Analyse des données massives

2eme MSTIC

2019-2020

Programme du cours théorique

Chapitre 1. Introduction à la programmation pour les grandes données

A) Quelles sont les grandes données ?

B) Comment la programmation des grandes données est-elle différente ?

Chapitre 2. Programmation d'accès aux données

A) Structures de données pour l'analyse des données

B) Outils de programmation d'accès aux données (par exemple R, SAS)

Chapitre 3. Programmation d'analyse de données

A) Programmation des statistiques descriptives

B) Programmation pour l'analyse avancée

C) Outil de programmation d'analyse de données, s (par exemple R, SAS)

Chapitre 4. Des paradigmes de programmation distribués

Chapitre 5. MapReduce

A) Outils de programmation distribuée pour le stockage des données et l'analyse des données (par exemple, Hadoop, Mahoot, Pig)

Chapitre 6. L'avenir des grandes données.

Chapitre 1. Introduction à la programmation pour les grandes données

A) Quelles sont les grandes données ?

Le phénomène « données massives »:

- **L'explosion quantitative des données numériques** a obligé les chercheurs à trouver de nouvelles manières de voir et d'analyser ces dernières.
- Il s'agit de découvrir **de nouvelles solutions** concernant **la capture, la recherche, le partage, le stockage, l'analyse et la présentation des données**.
- Ainsi est né le « **Big Data** ». Il s'agit d'un concept permettant de stocker un nombre indicible d'informations sur une base numérique.
- Selon les archives de la bibliothèque numérique de l'Association for Computing Machinery (ou ACM) dans des articles scientifiques concernant les défis technologiques à relever pour visualiser les « grands ensembles de données », cette appellation est apparue en octobre 1997.

A) Quelles sont les grandes données ?

Les données massives, c'est quoi ?

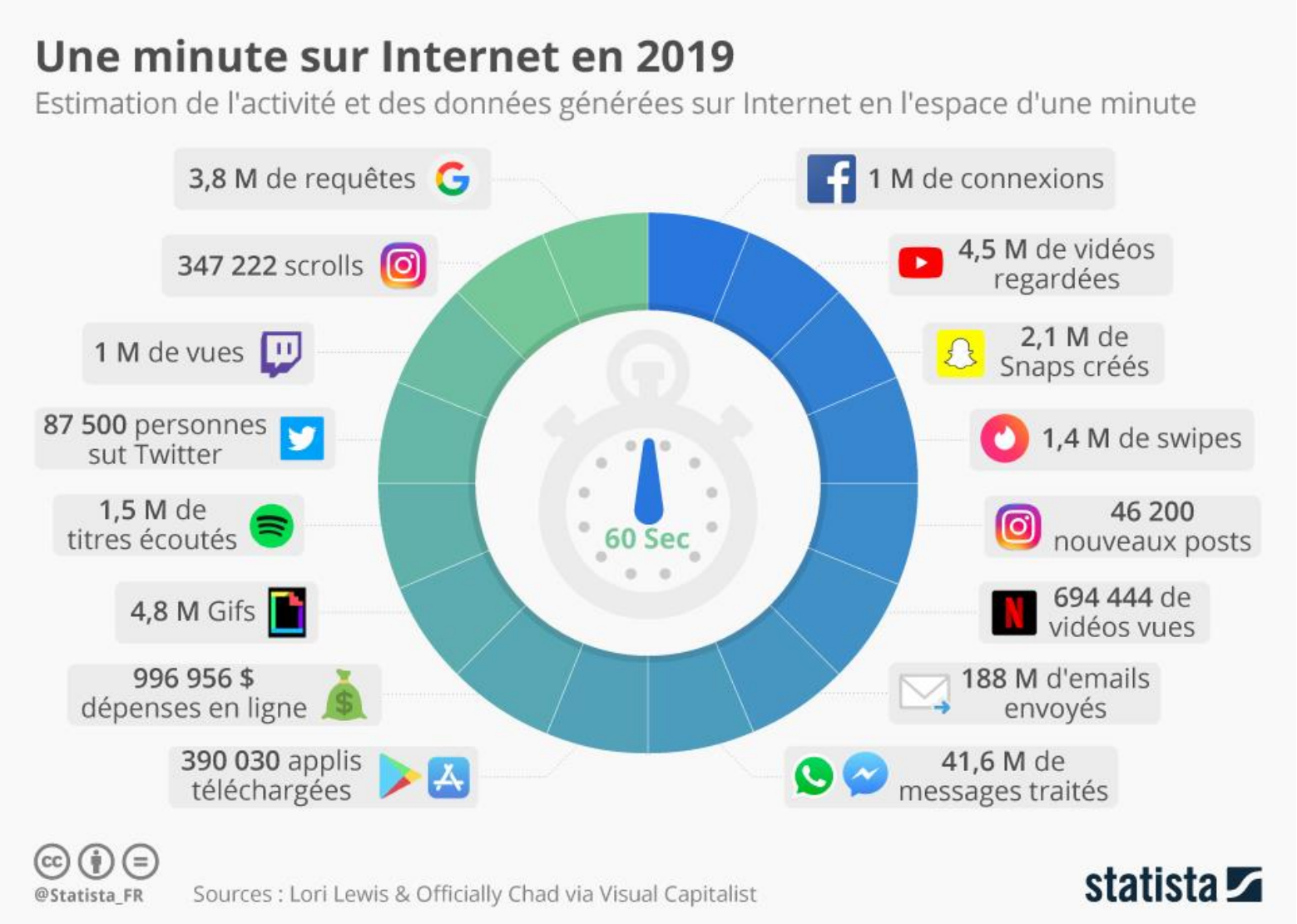
- Littéralement, ce terme signifie des **méga-données**, grosses données ou encore **big data**. Ils désignent un ensemble très volumineux de données qu'aucun outil classique de gestion de base de données ou de gestion de l'information ne peut vraiment travailler.

- En effet, nous procréons **environ 2,5 trillions d'octets de données tous les jours (voir la figure 1)**. Ce sont les **informations provenant de partout** : messages que nous nous envoyons, vidéos que nous publions, informations climatiques, signaux GPS, enregistrements transactionnels d'achats en ligne et bien d'autres encore. **Ces données sont baptisées Big Data ou volumes massifs de données**.

A) Quelles sont les grandes données ?

- Les géants du Web, au premier rang desquels Yahoo (mais aussi Facebook et Google), ont été les tous premiers à déployer ce type de technologie.
- Cependant, **aucune définition précise ou universelle ne peut être donnée au Big Data**. Etant un objet complexe polymorphe, sa définition varie selon les communautés qui s'y intéressent en tant qu'usager ou fournisseur de services.
- Une approche transdisciplinaire permet d'appréhender le comportement des différents acteurs : les concepteurs et fournisseurs d'outils (les informaticiens), les catégories d'utilisateurs (gestionnaires, responsables d'entreprises, décideurs politiques, chercheurs), les acteurs de la santé et les usagers.

FIGURE 1: VOLUMES DE DONNÉES PAR MINUTE SUR LE WEB (2019)



Discussion:

60 secondes sur Internet, c'est peu et pourtant…

Ce court laps de temps suffit à générer des millions de données à travers le monde.

Les données compilées et relayées par le site Visual Capitalist, en sont la preuve. En une minute:

- 3,8 millions de requêtes Google sont enregistrées,
- 41 millions de messages WhatsApp et Facebook Messenger traités,
- 1,5 million de titres écoutés sur Spotify…
- Etc.

A) Quelles sont les grandes données ?

- Le **Big data** ne dérive pas des règles de toutes les technologies, il est aussi un **système technique dual**. En effet, **il apporte des bénéfices** comme par exemple l'augmentation du nombre de clients, du chiffre d'affaires, ou encore la réduction des coûts.
- **Mais, il peut également générer des inconvénients tels que:**
  - **La difficulté à intégrer les outils Big Data aux technologies utilisées:** Les fournisseurs de base de données et d'outils analytiques traditionnels ne proposent pas d'outils Big Data. Il faut donc se tourner vers les nouveaux acteurs de ce dernier, et **accepter leur approche différente de la technologie**. Encore une fois, l'adaptation est le meilleure solution.

Inconvénients (suite)

- **Reproduire des processus existants avec des technologies différentes**

- **Philosophie speed-to-market:** Il est important pour les analystes d'être capables d'accéder aux données massives de l'entreprise. Il est indispensable qu'ils apprennent rapidement à **choisir et à maîtriser les outils adéquats**. Le marché évolue vite, et les entreprises doivent s'adapter à son rythme.

Inconvénients (suite)

- **Gouvernance des données:** l'un des risques majeurs des données massives provient de l'organisation du Data Lake. Une donnée en provenance d'une source peut accidentellement se combiner à une donnée d'une autre source et engendrer **une malencontreuse exposition de données**. Pour y remédier, il est important d'élaborer une stratégie de gouvernance des données en collaboration avec les employés responsables.
- **Sécurité des données et administration:** La sécurité des données est en toute logique le principal problème surveillé en premier lieu dans tous les domaines. Il existe **plusieurs techniques pour sécuriser ces données** : instaurer un périmètre de sécurité, le chiffrement des données stockées et en cours de transition, la configuration stricte de l'authentification, ou encore le choix judicieux entre un stockage interne ou sur le Cloud.

Devant tant d'informations, comment le mieux les gérer

- Il s'agit en fait du principal problème avec les mégadonnées. **La quantité énorme des informations est un des obstacles**. L'autre obstacle provient évidemment du niveau de certitude qu'on peut avoir sur une donnée.
- En effet, **quelques données de l'internet (marketing numérique** par exemple) peuvent être considérées comme **des informations « incertaines »**, l**e volume de données** associé au **manque de crédibilité** de celles-ci rend son exploitation plus complexe.
- Pour autant, **grâce aux algorithmes statistiques, des solutions existent**. C'est d'ailleurs, avant même de se demander s'il serait possible de collecter et stocker le big data, qu'on devrait toujours commencer par s'interroger de son aptitude **à les analyser et de leur utilité.**

Devant tant d'informations, comment le mieux les gérer

- Avec un but convenablement déterminé et des données d'une qualité suffisante, **les algorithmes et méthodes statistiques** permettent désormais de concevoir de la valeur alors que ce n'était pas encore faisable il y a encore quelques années. A ce propos, on peut distinguer d**eux types d'écoles dans le domaine prédictif à savoir l'intelligence artificielle ou « machine learning » et la statistique**. Ces deux secteurs bien qu'ils soient distincts se rejoignent finalement de plus en plus. De plus, ils peuvent être utilisés en simultanéité de manière vertueuse et intelligente pour mener à bien un projet.

Big Data : des innovations disruptives qui changent la donne

- **Le Big Data et les analytics** sont utilisés dans presque tous les domaines. Ils se sont même construits une place importante dans la société. **Ils se traduisent sous plusieurs formes** à ne citer que l'usage de statistiques dans le sport de haut niveau, **le programme de surveillance PRISM de la NSA,** la médecine analytique ou encore les algorithmes de recommandation d'Amazon.
- En entreprise particulièrement, l'usage d'outils Big Data et Analytics répond généralement à plusieurs objectifs comme l'**amélioration de l'expérience client, l'optimisation des processus et de la performance opérationnelle, le renforcement ou diversification du business model.**
- De **nouvelles opportunités** significatives de différenciation concurrentielle sont **générées par l'ère de la gestion d'importants volumes de données** et de leur analyse. Pour les organisations, plusieurs raisons peuvent les inciter à se tourner vers cette nouvelle administration de données à savoir **la gestion rentable des données, l'optimisation du stockage d'informations, la possibilité de faire des analyses programmables ou encore la facilité de la manipulation des données.**

B) Comment la programmation des grandes données est-elle différente ?

- **le Big Data** se présente comme une solution dessinée pour **permettre à tout le monde d'accéder en temps réel à des bases de données géantes**. Il vise à proposer un choix aux solutions classiques de bases de données et d'analyse.
- **Ainsi, ce concept regroupe une famille d'outils** qui répondent à une triple problématique dite la **règle des 3V (figure 2).** Il s'agit notamment d':
  - un **Volume** de données considérable à traiter,
  - une grande **Variété** d'informations (venant de diverses sources, non-structurées, organisées, Open…),
  - et un certain niveau de **Vélocité** à atteindre, autrement dit de fréquence de création, collecte et partage de ces données.

B) Comment la programmation des grandes données est-elle différente ?

Volume: Le volume correspond à la masse d'informations produite chaque seconde.

• Les données sont collectées de diverses sources : dont les transactions commerciales, médias sociaux et informations issues de capteurs ou de transactions de machine à machine.

• Selon des études, l'ensemble de toutes les données produites depuis le début des temps jusqu'à la fin de l'année 2008, conviendrait maintenant à la masse de celles qui sont générées chaque minute.

•Auparavant, le stockage de ces données aurait posé problème, mais cette tâche est désormais simplifiée par les nouvelles technologies comme Hadoop.

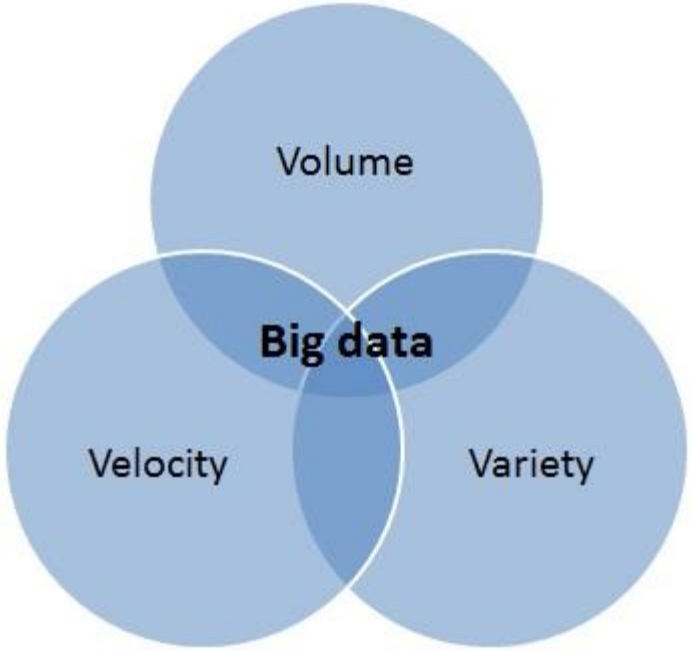


Figure 2: les 3V du Big data.

B) Comment la programmation des grandes données est-elle différente ?

- **Vitesse:** La vitesse décrit la fréquence à laquelle les données sont générées, capturées et partagées. Du fait des évolutions technologiques récentes, les consommateurs mais aussi les entreprises génèrent plus de données dans des temps beaucoup plus courts.
  - À ce niveau de vitesse, les entreprises ne peuvent capitaliser sur ces données que si elles sont collectées et partagées en temps réel. C'est précisément à ce stade que de nombreux systèmes d'analyse échouent.
  - S'ils peuvent seulement traiter les données par lots toutes les quelques heures par exemple, dans le meilleur des cas. Or, ces données n'ont alors déjà plus aucune valeur puisque le cycle de génération de nouvelles données a déjà commencé.

VITESSE (SUITE)

Exemple:

- si on diffuse des messages sur les réseaux sociaux, ils peuvent devenir « viraux » et se répandre en un rien de temps.
  - Il s'agit d'analyser les données au décours de leur lignée (appelé parfois analyse en mémoire) sans qu'il soit indispensable que ces informations soient entreposées dans une base de données.

B) Comment la programmation des grandes données est-elle différente ?

- **Variété:** Les données revêtent tous types de formats : des données numériques structurées dans des bases de données traditionnelles (**20%)** aux documents texte non structurés, en passant par les e-mails, les vidéos, les fichiers audio, les données boursières et les transactions financières.
- La technologie Big Data, permet de faire l'analyse, la comparaison, la reconnaissance, le classement des données de différents types comme des conversations ou messages sur les réseaux sociaux, des photos sur différents sites etc. Ce sont les différents éléments qui constituent la variété offerte par le Big Data.

VERS UN MODÈLE DE 5 V

Parmi les utilisateurs les plus enthousiastes du Big Data, on retrouve les gestionnaires et les économistes. Ces derniers le définissent par **la règle des 5V** (Volume, Velocity, Variety, Veracity, Value):

- **La_véracité:** la véracité concerne la fiabilité et la crédibilité des informations collectées.
  - Comme le Big Data permet de collecter un nombre indéfini et plusieurs formes de données, il est difficile de justifier l'authenticité des contenus.
  - Toutefois, les génies de l'informatique sont en train de développer de nouvelles techniques qui devront permettre de faciliter la gestion de ce type de données notamment par le W3C.
- **La valeur:** La notion de valeur correspond au profit qu'on puisse tirer de l'usage du Big Data.
  - Ce sont généralement les entreprises qui commencent à obtenir des avantages incroyables de leurs Big Data.

Programme du cours théorique

Chapitre 1. Introduction à la programmation pour les grandes données

A) Quelles sont les grandes données ?

B) Comment la programmation des grandes données est-elle différente ?

Chapitre 2. Programmation d'accès aux données

A) Structures de données pour l'analyse des données

B) Outils de programmation d'accès aux données (par exemple R, SAS)

Chapitre 3. Programmation d'analyse de données

A) Programmation des statistiques descriptives

B) Programmation pour l'analyse avancée

C) Outil de programmation d'analyse de données, s (par exemple R, SAS)

Chapitre 4. Des paradigmes de programmation distribués

Chapitre 5. MapReduce

A) Outils de programmation distribuée pour le stockage des données et l'analyse des données (par exemple, Hadoop, Mahoot, Pig)

Chapitre 6. L'avenir des grandes données.

CHAPITRE 2: PROGRAMMATION D'ACCÈS AUX DONNÉES

- Cette section traite de la problématique du stockage de très grands volumes de données. Dans un premier temps nous pointons les limites des bases de données relationnelles pour le stockage et la gestion des données massives, et évoquons l'apport du *Cloud Computing* (informatique dans les nuages).
- Puis nous soulignons l'intérêt pour le stockage et la gestion des données du modèle de programmation parallèle « **MapReduce** » et du f*ramework* libre «**Hadoop** » .
- Ensuite nous introduisons les différents modèles de bases de données dites NoSQL, constituant différentes solutions de stockage des méga-données. Pour finir nous évoquons quelques autres alternatives, notamment les bases de données NewSQL.

A) Structures de données pour l'analyse des données

Limites des bases de données relationnelles et *Cloud Computing*

- En matière de stockage de données, les bases de données relationnelles restent la référence. Ces outils, largement utilisés, garantissent le maintien des propriétés ACID (Atomicité, Cohérence, Isolation et Durabilité).
- Cependant, pour gérer de gros volumes de données, notamment dans un contexte d'entrepôt de données, les machines bases de données s'appuient sur une distribution des données sur différents disques permettant une parallélisation de l'exécution des requêtes.
- Alors que ces machines ne permettent pas de gérer des méga-données au-delà d'un certain volume.
- Aussi, différentes nouvelles solutions ont vu le jour. Toutes ces solutions reposent sur un stockage distribué (partitionné) des données sur les clusters.
- Mais, aucun système distribué ne peut assurer à la fois la cohérence, la disponibilité et la possibilité d'être partitionné.
- La conséquence est que, dans ces nouvelles solutions de stockage, un relâchement de ces propriétés sera nécessaire.

A) Structures de données pour l'analyse des données

- Le *cloud* est un ensemble de matériels, de raccordements réseau et de logiciels fournissant des services importants que des individus et des collectivités peuvent exploiter au besoin et depuis n'importe quel emplacement.
- Au lieu d'obtenir de la puissance de calcul par acquisition de nouveaux matériels et de logiciels, dans le *Cloud Computing*, le consommateur utilise une puissance de calcul mise à sa disposition sur une architecture d'un fournisseur via Internet (le stockage des données massives dans les clouds).
- Les besoins de stockage s'accroissant, de nouveaux serveurs sont déployés dans cette architecture de façon transparente pour l'utilisateur. Si le *cloud* permet d'appréhender la caractéristique de *volume* des données massives , les caractéristiques de *variété* et de *vélocité* ne le sont pas. Donc, le *cloud* étant davantage un support de stockage qu'une solution de gestion de données.

A) Structures de données pour l'analyse des données

Intérêt de MapReduce et de Hadoop

- Dans le stockage et la gestion des mégadonnées, le modèle de programmation parallèle « MapReduce » et le cadriciel libre « Hadoop » le mettant en œuvre s'avèrent déterminant.
- *MapReduce* est un patron ou modèle d'architecture de développement informatique, dans lequel sont effectués des calculs parallèles et souvent distribués sur des données pouvant être très volumineuses.
- Il repose sur deux fonctions : « *Map* » et « *Reduce* », empruntées aux langages de programmation fonctionnelle. De façon générale, la fonction *Map*, exécutée par un nœud spécifique, analyse un problème, le découpe en sous-problèmes, et ensuite délègue la résolution de ces sous-problèmes à d'autres nœuds de traitements pour être traités en parallèle, ceci à l'aide de la fonction *Reduce*. Ces nœuds font ensuite remonter leurs résultats au nœud qui les avait sollicités.

A) Structures de données pour l'analyse des données

- Ainsi le modèle MapReduce permet de manipuler de grandes quantités de données en les distribuant dans un cluster de nœuds pour être traitées.
- MapReduce a été rapidement utilisé par des sociétés intervenant sur le Web et possédant d'importants centres de traitement de données telles qu'Amazon ou Facebook.
- Notons que MapReduce est aussi de plus en plus utilisé dans le *Cloud Computing*. De nombreux *framework* implémentant MapReduce ont vu le jour, dont le plus connu est Hadoop.
- *Hadoop* (pour *High-Availability Distributed Object-Oriented Platform*), est un *framework* de référence libre et *open source*, intégrant MapReduce et permettant d'analyser, stocker et manipuler de très grandes quantités de données.
- Hadoop a été créé par Doug Cutting et fait partie des projets de la fondation logicielle Apache depuis 2009.
- Le noyau d'Hadoop est constitué d'une partie stockage consistant en un système de fichiers distribué, extensible et portable appelé HDFS (*Hadoop Distributed File System*), et d'une partie traitement appelée MapReduce.
- Hadoop fractionne les fichiers en gros blocs et les distribue à travers les nœuds du cluster. Pour traiter les données selon le modèle MapReduce, Hadoop transfère le code à chaque nœud qui traite les données dont il dispose. Cela permet de traiter un volume important de données plus rapidement et plus efficacement que dans une architecture super-calculateur classique.

A) Structures de données pour l'analyse des données

- Les bases de données NoSQL, adaptées au stockage et à la gestion des données massives, utilisent généralement le framework Hadoop pour analyser, stocker et manipuler ces dernières.
- Notons enfin que des *frameworks spécifiques* ont vu le jour permettant d'améliorer les performances de Hadoop, notamment dans les milieux hétérogènes, tant en termes de vitesse de traitement, qu'en termes de consommation électrique.

BASES DE DONNÉES NOSQL

- Le stockage des données massives nécessite un partitionnement. Or selon le théorème de CAP (pour *Consistency, Availability, Partition tolerance*), la gestion des données partitionnées, nécessairement garantie par un système distribué. Alors que, il est impossible d'assurer la cohérence et la disponibilité des données simultanément.
- Donc, les systèmes de gestion des données massives devront faire le choix entre la cohérence et la disponibilité. La disponibilité est généralement privilégiée dans le cas de l'exploitation de ces données.
- De plus, les systèmes relationnels imposent une structuration des données selon des schémas spécifiques, or les données massives sont en grande partie peu ou pas structurées.
- En conséquence, de nouveaux modèles de stockage de données ont vu le jour, et ont conduit à l'émergence des **bases de données NoSQL**. Notons que le terme NoSQL (proposé par Carl Strozzi) ne signifie pas le rejet du modèle relationnel (SQL), mais doit être interprété comme « *Not Only SQL* ».

Bases de données NoSQL

- Ces modèles ne remplacent pas les BD relationnelles mais sont une alternative, un complément apportant des solutions plus intéressantes dans certains contextes.
- Les systèmes NoSQL permettent une gestion d'objets complexes et hétérogènes sans avoir à déclarer au préalable l'ensemble des champs représentant un objet.
- Ils adoptent une représentation de données non relationnelle, sans schéma pour les données, ou avec des schémas dynamiques, et concernent des données de structures complexes ou imbriquées .
- Les systèmes NoSQL apportent une plus grande performance dans le contexte des applications Web avec des volumétries de données exponentielles.
- Ils utilisent une très forte distribution de ces données et des traitements associés sur de nombreux serveurs (*sharding*).
- Ensuite ces systèmes optent pour un partitionnement horizontal des données sur plusieurs nœuds ou serveurs (*consistent hashing*).
- Enfin, ils utilisent généralement pour cela des algorithmes de type « MapReduce », pour paralléliser tout un ensemble de tâches à effectuer sur un ensemble de données.

Bases de données NoSQL

- En conséquence du théorème de Brewer, les systèmes NoSQL font un compromis sur le caractère « ACID » des systèmes relationnels : pour plus de scalabilité horizontale (passage à l'échelle) et d'évolutivité, ils privilégient la disponibilité à la cohérence, ils ne possèdent généralement pas de gestion de transactions.
- Pour assurer le contrôle de concurrence, ils utilisent des **mécanismes de contrôle multi-version** (MVCC) ou des **horloges vectorielles** (*Vector-Clock*). Ils sont principalement utilisés dans des contextes où il y a peu d'écritures et beaucoup de lectures.
- L'adoption croissante des bases NoSQL par des grands acteurs du Web (Google, Facebook, Amazon, etc.), a conduit à une multiplication des offres de systèmes NoSQL, en grande partie en *Open-source*.
- Ces systèmes utilisent très souvent le *framework* Hadoop intégrant MapReduce et permettant d'analyser, stocker et manipuler de très grandes quantités de données.
- Enfin, il est important de noter que le système Hive construit sur Hadoop permettant d'écrire des requêtes dans le langage HQL (*Hive Query Language*) qui est proche du standard SQL.

Principaux modèles de bases de données NoSQL

Modèle orienté « clé-valeur »

- Dans ce modèle (figure 3), les données sont stockées sous la forme de grandes tables de hashage distribuées, permettant de facilement passer à l'échelle.
- Les données sont simplement représentées par un couple clé-valeur. La valeur peut être une simple chaîne de caractères, un objet sérialisé…
- Cette absence de structure et de typage a un impact important sur le requêtage.
- Ainsi la complexité d'une requête qui sera dans un système relationnel prise en charge par le langage SQL, sera dans ce modèle NoSQL prise en charge par l'applicatif interrogeant la base de données, la communication avec la base se limitant généralement à des ordres tels que PUT, GET et DELETE.
- Les systèmes NoSQL orientés clé-valeur les plus connus sont Memcached, Amazon's Dynamo, Redis, Riak et Voldemort créé par LinkedIn.

Modèle orienté « clé-valeur »

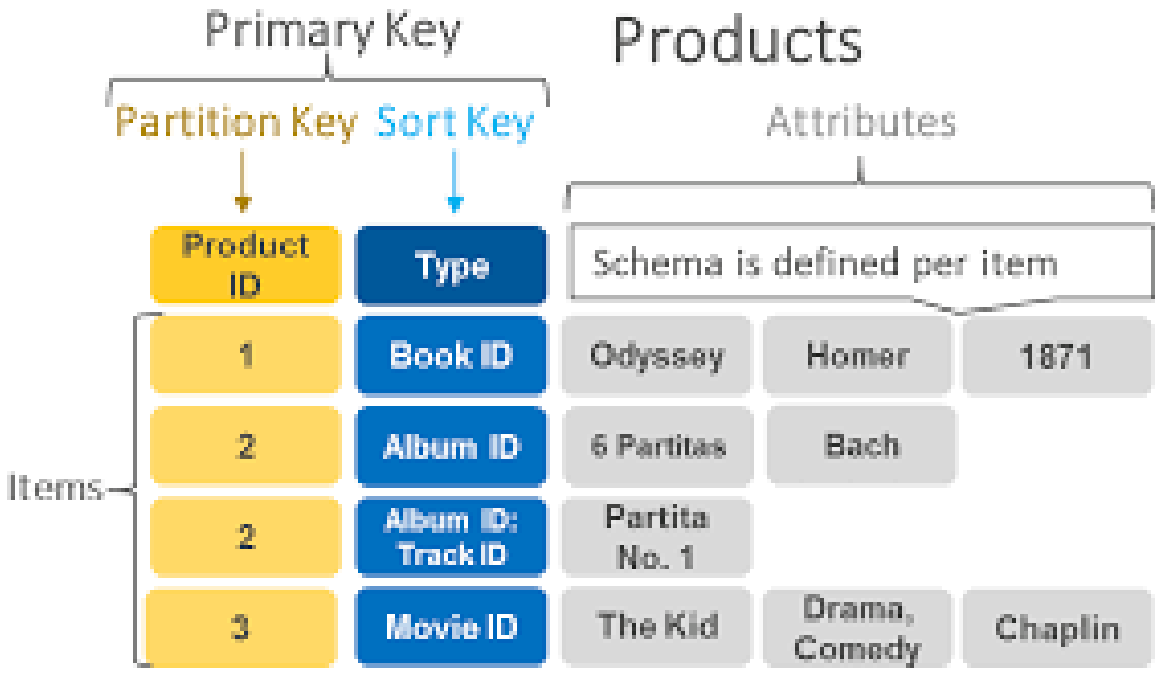


FIGURE 3: Le modèle orienté « clé-valeur »

Principaux modèles de bases de données NoSQL

Modèle orienté « documents »

- Ce modèle (figure 4) se base sur le paradigme clé-valeur précédent ; cependant dans ce nouveau modèle la valeur est un document de type JSON ou XML.
- Ainsi, dans ce modèle, les données sont stockées à l'intérieur de documents. Un document peut être vu comme un n-uplet d'une table dans le monde relationnel, à la différence toutefois que les documents peuvent avoir une structure complètement différente les uns des autres.
- L'avantage est de pouvoir récupérer, via une seule clé, un ensemble d'informations structurées de manière hiérarchique. La même opération dans le monde relationnel impliquerait plusieurs jointures.
- Les systèmes NoSQL orientés documents les plus connus sont CouchDB d'Apache, RavenDB (destiné aux plateformes .NET/Windows avec la possibilité d'interrogation via LINQ) et MongoDB.

MODÈLE ORIENTÉ « DOCUMENTS »

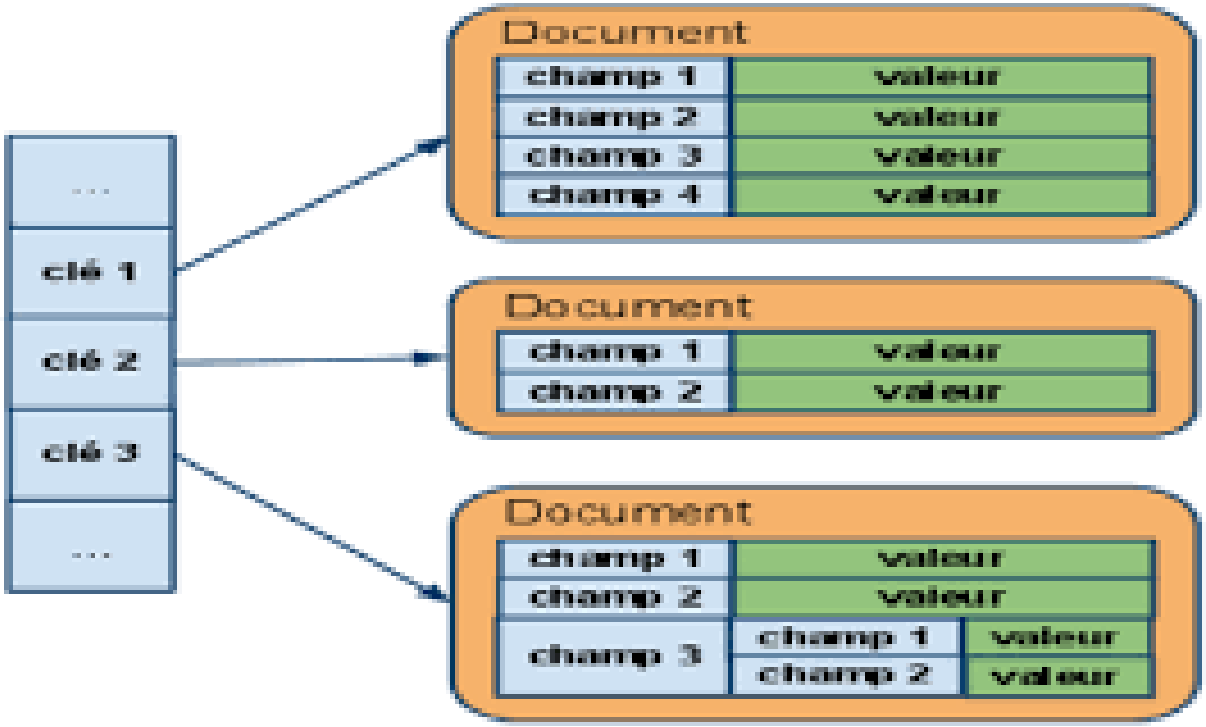


FIGURE 4: Le modèle orienté « documents »

PRINCIPAUX MODÈLES DE BASES DE DONNÉES NOSQL

Modèle orienté « colonnes »

- Ce modèle (figure 6) ressemble à première vue à une table du modèle relationnel, du fait que les attributs sont regroupés en famille de colonnes.
- Ainsi, deux attributs qui sont fréquemment interrogés ensemble seront stockés au sein d'une même famille de colonnes.
- Cependant la différence est que, dans cette base NoSQL orientée colonnes, le nombre de colonnes est dynamique, alors que dans une table relationnelle, le nombre de colonnes est fixé dès la création du schéma de la table.
- De plus, dans ce modèle, contrairement au modèle relationnel, le nombre de colonnes peut varier d'un enregistrement à un autre, ce qui évite de retrouver des colonnes ayant des valeurs inconnues (*Null Value*).
- Les systèmes NoSQL orientés colonnes les plus connus sont principalement HBase, implémentation *Open Source* du modèle BigTable développé par Google, et Cassandra, projet Apache qui respecte l'architecture distribuée de Dynamo d'Amazon, et le modèle BigTable de Google.

Modèle orienté « colonnes »

Stockage orienté lignes

id	type	lieu	spec	intérêts
Nicolas	prof	CNAM	BDD, NoSQL	BZH, Star Wars
Régis		OC	Machine Learning, Dev	escalade, nouilles chinoises
Luc	resp formation OC	OC	formation, audiovisuel	
Céline	prof	CentraleSupelec	Ontologie, logique formelle, visualisation	

Stockage orienté colonnes

id	type
Nicolas	prof
Céline	prof
Luc	resp formation OC

id	lieu
Céline	Centrale Supelec
Nicolas	CNAM
Régis	OC
Luc	OC

id	spec
Nicolas	BDD
Nicolas	NoSQL
Régis	Machine Learning
Régis	Dev
Luc	formation
Luc	audiovisuel
Céline	Ontologie
Céline	logique formelle
Céline	visualisation

id	intérêts
Nicolas	BZH
Nicolas	Star Wars
Régis	escalade
Régis	nouilles chinoises

FIGURE 5: Le modèle orienté « colonnes»

Principaux modèles de bases de données NoSQL

Modèle orienté « graphe »

- Ce modèle (figure 4) qui repose sur la théorie des graphes, permet de représenter les données sous la forme de graphes.
- Le modèle s'appuie sur la notion de nœuds, de relations et de propriétés qui leur sont rattachées.
- Les entités sont alors les nœuds du graphe et les relations que partagent les entités sont alors des arcs qui relient ces entités.
- Ce modèle est notamment adapté au traitement des données des réseaux sociaux.
- Notons que les systèmes NoSQL orientés graphe trouvent un certain intérêt pour des applications dans le domaine du Web Sémantique, dans la gestion de bases de données de triplets RDF (*triple-stores*), permettant de stocker des connaissances ou ontologies, un triplet étant une arête d'un graphe.
- Les systèmes NoSQL orientés graphe les plus connus sont Neo4J, Infinite Graph, OrientDB.

MODÈLE ORIENTÉ « GRAPHE »

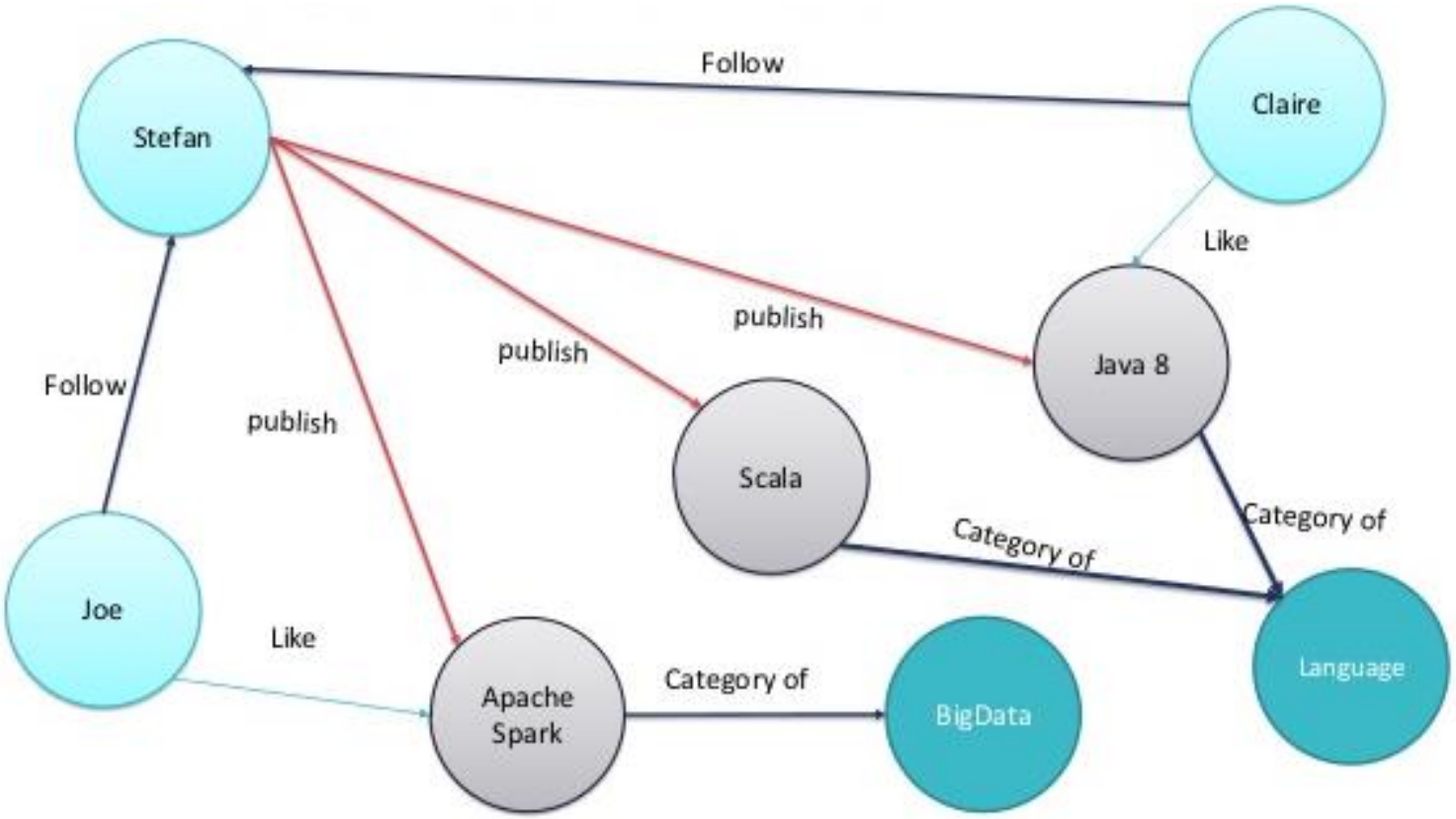


FIGURE 6: Le modèle orienté « graphe»

Alternatives au NoSQL : bases de données NewSQL

- Pour pallier ces limitations et « réconcilier » le monde SQL et le monde NoSQL, de nouvelles architectures de stockage des données sont apparues récemment, regroupées sous le terme de **systèmes NewSQL**.
- Ces systèmes ont pour objectif une amélioration des performances des systèmes relationnels grâce à de nouveaux moteurs de stockage, des technologies transparentes de fragmentation, de nouveaux logiciels et matériels.
- Les systèmes NewSQL doivent permettre une interrogation des données via SQL tout en garantissant des performances et un passage à l'échelle similaires aux bases de données NoSQL.
- De plus ces systèmes cherchent à préserver les propriétés ACID.
- Parmi les systèmes NewSQL, on peut citer Clustrix, NuoDB, VoltDB ou encore F1, récemment proposé par Google.
- Des produits Data Grid/Cache, sont aussi en émergence. Ils ont pour objectif une amélioration des performances notamment par un stockage des données persistantes en cache, une réplication des données distribuées, et un calcul exploitant le Grid.

Alternatives au NoSQL : bases de données NewSQL

- Mais les systèmes relationnels n'ont pas dit leurs derniers mots. Ainsi Stonebraker et al. (2010) comparent les systèmes NoSQL basés sur MapReduce aux systèmes relationnels parallèles commercialisés comme Terradata, Aster Data… Dans cette comparaison, ces derniers restent les plus performants au niveau du traitement des requêtes et des langages d'interrogation et interfaces qu'ils fournissent. Cependant avec MapReduce les systèmes NoSQL excellent dans les processus ETL (*Extract, Transform and Load*), et dans l'analyse d'ensembles de données semi-structurées en lecture seule. Enfin l'émergence de moteurs analytiques basés sur MapReduce a un impact sur les produits systèmes relationnels parallèles commercialisés. Ainsi certains systèmes, comme Aster Data, permettent maintenant l'invocation de fonctions MapReduce sur les données stockées dans la base de données comme une partie d'une requête SQL. La fonction MapReduce apparaît dans la requête comme une table résultante à composer avec d'autres opérateurs SQL. D'autres éditeurs de systèmes relationnels fournissent des utilitaires pour déplacer des données entre des moteurs MapReduce et leurs moteurs relationnels, ce qui facilite notamment le passage des données structurées, distillées en analyse sur la plateforme MapReduce, dans le système relationnel.

CHAPITRE 3
ANALYSE DES DONNÉES MASSIVES

- Dans cette section, nous nous intéressons à la problématique de la nature de l'analyse de très grands volumes de données qui dépend de la nature et de la structure de ces dernières
- Des différentes analyses mettent en œuvre divers algorithmes relevant de la fouille de données (*Data Mining*), de l'apprentissage machine automatique (*Machine Learning*), de l'aide à la décision, et voire de la visualisation.
- Dans ce chapitre:
  - nous soulignons tout d'abord l'intérêt de l'apprentissage automatique pour l'analyse de ces méga-données.
  - Ensuite nous distinguerons
    - l'analyse de méga-données stockées par exemple dans des systèmes NoSQL, et
    - l'analyse de méga-données échangées et émises en continu, par exemple des données en flots, qu'il n'est pas envisageable de stocker du fait de leur volume.
  - Aussi nous illustrerons quelques types d'analyses associées à des grands types de données massives: les données principalement composées de données numériques, données massives textuelles, méga-données issues du Web, liées à des réseaux, et enfin liées aux mobiles.
- Comme nous l'évoquons plus loin, chacun de ces types d'analyse possède ses propres caractéristiques et utilise des technologies plus ou moins spécifiques.

3.1 Intérêt de l'apprentissage automatique

- D'une façon générale, l'apprentissage automatique consiste à déterminer automatiquement un modèle formel, décrivant les données disponibles et permettant un certain niveau de généralisation sur des données nouvelles.
- L'objectif d'une méthode d'apprentissage est de déterminer le modèle qui minimise les erreurs de généralisation, c'est-à-dire qui permet d'obtenir une classification la plus exacte possible de données nouvelles.
- L'apprentissage s'effectue sur un jeu de données réduit.

3.1 Intérêt de l'apprentissage automatique

- Toute la difficulté réside dans l'obtention d'un bon équilibre entre qualité de la classification sur les données exemples (ce sur quoi l'apprentissage est optimisé) et sur les données nouvelles, par définition inconnues au moment de l'apprentissage.
- Si l'apprentissage est essentiellement statistique, via des modèles probabilistes dans les systèmes actuels, il peut aussi être symbolique par l'induction de règles, toujours à partir d'exemples, ou mixte comme c'est le cas avec les arbres de classification qui produisent, par analyse statistique, des règles de classification symboliques et compréhensibles par un opérateur humain.
- Par opposition, les méthodes d'apprentissage statistique produisent en effet des modèles « boîte noire », trop complexes pour être lisibles par un humain.

3.1 Intérêt de l'apprentissage automatique

- Relevons enfin que le coût de mise en œuvre des méthodes d'apprentissage réside essentiellement dans la nécessité de disposer de données annotées manuellement du moins dans le cadre de l'apprentissage classique dit « supervisé ».
- Une des grandes nouveautés de ces dernières années est l'apprentissage par transfert qui exploite des données très diverses, y compris non annotées, pour apprendre des tâches spécifiques, en quantité plus ou moins grande selon les méthodes choisies, mais nécessite aussi le paramétrage des méthodes, leur test et la mise à jour des modèles.
- La mise en œuvre de ces méthodes d'apprentissage suit un cheminement très différent des méthodes expertes traditionnelles qui souffrent d'être très spécifiques, peu robustes et surtout peu capables de passer à l'échelle des données massives et de tirer profit de ces gisements d'information.

3.2 Analyse de méga-données stockées

- Comme nous l'avons dit précédemment, le stockage et l'exploitation des grandes données nécessitent un partitionnement des données et aussi une distribution des traitements, des algorithmes, nécessaires à l'accès et à la gestion de ces données. Il en sera de même pour les traitements et les algorithmes d'analyse qui seront aussi distribués en utilisant ou non MapReduce.
- L'analyse des données massives nécessite la mise en œuvre de traitements et d'algorithmes, notamment d'apprentissage automatique et de fouille de données, qui doivent aussi être distribués pour être efficaces.
- Ainsi sur la base d'Hadoop, s'est développé le projet *Mahout* fournissant des versions distribuées de plusieurs algorithmes standards d'apprentissage automatique et de fouille de données, comme :
  - des algorithmes de factorisation de matrices, utilisés par exemple dans les systèmes de recommandation,
  - des algorithmes de classification tels que les k-means qui permettent d'organiser une collection de documents (ou plus généralement d'objets représentés sous formes de vecteurs) ou encore
  - des algorithmes de catégorisation tels que les forêts aléatoires (*Random Forest*) ou les classifieurs bayésiens.

3.2 Analyse de méga-données stockées

- La mise en œuvre de ces algorithmes distribués d'analyse de données, incluant la fouille, l'apprentissage, l'aide à la décision, et la visualisation, nécessite de les réécrire pour en proposer une version distribuée, mais aussi nécessite des environnements matériels spécifiques permettant de les exécuter en mode distribué, par exemple des machines multi-cœurs très puissantes ou des grilles de calcul (Grid computing).
- Les limites actuelles à la distribution des algorithmes d'analyse des données massives incluant la fouille, l'apprentissage, l'aide à la décision, la visualisation, résident tant dans la difficulté algorithmique, que dans l'architecture matérielle d'exécution.
- Ainsi la plupart des algorithmes de fouille ne se distribuent pas facilement, et pas nécessairement avec une approche de type MapReduce, et il est parfois nécessaire de trouver de nouvelles techniques pour réaliser un parallélisme efficace.
- Soulignons aussi le manque de langages standardisés devant permettre aux développeurs d'utiliser facilement les implantations parallèles existantes et d'en proposer de nouvelles.

3.3 Analyse de flots de données

- Les données massives ne concernent pas seulement les données stockées, par exemple dans des systèmes NoSQL. Elles concernent aussi les données échangées et émises en continu, comme des données en flots sur des médias en ligne, des données en provenance de capteurs, ou encore des relevés d'expérimentation dans le domaine de la physique.
- Lorsqu'il s'agit de requêter un flux de données continu, rapide et sans fin, il n'est pas envisageable d'interroger la totalité du flux, ce qui pourrait avoir pour conséquence de stopper le flux. De nouveaux algorithmes ont donc été optimisés en temps de traitement, et en occupation mémoire, pour répondre à cette contrainte d'exploration et d'analyse de données.
- Parmi les techniques les plus utilisées dans la fouille de flots de données, citons celles permettant de construire des résumés ou synopsis. Ces techniques n'explorent pas le flot entier, mais interrogent des données sélectionnées dans le flux, on accepte ainsi des résultats avec une certaine approximation (avec un taux d'erreur).
- Les fenêtres temporelles (time window) sont une de ces techniques travaillant sur un ensemble restreint du flux pour en extraire des motifs porteurs de connaissance (items, itemsets, et motifs séquentiels ou *Sequential patterns*).
- D'autres techniques comme les histogrammes, la compression, les sketches, l'échantillonnage statistique, permettent aussi de créer des résumés de flux de données et d'effectuer des fouilles sur ces résumés. Les résumés servent aussi à « ralentir » le flot de données quand il est trop rapide pour les machines effectuant l'analyse.

3.3 Analyse de flots de données

- Les grands défis que posent les flots de données résident tout d'abord dans le fait que les algorithmes nécessaires pour les traiter ne disposent que d'un espace mémoire réduit, et qu'il n'est possible de stocker qu'une faible proportion des données reçues.
- Ensuite ces algorithmes disposent d'un temps limité pour effectuer les traitements voulus, ils ne peuvent généralement effectuer qu'une seule passe sur les données.
- Cette complexité augmente si l'on cherche des résumés des données plus riches ou si les données possèdent une structure complexe, par exemple sous forme de graphes, comme dans les réseaux sociaux. Ainsi le déploiement d'algorithmes de prédiction de diffusion, dans de tels graphes pose d'importants problèmes algorithmiques .
- La nature dynamique des données en ligne a, de plus ,donné un nouvel essor aux travaux sur les séries temporelles, tels que la proposition de nouvelles méthodes pour réaliser des tâches de classification, supervisée ou non, et de prédiction sur les séries temporelles.
- Elle a aussi remis en avant les travaux sur la construction incrémentale de modèles, ainsi que ceux liés à la visualisation des données le défi étant ici de proposer des méthodes de visualisation qui permettent d'avoir un bon aperçu des données traitées sans trop sacrifier à la qualité du résumé proposé.

3.4 Analyse de données

- L'analyse des données repose principalement sur le principe de fouille de données et l'analyse statistique. La plupart de ces techniques reposent sur les technologies commerciales tels que les SGBD relationnels, les entrepôts de données, les ETL, l'analyse OLAP et l'analyse des processus.
- Depuis la fin des années 1980, les chercheurs en IA (Intelligence Artificielle), en algorithmique, et en base de données ont développé divers algorithmes d'exploration de données. Dans la conférence internationale ICDM 2006 en fouille de données, les dix algorithmes d'exploration de données les plus importants ont été identifiés, ceci sur la base d'experts, de nombre de citations, et sur une enquête communautaire.
- Par ordre d'importance décroissante on trouve : les arbres de décision (C4.5), la classification par k-moyenne (clustering de type k-means), la classification supervisée par approche discriminante et fonctions noyaux de type SVM (*Support Vector Machine*), Apriori, l'estimation automatique de distribution de probabilités par approche EM (*Expected Maximisation*), l'estimation de scores d'autorité ou de popularité par marche aléatoire au sein de graphes comme l'algorithme Google PageRank, la classification au moyen de classifieurs simples mais « boostés » comme AdaBoost, la catégorisation par calcul de similarités avec la méthode des kNN (k plus proches voisins), la classification probabiliste bayésienne naïve (*Naive Bayes*), et les arbres de classification CART.

3.4 Analyse de données

- Ces algorithmes couvrent:
  – la **classification** qui est l'attribution automatique d'un individu à une classe pré-existante ;
  – le **clustering**, regroupement automatique d'individus au sein d'un certain nombre de classes a priori inconnues ;
  – la **régression** qui est une estimation automatique d'une fonction mathématique permettant de faire correspondre des entrées, par exemple des vecteurs décrivant des individus et des sorties, ou encore des classes ou des valeurs numériques ;
  – l'**analyse d'association**, entre individus ou entre variables ;
  – l'**analyse de réseaux**, ou graphes.
- Notons que la plupart de ces algorithmes de fouille de données sont maintenant intégrés dans des systèmes de fouille de données commercialisés (Matlab notamment) ou *open source* (tels que Weka, l'environnement R ou SciKit pour Python). À ces algorithmes de base, il faut rajouter les développements récents consécutifs de la remise au goût du jour des approches d'apprentissage automatique par réseaux de neurones. Couplés en sortie à des classifieurs de type régression linéaire ou logistique ou à des classifieurs bayésiens, ils permettent d'apprendre des représentations riches d'individus en très grande quantité.

3.4 Analyse de données

- Issu de l'approche connexionniste (réseaux de neurones – réseaux convolutifs), l'**apprentissage profond** (*Deep Learning*) est actuellement l'objet d'un fort engouement des communautés scientifiques et d'ingénieurs qui proposent de nombreuses variantes et des architectures ouvertes (par exemple TensorFlow de Google). Difficile à paramétrer (architectures récurrentes ou non, nombre de couches, taille des vecteurs représentant les individus, fonctions d'activation…) et nécessitant un très grand nombre de données en exemple, l'apprentissage profond a, cependant, produit des résultats significativement meilleurs que toutes les autres méthodes pour la fouille d'images, la reconnaissance de la parole ou de caractères.
- Sur le plan technique, ces méthodes peuvent être mises en œuvre de façon massivement parallèle via l'exploitation de cartes GPU. Du fait des succès obtenus par les communautés de chercheurs en fouille de données et en analyse statistique, l'analyse des données continue d'être un domaine de recherche très actif, principalement basé sur des modèles mathématiques bien fondés et des algorithmes puissants, des techniques telles que les réseaux bayésiens, les modèles de Markov cachés (HMM), les machine à vecteur support (SVM), et les modèles d'ensemble.
- L'apprentissage statistique a été appliqué à de l'analyse de données, de textes et du Web.

3.4 Analyse de données

- Au-delà, c'est l'approche générale qui fait aussi l'objet de variantes plus ou moins efficaces selon les contextes et les données disponibles et manipulées. Supervisé (à base d'exemples pré-étiquetés ou classés), semi-supervisé (combinaison d'exemples étiquetés et d'exemples non étiquetés pour l'apprentissage) ou non supervisé (apprentissage d'un modèle à partir des seules données non étiquetées), l'apprentissage peut aussi être actif (sélection automatique des meilleurs exemples à étiqueter manuellement parmi un ensemble très important de données non étiquetées) ou 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).
- D'autres techniques d'analyse de données ont vu le jour pour explorer des données spécifiques, de la fouille de données séquentielles, temporelles ou spatiales, à la fouille de flux de données à haut débit et les données de capteurs. Les préoccupations croissantes de respect de la vie privée dans diverses applications d'e-commerce, e-gouvernement, et de santé ont conduit à l'émergence de techniques de fouille de données respectant cette vie privée (*privacy-preserving data mining*), utilisant généralement des techniques d'anonymisation, qui sont des méthodes dirigées par les données, ou des méthodes dirigées par les processus définissant comment les données peuvent être accédées et utilisées.
- Citons aussi la fouille de processus (*process mining*), basée sur l'analyse de données événementielles, et permettant la découverte de nouveaux processus ou le contrôle de conformité de processus, en exploitant des journaux d'événements (*event logs*) de plus en plus disponibles dans les organisations quel que soit le domaine, de l'industrie à la santé.

3.5 Analyse de textes

- Une partie importante du contenu non-structuré recueilli par une organisation est en format textuel, qu'il s'agisse de la communication par e-mail et des documents d'entreprise ou de pages Web et du contenu des médias sociaux.
- L'**analyse de texte** (*Text analytics*) relève de la **recherche d'information** (RI), de la **fouille de texte** (*Text Mining*) et de la **linguistique informatique**. Dans la RI, la représentation des documents et le traitement des requêtes sont les fondements de l'élaboration du modèle vectoriel, du modèle booléen, et du modèle probabiliste, qui sont à la base de l'exploitation des bibliothèques numériques modernes, des moteurs de recherche et des systèmes de recherche d'entreprise.
- En linguistique informatique, on dispose maintenant de techniques de **traitement automatique du langage naturel** (TALN ou NLP – *natural language processing*), principalement statistiques, mais aussi symboliques, pour l'acquisition lexicale et l'extraction de terminologie, la désambiguïsation sémantique, la reconnaissance d'entités nommées (noms propres, dates, quantités…), l'étiquetage syntaxique.
- En plus de la représentation de documents et de requêtes, des modèles d'utilisateur par retour de pertinence tenant compte des comportements et de l'expression de jugements (*relevance feedback*) sont également utilisés dans l'amélioration des performances de recherche.
- Notons que les moteurs de recherche actuels utilisent de plus en plus des techniques de TALN pour réduire l'impact de l'utilisation d'un vocabulaire varié entre les requêtes et les documents, pour exploiter des requêtes plus complexes ou tout naturellement pour mieux appréhender la sémantique des contenus.

3.5 Analyse de textes

- Tirant parti de la puissance des méga-données en apprentissage, et du TALN statistique pour construire des modèles de langue (distributions de probabilités décrivant comment les mots apparaissent les uns par rapport aux autres), les techniques d'analyse de textes ont été utilisées dans plusieurs domaines émergents, comme l'extraction d'information, les systèmes de question réponse (question en langue naturelle et extraction automatique des réponses au sein de de documents textuels) et en analyse d'opinions et de sentiments.
- L'**extraction d'information** vise à extraire automatiquement des types spécifiques d'informations à partir de documents. Elle peut être vue comme un moyen de structurer automatiquement des phrases ou des documents.
- La tâche de **reconnaissance d'entités nommées** (REN ou NER pour *named entity recognition*) est un processus qui identifie les éléments atomiques dans le texte, et les classe en catégories prédéfinies (par exemple, noms, lieux, dates).
- Les techniques de NER ont été développées avec succès notamment pour l'analyse des nouvelles (*news*) et dans l'extraction d'information dans le domaine biomédical. Actuellement les méthodes statistiques permettent d'extraire plus de 90 % des entités nommées. Cependant l'extraction de relations entre ces entités nommées est plus difficile, nécessite plus de sémantique, surtout lorsqu'il s'agit de relations n-aires (extraction d'événements).
- Les « *topic models* » correspondent à une famille d'algorithmes permettant de découvrir les principaux thèmes qui imprègnent une grande collection non structurée de documents. Dans ce cas, un thème n'est pas représenté par un mot mais par un vecteur de mots, chacun associé à un score.

3.5 Analyse de textes

- Plusieurs variantes ont été proposées, parmi lesquelles les *Author Topic Models*, qui permettent d'identifier les thèmes qui caractérisent les auteurs et les *Sentiment Topic Models* qui proposent de différencier les thèmes qui font l'objet de sentiments positifs des autres.
- Les **systèmes Question-Réponse** (*Question answering systems*) reposent sur des techniques du TALN, de la RI, et l'interaction homme-machine (IHM). Principalement conçus pour répondre à des questions factuelles (de type « qui, quoi, quand et où »), ces systèmes impliquent des techniques différentes pour l'analyse de la question exprimée par l'utilisateur en langue naturelle, la recherche de la source (quels sont les documents les plus susceptibles de contenir les réponses cherchées), l'extraction de la (ou des) réponse(s) et leur présentation aux utilisateurs. Les récents succès de Watson d'IBM et de Siri d'Apple ont mis en évidence le potentiel de ces systèmes Question-Réponse ainsi que des opportunités de commercialisation dans de nombreux domaines d'application, notamment l'éducation, la santé, et la défense.
- La **fouille ou analyse d'opinion** ou de sentiments (*sentiment analysis*) se réfère aux techniques pour extraire, classer, comprendre et évaluer les sentiments exprimés dans diverses sources en ligne, dans des commentaires sur les médias sociaux, et dans d'autres contenus générés par les utilisateurs. L'analyse de sentiments est l'objet de plusieurs variantes destinées à estimer l'affect, la subjectivité, et d'autres états émotionnels dans les textes en ligne. Le Web 2.0 et le contenu des médias sociaux ont créé de nombreuses opportunités passionnantes pour comprendre les opinions du grand public et des consommateurs en ce qui concerne les événements sociaux, les mouvements politiques, les stratégies d'entreprise, les campagnes de marketing, et les préférences de produits.

3.5 Analyse de textes

- L'**analyse de textes** offre également des opportunités et des défis de recherche importants dans plusieurs domaines plus ciblés, y compris l'analyse web « stylométrique » pour l'attribution d'auteur (détection de plagiat), l'analyse multilingue pour les documents web, et la visualisation à grande échelle de collections , par exemple l'environnement logiciel TXM (textométrie) ou Gephi pour les graphes. N'oublions pas le résumé automatique de textes, avec notamment ses approches extractives utilisant des techniques statistiques, appliquées à un seul document ou à un ensemble de documents.
- Dans le domaine multimédia, ces approches de fouille de texte sont exploitées via des modules de transcription automatique de la parole en texte, de reconnaissance du locuteur pour distinguer les intervenants dans un flux audio et structurer le document en sortie ou d'analyse d'images pour différencier des plans dans une vidéo.
- La prise en compte de critères extra-linguistiques (gestuelle, prosodie…) dans l'analyse de contenus multimédia fait l'objet de recherches intensives partant du constat que la compréhension humaine du langage ne peut être pleinement dissociée de son environnement.
- Enfin, comme pour l'analyse de données, l'analyse de textes tire profit d'implémentations orientées « Big Data » autour de MapReduce et d'Hadoop, des services du *cloud*, des bases NoSQL (par exemple Apache Solr) et des modules matériel de type GPU (par exemple Word2Vec).

3.6 Analyse du Web

- Depuis près de dix ans, l'analyse du Web (*Web Analytics*) constitue un thème de recherche très actif avec de nombreux challenges et opportunités. L'analyse du Web regroupe les méthodes et technologies relatives à la collecte, la mesure, l'analyse et la présentation des données utilisées dans les sites et applications Web.
- L'analyse du Web n'a cessé de croître, et est passée d'une simple fonction HTTP de journalisation du trafic, à une suite plus complète d'outils permettant le suivi, l'analyse et la création de rapports sur des données d'utilisation sur le Web. L'industrie et le marché de l'analyse du Web sont en plein essor.
- Elle s'appuie principalement sur les avancées en fouille de données, en recherche d'information (RI), et en traitement automatique des langues naturelles (TALN) déjà utilisé en l'analyse de textes.
- Les sites Web basés sur http et html inter-reliés, les moteurs de recherche sur le Web, et les systèmes d'annuaire pour localiser le contenu Web, ont contribué à développer des technologies spécifiques pour l'exploration des sites sur le Web (robots d'indexation – *web crawler* ou *web spider*), la mise à jour des pages Web, le classement des sites Web, et l'analyse des logs de recherche, maintenant intégrés dans les systèmes de recommandation.
- Cependant, l'analyse du Web est devenue encore plus excitante avec la maturité et la popularité des services Web et l'arrivée des systèmes du Web 2.0 dans le milieu des années 2000.
- Basés sur XML et les protocoles Internet http et smtp, les services Web offrent une nouvelle façon de réutiliser et d'intégrer des systèmes tiers ou existants. De nouveaux types de services Web et leurs API associées (interface de programmation d'application) permettent aux développeurs d'intégrer facilement des contenus divers issus de différents « *web-enabled* » systèmes.

3.6 Analyse du Web

- Citons par **exemple** REST (*Representational State Transfer*) pour invoquer des services à distance, RSS (*Really Simple Syndication*) pour le « *pushing* » de nouvelles, JSON (*JavaScript Object Notation*) pour les échanges légers de données, et enfin AJAX (*asynchronous JavaScript + XML*) pour l'échange de données et d'affichage dynamique.
- Ces modèles de programmation légers permettent la syndication et la notification de données, ainsi que les agrégations (*mashups*) de contenus multimédia (Flickr, Youtube, Google Maps) à partir de différentes sources, un processus web similaire au processus ETL (*Extraction, Transformation and Loading*) des entrepôts de données. La plupart des fournisseurs d'e-commerce ont fourni des API pour accéder à leur produit et au contenu de leur clientèle.
- Les services Web et les API continuent de fournir un flux impressionnant de nouvelles sources de données pour les mégadonnées. Une importante composante émergente dans la recherche en analyse du Web est le développement de plateformes et services de *cloud computing*, qui comprennent des applications, des logiciels système et du matériel fournis comme services sur Internet.

3.6 Analyse du Web

- Basé sur une architecture orientée services (SOA), sur la virtualisation des serveurs et sur l'informatique utilitaire (*utility computing*), le *cloud computing* peut être offert en tant que logiciel comme service (SaaS), l'infrastructure en tant que service (IaaS), ou plateforme en tant que service (PaaS).
- Les recherches en analyse du Web englobent maintenant la recherche et la fouille sociale, les systèmes de réputation, l'analyse des médias sociaux, et la visualisation Web.
- De plus, les ventes aux enchères sur le Web, la monétisation d'Internet, le marketing social et la confidentialité/sécurité du Web sont quelques-uns des axes de recherche prometteurs liés à l'analyse du Web.
- Beaucoup de ces nouveaux domaines de recherche peuvent compter sur les progrès dans l'analyse des réseaux sociaux, l'analyse de texte, et même dans la recherche en modélisation économique.

Chapitre 4
Des paradigmes de programmation distribués

- En général, la programmation distribué essaye d'utiliser la concurrence dans un programme, afin de réduire le temps de réponse pour résoudre un problème ou d'augmenter la taille des problèmes qui peuvent être résolus. La programmation distribué est donc plus performante pour solutionner un problème posé.
- Les modèles de programmation distribuée parallèle dépendent fortement des architectures parallèles sous-jacentes. Car nous ne pouvons pas programmer une machine avec une mémoire partagée de la même manière qu'une machine avec une mémoire distribuée.

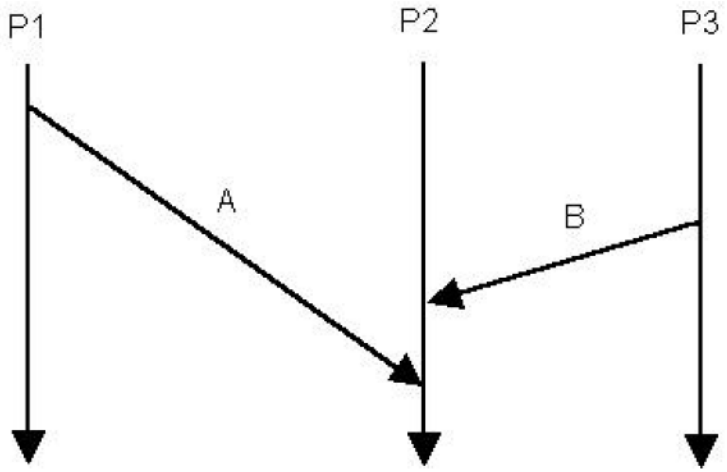
Chapitre 4
Des paradigmes de programmation distribués

- Ainsi pour la programmation parallèle et même distribué, nous ne pouvons pas appliquer les mêmes principes qu'en programmation séquentielle. En programmation parallèle le programmeur doit décider s'il applique le parallélisme de données, c'est à dire si on fait la même chose sur les données différentes, ou s'il opte pour un parallélisme de contrôle sur l'ordonnancement des tâches ou encore s'il applique les deux parallélismes simultanément.

Chapitre 4
Des paradigmes de programmation distribués

- Ainsi, nous pouvons définir différents grains de parallélisme qui peuvent aller du niveau du programme jusqu'au niveau des bits. D'autres problèmes sont dus à des communications et synchronisations de tâches parallèles. Dés lors, avec des délais introduits par des réseaux et l'absence du temps global (voir figure suivante) dans un système distribué, l'exécution d'un programme parallèle n'est pas totalement déterministe.
- En effet, par exemple suite à la charge d'un réseau ou encore à la différence des distances entre les lieux d'exécution des tâches parallèles, les communications ne peuvent pas garantir l'ordre causal des messages, ce qui pose entre autre un problème pour le débogage et la vérification sémantique des programmes parallèles.

Modèles dChapitre 4
Des paradigmes de programmation distribué



- *Visualisation de l'absence de temps global externe (avec différence des distances et différents délais de réseau). "Un message A est plus ancien qu'un message B si A et B arrivent sur le même processeurs P*

PARADIGME DE PROGRAMMATION *MULTI-THREAD*

- La programmation distribué parallèle la plus simple consiste à avoir plusieurs *threads* ou flots d'instruction qui s'exécutent en manipulant des données stockées dans une mémoire commune. Elle est donc naturellement le type de programmation le plus employé sur les machines SMP.
- La programmation peut se faire avec des outils spécialisés comme les bibliothèques implantant la norme POSIX. Elle peut aussi utiliser des langages connus qui intègrent les *threads* dans leur sémantique, comme Java. La difficulté de cette programmation est de garantir la sémantique correcte du programme en synchronisant les accès à la mémoire entre calculs se déroulant de manière concurrente.
- Pour obtenir des performances, il suffit d'avoir toujours suffisamment de calculs concurrents à faire pour occuper les différents processeurs. Par exemple, on se base sur le principe du *multi-threding* où chaque processeur exécute en alternance plusieurs *threads*. Après un certain temps de calcul ou quand un *thread* passe dans un état d'attente le processeur bascule alors à un autre *thread* qui est en attente d'exécution.

L'avantage du *multi-threading*

- Utilisable sur des architectures sans mémoire partagée où la communication et des caractéristiques réseau comme le délai jouent des rôles plus importants.
- Avec le principe de *multi-threading* il est alors facile de combler par exemple l'attente d'une synchronisation ou d'une communication, comme l'attente d'une donnée d'un autre *thread*, par d'autres calculs (principe *asynchrone*) ou de remplir le temps de la communication en terme de délai du réseau par des calculs (principe *non-blocking*) en implémentant des *threads* qui gèrent les protocoles de communication.
- Cette utilisation du *multi-threading* est aussi connu dans la littérature sous le nom de parallélisme de tâche (ou parallélisme fonctionnel) et consiste donc de décomposer un programme en plusieurs tâches qui peuvent alors être exécutées simultanément.

Désavantage du *multi-threading*

- le sur-coût qui est induit à cause du basculement entre *threads*. En effet, on doit enregistrer l'environnement d'exécution du *thread* qu'on veut mettre en attente et chargé les registres du processeur avec le contexte d'exécution du nouveau *thread*.
- Cependant, il existe des architectures avec des jeux de registres qui permettent de garder plusieurs environnements d'exécution de *threads* à la fois.
- Une autre solution pour diminuer le sur-coût du passage entre les *threads* sur un processeur est de créer un environnement qui implémente un passage entre des *threads* qui est plus performant et spécialisé à une classe de problème.
- Ainsi, l'environnement cache le *multithreading* devant le système exploitation qui, lui, doit fournir de la gestion des *threads* beaucoup plus complet et dès lors plus lourds.
- Les *threads* pris en charge par un tel environnement sont souvent appelés *user-level threads*.

Le paradigme *message-passing*

- La programmation d'une machine à mémoire distribuée impose plusieurs contraintes. Il faut exprimer explicitement quelles vont être les données transmises et à quels moments les transmissions doivent se faire. La façon standard de programmer est d'utiliser une bibliothèque d'échanges de messages comme MPI ou PVM , qui permet de s'abstraire de la réalité physique de la machine ou du réseau, et de se concentrer sur les échanges de données. En effet, ces bibliothèques sont implémentées pour la plupart des architectures parallèles et permettent donc d'écrire en principe des programmes portables et indépendants des machines.
- Dans le paradigme de programmation *message-passing*, les programmeurs voient leurs programmes comme une collection de processus avec une mémoire privée. En général, en *message-passing* on travaille avec autant de processus que de processeurs. Ainsi, l'environnement de *message-passing* peut affecter un processus à un processeur et l'échange de message entre processus revient à un échange de message entre processeurs. C'est aussi l'environnement qui spécifie un identificateur pour chaque processus.

Le paradigme message-passing

- La communication entre processus et l'échange de données se fait en général avec deux primitives fournies par les bibliothèques d'échange de messages. La première est la primitive d'envoi send. Sa syntaxe est toujours proche de la suivante:

  - send(buffer, taille, destinataire, type)

– buffer est une référence en mémoire qui indique où trouver le premier octet du message à envoyer

– taille est un entier indiquant la taille du message

– destinataire est l'identification du processus destinataire

– type est le type du message, souvent un entier. Il permet de classer les messages du point de vue de l'application.

Le paradigme message-passing

- La deuxième, le complémentaire de send, est la primitive de réception receive. Sa syntaxe est proche de :

receive(buffer, taille, expéditeur, type)

– buffer est une référence en mémoire qui indique où devra être déposé le premier octet du message

– taille est un entier indiquant la taille du message

– expéditeur est l'identification du processus l'expéditeur

– type est le type du message attendu.

Un message peut seulement être envoyé si l'expéditeur appelle explicitement la primitive send et le destinataire sa primitive receive.

Le paradigme message-passing

- Ces deux primitives sont en général disponibles en plusieurs modes de communication. Citons ici quatre modes de comportement possibles dans une communication synchrone, asynchrone, bloquante et non-bloquante.
- Par exemple, les primitives par défaut en MPI sont asynchrones et bloquantes.
  - Bloquant veut dire que l'appel à send ou receive retourne uniquement si le buffer associé est libéré, c'est à dire pour le send que l'outil de communication a "copié" le message pour l'envoi et pour le receive que le message reçu se trouvent dans le buffer (voir figure 2).
  - Et asynchrone exprime ici le fait que send ne sait pas si receive a reçu le message ou pas.

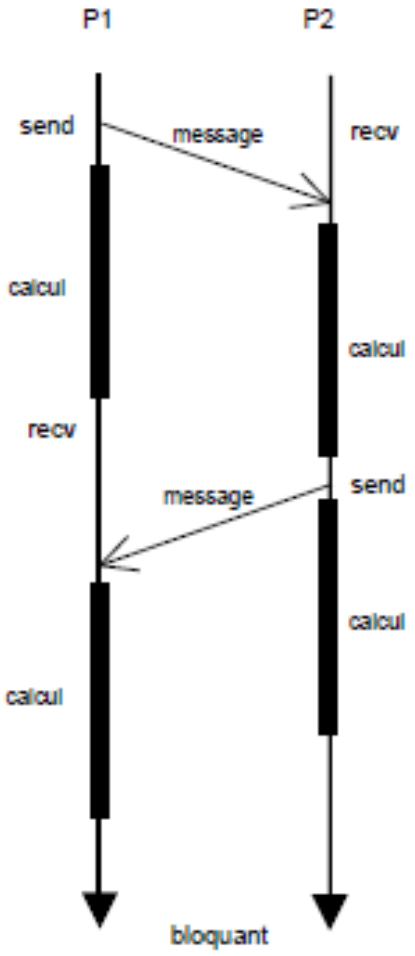
Le paradigme message-passing

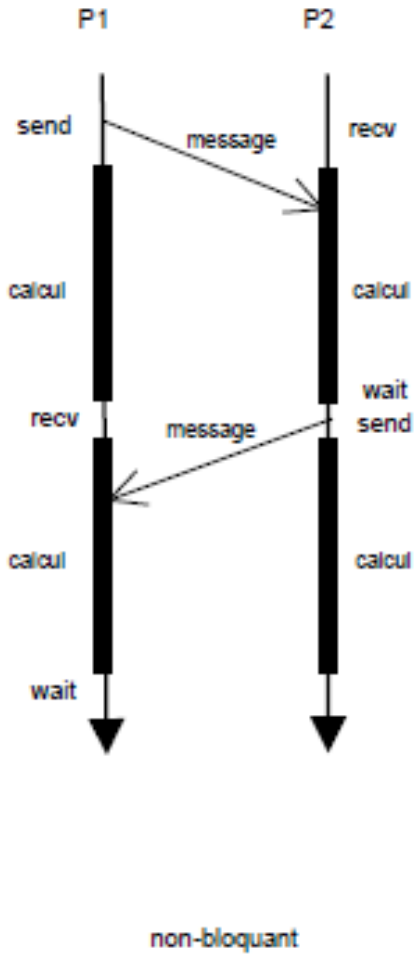
- En général, les bibliothèques fournissent aussi des primitives pour le non-bloquant et le synchrone. Le principe non-bloquant est introduit pour augmenter les performances du système.
- En effet, les primitives non-bloquantes retournent directement et rendent ainsi possible de faire des calculs pendant la communication. Dans le cas du send il y un risque d'écraser le buffer avant que le message ne soit envoyé.
- Le receive est complété par un appel de wait qui est bloquant jusqu'à ce que le message soit disponible dans le buffer qui est spécifié par receive. Synchrone exprime ici le fait que l'expéditeur est sûre que le message est arrivé chez le destinataire, c'est à dire que le send retourne uniquement si le receive correspondant est terminé.

Le paradigme message-passing

- En plus des deux primitives principales, les modèles de *message-passing* fournissent en général aussi des moyens de communication plus globaux ou plus performants adaptés à des cas particuliers. Ce sont les communications collectives. Par exemple, pour envoyer un message à tous les processus il existe un appel broadcast (diffusion général) et pour récolter des résultats d'un calcul de tous les processus chez un processus un appel reduce.
- Néanmoins, les communications sur des machines à mémoire distribuée sont toujours assez coûteuses en temps par rapport à leur puissance de calcul. Cela induit que la programmation parallèle devient plus complexe : il faut essayer de minimiser les communications, dans le même temps, que de partager les données et les activités pour maintenir les processeurs occupés.

Le paradigme message-passing





Chapitre 5
MapReduce

- Veuillez consulter l'expose ainsi que les informations présentées dans les chapitres précédents.

Chapitre 6
L'avenir des grandes données

- Les données massives 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 d'autres systèmes pour extraire des informations qui nous concernent…. Donnons, d'une façon non exhaustive, quelques exemples d'usage des méga-données dans différents grands domaines d'activité.

6.1 Domaine de la recherche scientifique

- Dans le domaine scientifique et technique, les scientifiques et ingénieurs font face à des méga-données notamment générées automatiquement par des capteurs ou instruments de mesure.
  - Par **exemple** dans le domaine de l'astronomie, en huit ans (2000-2008), le Sloan Digital Sky Survey, un grand programme d'observation astronomique, a enregistré 140 téraoctets d'images (140.1012). Mais il ne faudra que cinq jours à son successeur, le LSST (Large Synoptic Survey Telescope) pour acquérir ce volume.
  - En physique, dans sa quête du boson de Higgs, le grand collisionneur de hadrons (LHC) a amassé de son côté, chaque année, près de 15 pétaoctets de données (15.1015), l'équivalent de plus de 3 millions de DVD.
  - En recherche médicale, les technologies associées aux méga-données ont permis des avancées spectaculaires dans l'analyse du génome humain : alors qu'il a fallu dix ans, et plus de 2 milliards d'euros pour réaliser le premier séquençage humain complet, il est maintenant possible d'en réaliser un en quelques jours et pour environ mille euros. Ces connaissances sur le génome, couplées à d'autres, permettent de mieux comprendre l'évolution de pathologies, d'améliorer les mesures de prévention ou encore les protocoles de soins.

3.2 Domaine de la santé

- Concernant le domaine de la santé, dans le rapport rendu public « *The big data revolution in healthcare* », McKinsey évalue entre 300 et 450 milliards de dollars les sommes que le « big data » pourrait faire économiser au système de santé américain, sur un total de 2 600 milliards.
- Ces économies concernent notamment la prévention, avec un suivi des patients les incitant à changer leurs habitudes, le diagnostic, en aidant les médecins à choisir le traitement le plus approprié, le personnel médical, en déterminant si le patient a besoin d'une infirmière, d'un généraliste ou d'un spécialiste, la maîtrise des coûts, à la fois en automatisant les procédures de remboursement et en détectant les fraudes, et enfin l'innovation, à travers les multiples apports du calcul intensif à la compréhension du vivant et à l'amélioration des traitements.
- De même grâce aux mégadonnées, il est possible de mieux prévenir certaines maladies ou épidémies, ou d'améliorer le traitement des patients. Ainsi en analysant les recherches des internautes sur Google, une équipe est parvenue à détecter plus rapidement l'arrivée des épidémies de grippe.
- Autre exemple, en s'intéressant aux données disponibles sur Facebook, des chercheurs ont détecté les adolescents ayant des comportements à risque pour cibler les campagnes de prévention.

6.3 Domaine socio-économique et politique

- Dans le domaine socio-économique, de façon générale, les mégadonnées peuvent être utilisées pour simplifier ou adapter des services offerts, ceci en écoutant mieux les usagers, en comprenant mieux leurs modes d'utilisation de ces services.
- Ainsi Google Analytics propose par exemple aux entreprises, comme aux administrations publiques, d'améliorer la conception de leur site internet par l'analyse des visites des internautes.
- Dans l'éducation, avec le télé-enseignement (dont les *Massive Open Online Courses* – MOOC), le traitement de mégadonnées permet d'analyser les activités des élèves (temps consacré, façon de suivre les programmes, arrêt-retour dans les vidéos pédagogiques, recherches Internet parallèles, etc.) pour améliorer les modes d'enseignement. L'analyse des mégadonnées permet aussi de mieux comprendre les sentiments ou les besoins des citoyens.
- Ainsi lors de la campagne de réélection de Barack Obama en 2012, les conseillers ont analysé en temps réel les messages sur Twitter pour adapter en direct le discours du président.

6.4 Domaine du transport et de l'énergie

- Dans le domaine des transports, les déplacements des populations peuvent être modélisés pour adapter les infrastructures et les services (horaires des trains, etc.).
- À cette fin, les données provenant des *pass* de transports en commun, des vélos et des voitures partagées, mais aussi de la géolocalisation (données cellulaires et systèmes de localisation par satellites) de personnes ou de voitures, sont utilisées.
- Dans le domaine de l'énergie et du développement durable, les systèmes de compteurs intelligents (électricité, gaz, eau) génèrent des mégadonnées qui permettent de rationaliser la consommation énergétique.
- En plus d'offrir aux citoyens la possibilité de mieux contrôler leur consommation, ces compteurs permettent de couper à distance, avec l'accord des clients, l'alimentation d'équipements pour éviter les surcharges du réseau.
- Dans le transport aérien, en associant les données issues de capteurs sur les avions à des données météo, on peut modifier les couloirs aériens pour réaliser des économies de carburant, on améliore la conception, la maintenance des avions ou leur sécurité.