France.

2. Exploitation des Data Centers

- **Data Centers et le Cloud Computing**
- **Sharding et consistent hashing**
- **Map Reduce,**
- **MVCC et Vector-clock**
- **HADOOP**

Data Centers (0)

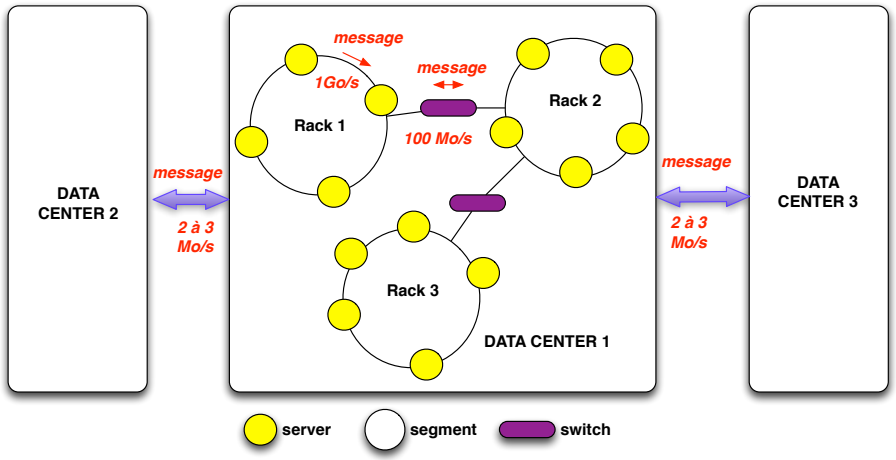
Permet de stocker des mégadonnées

Yahoo ! data centers

Data Centers : organisation et communications (1)

- Utilisent des **LANs** (Local Area Networks) avec **3 niveaux de communication** :
  - Les **serveurs** sont regroupés en « **Racks** » : *liaison réseau rapide, environ 1Go/sec*
  - un « **Data center** » consiste en un grand nombre de « **racks** », interconnectés par des **routeurs** (switches) : *liaison à 100 Mo/sec*
  - entre différents « **Data centers** » : *communication internet à 2-3 Mo/sec*
- Les **serveurs** communiquent par *envoi de messages*, ils ne partagent pas de disque ni de ressource de traitement = ***architecture « shared nothing »***

Data Centers : organisation et communications (2)



Data Centers : dimensions

- **Ex. : Data center de Google (début 2010) :**
  - Un « **Data center** » Google contient entre **100 et 200** « **Racks** », chacun contenant **40 serveurs**
  - environ **5000 serveurs par** « **Data-center** » pour un total de **1 millions de serveurs** (estimation d'après la consommation électrique) ;
  - Estimation Garner de **2,5 millions de servers pour Google en 2016** …
- **Ex. : Data center de Facebook (2010) :**
  - **2500 cpu** (serveurs)
  - **1 PetaByte d'espace disque** (= milleTerabytes = 1 million de Gigabytes)
  - Plus de **250 Gigabytes de données compressées** (plus de 2 Terabytes non compressés)

Gestion distribuée des mégadonnées

Les mégadonnées nécessitent un stockage et une gestion des données **fortement distribuées** sur des clusters (serveurs, data centers, …)

On distingue 2 stratégies de traitement des mégadonnées :

- **Par distribution des traitements (« *scaling* » *des traitements*) :**
  - **on distribue ces traitements** sur un nombre de machines important afin d'absorber des charges très importantes
  - on **envoie les données** aux endroits appropriés, et on enchaîne les exécutions distantes (scénario type **Workflow** implémenté avec des **Web services**)

⇨ **Par distribution des données (« *scaling* » *des données*) :**

- on **distribue les données** sur un nombre important de serveurs afin de stocker de très grands volumes de données – (***sharding & consistant hashing***)
- on « **pousse** » **les programmes vers ces serveurs** (plus efficace de transférer un petit programme sur le réseau plutôt qu'un grand volume de données – (modèle de programmation parallèle ***MapReduce***).

Stockage des mégadonnées : le Cloud Computing

Le **cloud** :

- une architecture composée de *matériels de stockage - data centers*, de *réseau* et de *logiciels* **fournissant des services** que des utilisateurs peuvent exploiter depuis n'importe où (cloud computing).
- un support de stockage des mégadonnées, si les besoins de stockage s'accroissent, de nouveaux serveurs sont déployés dans cette architecture de façon transparente pour l'utilisateur.

Pour le stockage des mégadonnées sur le cloud sont généralement utilisés :

- le modèle de programmation parallèle « **Map Reduce** »
- le quadriciel libre « **Hadoop** » le mettant en œuvre.

- Pour la gestion des mégadonnées sur le cloud sont utilisés des systèmes de gestion de base de données (SGBD) spécifiques de type **NoSQL**

Exploitations des Data Centers

Principales spécificités :

- Le « **Sharding** » : un partitionnement des données sur plusieurs serveurs,
- Le « **Consistent hashing**» : un partitionnement des données sur plusieurs serveurs eux-mêmes partitionnés sur un segment,
- Le « **Map Reduce** » : un modèle de programmation parallèle permettant de paralléliser tout un ensemble de tâches à effectuer sur un ensemble de données,
- Le « **MVCC** » : pour « Contrôle de Concurrence Multi-Version », est un mécanisme permettant d'assurer le contrôle de concurrence,
- Le « **Vector-Clock**» : ou horloges vectorielles permet des mises à jours concurentes en datant les données par des vecteurs d'horloge.

Exploitations des Data Centers: Sharding

Le « Sharding » : un ensemble de techniques qui permet de répartir les données sur plusieurs machines pour assurer la scalabilité de l'architecture.

Mécanisme de partitionnement horizontal (par tuples) des données dans lequel les objets-données sont **stockées** sur des **nœuds serveurs différents** en **fonction d'une clé** (ex : fonction de hachage)

*Ex : **partition = hash(o) mod n** avec o = objet-données à placer et n= nb de nœuds serveur disponibles*

- les données peuvent être aussi partitionnées afin que :
  - les **objets-données** accédés ou mis à jour **en même temps**, résident **sur le même nœud**
  - la **charge** soit **uniformément répartie** entre les **nœuds**
  - pour cela des **objet-données** peuvent être **répliquées**
- certains systèmes utilisent aussi un **partitionnement vertical** (par colonnes) dans lequel des parties d'un enregistrement sont stockées sur différents serveurs.

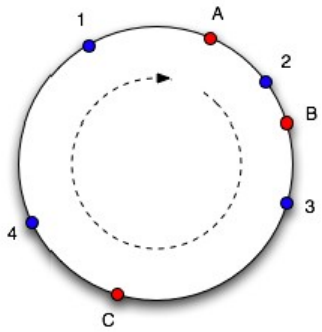
Exploitations des Data Centers: Consistent hashing (1)

« Consistent hashing » :

Mécanisme de partitionnement (horizontal) dans lequel les *objet-données* sont *stockés* sur des *nœuds-serveurs différents* en utilisant la **même fonction de hachage** à la fois pour le *hachage des objets* et le *hachage des nœuds :*

- nœuds et objets sont associés par une même fonction de hachage, et imaginés être placés sur un anneau (rack/cluster de serveurs)
  Ex : A, B, C sont des nœuds (serveurs) et 1, 2, 3, 4 sont des objets.
- un ***objet*** est associé au ***premier nœud*** serveur dans le ***sens horaire*** :
  Ex : les objets 4 et 1 sont associés au nœud A ; 2 à B ; 3 à C.

D'après Strauch



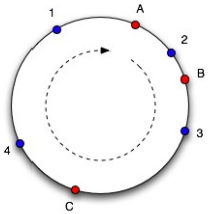
Exploitations des Data Centers: Consistent hashing (2)

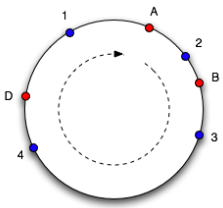
Suite …

- quand un ***nœud quitte*** l'anneau, les objets qui lui sont liés sont alors associés à leur ***nœud adjacent dans le sens horaire*** :
  Ex : le nœud C quitte l'anneau, 3 est alors associé avec 4 et 1 au nœud A

- quand un ***nœud entre*** dans l'anneau, il est placé (haché) sur l'anneau et ***des objets lui seront associés*** selon sa place dans l'anneau :
  Ex : le nœud D entre dans l'anneau, les objets 3 et 4 lui sont associés

D'après Strauch





Exploitations des Data Centers: Map Reduce (1)

Map-Reduce :

- **Modèle de programmation parallèle** (framework de calcul distribué) pour le traitement de **grands ensembles de données**
- **développé par Google** pour le traitement de ***gros volumes de données*** en ***environnement distribué*** :
  - permet de *répartir la charge* sur un *grand nb de serveurs* (cluster)
  - *abstraction quasi-totale de l'infrastructure matérielle* **:** gère entièrement, de façon transparente le *cluster*, la *distribution de données*, la *répartition de la charge*, et la *tolérance aux pannes*
  - *ajouter* des machines *augmente la performance* (plug & play, scalable-friendly)
- la **librairie MapReduce** existe dans plusieurs langages (C++, C#, Erlang, Java, Python, Ruby…)

Exploitations des Data Centers: Map Reduce (2)

Divers usages de Map-Reduce :

- **Utilisé par les grands acteurs du Web** notamment pour : *construire les index* (Google Search), *détection de spam* (Yahoo), *Data Mining* (Facebook) …
- **Mais aussi pour** :
  - De *l'analyse d'images astronomique*, de la *bioinformatique*, de la *simulation métrologique*, de *l'apprentissage automatique* (Machine Learning), des *statistiques*, …
  - *le calcul de la taille de plusieurs milliers de documents*
  - *trouver le nb d'occurrences d'un pattern* dans un très grand volume de données
  - *classifier de très grands volumes de données* provenant par exemple de paniers d'achats de clients (Data Mining)
  - …

Exploitations des Data Centers: Map Reduce (3)

Usage de MapReduce en gestion de données :

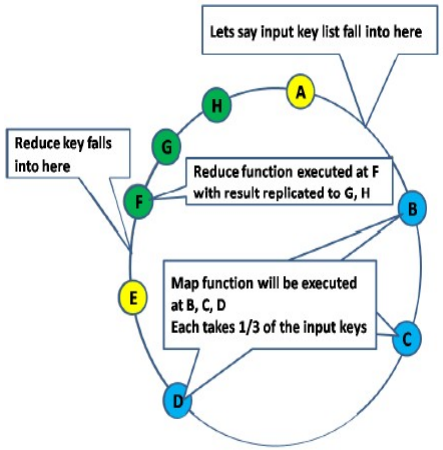
- **Principales opérations** à faire sur des grands ensembles de données :
  1. **Itérer** sur un grand nombre d'enregistrements
  2. **Extraire** quelque chose ayant un intérêt de chacun de ces enregistrements
  3. **Regrouper** et **trier** les résultats intermédiaires
  4. **Agréger** tous ces résultats ensemble
  5. **Générer** le résultat final

- Dans le **modèle de programmation MapReduce**, le développeur implémente 2 fonctions : la fonction **Map** et la fonction **Reduce**
  - *opérations 1 et 2* : **fonction Map traite une paire clé/valeur** et génère un ensemble de paires de clés intermédiaires/valeurs
  - *opérations 3, 4 et 5* : **fonction Reduce fusionne toutes les valeurs intermédiaires** associées avec la même clé intermédiaire.

Exploitations des Data Centers: Map Reduce (4)

- Appliqué à la BD, MapReduce traite un **ensemble de clés** en appliquant les fonctions **Map** et **Reduce** aux **nœuds** de stockage qui appliquent localement la fonction **Map** aux **clés** qui doivent être traitées et qu'ils possèdent
- les ***résultats intermédiaires*** peuvent être ***hachés comme des données ordinaires et traitées par les nœuds suivants dans le sens horaire***, qui appliquent la fonction Reduce aux résultats intermédiaires et produisent les résultats finaux
- du fait du ***hachage des résultats intermédiaires***, aucun **coordinateur** est utilisé pour diriger les nœuds de traitement vers ces **résultats**

D'après Strauch



Exploitations des Data Centers: Map Reduce (5)

Fonctionnement de MapReduce :

- Traite des *données en entrée* pour en fournir des *résultats en sortie*
- Traitement constitué de *plusieurs tâches* dont chacune est *traitée indépendamment,* puis leurs *résultats sont combinés*
- On distingue 3 opérations majeures :
  - **Split** correspond à une opération de *découpage*
  - **Compute** correspond à une opération de *calcul*
  - **Join** correspond à l'opération de *regroupement* du résultat

- Dans le modèle de programmation MapReduce, le développeur implémente 2 fonctions :
  - la fonction **Map**
  - la fonction **Reduce**.

Exploitations des Data Centers: Map Reduce (6)

Fonction Map : prend en entrée un ensemble de « Clé, Valeurs » et retourne une liste intermédiaire de « Clé1, Valeur1 » :

```
Map(key,value) -> list(key1,value1)
```

Fonction Reduce : prend en entrée une liste intermédiaire de « Clé1, Valeur1 » et fournit en sortie une ensemble de « Clé1, Valeur2 » :

```
Reduce(key1,list(value1)) -> value2
```

L'algorithme MapReduce s'exécute en **5 phases** :

1. La phase **Initialisation**
2. La phase **Map**
3. La phase **regroupement** (**Shuffle**)
4. La phase de **Tri**
5. La phase **Reduce**

Exploitations des Data Centers: Map Reduce (7)

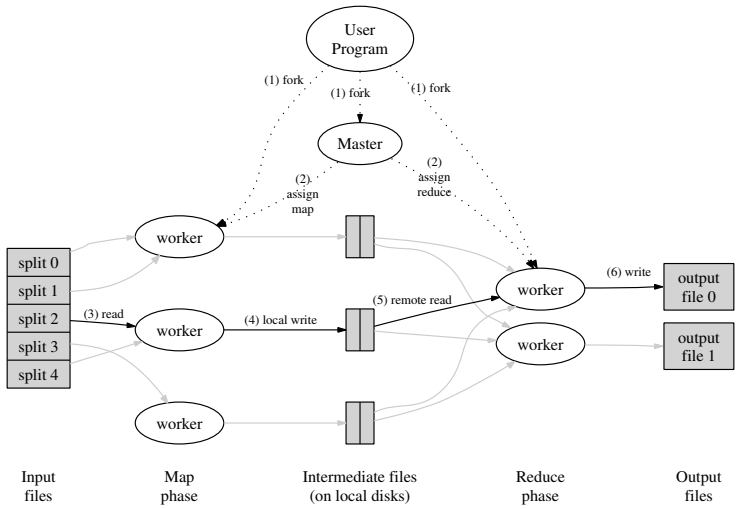
Fonctionnement de MapReduce :

- Lorsque l'application MapReduce est lancée, elle crée un composant « ***Master*** » responsable de la distribution des données et de la coordination de l'exécution de différentes unités de travail ou « ***Workers*** ».
- Le ***Master*** attribue aux ***Workers*** les tâches ***Map*** et ***Reduce***
- Un ***Worker*** possède **3 états**:
  - ***idle*** : il est *disponible* pour un nouveau traitement
  - ***in-progress*** : un *traitement est en cours* d'exécution
  - ***completed*** : il a *fini un traitement*, il en *informe* alors le *Master* de la taille, de la localisation de ses fichiers intermédiaires
- Le ***Master*** gère la synchronisation, la réorganisation, le tri et le regroupement des données :

  lorsqu'un ***Worker*** de type ***Map*** a fini son traitement, le ***Master*** regroupe, trie et renvoie le résultat à un ***Worker*** de type ***Reduce***

Exploitations des Data Centers: Map Reduce (8)

Soit :



Exploitations des Data Centers: Map Reduce (9)

Exemple de BD distribuée :

- BD de 1To de données
- 1800 serveurs
- les données sont découpées en 15000 morceaux d'environ 64Mo
- un seul serveur effectue la réduction (afin d'avoir les résultats dans un seul fichier)

Performances de MapReduce sur cette BD :

- démarrage relativement lent dû au temps de propagation du programme (±1mn).
- les Maps sont tous finis au bout d'environ 80s
- l'opération se termine en 150s.
- on atteint un pic de lecture de 30Go/s

Exploitations des Data Centers: Map Reduce (10)

Exemple :

On souhaite **calculer le nombre de mots dans un document** :

La fonction ***Map, « mapFunc »*** est alors :

```
mapFunc(String key, String value):
// key: id du document;
// value: contenu du document
for each word w in value
EmitIntermediate (w, 1)
```

mapFunc a 2 arguments :

- la *clé (**key**) identifiant le document* dont on souhaite compter les mots
- le *contenu du document (**value**)*

l'ensemble des valeurs est parcouru par une boucle « for each » :

- pour chaque mot identifié, on appelle la méthode ***EmitIntermediate***
- elle place dans une zone intermédiaire le **mot** et la valeur **1** (correspondant à une occurrence).

Exploitations des Data Centers: Map Reduce (11)

La fonction ***Reduce « reduceFunc »*** est :

```
reduceFunc(String key, Iterator values)
// key: un mot;  values: une liste de valeurs
result = 0
for each count v in values
result += v
EmitResult(result)
```

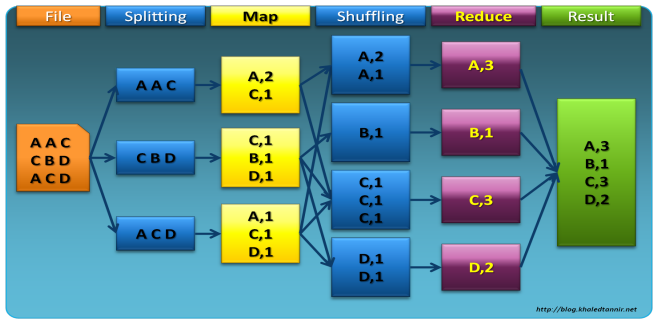
- **reduceFunc** a 2 arguments :
  - la ***clé*** (***key***) correspond à un *mot identifié par la fonction **mapFunc***
  - une ***liste de valeurs*** contenant le *nb d'occurrences de ce mot*
- la liste des valeurs correspond à l'ensemble des paires (mot, count) mises en zones intermédiaires par la fonction ***mapFunc***
- pour comptabiliser tous les mots du document, on initialise la variable ***result*** à 0, puis on itére sur l'ensemble des valeurs de la liste, puis chaque valeur est rajoutée à la variable ***result***
- à la fin de l'itération, la variable ***result*** contient le total du nb d'occurrence et le résultat est renvoyé par la fonction ***EmitResult***

Exploitations des Data Centers: Map Reduce (12)

Soit un fichier document contenant 3 lignes composée chacune de 3 mots parmi les mots {A, B, C, D}.

Il s'agit de compter tous les mots contenus dans ce fichier.

Le traitement MapReduce effectué pour compter les mots du document est :



Exploitations des Data Centers: Map Reduce (13)

Les étapes réalisées sont :

- **L'étape File** : on lit le fichier document en entrée et on initialise les différents « *Workers MapReduce* »
- **L'étape Splitting** : on distribue les données à traiter sur les différents noeuds du cluster de traitement
- **L'étape Map** : on effectue le compte de chacune des lettres et ceci en local sur chaque noeud du cluster de traitement
- **L'étape Suffling** : on regroupe toutes les lettres ainsi que leur compte à partir de tous les noeuds de traitement
- **L'étape Reduce** : on effectue le cumule de toutes les valeurs de chaque lettre
- **L'étape Result** : on agrège tous les résultats des différentes étapes Reduce et on retourne le résultat final.

Exploitations des Data Centers: MVCC

Contrôle de Concurrence Multi-Version (MVCC) :

- Méthode de **contrôle de concurrence** couramment utilisée par les SGBD pour gérer des accès simultanés à la base de données avec mises à jour
- Dans une **BD NoSQL**, la gestion des mises à jour est faite :
  - non pas en *supprimant* une fraction contenant les données avant modification et en la *remplaçant* par une fraction contenant les données modifiées
  - mais en **marquant** les anciennes données comme **obsolètes** et en **ajoutant** une **nouvelle version** contenant les nouvelles données
  - il existe ainsi **plusieurs versions enregistrées**, une seule est la plus récente
- nécessite généralement un **balayage périodique** pour **supprimer** les objets de données obsolètes.

Exploitations des Data Centers: Vector-clock

Horloges vectorielles (Vector-clocks) :

- Les ensembles de données répartis sur nœuds peuvent être lus et modifiés sur chaque nœud et aucune cohérence stricte n'est assurée par des protocoles de transactions distribuées

  Problème** : comment faire des **modifications concurrentes
- Une **solution** : les **horloges vectorielles** :
  - un ***vecteur d'horloge*** est défini comme un n-uplet V [0], V [1], ..., V[n] des *valeurs d'horloge* à partir de chaque noeud.
  - **à tout instant le noeud i maintient un vecteur d'horloge** représentant son état et celui des autres nœuds répliques : (Vi [0] = valeur de l'horloge du premier noeud, Vi [1] = valeur de l'horloge du deuxième noeud, ... Vi [i] = sa propre valeur d'horloge, ... Vi [n] = valeur de l'horloge du dernier nœud)
  - les valeurs d'horloge peuvent être de réelles « timestamps » dérivées d'une horloge locale de nœud, du numéro de version/révision …

HADOOP

- Hadhoop pour « **High-Availability Distributed Object-Oriented Platform** », est un framework de référence, libre et open source

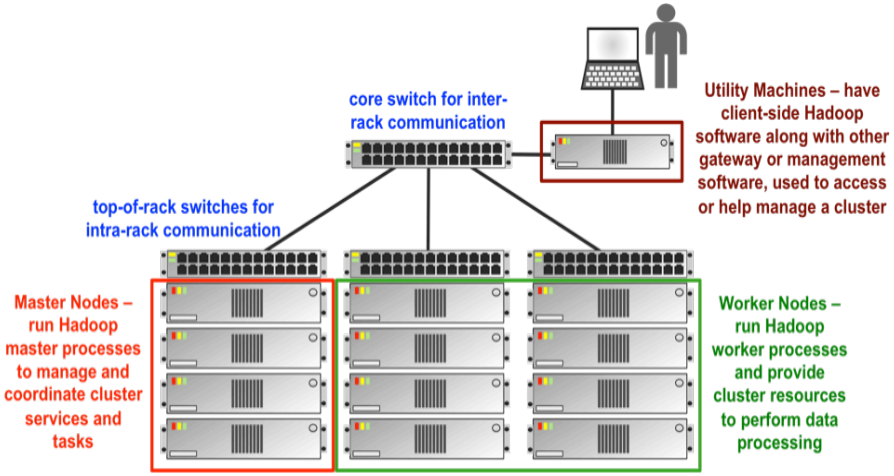
C'est un **système distribué** qui permet **d'analyser**, **stocker** et **manipuler** de **très grandes quantités de données** (Big Data).

- Hadoop a été créé par **Doug Cutting en 2002** pour les besoins du projet « Apache Nutch », intégrant **MapReduce** suite à la sortie de l'article de Google en 2004,
- Notons que Yahoo ! est un contributeur majeur au projet Hahoops
- Hadhoop est depuis 2008 un **projet indépendant de la fondation Apache**
- Il est utilisé par les géants du web comme **Yahoo**!, **Twitter**, **LinkedIn**, **eBay**, **Amazon**, …

HADOOP : Fonctionnement (1)

- Hadoop n'a d'intérêt que pour gérer des données de très grande taille dans un environnement composé de très nombreuses machines (Data Denters) :
- Hadoop :
  - ***fractionne*** les fichiers en gros blocs,
  - ***distribue*** ces blocks à travers les nœuds du cluster,
  - comprend plusieurs composants, notamment :
    - les nœuds maîtres (**Master nodes**),
    - les nœuds travailleurs (**Worker nodes** – ou Slave nodes).

HADOOP : Fonctionnement (2)



HADOOP : Composition du Framework

Le framework Hadoop se compose des modules suivants :

Hadoop Distributed File System (HDFS)	Système de gestion de fichiers distribués permettant de stocker les données sur les machines du cluster
Hadoop Common	Contient les bibliothèques et les utilitaires nécessaires aux autres modules Hadoop
Hadoop YARN (Yet Another Resource Negotiator)	Une plate-forme chargée de la gestion des ressources informatiques du clusters et de les utiliser pour la planification des applications des utilisateurs
Hadoop MapReduce	Une implémentation du modèle de programmation MapReduce pour le traitement des données à grande échelle

Hadoop réfère aussi à son écosystème et à un ensemble des logiciels comme : *Apache Pig, Apache Hive, Apache HBase, Apache Phoenix, Apache Spark, Apache ZooKeeper, Cloudera Impala, Apache Flume, Apache Sqoop, Apache oozie et Apache Storm.*

3. Stockage et gestion des mégadonnées *(Big Data Engineering – BDE)*

- **Limites des bases de données relationnelles**
- **Intérêt de Map Reduce et de Hadoop**
- **Les bases de données NoSQL**
- **Grands modèles de bases de données NoSQL**

Problématique générale

Objectif général : Soutien des activités de **gestion des mégadonnées** : ***stockage*** et ***traitement***

- Activités influencés par de **nombreux facteurs** notamment :
  - **Extensibilité** (*scalability*) : exigence fondamentale compte tenu de la taille toujours croissante des données. Toute solution de stockage doit pouvoir *s'adapter de manière optimale à la croissance des données*.
  - **Disponibilité** (availability) : exigence cruciale que les données puissent être *accessibles de façon fréquente et efficace.*
  - **Sécurité** : les données volumineuses peuvent concerner l'organisation, la *sécurité de leurs données est une grande préoccupation*.
  - **Intégration** : une *intégration les mégadonnées avec des applications et technologies diverses* est souvent nécessaire pour créer une application complète.

Limites des BD relationnelles

- En matière de stockage de données, les **BD relationnelles** restent la **référence**, en garantissant le maintien des **propriétés ACID (Atomicité, Cohérence, Isolation et Durabilité)** pour une **intégrité complète de la BD**.
- Pour gérer de gros volumes de données (contexte d'entrepôt de données), fidèle au modèle relationnel, les **machines bases de données** sont une solution (ex :*Terradata™*), s'appuient sur une *distribution des données sur de nombreux disques* permettant une *parallélisation de l'exécution des requêtes*.
- Cependant ces machines ne peuvent gérer des mégadonnées au delà d'un certain volume.

⇨ Recours à des systèmes de stockage de données **massivement distribuées** sur des « **Data Centers** » (et non des disques seuls)

⇨ Pour le stockage et la gestion des données sur ces Data Centers, de nouvelles solutions ont vu le jour, reposant sur un stockage **massivement distribué (partitionné) des données sur des serveurs** : **les systèmes NoSQL**

Stockage des mégadonnées : les BD NoSQL

Face aux limites des Bases de données relationnelle dans le stockage et la gestion des mégadonnées, de **nouvelles solutions** ont vu le jour :

- permettant une **meilleure scalabilité** (passage à l'échelle) dans des **contextes fortement distribués,**
- permettant une **gestion d'objets complexes et hétérogènes** sans avoir à déclarer au préalable l'ensemble des champs représentant un objet,
- regroupées derrière le terme **NoSQL** (proposé par **Carl Strozzi**), ne se substituant pas aux SGBD Relationnels mais les **complétant** en comblant leurs **faiblesses (Not Only SQL).**

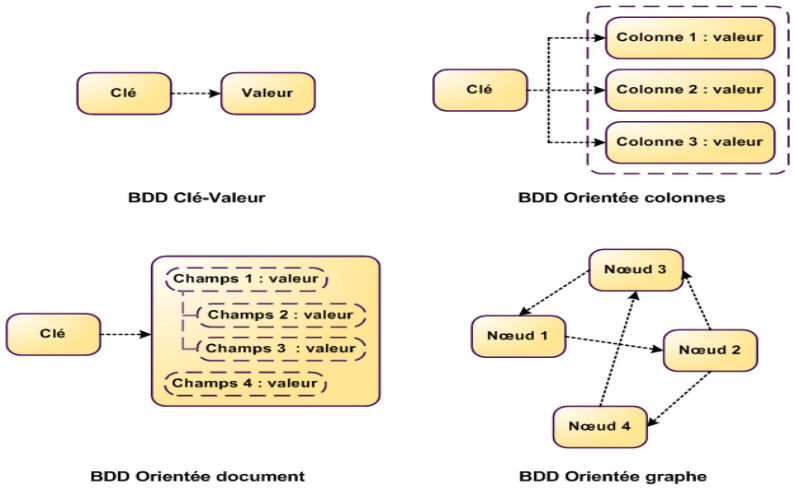
⇨ ***Solutions très efficaces pour stocker et exploiter les mégadonnées, … cependant dans des contextes particuliers …***

Types de BD NoSQL (2)

Stocker les informations de la **façon la mieux adaptée** à leur représentation => **différents types de BD NoSQL :**

- **type « Clé-valeur / Key-value » :** basique, chaque objet est identifié par une clé unique constituant la seule manière de le requêter
  - ⇨ ***Systèmes : Voldemort, Redis, Riak, …***
- **type « Colonne / Column »** : permet de disposer d'un très grand nb de valeurs sur une même ligne, de stocker des relations « one-to-many », d'effectuer des requêtes par clé (adaptés au stockage de listes : messages, posts, commentaires, ...)
  - ⇨ ***Systèmes : HBase, Cassandra, Hypertable, …***
- **type « Document »** : pour la gestion de collections de documents, composés chacun de champs et de valeurs associées, valeurs pouvant être requêtées (adaptées au stockage de profils utilisateur)
  - ⇨ ***Systèmes : MongoDBn CouchDB, Couchbase, …***
- **type « Graphe »** : pour gérer des relations multiples entre les objets (adaptés au données issues de réseaux sociaux, …)
  - ⇨ ***Systèmes : Neo4j, OrientDB, …***

Types de BD NoSQL (2)



Bilan sur la gestion des mégadonnées

Stockage des données : bases de données NoSQL

- Apportent des **solutions efficaces** tant sur le volume que sur la variété des mégadonnées
- **Complexité des traitements** : pas de langage puissant de requêtage et d'exploitation comme SQL, mais des langages propriétaire (sauf BD graphes : *Sparql*, *Cypher*)
- **Relâchement de la Cohérence** (ACID -> BASE) :
  - permet un grain de performances et un passage à l'échelle facilité
  - mais peut être critique pour certaines applications (ex : opérations financières) et alors nécessiter le développement de couches logicielles supplémentaires
- **Technologie peu familière** et une **communauté encore réduite**
- **Beaucoup de solution open-source** : encore très peu de support client.

Distribution intensive par usage :

- du quadriciel HADOOP
- du modèle de programmation parallèle Map Reduce

3. Analyse des mégadonnées *(Big Data Analytics – BDA)*

- **Problématique**
- **Analyse des données et apprentissage automatique**
- **Analytique des données, des flots de données, de textes, du Web, …**

Problématique générale

Objectif général : découvrir par analyse des mégadonnées des modèles (connaissances) enfouis et en tirer des enseignements utiles

- La nature de cette analyse dépend de la **nature** et de la **structure** des mégadonnées,
- Appelé aussi « **analytique** », traduction de « *analytics* »
- Ces différentes analyses mettrent en œuvre divers algorithmes relevant de :
  - des **statistiques** et de **l'analyse de données,**
  - la **fouille de données** (Data Mining),
  - de **l'apprentissage (machine) automatique** (Machine Learning),
  - de l'aide à la décision,
  - de la visualisation
  - …

Apprentissage automatique

L'apprentissage automatique (Machine Learning) est l'un des principaux domaines de l'intelligence artificielle (IA) :

- Il permet de réaliser des *algorithmes accomplissant des tâches difficiles* à programmer de façon traditionnelle (ou au prix d'un effort/temps important).
- Ces algorithmes utilisent les données représentatives (*en entrée*) du problème que l'on cherche à résoudre pour « apprendre à la machine » à le résoudre (produire une *sortie adéquate*).

  Ex :

  *On présente à l'algorithme (modèle) des exemples de la tâche à réaliser, où sont fournies à la fois les **entrées** (des images par exemple) et les **sorties** (résultats) souhaitées (par exemple voiture, avion, etc.).*

  *Le but de l'algorithme est alors de **rechercher une règle générale** qui fait correspondre les entrées aux sorties lors de l'apprentissage, mais aussi de trouver les bonnes sorties pour entrées n'ayant pas servi à l'apprentissage (**généralisation**)*

Analytique de données et apprentissage

- **Caractérisation de l'apprentissage :**
  - **supervisé** (à base d'exemples pré-étiquetés ou classés),
  - **semi-supervisé** (apprentissage à partir d'une combinaison d'exemples étiquetés et d'exemples non étiquetés)
  - **non supervisé** (apprentissage d'un modèle à partir des seules données non étiquetés)
  - **actif** (sélection automatique des meilleurs exemples à étiqueter manuellement parmi un ensemble très important de données non étiquetés)
  - **par renforcement** (le système apprend au fur et à mesure que l'on s'en sert, à la manière d'un joueur qui gagne en expérience).
- **Apprentissage statistique**
  - basé sur des *modèles mathématiques* bien fondés et des *algorithmes puissants*,
  - Ex : réseaux bayésiens, modèles de Markov cachés (HMM), machines à vecteur support (SVM), …
- **Apprentissage profond** – *Deep Learning* (réseaux de neurones) :
  - *difficiles à paramétrer* (architectures récurrentes ou non, nombre de couches, taille des vecteurs représentant les individus, fonctions d'activation...)
  - *nécessitent beaucoup d'exemples*, et traitement massivement parallèle par cartes graphiques (GPU).

Analytique de données : tâches et techniques

- **2 tâches principales :**
  - la **classification** *(attribuer des classes)*
  - **le clustering** *(catégoriser, trouver des classes)* et
- **Principales techniques utilisées :**
  - **Arbres de décision** (ex : C4.5) classification supervisée
  - **K-mean** clustering
  - **SVM** (Support Vector Machine) : classification supervisée par approche discriminante et fonctions noyaux
  - **Naive Bayes** : classification probabiliste Bayésienne Naïve
  - **Apriori** : estimation automatique de distribution de probabilités par Expected Maximisation
  - **Réseaux de neurones** …
  - **…**

⇨ **Principal problème : parallélisations des algorithmes …**

⇨ **Projet *Mahout*** (https://mahout.apache.org): basé sur *Hadoop* il fournit des *versions distribuées de plusieurs algorithmes standards d'apprentissage automatique et de fouille de données*.

Analytique de données : autres techniques …

- **Techniques de fouille pour l'exploration de données spécifiques :**
  - Données *séquentielles*,
  - Données *temporelles*
  - Données *spatiales*
  - …
- **Techniques de fouille de processus (Process Mining) :**
  - Basée sur *l'analyse de données évènementielles*
  - Permettent la *découverte de nouveaux processus*, le *contrôle de conformité de processus*, …
  - Exploitent des *journaux d'évènements* (event logs) de plus en plus disponibles dans les organisations quel que soit leur domaine, de l'industrie à la santé [30].
  - …
- **Autres techniques …**

Analytique de flots de données

- **Données échangées et émises en continu :**

Ex : données en flots sur des médias en ligne, données en provenance de capteurs, relevés d'expérimentation dans le domaine de la physique, …

⇨ *Pas envisageable d'interroger la totalité du flux de données continu, ce qui pourrait avoir pour conséquence de stopper le flux.*

⇨ **nouveaux algorithmes optimisés** en temps de traitement et en occupation mémoire :

- Permettant de construire des ***résumés*** / ***synopsis*** du flux de données
- L'interrogation de ces *résumés* / *synopsis* conduit à résultats avec une certaine ***approximation***.

Ex : technique des « fenêtres temporelles » travaillant sur un ensemble restreint du flux pour en extraire des motifs (items, itemsets, et motifs séquentiels ou Sequential patterns) porteurs de connaissance.

Analytique de textes (Text Analytics) – (1)

- **Texte** = part importante du contenu non-structuré dans les organisations (documents, e-mail, pages Web, contenu des médias sociaux, …)
- **Analytique de texte** (Text analytics) s'appuie sur :
  - la *fouille de texte* (Text Mining),
  - la *recherche d'information* (RI),
  - la *linguistique informatique* (techniques de traitement automatique du langage naturel (TALN), symboliques, statistiques, et/ou utilisant l'apprentissage profond (plongement de mots – Word Embedding)
- **Concernent différentes tâches :**
  - *Extraction d'information : entités nommées (noms propres, dates, quantités ...) et relations entre ces entités nommées*
  - *Désambiguïsation sémantique*
  - *Etiquetage syntaxique*
  - *Résumé automatique (mono et multi document(s)*
  - *Simplification de textes*
  - *Analyse d'opinions et de sentiments*
  - …
- **Moteurs de recherche** utilisent de plus en plus de techniques de TALN pour mieux rapprocher *requêtes* et *documents*, traiter les *requêtes complexes* et appréhender la *sémantique des contenus*.

Analytique du Web (Web Analytics)

- **Thème de recherche très actif** avec de nombreux challenges et opportunités
- Concerne les *méthodes* et *technologie* relatives à la *collecte*, la *mesure*, *l'analyse* et la *présentation* des données utilisées dans les sites et applications Web
- S'appuie principalement sur les avancées en *Analytique de textes*
- Englobe la *recherche* et la *fouille sociale*, les *systèmes de réputation*, *l'analyse des médias sociaux,* et la *visualisation Web*, la *ventes aux enchères* sur le Web, la *monétisation d'Internet*, le *marketing social* et la *confidentialité/sécurité* du Web
- Conduit au développement de **plates-formes et services de Cloud Computing** (applications, logiciels système et du matériel fournis comme services sur Internet) :

  Ex1 : ***Amazon Elastic Compute Cloud (EC2***) permet aux utilisateurs de louer des ordinateurs virtuels, sur lesquels ils peuvent exécuter leurs propres applications informatiques. Son Simple Storage Service (S3) offre un service de stockage en ligne de Web.

Axes de recherche prioritaires en exploitation des mégadonnées

- **Calcul parallèle** en utilisant les *technologies du cloud computing, Hadoop, Map-Reduce*, …
- **Méthodes de planification** *(scheduling method)* pour gérer les ressources de calcul de la plate-forme sur le cloud pour analyser rapidement des mégadonnées
- **Méthodes efficaces pour la réduction de temps de calcul** des entrées, des comparaisons, des échantillonnages pour l'analyse des mégadonnées
- **Méthodes d'extraction de connaissance** des mégadonnées
- **Méthodes d'analyse des connaissances** obtenues des mégadonnées
- **La fusion d'informations** permettant la combinaison d'informations provenant de différentes ressources externes pour améliorer l'analyse des mégadonnées
- **Algorithmes métaheuristiques** (déjà utilisés pour trouver des solutions approximatives dans un délai raisonnable) pour l'exploration des mégadonnées
- **Méthodes d'analyse de réseaux sociaux** pour prédire le comportement d'utilisateurs pour élaborer des stratégies vis à vis de ces utilisateurs.
- **Sécurité**, **respect de la vie privée**, et **protection** des mégadonnées
- …