Résumé détaillé : Partie 1 - Big Data

Définition du Big Data

Le Big Data, aussi appelé "mégadonnées" ou "données massives", désigne des ensembles de données si volumineux, rapides et complexes qu'ils dépassent les capacités des outils classiques de gestion et d'analyse. Il touche de nombreux domaines : navigation Web, réseaux sociaux, objets connectés, etc. D'ici 2025, on prévoit 75 milliards d'objets connectés.

Les facteurs déclencheurs de l'ère du Big Data

1. **Torrent croissant de données :**
    - Doublement des données digitales tous les deux ans.
    - Augmentation de 40 % des données mondiales entre 2013 et 2025.
    - Génération continue à un rythme rapide.
2. **Capacités de stockage :**
    - Stockage à grande échelle, moins coûteux et très performant.
3. **Informatique avancée :**
    - Algorithmes distribuées.
    - Base de données NoSQL adaptées aux données complexes.

Les caractéristiques principales du Big Data : Les 5V

1. **Volume :** Quantité massive de données générées chaque seconde.
2. **Vélocité :** Vitesse de génération et de traitement des données.
3. **Variété :** Diversité des formats de données :
    - Structurées : données relationnelles.
    - Semi-structurées : XML, JSON.
    - Non-structurées : images, vidéos, fichiers texte.
4. **Véracité :** Fiabilité et exactitude des données.
5. **Valeur :** Potentiel économique et analytique des données.

Disciplines et technologies du Big Data

1. **Mathématiques et Informatique :**
    - Algorithmes d'apprentissage automatique, statistiques et fouille de données.
    - Cœur mathématique du Big Data, souvent identifié comme la "Data Science".
2. **Informatique distribuée :**

    - Analyse de données à grande échelle avec des systèmes distribués (e.g., MapReduce).
    - Traitement au plus près des données pour réduire les délais.
3. **Informatique parallèle :**
    - Utilisation de clusters de calcul (PC multi-cœurs, GPU) pour accélérer le traitement.
    - Essentiel pour des techniques comme le Deep Learning.
4. **Bases de données NoSQL :**
    - Adaptées pour gérer des données non structurées et semi-structurées à grande échelle.
    - Offrent des performances optimisées pour des requêtes rapides.

Applications typiques du Big Data

1. **Analyse comportementale :**
    - Traces des transactions, activités mobiles, accès web.
    - Détection de comportements pour des systèmes de recommandation ou analyses prédictives.
2. **Maintenance prédictive :**
    - Analyse des signaux captés par des capteurs industriels.
    - Identification en temps réel des prémisses de défaillances.
3. **Analyse de réseaux sociaux :**
    - Étude des relations et influences entre individus ou groupes.
    - Utilisation de graphes pour comprendre les interactions.

Sources de données du Big Data

- Internet et mobiles.
- Réseaux sociaux.
- Géolocalisation.
- IoT (Internet des Objets) et capteurs.
- Cloud computing.
- Médias en streaming.

Défis du Big Data

1. **Hétérogénéité des données :**
    - Données complexes provenant d'applications et utilisateurs différents.
    - Liens explicites (URLs) ou implicites à extraire.

2. **Infrastructure :**
    - Nécessité de clusters/serveurs multiples pour stocker et traiter les données.
    - Algorithmes pour la gestion distribuée et le calcul à grande échelle.

Résumé

Le Big Data repose sur la gestion efficace de volumes de données massifs, générés rapidement et provenant de sources diverses. Il nécessite des technologies avancées (NoSQL, informatique parallèle et distribuée) et des approches analytiques sophistiquées pour offrir des insights exploitables dans des domaines variés comme les réseaux sociaux, l'industrie, ou le commerce.

Partie 2. Hadoop

1. Hadoop

Hadoop est un **framework open-source** développé en Java, destiné au développement d'applications **distribuées et scalables**. Il permet la gestion de milliers de nœuds et de pétaoctets de données.
Hadoop s'inspire de deux concepts principaux :

1. **Google File System (GFS)** : système de fichiers distribué.
2. **MapReduce** : modèle de programmation pour le traitement distribué des données.

---

2. HDFS (Hadoop Distributed File System)

- Le HDFS est le système de fichiers distribué utilisé par Hadoop.
- Les données sont découpées en **blocs partitionnés** et **répliquées sur trois nœuds** (deux sur le même support et un sur un autre).
- Structure des composants :
    - **NameNode** : gère l'arborescence des fichiers et l'espace de nommage.
    - **Secondary NameNode** : sauvegarde périodique des métadonnées pour suppléer au NameNode en cas de panne.
    - **DataNode** : stocke les blocs de données et communique avec le NameNode.
    - **JobTracker** : coordonne les tâches MapReduce.
    - **TaskTracker** : exécute les tâches MapReduce sur les nœuds.

Commandes HDFS

- Stocker un fichier : `hadoop fs -put fichier.txt /data_input/`
- Récupérer un fichier : `hadoop fs -get /data_input/fichier.txt fichier_local.txt`
- Créer un répertoire : `hadoop fs -mkdir /data_input`
- Supprimer un fichier : `hadoop fs -rm /data_input/fichier.txt`

Inconvénients de HDFS

- **NameNode unique** : point de défaillance unique (atténué dans les versions récentes).
- Optimisé pour des **lectures concurrentes**, mais les écritures simultanées sont moins performantes.

---

3. MapReduce

MapReduce est un paradigme qui permet de traiter de grandes quantités de données de manière **distribuée** et **parallèle**.

Fonctionnement

- **Étape Map** : divise les données d'entrée en **couples clé/valeur**.
- **Étape Reduce** : applique un traitement aux valeurs associées à une clé pour produire un résultat.
- **Étapes principales** :
    1. **Split** : diviser les données en fragments.
    2. **Map** : générer des couples clé/valeur.
    3. **Shuffle & Sort** : grouper les couples par clé.
    4. **Reduce** : traiter les groupes pour produire le résultat final.

Avantages

- Programmation simplifiée.
- Gestion automatique des pannes, de la répartition du travail et de la synchronisation.

---

4. Écosystème Hadoop

Hadoop dispose d'un écosystème riche comprenant des outils complémentaires :

HDFS

- Système de stockage distribué pour le Big Data.
- Divise les fichiers entre les nœuds pour un accès parallèle.

YARN (Yet Another Resource Negotiator)

- Gestionnaire de ressources pour Hadoop.
- Supporte plusieurs frameworks (MapReduce, Spark, Flink).
- Composants :
    - **Resource Manager** : gère les ressources globales (mémoire, CPU).
    - **Application Manager** : suit l'exécution des programmes.
    - **NodeManager** : exécute les tâches sur chaque nœud.

Autres outils

- **Hive** : langage de requêtes de type SQL pour l'analyse.
- **Pig** : script de flux de données.
- **HBase** : base de données NoSQL pour des données en tables clairsemées.
- **ZooKeeper** : gestion de configuration pour systèmes distribués.
- **Spark** : traitement rapide des données en mémoire.
- **Giraph** : analyse de graphes sociaux.

---

5. Exemple YARN

- Un client soumet un programme au **ResourceManager**.
- L'**Application Manager** exécute le programme dans un container dédié.
- Les **NodeManagers** attribuent et exécutent les tâches, signalant leur progression à l'Application Master.