République Algérienne Démocratique et Populaire
Ministère de l'enseignement supérieur et de la recherche scientifique

UNIVERSITE Dr. TAHAR MOULAY SAIDA

FACULTE : TECHNOLOGIE

DEPARTEMENT : INFORMATIQUE

MEMOIRE DE MASTER

OPTION : sécurité informatique et cryptographie

Thème

Une approche de Dé-identification des données pour la préservation de la vie privée des Big Data

Présenté par :
Taouche sabrina
Benyamina khadidja

Encadré par :
Mr A.Zahaf

Promotion: 2017-2018

Remerciements

D'abord Nous Remercions Dieu le tout puissant qui nous guidé à réaliser ce travail.

Nous adressons nos sincères gratitudes à notre encadreur « Dr. Ahmed ZAHAF » pour son suivi et son aide.

Nos remerciements s'adressent aussi à tous nos enseignons du département d'informatique.

A tous ceux qui nous ont aidé et encouragé de près ou de loin.

Table des matières

Résumé

La Publication des données personnelles sans pour révéler des informations sensibles est un problème important. Au cours de ces dernières années, différentes approches ont été proposées comme solution. Dans le même cadre, le présent travail introduit une approche qui se base sur la préservation de la confidentialité lors de la publication et l'analyse des données. Cette approche fournit une meilleure protection de la vie privée des données sans affecter leurs usage et exploitation par les utilisateurs.

Mots clés : vie privée, Dé-identification des données, Big Data, Hadoop, Mapreduce

Liste des figures

Liste des tableaux

Introduction générale

L'augmentation récente du nombre de dépôts de données volumineux par les entreprises et les gouvernements a donné plus de crédit au développement de systèmes décisionnels fondés sur l'information grâce à l'exploration de données.

Par exemple, tous les hôpitaux autorisés en Californie doivent soumettre à la California Health Facilities Commission des données spécifiques à chaque personne (Zip, date de naissance, dates d'admission et de sortie, principale langue parlée) pour que ces données soient disponibles. Les parties intéressées (par ex : assurance, chercheurs) pour promouvoir l'accessibilité équitable des soins de santé pour la Californie .En 2004, le Comité consultatif sur les technologies de l'information du président des États-Unis publiait un rapport intitulé Révolutionner les soins de santé par la technologie de l'information, afin de mettre en place un système de dossier médical électronique national pour promouvoir et encourager le partage des connaissances médicales. Dans tout le système d'aide à la décision clinique assisté par ordinateur. La publication de données est également bénéfique dans de nombreux autres domaines. Par exemple, en 2006, Netflix (société de location de DVD en ligne) a publié 500 000 données d'audience de films pour encourager la recherche afin d'améliorer la précision des recommandations de films sur la base de préférences personnelles. Depuis octobre 2012, les gouvernements du Canada et des États-Unis ont lancé un projet pilote intitulé «Projet pilote d'entrée / sortie», pour partager les données biographiques des voyageurs qui traversent la frontière entre les États-Unis et le Canada. Agence (ASFC) et le Département of Homeland Security (DHS). Ceci est un exemple de partage de données entre deux gouvernements.

Des volumes considérables de données sont créés tous les jours à partir des données utilisateur générées automatiquement sur Internet. Réseaux sociaux, appareils mobiles, messagerie électronique, blogs, vidéos, transactions bancaires et autres interactions utilisateur pilotent désormais les campagnes Marketing, les études sociodémographiques, les enquêtes de polices et les intentions électorales, en établissant une nouvelle dimension appelée Big Data.

Les moteurs de base de données basés sur le standard SQL et créés dans les années 1970 ont de bonnes performances lors du traitement de petites quantités de données relationnelles mais ces outils sont très limités face à l'expansion des données en volume et en complexité. Le traitement MPP créé initialement au début des années 1980 a amélioré légèrement les indicateurs de performance pour les volumes de données complexes. Cependant, ce traitement n'a pas pu être utilisé pour le traitement des données non-relationnelles à expansion permanente.

Des outils puissants sont requis pour stocker et exploiter ces données en expansion quotidienne dans le but de soumettre un traitement simple et fiable des données récoltées des utilisateurs. Des résultats rapides et de bonne qualité sont attendus. Pour les industriels et les décideurs en général, ces résultats sont aussi importants que les plus lourds investissements métier. Les opérateurs de modélisation traditionnels sont confrontés à leurs limitations dans ce défi, puisque les informations se multiplient en volume et complexité, une chose qui actuellement ne peut être gérée que par des techniques de modélisation non-relationnelles. Hadoop MapReduce est considéré comme la technique de traitement la plus efficace, comparée aux bases de données SQL et au traitement MPP. Hadoop dispose d'une performance proportionnelle à la complexité des données volumineuses. C'est un outil efficace pour résoudre les problèmes de données massives mais c'est aussi un concept qui a changé l'organisation des systèmes de traitement en large échelle. Cependant malgré le succès qu'il a eu, ce modèle n'a pas encore atteint son aspect final en tant que solution informatique mature. Au contraire, il s'agit d'un point de départ vers d'autres perspectives.

Le traitement des Big data met en évidence un compromis entre la facilité de traitement et sécurité-confidentialité des données et des calculs. Plus précisément, le déploiement de MapReduce dans un cloud public gère les ressources. Mais, un cloud public ne garantit pas la sécurité rigoureuse et l'intimité des données stockées [7].

Les aspects de sécurité dans le contexte de MapReduce sont cruciaux pour authentifier et autoriser utilisateurs, audit-confidentialité-intégrité des données et calcul, disponibilité des mappeurs et des réducteurs, et vérification des résultats. La sécurité de MapReduce assure une fonctionnalité légitime du Framework. Un framework MapReduce sécurisé gère les attaques suivantes: attaques sur authentification (attaques d'usurpation d'identité et de rejeu), attaques contre la confidentialité (écoute et attaques de type "man-in-the-middle"), altération des données (modification des données d'entrée, sorties intermédiaires et les sorties finales), la falsification matérielle, la falsification logicielle (modification des mappeurs et des réducteurs), déni de service, interception-libération de données ainsi que des calculs, et analyse de communication[7].

Au contraire, les aspects de la vie privée supposent une fonctionnalité légitime du cadre et sont donc construit sur le dessus de la sécurité. En plus du cadre fonctionnant correctement, la vie privée dans le contexte de MapReduce est une capacité de chaque partie participante (fournisseurs de données, fournisseurs de cloud et utilisateurs) à empêcher d'autres parties, éventuellement antagonistes, d'observer des données, des codes, des calculs et des extrants.

Afin d'assurer la confidentialité, un algorithme MapReduce dans les nuages publics cache le stockage de données ainsi que le calcul aux nuages publics et aux utilisateurs adversaires. Distinction supplémentaire entre sécurité et la vie privée est que la sécurité est beaucoup plus un problème binaire,

c'est-à-dire que l'attaque réussit ou non, alors que Dans un contexte de confidentialité, il existe un compromis entre la confidentialité des données et l'utilisation du cadre [7].

L'objectif de ce mémoire est de traiter une problématique universelle et d'actualité qui est la comment préservé de la vie privée dans les big data, l'idée de base est de proposer une approche de dé-identification des données. La dé-identification sert à appliquer deux mécanismes/ la généralisation des valeurs et la suppression des données. Les nouvelles données diffèrent totalement des données originales ce qui implique une grande protection. Nous mesurons la perte d'information après dé-identification sur deux sur deux ensembles de données distinctes. Puis, nous démontrons le fonctionnement de notre approche sur un ensemble de requêtes types.

Notre mémoire est organisé en quatre chapitres :

Dans le chapitre 1 nous présentons une vue générale sur Big Data, et architecture de system Big Data, Intérêts des Big Data, enjeux de Big Data, cas d'usage du Big Data, limites des Big Data, avenages ; Puis il est venu Hadoop pour remplacer Big Data Nous avons essayé d'en donner un bref historique, et une introduction complète à celui-ci,et architecture de Hadoop nous avons également mentionné les principaux composants HDFS et MapReduce aussi bien composent apache hadoop.

Dans Le chapitre 2 Nous avons parlé tous les détails sur Confidentialité Conservation de la publication et de l'analyse des données , définition complète de La confidentialité différentielle dans la vies privées , Il y a Mécanismes de confidentialité différentielle Compter sur eux confidentialité différentielle lesquels sont deux mécanisme de Laplace et Le mécanisme exponentiel ,et les aspects de la vie privée dans MapReduce , La principale chose sur laquelle repose la vie privée est la confidentialité c'est la confidentialité des données avec les utilisateurs adversaires

Dans le chapitre 3 présents problèmes de la vie privée des données contre un utilisateur malveillant résoudre son propre problème et nous proposons deux mécanismes pour la dé-identification : la généralisation et la suppression, nous offrons Evaluation et Métrique de perte.

Dans le chapitre 4 nous présentons notre approche L'architecture de notre application, En fin nous terminons par une brève conclusion qui résume notre travail et présente quelques perspectives.

Chapitre 1

Chapitre 1

Big Data

1.1 Introduction

De nos jours, l'humanité produit chaque année un volume d'informations numériques de l'ordre du Zettaoctet. Soit presque autant d'octets qu'il existe d'étoiles dans l'Univers ! En 2010 déjà, Eric Schmidt, le patron de Google, estimait que « tous les deux jours, nous produi sons autant d'informations que nous en avons générées depuis l'aube de la civilisation jusqu'en 2003 ». Des chiffres et des phrases chocs sur le sujet qui agite les mondes scientifique et économique aujourd'hui : les très grands volumes de données. Les chiffres parlent d'eux-mêmes. Chaque seconde, plus d'une heure de vidéo est mise en ligne sur YouTube, et plus de 1,5 million d'e-mails sont envoyés. Nous somme entré dans l'ère du Big Data. ***[1]***

Ce chapitre présent la notion du big data : Architecture**,** Intérêts, les enjeux, cas d'usage du big data. Nous présentons ensuite les limites des big data ainsi que son avantage.

Dans l'aspect pratique big data à été implémente a travers une panoplie de Framework dont hadoop est l'un d'entre eux.

1.2 Définitions

Big Data littéralement les « grosses données », « mégadonnées » ou encore « données massives » c'est l'appellation couramment utilisée par les spécialistes pour évoquer les grandes masses de données numériques. Il désigne des ensembles de données qui deviennent tellement volumineux qu'ils en deviennent difficiles à travailler avec des outils classiques de gestion de base de données ou de gestion de l'information. Le Big Data se caractérise par trois dimensions (3V) qui sont le Volume, la Variété et la Vélocité, certains auteurs ont rajoutés d'autres V comme la Valeur et la Visualisation.

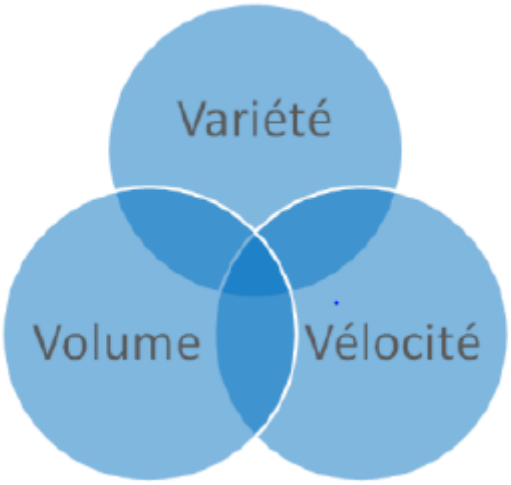


Figure 1.1 – 3V du Big Data. [2]

1. **Volume** : désigne la masse de données collectées (giga-octets, téraoctets).

2. **Variété** : désigne l'origine variée des sources de données qui sont soit

Structurées ou non structurées (images, mails, données de géo-localisation).

3. **Vélocité** : représente à la fois la fréquence à laquelle les données sont générées, capturée set partagées et mises à jour**. [2]**

1.3 Architecture des systèmes Big Data

On distingue principalement les couches suivantes :

- **Couche d'infrastructure :** les serveurs (physique et virtuel).
- **Couche de stockage :** les données seront stockées soit dans une base NoSQL, ou bien directement dans le système de fichier distribué ou les Datawarehouse.
- **Couche du traitement :** on trouve dans cette couche les outils de traitement et analyse des données comme MapReduce et Pig**. [3]**

1.4 Intérêts des Big Data

L'utilisation des Big Data pourrait impacter fortement le monde de l'entreprise et ce de façon méliorative, ainsi les entreprises pourront :

- Améliorer la prise de décision.

- Réduire les coûts d'infrastructures informatiques via l'utilisation des serveurs standards et des logiciels open source.

- Développer la réactivité et l'interactivité à l'égard des clients.
- Améliorer les performances opérationnelles.

1.5 Enjeux des Big Data

Le Big Data apparaît comme le challenge technologique des années 2010-2020. Dépassant les domaines techniques et informatiques, le Big Data suscite un vif intérêt auprès des politiciens, des scientifiques et des entreprises. Les enjeux du Big Data touchent plusieurs secteurs d'activités.

1.5.1 Enjeux techniques

Les enjeux techniques s'articulent autour de l'intégration, le stockage, l'analyse, l'archivage, l'organisation et la protection des données.

1.5.2 Enjeux économiques

D'après le cabinet de conseil dans le marketing IDC, « le marché du Big Data représentera 24 milliards de dollars en 2016, avec une part de stockage estimée à 1/3 de ce montant ». Il va sans dire que la « donnée » est le nouvel or noir du siècle présent, les spécialistes s'accordent déjà sur le fait que le Big Data sera l'arme économique de demain pour les entreprises et se présentera comme un levier qui fera la différence.

Les entreprises collectent de plus en plus d'information en relation avec leurs activités (production, stockage, logistique, ventes, clients, fournisseurs, partenaires, etc), toutes ces informations peuvent être stockées et exploitées pour stimuler leur croissance. Les Big Data permettent :

- D'améliorer les stratégies marketing et commerciale.
- D'améliorer et entretenir la relation client.
- De fidéliser la clientèle.
- De gagner de nouvelles parts de marché.
- De réduire les coûts logistiques.
- De favoriser la veille concurrentielle.

Le client est un acteur majeur dans ce contexte. Jusqu'à présent, la vente consistait à se demander « J'ai un produit, à qui vais-je pouvoir le vendre? ». A l'ère du Big Data, nous devons changer le paradigme pour dire « J'ai un client, de quoi a-t-il besoin aujourd'hui ? ». En connaissant mieux son public, à

travers ses achats, ses activités sur Internet, son environnement, les commerçants peuvent améliorer l'expérience-client, exploiter la recommandation, imaginer le marketing prédictif (le marketing prédictif regroupe les techniques de traitement et de modélisation des comportements clients qui permettent d'anticiper leurs actions futures à partir du comportement présent).

1.5.3 Enjeux juridiques

Le principal enjeu juridique dans un contexte où les utilisateurs sont souvent des « produits », reste la protection de la vie privée. **[4]**

1.6 Cas d'usage du Big Data

Le Big Data couvre de nombreux domaines d'applications telles que l'industrie, la distribution, les banques, l'assurance, le transport, loisirs et le télécom. Des exemples sont cités ci-dessous :

1.6.1 Transports

- ***Contrôle du trafic :*** exploitation de données de tous types (GPS, Radars, sondes) afin de fluidifier le trafic et d'évaluer précisément le temps de transport d'un point à un autre.
- ***Planification des voyages :*** mise à disposition du citoyen de données jusque là réservées aux administrations (gagner du temps / réduire le coût).
- ***Systèmes de transport intelligents (ITS) :*** les applications des NTIC (Nouvelles Technologies de l'Information et de la Communication) destinées au domaine des transports. Parmi les thématiques d'actualité exposé durant le 2 ème congrès mondial des Systèmes de Transport Intelligents 3 nous citons comme exemple : les véhicules autonomes, les véhicules coopératifs et les systèmes de positionnement par satellite.

1.6.2 Santé

Exploitation des données à des fins d'études épidémiologiques, un cas d'utilisation est l'exemple du site « Openhelth.fr » qui affiche en temps réel des informations sur la santé des Français et des cartes en rapport (épidémies, allergie).

Exploitation des données stockées depuis des années, jamais exploitées, qui permettraient de comprendre des liens de cause à effet «legacy data». Suivi des patients (dossier médical du patient).

1.6.3 Économie

Connaissance des clients, actions personnalisées et ciblées, amélioration de la satisfaction.

Accélération des temps d'analyse des données clients pour l'identification des comportements atypiques, - Ciblage marketing (ex. micro segmentation), – Analyse prédictive de l'acte d'achat.

1.6.4 Recherche

En TALN, deux approches coexistent : les technologies « speech-to-text » (transcription automatique de discours livrés sous forme orale) et les technologies de « machine translation » (traduction automatique de discours écrits).

Dans le domaine de l'Image Processing (traitement automatique de l'image), deux secteurs émergent : l'indexation automatique de flux d'images et de fichiers vidéo, de la reconnaissance faciale et de la reconnaissance d'objets***. [5]***

1.7 Limites des Big Data

Si le terrain de jeu du Big Data est loin d'être restreint, il n'est pas sans limites. Elles tiennent, en premier lieu, à la nature des données et aux traitements envisagés, et lorsqu'il est question des données personnelles, la vigilance est nécessaire. Dans certains pays, le traitement des données à caractère personnel est régi par des dispositions particulières, ce qui n'est pas le cas dans la majorité des pays.

Nous sommes arrivés à un point où la protection des données personnelles, portée à la défense des libertés fondamentales de l'individu, est en train de devenir un argument économique. L'enjeu étant dorénavant, d'élaborer le cadre normatif le plus attractif pour le développement de l'économie numérique et des échanges de données, ceci nécessite d'être vigilent dans un contexte de forte concurrence entre les puissances économiques.

L'autre préoccupation provient de la sécurité : une faille minuscule peut menacer des quantités de données considérables. Si les utilisateurs perdent confiance dans l'utilisation de leurs informations, c'est donc tout l'édifice du big data qui risque de s'écrouler. Pour éviter cela, par exemple en Europe, la commission européenne a présenté, en début 2012, un règlement qui vise à protéger davantage les utilisateurs. Ce texte devrait être voté en 2014 pour une application en 2016, il obligera les entreprises à demander le consentement explicite de l'utilisateur avant de collecter ses données. (Haas, 2013). [**4]**

1.8 Avantages

Plusieurs avantages peuvent être associés à une architecture Big Data, nous pouvons citer par exemple :

- ***Evolutivité (scalabilité) :*** Quelle est la taille que devra avoir votre infrastructure? Combien d'espace disque est nécessaire aujourd'hui et à l'avenir ? le concept Big Data nous permet de s'affranchir de ces questions, car il apporte une architecture scalable.
- ***Performance :*** Grâce au traitement parallèle des données et à son système de fichiers distribué, le concept Big Data est hautement performant en diminuant la latence des requêtes.

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

L'architecture Big Data apporte ses propres mécanismes de haute disponibilité.

II Hadoop

1.9 Historique

En 2004, Google a publié un article de recherche présentant son algorithme MapReduce, conçu pour réaliser des opérations analytiques à grande échelle sur un grand cluster de serveurs, et sur son système de fichier en cluster GFS. Doug Cutting, qui travaillait alors sur le développement du moteur de recherche libre Apache Lucene et butait sur les mêmes problèmes de volumétrie de données qu'avait rencontré Google, s'est alors emparé des concepts décrits dans l'article du géant de la recherche et a décidé de répliquer en open source les outils développés par Google pour ses besoins. Employé chez Yahoo, il s'est alors lancé dans le développement de ce qui est aujourd'hui le projet Apache Hadoop. **[3]**

1.10 Présentation Hadoop

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

Hadoop a été conçu par Doug Cutting en 2004, également à l'origine du moteur Open Source Nutch. Doug Cutting cherchait une solution pour accroître la taille de l'index de son moteur. Il eut l'idée de créer un Framework de gestion de fichiers distribués.
Yahoo! en est devenu ensuite le principal contributeur, le portail utilisait notamment l'infrastructure pour supporter son moteur de recherche historique. Comptant plus de 10 000 clusters Linux en 2008, il s'agissait d'une des premières architectures Hadoop digne de ce nom. Crée spécialement pour les gros volumes. Facebook pour l'analyse des logs, Google pour l'analyse des requêtes.
Il est caractérisé par :

- ***Robuste :*** si un nœud de calcul tombe, ses tâches sont automatiquement réparties sur d'autres nœuds. Les blocs de données sont également répliqués.
- ***Coût :*** il optimise les coûts via une meilleure utilisation des ressources présentées.

- ***Souple :*** car il répond à la caractéristique de variété des données en étant capable de traiter différents types de données.
- ***Virtualisation :*** ne plus se reposer directement sur l'infrastructure physique (baie de stockage coûteuse), mais choisir la virtualisation de ses clusters Hadoop**. [4]**

1.11 Architecture

Hadoop est principalement constitué de quatre composants :

Hadoop Distributed File System (HDFS)

Un système de fichiers distribués qui fournit un accès haut-débit aux données de l'application.

Hadoop YARN

Un Framework pour la planification des tâches et la gestion des ressources du cluster.

Hadoop MapReduce

Un système basé sur YARN pour le traitement parallèle des gros volumes de données.

Hadoop Common

Les utilitaires communs qui supportent les autres modules d'Hadoop.

Plus concrètement l'écosystème Hadoop comprend de nombreux autres outils couvrant le stockage et la répartition des données, les traitements distribués, l'entrepôt de données, le workflow, la programmation, sans oublier la coordination de l'ensemble des composants. On parle des outils comme Hive, Pig, Hbase, Flume**. [3]**

Figure 1.2 Hadoop v2: Architecture [3]

1.12 HDFS

Hadoop utilise un système de fichiers virtuel qui lui est propre : le HDFS (Hadoop Distributed File System). HDFS est un système de fichier distribué, extensible et portable inspiré par le Google File System (GFS).

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

Toutefois, HDFS se démarque d'un système de fichiers classique pour les principales raisons suivantes :

- HDFS n'est pas dépendant du noyau du système d'exploitation. Il assure une portabilité et peut être déployé sur différents systèmes d'exploitation. Un des se inconvénients est de devoir solliciter une application externe pour monter une unité de disque HDFS.
- HDFS est un système distribué sur un système classique, la taille du disque est généralement considérée comme la limite globale d'utilisation. Dans HDFS, chaque nœud d'un cluster correspond à un sous-ensemble du volume global des données du cluster. Pour augmenter ce volume global, il suffira d'ajouter de nouveaux nœuds. On retrouvera également dans HDFS, un service central appelé NameNode qui aura la tâche de gérer les métadonnées.
- HDFS utilise des tailles de blocs largement supérieures à ceux des systèmes classiques. Par défaut, la taille est fixée à 64 Mo. Il est toutefois possible de monter à 128 Mo, 256 Mo, 512 Mo voire 1 Go. Alors que sur des systèmes classiques, la taille est généralement de 4 Ko, l'intérêt de fournir des tailles plus grandes permettant de réduire le temps d'accès à un bloc. Notez que si la taille du fichier est inférieure à la taille d'un bloc, le fichier n'occupera pas la taille totale de ce bloc.
- HDFS fournit un système de réplication des blocs dont le nombre de réplications est configurable. Pendant la phase d'écriture, chaque bloc correspondant au fichier est répliqué sur plusieurs nœuds. Pour la phase de lecture, si un bloc est indisponible sur un nœud, des copies de ce bloc seront disponibles sur d'autres nœuds.

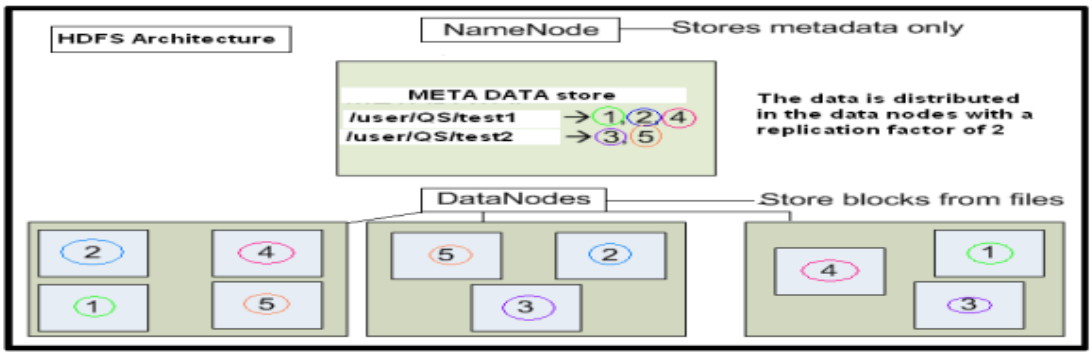


Figure1.3 L'architecture d'HDFS [4]

HDFS définit deux types de nœuds :

Le nœud principal ou NameNode

Il se caractérise par :

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

Le nœud de données ou DataNode

Il se caractérise par :

- Stocke des blocs de données dans le système de fichier local.
- Maintenir des métadonnées sur les blocs possédés.
- Serveur de bloc de données et de métadonnées pour le client hdfs.

Secondary NameNode

Le NameNode dans l'architecture Hadoop est un point unique de défaillance. Si ce service est arrêté, il n'y a pas moyen de pouvoir extraire les blocs d'un fichier donné. Pour résoudre ce problème, un NameNode secondaire appelé Secondary NameNode a été mis en place dans l'architecture Hadoop. Son rôle consiste à :

- Télécharger régulièrement les logs sur le NameNode.
- Crée une nouvelle image en fusionnant les logs avec l'image HDFS.
- Renvoie la nouvelle image au NameNode.

1.12.1 Lecture d'un fichier HDFS

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

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

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

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

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

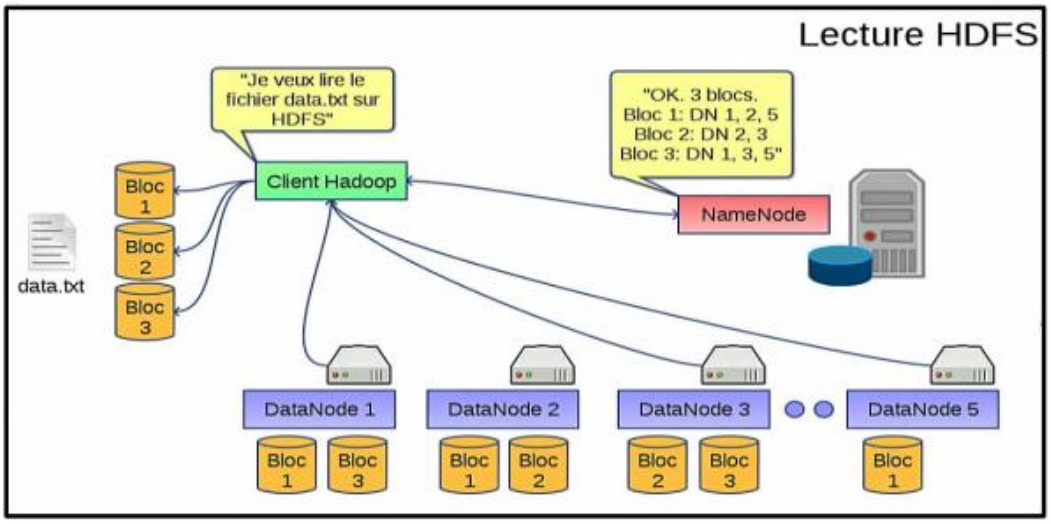


Figure 1.4 Lecture d'un fichier HDFS. [4]

1.12.2 Ecriture dans un fichier ou volume HDFS

Pour écrire un fichier au sein d'HDFS:

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

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

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

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

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

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

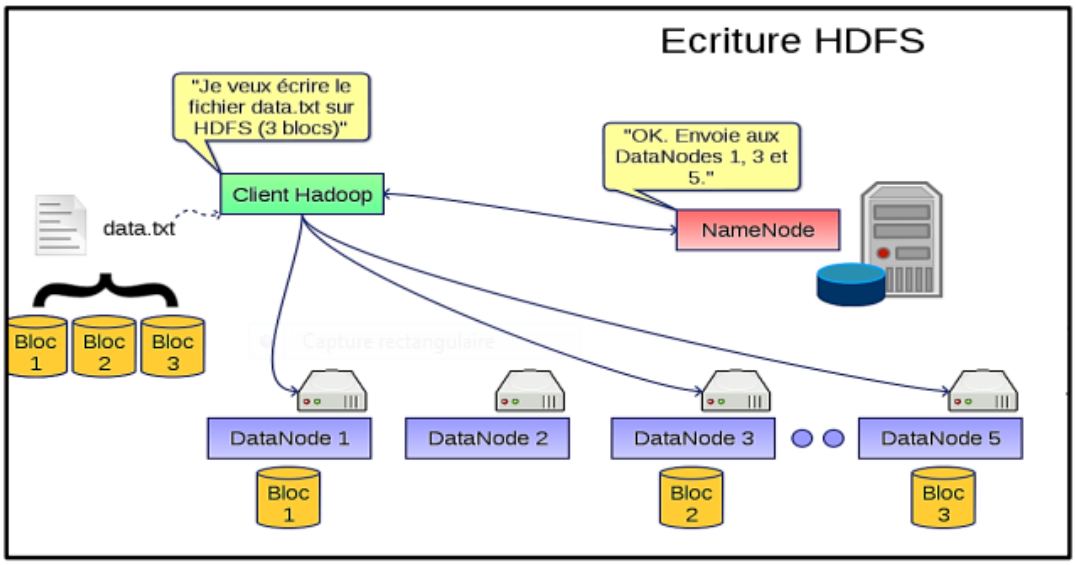


Figure1.5 Processus d'écriture dans un volume ou fichier HDFS. [4]

1.13 MapReduce

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

Le modèle de programmation fournit un cadre à un développeur afin d'écrire une fonction Map et une fonction Reduce. Tout l'intérêt de ce modèle de programmation est de simplifier la vie du développeur. Ainsi, ce développeur n'a pas à se soucier du travail de parallélisation et de distribution du travail. MapReduce permet au développeur de ne s'intéresser qu'à la partie algorithmique. **[4]**

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

- La première, **MAP**, va transformer les données d'entrée en une série de couples clef /valeur. Elle va regrouper les données en les associant à des clefs, choisies de telle sorte que les couples clef/valeur aient un sens par rapport au problème à résoudre. Par ailleurs, cette opération doit être parallélisable: on doit pouvoir découper les données d'entrée en plusieurs fragments, et faire exécuter l'opération MAP à chaque machine du cluster sur un fragment distinct. La fonction Map s'écrit de la manière suivante : **Map (clé1, valeur1) → List (clé2, valeur2).**
- La seconde, **REDUCE**, va appliquer un traitement à toutes les valeurs de chacune des clefs distinctes produite par l'opération MAP. Au terme de l'opération REDUCE, on aura un résultat pour chacune des clefs distinctes. Ici, on attribuera à

chacune des machines du cluster une des clefs uniques produites par MAP, en lui donnant la liste des valeurs associées à la clef. Chacune des machines effectuera alors l'opération REDUCE pour cette clef. La fonction Reduce s'écrit de la manière suivante : **Reduce (clé2, List (valeur2)) → List (valeur2).**

L'exemple classique est celui du **WordCount** qui permet de compter le nombre d'occurrences d'un mot dans un fichier. En entrée l'algorithme reçoit un fichier texte qui contient les mots suivants **voiture la le elle de elle la se la maison voiture** Dans notre exemple, la clé d'entrée correspond au numéro de ligne dans le fichier et tous les mots sont comptabilisés à l'exception du mot «se». Le résultat de la fonction Map est donné ci-dessous.

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

Avant de présenter la fonction Reduce, deux opérations intermédiaires doivent être exécutées pour préparer la valeur de son paramètre d'entrée. La première opération appelée **shuffle** permet de grouper les valeurs dont la clé est commune. La seconde opération appelée **sort** permet de trier par clé. A la différence des fonctions Map et Reduce, shuffle et sort sont des fonctions fournies par le Framework Hadoop, donc, il n'a pas à les implémenter.

Ainsi, après l'exécution des fonctions shuffle et sort le résultat de l'exemple est le suivant :

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

Suite à l'appel de la fonction Reduce, le résultat de l'exemple est le suivant :

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

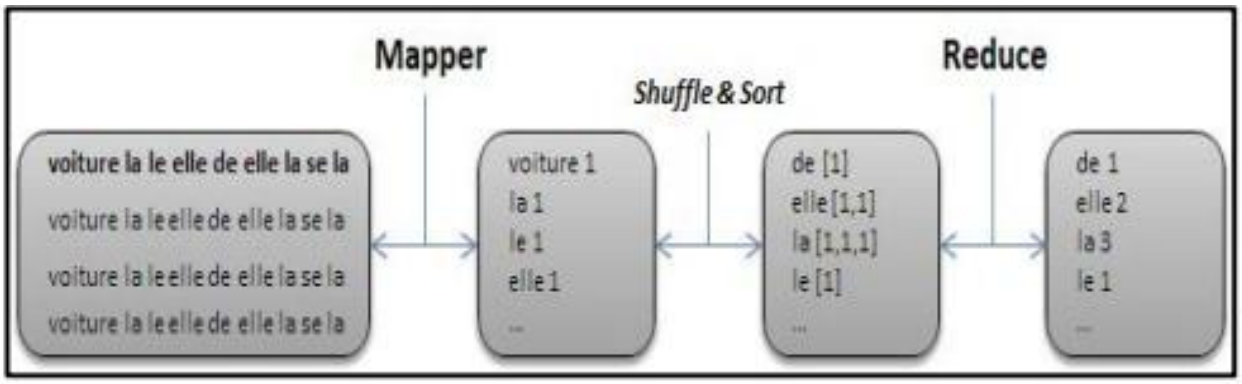


Figure 2.3: Exemple d'un programme MapReduce (WordCount) [4]

1.14 Composant Apache Hadoop

1.14.1 Hbase

Hbase est un système de gestion de bases de données non relationnelles distribuées, écrit en Java, disposant d'un stockage structuré pour les grandes tables. C'est une base de données NoSQL, orientée colonnes. Utilisé conjointement avec HDFS, ce dernier facilite la distribution des données de Hbase sur plusieurs noeuds. Contrairement à HDFS, HBase permet de gérer les accès aléatoires read/write pour des applications de type temps réel.

1.14.2 HaCatalog

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

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

1.14.3 Hive

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

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

- **UDF (User DefinedFunction) :** qui prennent une ligne en entrée et retournent une ligne en sortie. Exemple : mettre une chaîne de caractère en minuscule et inversement.
- **UDAF (User DefinedAggregateFunction)** : qui prennent plusieurs lignes en entrée et retournent une ligne en sortie. Exemple : somme, moyenne, max.
- **UDTF (User Defined Table Function) :** qui prennent une ligne en entrée et retournent plusieurs lignes en sortie. Exemple : découper une chaîne de caractère en plusieurs mots.

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

1.14.4 Pig

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

1.14.5 Sqoop

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

1.14.6 Flume

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

1.14.7 Oozie

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

1.14.8 Zookeeper

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

1.14.9 Ambari

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

1.14.10 Mahout

Mahout est un projet de la fondation Apache visant à créer des implémentations d'algorithmes d'apprentissage automatique et de data mining.

1.14.11 Avro

Avro est un format utilisé pour la sérialisation des données.

Le caractère open source de Hadoop a permis à des entreprises de développer leur propre distribution en ajoutant des spécificités. **[5]**

1.15 Conclusion

Le taux des informations disponibles dans le web ne cesse de croitre d'un jour à un autre. Cette croissance a conduit à une explosion énorme donnant la naissance de ce qui est appelé aujourd'hui Big Data. Ce chapitre a été adressé à la présentation générale du concept de Big Data, ses enjeux et les techniques utilisées pour assurer ses services notamment le framework Hadoop.

Chapitre 2

Chapitre 2

La vie privée

2.1 Introduction

Les progrès des nouvelles technologies de l'information et de la découverte de connaissances, permettant la surveillance généralisée à grande échelle sur des volumes de données massives, avaient soulevé la nécessité de telles techniques permettant un avancement sans compromettre la vie privée des utilisateurs. De nos jours, la vie privée est l'un des problèmes critiques associés à la croissance du Big Data et la propagation de l'information. Il y a une inquiétude croissante sur la confidentialité des informations. Le monde de PVP ne compte pas seulement à éviter l'observation ou de cacher des substances personnelles et des relations, mais il signifie la capacité de partage des informations sélectivement et non publiquement. Chaque jour, des océans de données sont recueillis sur la planète via des capteurs sans fil et des dispositifs intelligents. Ces données sont en partie sensibles. D'autres parties des données peuvent être considérées comme sensibles dans certains cas, bien que non dans d'autres cas. Les gens et les chercheurs ont pu être embrouillés avec la sécurité et la préservation de la vie privée.

Dans ce chapitre, nous allons passer en revue la notion et les techniques de la vie privée lors de la publication et l'analyse des données dans section 2. La section 3 présente la vie privée différentielle qui constitue actuellement un modèle standard pour la vie privée. Dans la section 4, nous citons les nouveaux défis de la vie privée rencontrés dans le contexte du Big data puis nous présentons quelques solutions qui concernent la préservation de la vie privée lors de la publication et l'analyse des données.

2.2 Préservation de vie privée lors de la publication et l'analyse des données

Au cours des deux dernières décennies, les informations numériques collectées par les entreprises, les organisations et les gouvernements ont abouti à un grand nombre d'ensembles de données, et la vitesse de collecte de ces données a augmenté de façon spectaculaire au cours des dernières années. Un collecteur de données, également connu sous le nom de conservateur, est chargé de publier les données pour une analyse.

La figure 2.1 illustre le processus de publication et d'analyse des données, dans lequel il existe environ deux étapes. Dans la première étape, étape de la collecte de données, les individus soumettent leurs informations personnelles à un ensemble de données. La quantité de données est collectée dans leur domaine relatif, tel que les données médicales, les données des banques, les données de réseaux sociaux. Le conservateur a la gestion complète de cet ensemble de données. La deuxième étape est la phase de publication ou d'analyse des données. La publication de données vise à partager des ensembles de données ou des résultats de requête avec des utilisateurs publics. Dans certains documents, ce scénario est connu sous le nom de partage de données ou de diffusion de données. L'analyse des données fournit des modèles de données aux utilisateurs publics. Il peut être associé à des algorithmes particuliers d'apprentissage automatique ou d'exploration de données. La publication et l'analyse de données apportent des avantages sociaux, tels que la fourniture de meilleurs services, la publication de statistiques officielles, la fourniture de tâches d'exploration de données ou d'apprentissage automatique.

Comme le montre la figure 2.1, la plupart des ensembles de données collectés sont liés et contiennent des informations privées ou sensibles. La violation de la confidentialité peut se produire dans les deux étapes. Si les commissaires ne sont pas dignes de confiance, les renseignements personnels seront divulgués directement à l'étape de la collecte des données. Même si les conservateurs peuvent faire confiance et appliquer plusieurs techniques d'anonymisation simples, lorsqu'ils publient des informations agrégées au public, les informations personnelles peuvent être divulguées car le public n'est normalement pas digne de confiance .La préservation de la vie privée est donc devenue une question urgente qui doit être traitée. **[6]**

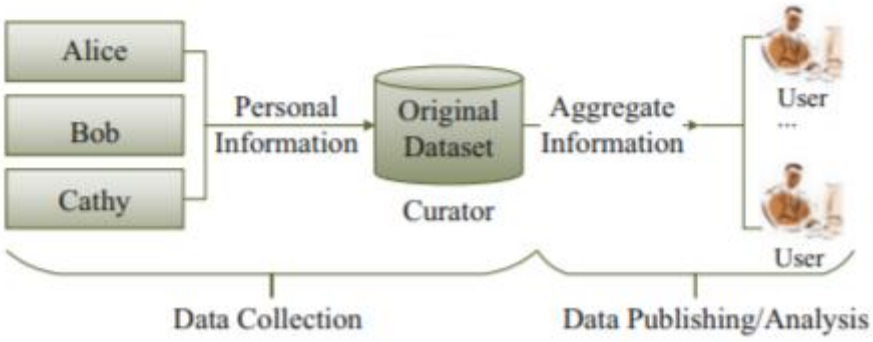


Figure2.1 Confidentialité Conservation de la publication et de l'analyse des données [6]

2. 3 La confidentialité différentielle

Il y a le modèle de confidentialité différentiel, qui est un solide modèle de protection de la vie privée qui fournit une garantie de confidentialité prouvable pour les individus. Il suppose que même si un adversaire connaît tous les autres enregistrements d'un ensemble de données sauf un enregistrement, il ne peut toujours pas inférer les informations contenues dans cet enregistrement inconnu. En d'autres termes, même l'adversaire apprend à connaître l'information de base maximale à l'exception de l'enregistrement qu'il / elle veut savoir, il / elle ne peut pas identifier le dossier spécifique. Sous cette hypothèse, la vie privée différentielle prouve théoriquement qu'il y a une faible probabilité que l'adversaire trouve l'enregistrement inconnu. Par rapport à la vie privée précédente modèles, la vie privée différentielle peut résister avec succès à l'attaque de fond et fournir une garantie de confidentialité prouvable.

Les premiers pas vers une vie privée différentielle ont été faits en 2003. Mais ce n'est qu'en 2006 que Dwork et al. Discuté de la technologie dans sa forme actuelle.

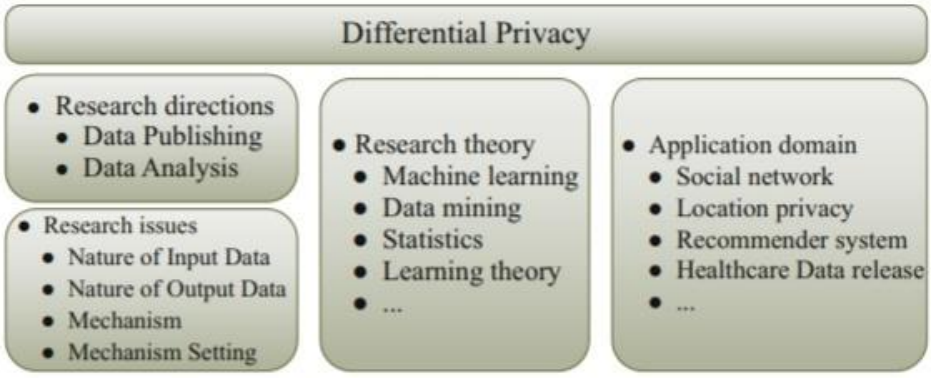


Figure 2.2 Composants clés associés à la confidentialité différentielle [6]

Depuis lors, un important travail a été développé par des scientifiques du monde entier. La vie privée différentielle attire l'attention des universitaires. En 2012, Microsoft publie un livre blanc intitulé Confidentialité différentielle pour tout le monde, visant à traduire cette recherche en nouvelles technologies améliorant la protection de la vie privée.

La confidentialité différentielle a récemment été considérée comme un modèle prometteur de préservation de la vie privée. L'intérêt dans ce domaine est très élevé et la notion couvre plusieurs domaines de recherche, allant de la communauté de la vie privée aux communautés de la science des données, y compris l'apprentissage automatique, l'exploration de données, les statistiques et la théorie de l'apprentissage. La figure 1.3 montre les composants clés associés à la recherche différentielle sur la vie privée. Il existe à peu près deux grandes directions sur la recherche différentielle en matière de protection de la vie privée, la publication de données et l'analyse de données. Dans les deux sens, les

principaux problèmes de recherche comprennent la nature des données d'entrée et de sortie, la conception du mécanisme et les paramètres. Les méthodes de recherche et les théories impliquent l'apprentissage automatique, l'exploration de données, les statistiques et le chiffrement. Beaucoup de travail a été effectué dans un certain nombre de domaines d'application, y compris le réseau social, la confidentialité de la localisation, les systèmes de recommandation et d'autres applications. **[6]**

2.3.1 Définition de la confidentialité différentielle

Une définition formelle de la confidentialité différentielle est présentée ci-dessous:

Définition :(ε, δ)-Protection de la vie privée) Un mécanisme randomisé $\boldsymbol{M}$ donne.(ε, δ) -différentiel différentiel pour chaque ensemble de sorties$\boldsymbol{S}$, et pour tout voisinage ensembles de données de $\boldsymbol{D}$ et$\boldsymbol{D'}$, si $\boldsymbol{M}$ satisfait:

$$\boldsymbol{Pr[M(D) \epsilon S] \leq exp\ (\epsilon).\ Pr\left[M\left(D'\right) \epsilon S\right] + \delta}$$

La figure 3.1 montre le mécanisme sur les ensembles de données voisins. Pour une sortie particulière, le rapport sur deux probabilités est borné par $\boldsymbol{e^{\epsilon}}$ Si $\boldsymbol{\delta = 0}$, le mécanisme randomisé $\boldsymbol{M}$ donne la confidentialité $\boldsymbol{\epsilon}$-différentielle par sa définition la plus stricte. (ε, δ) -différence discrétionnaire fournit la liberté de violer la stricte confidentialité $\boldsymbol{\epsilon}$-différentielle pour une faible probabilité événements. La confidentialité $\boldsymbol{\epsilon}$-différentielle est généralement appelée confidentialité différentielle pure, alors que (ε, δ) −la confidentialité différentielle avec $\boldsymbol{\delta > 0}$ est appelée confidentialité différentielle approximative. **[6]**

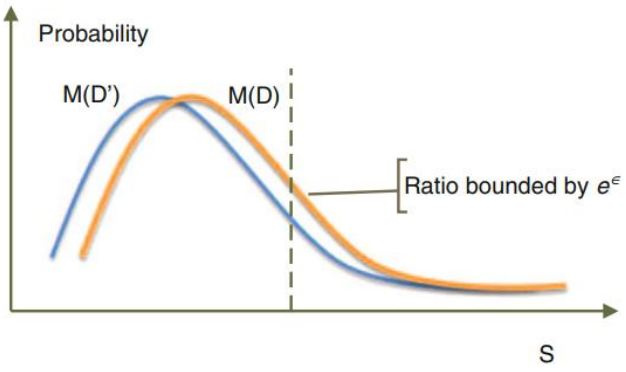


Figure 2.3 Distribution des réponses aux requêtes [8]

2.4 Mécanismes de confidentialité différentielle

2.4.1 Mécanisme de Laplace

Dans cette section, nous présentons l'un des mécanismes les plus élémentaires de la confidentialité différentielle.

Le mécanisme de Laplace consiste à ajouter du bruit aléatoire qui s'adapte à Laplace distribution avec moyenne 0 et échelles $\frac{GS((f)}{\epsilon}$ et ajouter indépendamment à chaque réponse à une requête, s'assurant ainsi que chaque requête est perturbée de manière appropriée.

Pour analyser le mécanisme de Laplace, nous devons d'abord définir la distribution de Laplace.

Définition *Distribution Laplace*

La distribution de Laplace est caractérisée par l'emplacement θ(n'importe quel nombre réel) et échelle λ (doit être supérieur à 0) avec la fonction de densité de probabilité suivante:

$$f(x \mid \theta, \lambda) = \frac{1}{2\lambda} \exp(-\frac{|x-\theta|}{\lambda})$$

Définition *Le mécanisme de Laplace*

Compte tenu de toute fonction $f: D^n \rightarrow R^k$, le mécanisme de Laplace est défini comme:

$$\mathcal{ML}(x, f(.), \epsilon) = f(x) + (y_1, \ldots, y_k)$$

2.4 .2 Le mécanisme exponentiel

Un bruit réel est ajouté à la réponse réelle quand il s'agit de Laplace Mécanisme. Néanmoins, il est connu que toutes les fonctions de requêtes ne peuvent pas retourner des valeurs numériques à leur sortie tout le temps. Par conséquent, McSherry et Talwar ont proposé une méthode plus générale qui peut être appliquée pour répondre à des requêtes non numériques dans une matière différentielle.

Étant donné une fonction qualité $q : D \times R \rightarrow \mathbb{R}$, le mécanisme exponentiel sélectionne un sortie de d avec n éléments du domaine D et une plage arbitraire R, basée sur le score qui représente la qualité de r en d.

La sortie finale serait proche du choix idéal sur q puisque le mécanisme nomme des probabilités exponentiellement plus élevées d'être sélectionné pour les sorties supérieures.

Définition *(Le mécanisme exponentiel)*

Étant donné une base de données $D \epsilon X^n$, et une fonction de qualité q par rapport à D, la sensibilité globale de s être $GS_q = \max_{D1,D2,r \epsilon R} |q(D_1, r) - q(D_2, r)|$

Et plage de requête$\boldsymbol{R}$, le mécanisme exponentiel $\mathbf{M_E(D, q, R)}$ donne la sortie $\boldsymbol{r \epsilon R}$ basé sur la probabilité:

$$\mathbf{Pr}[\boldsymbol{M(D, r, R)} = \boldsymbol{r}] \propto \boldsymbol{exp}\left(\frac{\boldsymbol{\epsilon q}\,(\mathbf{D}, \mathbf{r})}{\mathbf{2G}\boldsymbol{S_q}}\right)$$

Lorsque la gamme du mécanisme exponentiel est super-polynomiale dans les paramètres naturels du problème, il peut définir une distribution complexe sur un grand domaine arbitraire et il peut donc ne pas être possible de l'appliquer efficacement.

Le mécanisme médian est un mécanisme interactif différentiel interactif qui répond aux requêtes de prédicat arbitraires $\boldsymbol{f_1}, \ldots\ldots, \boldsymbol{f_k}$ qui arrivent à la volée sans les futures requêtes de connaissances, où k pourrait être grand ou même super-polynomial. Il effectue beaucoup mieux que les autres mécanismes (par exemple, le mécanisme de Laplace) quand il vient répondre à plus de requêtes de manière exponentielle et donne des contraintes fixes. **[6]**

2.5 Aspects de la vie privée dans MapReduce

La confidentialité garantit que les données sensibles ne sont pas exposées à des utilisateurs non autorisés et à des intrus (c'est-à-dire des fournisseurs de cloud, d'autres fournisseurs de données, des utilisateurs de MapReduce ou des adversaires). Notez que les fournisseurs de données sont intéressés à autoriser certaines sortes de calculs sur les données, mais il est également nécessaire de préserver la violation des données sensibles. Les données sensibles dans ce cas sont spécifiques à chaque cas et peuvent être des enregistrements personnels avec des informations d'identification (informations personnelles identifiables PII), spécifiques à l'organisation informations. Dans cette section, nous présentons un bref résumé des aspects relatifs à la confidentialité dans le Cloud général informatique, les exigences de confidentialité dans MapReduce, puis, passer en revue certaines solutions existantes pour la vie privée dans MapReduce.

2.5.1 Défis de confidentialité dans MapReduce Computing

Le Cloud Computing et le déploiement de MapReduce sur les Clouds publics présentent de nouveaux défis en matière de confidentialité des données. Ici, nous décrivons les défis de la vie privée de l'informatique en nuage dans le contexte de MapReduce et les divisons en quelques cas selon les comportements contradictoires des nuages publics et des utilisateurs.

- **Protection de la confidentialité des données par les fournisseurs de Cloud contradictoires**.

Un utilisateur peut conserver des données privées dans des Clouds publics en raison de leur volume ou de leur facilité de calcul sur les Clouds publics, tout en visant à préserver la confidentialité des données. Dans ce contexte, nous considérons un fournisseur de Cloud contradictoire qui peut observer les utilisateurs données et code MapReduce, mais ne devrait pas changer les requêtes ou les résultats des utilisateurs. Assurer la confidentialité en présence d'un fournisseur de Cloud contradictoire qui peut modifier ou supprimer des données et des calculs est un défi insurmontable. L'objectif de la vie privée en présence de nuages adversaires est de minimiser fuite de données au fournisseur de Cloud tout en permettant aux utilisateurs d'effectuer des opérations sur les données. La majorité du travail dans ce domaine est basé sur le cryptage des données des utilisateurs et la recherche d'un moyen d'autoriser les opérations sur les données cryptées dans le nuage.

- **Protection des données des utilisateurs adversaires.**

Les fournisseurs de données permettent aux utilisateurs d'exécuter des travaux MapReduce sur leurs données via des fournisseurs de Cloud, mais souhaitent également contrôler et préserver la confidentialité des données. Par exemple, bien que le revenu annuel moyen puisse être calculé, le revenu de cette personne devrait rester secret. Les solutions à ce cas d'utilisation sont basées sur l'anonymisation des données (c.-à-d. valeurs sensibles pouvant identifier les individus), en ajoutant du bruit aléatoire aux données, ou des résultats de calcul pour masquer les valeurs réelles (par exemple en utilisant une confidentialité différentielle).

- **Multi utilisateurs sur un seul nuage public.**

Un fournisseur de Cloud public et un fournisseur de données devraient permettre à plusieurs utilisateurs d'effectuer leurs calculs sans fuite de données. Par exemple, une organisation peut conserver toutes ses données dans des Clouds publics, et plusieurs utilisateurs traitent et accèdent à certaines parties des données pour lesquelles ils sont autorisés. Un hôpital peut stocker des données de tous les patients sur des nuages. Il existe plusieurs groupes d'utilisateurs, par exemple des médecins, des compagnies d'assurance, des patients et des sociétés de recherche pharmaceutique, qui doivent accéder à certaines parties des données. Dans ce cas, le cadre de confidentialité doit garantir que chaque utilisateur est en mesure d'accéder à toutes les données requises, mais également que les utilisateurs ne peuvent pas accéder aux parties de données pour lesquelles ils ne sont pas autorisés pour. Ceci est habituellement résolu par des mécanismes

d'authentification et d'autorisation. La situation devient plus complexe car les données sont fournies par un certain nombre de fournisseurs de données, chacun avec des exigences de confidentialité différentes. Dans de tels Clouds publics, un utilisateur accusatoire peut également accéder aux données d'un autre utilisateur en injectant un mapper ou un réducteur malveillant qui exploite les problèmes de sécurité existants. Diverses solutions sont suggérées pour la confidentialité du Cloud Computing. Cependant, toutes les solutions existantes pour la confidentialité du Cloud Computing ne peuvent pas être utilisées pour la confidentialité dans MapReduce, en raison d'un certain nombre de contraintes et de défis supplémentaires dans MapReduce, nécessitant ainsi une adaptation ou un changement de ces solutions.

2.5.2 Exigences de confidentialité dans MapReduce

MapReduce décompose de manière inhérente les fournisseurs de données, les fournisseurs de Cloud et les utilisateurs qui exécutent des requêtes sur des données. En se référant à la structure de nuage décrite dans la figure 4, les fournisseurs de données téléchargent des données vers le fournisseur de Cloud, et les utilisateurs du Cloud effectuent des requêtes sur les données. Cependant, malgré la séparation entre les différentes entités, il reste difficile de garantir la confidentialité de ces paramètres. Ici, nous fournissons des exigences de confidentialité dans le Framework MapReduce, déployées sur le Cloud hybride ou le Cloud public.

- **Protection des fournisseurs de données.**

Dans un environnement où les données sont téléchargées vers le Cloud par différents fournisseurs de données, chaque fournisseur de données peut avoir des exigences de confidentialité différentes. Le fournisseur de Cloud doit s'assurer que ces exigences de confidentialité sont respectées même en présence d'utilisateurs adversaires. De plus, différents fournisseurs de données peuvent exiger un niveau de confidentialité différent pour divers ensembles de données. Le cadre de confidentialité devrait permettre l'adaptation des niveaux de confidentialité à ces exigences.

- **Fournisseurs de Cloud non fiables.**

Comme un fournisseur de Cloud contradictoire peut effectuer n'importe quel calcul sur des données pour révéler des données, modifier des données et produire de mauvaises sorties, les données doivent être protégées des fournisseurs de Cloud. En plus de protéger les données des fournisseurs de services de Cloud, le cadre de confidentialité doit aussi être en mesure de protéger les calculs effectués. À titre d'exemple, considérons un utilisateur demandant des informations spécifiques. Même si les résultats de données ne sont pas communiqués au fournisseur de Cloud, il est possible d'apprendre l'intention de l'utilisateur d'observer les calculs effectués. En termes de MapReduce, il peut être nécessaire que les mappeurs et les réducteurs travaillent sur des données cryptées.

- **Utilisation et compromis de confidentialité.**

Un fournisseur de données peut crypter des données d'une manière qu'aucune information ne peut être apprise de lui. Cependant, cela empêchera également l'utilisateur d'effectuer des calculs sur les données, et donc, diminue l'utilisation de MapReduce. En tant que tel, le cadre de confidentialité de MapReduce doit fournir une utilisation maximale tout en préservant la confidentialité des données selon les exigences des fournisseurs de données.

- **Efficacité.**

Dans la plupart des Clouds publics, les utilisateurs sont tarifés pour l'utilisation et le stockage. Par conséquent, le cadre de confidentialité doit être efficace en termes de consommation de CPU et de mémoire, et en termes de stockage requis. Si le cadre de protection de la vie privée entraîne des frais généraux élevés, il pourrait être plus rentable d'effectuer des calculs sur le cloud privé, où la sécurité physique résout les problèmes de confidentialité.

2.5.3 Modèles adverses pour la confidentialité de MapReduce

Tous les modèles contradictoires mentionnés sont applicables à la vie privée de MapReduce avec de petits changements. Ci-dessous, nous expliquons l'adaptation requise dans les définitions des adversaires et comment ils peuvent être appliqués dans les paramètres de confidentialité.

- ***Honnête mais Curieux adversaire.***

Ce type d'adversaire s'applique principalement aux fournisseurs de Cloud. Les fournisseurs de Cloud curieux peuvent violer la confidentialité des données et des calculs MapReduce très facilement, puisque le cluster entier est sous le contrôle des fournisseurs de Cloud, qui ont tous les types d'accès privilégié aux données et aux nœuds de calcul. Il est important de noter qu'en réalité, les fournisseurs de Cloud curieux ne sont pas nécessaires adversaires par choix, mais plutôt pourrait être conforme par la loi de la cour, les règlements et les demandes gouvernementales.

- ***Un adversaire malveillant.***

Ce type d'adversaire s'applique à un utilisateur qui essaie d'apprendre, modifier ou supprimer des informations à partir des données en émettant diverses requêtes. En général, les fournisseurs de Cloud ne sont pas supposés être malveillants, comme assurer la confidentialité avec les fournisseurs de Cloud malveillants nécessite un haut niveau de confidentialité des mesures qui réduisent considérablement l'utilisation du cadre.

- ***Adversaire bien informé.***

Un adversaire bien informé s'applique à la fois à un fournisseur de Cloud et à un utilisateur, qui essaient d'apprendre, de modifier ou de supprimer des informations. Un adversaire bien informé est supposé avoir une connaissance complète du Framework MapReduce, la structure du nuage, et est capable d'utiliser un algorithme ou un inconvénient de la cryptographie. En d'autres termes, il n'y a pas de «sécurité par l'obscurité».

- ***Réseau et adversaire de nœud.***

Comme opposé au réseau et aux adversaires d'accès aux nœuds dans les modèles contradictoires de sécurité, un fournisseur de Cloud travaillant comme un adversaire de réseau et de nœud a tous les privilèges l'accès aux nœuds de calcul et à l'infrastructure Cloud complète. Un exemple réel d'un tel adversaire est un employé de fournisseur de Cloud qui enfreint les informations sensibles les plus clairement montrées par l'affaire Edward Snowden. Il est impossible de cacher un calcul MapReduce ou des données de ce type de adversaire **[7]**

2.6 La confidentialité des données avec les utilisateurs adversaires

Dans de nombreuses applications, les fournisseurs de données et les utilisateurs de données sont des parties différentes et peuvent être complètement séparés. Des exemples de telles applications sont les fournisseurs de services de santé, les sociétés pharmaceutiques et les fournisseurs de données génomiques. Dans ces cas, il est clairement nécessaire de fournir un accès aux données aux parties externes à des fins de recherche, de surveillance, ou le partage des connaissances tout en assurant une protection suffisante des données pour les fournisseurs de données. Cependant, à ces fins, la gestion des contrôles d'accès ou le cryptage des données ne suffit pas. L'anonymisation des données est une solution prometteuse pour garantir la confidentialité des données sur les Clouds publics. Il fonctionne en masquant les identifiants de données, c'est-à-dire les attributs qui permettent l'identification d'individus spécifiques, en changeant l'information à certaines valeurs, en insérant des enregistrements et en supprimant des informations. [7].

Une autre notion de protection de la vie privée, appelée confidentialité différentielle L'idée de la confidentialité différentielle est de s'assurer qu'un ajout ou un retrait d'un seul élément de jeu de données n'affecte pas substantiellement le résultat des calculs. En d'autres termes, l'adversaire ne peut pas distinguer les résultats avec et sans un élément de jeu de données spécifique. Les moyens les plus courants pour atteindre la confidentialité différentielle sont l'ajout de bruit aléatoire (spécifique) aux données sensibles ou aux calculs, cacher une existence de tout enregistrement individuel. La confidentialité différentielle est actuellement considérée comme la norme de fait des données privées

la publication car elle fournit une garantie de confidentialité assez forte car elle ne dépend pas des informations auxiliaires connues de l'adversaire ou du pouvoir de calcul. [7].

Des méthodes supplémentaires de préservation de la confidentialité pour MapReduce sont des solutions basées sur le chiffrement-déchiffrement et une solution basée sur la responsabilité.

Les auteurs [7].ont présenté un nouveau Map pour les calculs MapReduce basés sur l'anonymisation des données. Le Framework introduit une nouvelle couche, appelée Privacy-Preserving Layer (PPL), voir Figure (2.4), qui existe entre les données d'origine et l'infrastructure MapReduce pour l'exécution d'un travail affecté. La couche PPL prend les exigences de confidentialité et les données d'origine en entrée. La couche peut ensuite appliquer différentes approches d'anonymisation en fonction des exigences de confidentialité des fournisseurs de données. Cela permet une flexibilité dans le choix de différents mécanismes de confidentialité dans un cadre unique.

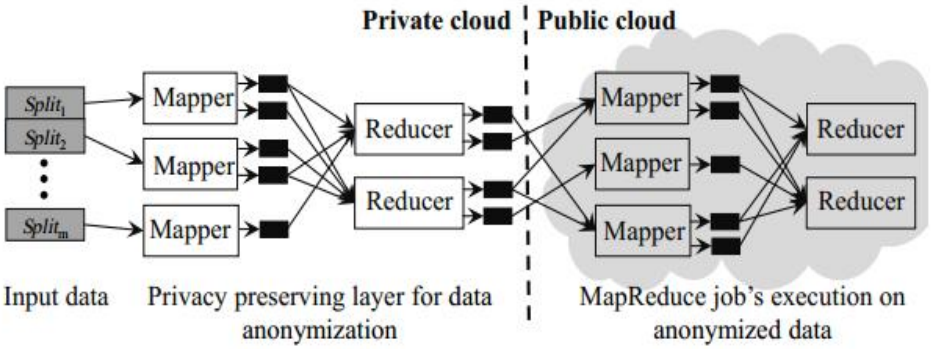


Figure 2.4 Cadre MapReduce avec couche préservant la confidentialité [7]

Plus précisément, la couche PPL se compose de quatre modules principaux, comme suit:

***Interface de spécification de confidentialité (PSI*)**: prend plusieurs paramètres comme spécifications de confidentialité.

Anonymisation de données (DA): effectue l'anonymisation de données à l'aide d'algorithmes d'anonymisation basés sur MapReduce et de spécifications de confidentialité

Mise à jour des données (DU): anonymisation de nouvelles données sans anonymisation des données à partir de zéro.

Gestion des ensembles de données anonymes (ADM): fournit des méthodes pour stocker des données anonymisées dans des Clouds publics sans enfreindre la confidentialité des données.

Airavat est le premier système combinant le contrôle d'accès obligatoire (MAC) et la confidentialité différentielle pour assurer la confidentialité des données conformément à la définition de confidentialité différentielle des utilisateurs non fiables. Airavat permet aux utilisateurs de soumettre des tâches MapReduce avec des mappeurs et des réducteurs personnalisés choisis dans un ensemble prédéfini de réducteurs approuvés. Le contrôle d'accès obligatoire garantit que les mappeurs non fiables ne fuient pas les données à l'extérieur du système via un réseau ou un système de fichiers. Les

exigences de confidentialité différentielle sont assurées sur les sorties intermédiaires en leur ajoutant un bruit aléatoire. Afin de maximiser l'utilisation du système et de minimiser la quantité de bruit ajouté, l'utilisateur doit fournir une gamme de valeurs de sortie pour chaque mappeur fourni. Si la sortie dépasse la plage, elle est mappée à une valeur aléatoire dans la plage. Naturellement, le système nécessite une confiance totale dans le fournisseur de Cloud qui implémente le protocole et assure l'intégrité de ses composants.

- ***Problèmes d'utilité***

Toutes les méthodes d'anonymisation des données ont un compromis inhérent entre l'utilisation et intimité. Étant donné que les méthodes d'anonymisation des données offrent un haut niveau de confidentialité des données, l'utilisation du système peut diminuer. En outre, il peut être difficile d'assurer des données anonymisées après avoir effectué des opérations sur celui-ci. Par exemple, la jonction d'une relation ayant des données anonymisées avec une autre relation ayant des données non anonymisées peut révéler des données de la première relation. **[7]**

2.7 Conclusion

Dans ce chapitre, nous avons présenté la préservation de la confidentialité lors de la publication et l'analyse des données. La préservation de la vie privée a pour objectif de permettre de protéger la confidentialité des données personnelles d'un individu tout en conservant l'utilité des données anonymes. Nous avons aussi détaillé la confidentialité différentielle qui constitue un modèle standard pour la préservation de la vie privée. Nous avons présenté les nouveaux défis exigés pour préserver la vie privé dans un environnement Big data. A la fin du chapitre, nous avons présenté quelques solutions qui concernent la préservation de la vie privée contre les utilisateurs malveillants dans le contexte Big Data.

Chapitre 3

Chapitre 3

Dé-identification des Données

3.1 Introduction

L'objectif de ce chapitre est de présenter notre solution du problème de la vie privé des données contre un utilisateur malveillant dans un environnement Map-Reduce. Notre solution consiste à dé-identifier les données privées avant les stocker chez un Cloud Map-Reduce. Nous proposons deux mécanismes pour la dé-identification : la généralisation et la suppression. La généralisation sert à remplacer une valeur par une autre plus générale, La suppression consiste à supprimé des informations sensibles.

Dans la section 2, nous présentons le modèle d'utilisateur malveillant contre lequel nous préservons la vie privé des données. A travers des exemples, nous montrons les types d'attaques possibles de ce type d'utilisateurs malveillants.

Dans la section 3, nous présentons notre solution au problème contre ce type d'utilisateurs malveillant. Nous présentons deux mécanismes : la généralisation et la suppression pour la dé-identification des données. Nous démontrons l'application de ces mécanismes sur les exemples vus dans la section 2. Dans la section 4, nous présentons l'architecture de notre application puis nous déterminons l'emplacement de notre application relativement aux différents acteurs intervenant dans le processus de publication et d'analyse des données dans un contexte Big Data. Nous menons une expérimentation pour évaluer notre approche dans la section 5. Puis nous conclusion le chapitre dans la section 6

3.2 Modèle d'adversaire

- ***Adversaire malveillant***

Ce type d'adversaire s'applique à un utilisateur qui tente d'apprendre, de modifier ou de supprimer des informations à partir des données en émettant diverses requêtes. En général, les fournisseurs de Cloud ne sont pas supposés être malveillants, car assurer la confidentialité des fournisseurs de Cloud malveillants nécessite un niveau élevé de mesures de confidentialité qui réduisent considérablement l'utilisation du Framework.

La figure 3.1 montre un modèle d'attaque de base. Supposons qu'un conservateur souhaite préserver la confidentialité pour $\boldsymbol{n}$ enregistrements dans un jeu de données $\boldsymbol{D}$. Cependant, un attaquant a toutes les informations sur $\boldsymbol{n_1}$ enregistre dans le jeu de données $\boldsymbol{D}$ sauf le $\boldsymbol{n}$ enregistrement. Ces enregistrements $\boldsymbol{n_1}$ peuvent être définis comme Informations d'arrière-plan Il / elle peut faire une requête sur l'ensemble de données $\boldsymbol{D}$ pour obtenir l'agrégat informations sur $\boldsymbol{n}$ enregistrements dans $\boldsymbol{D}$. Après avoir comparé la différence entre le résultat de la requête avec les informations de base, l'attaquant peut facilement identifier les informations de enregistrer.

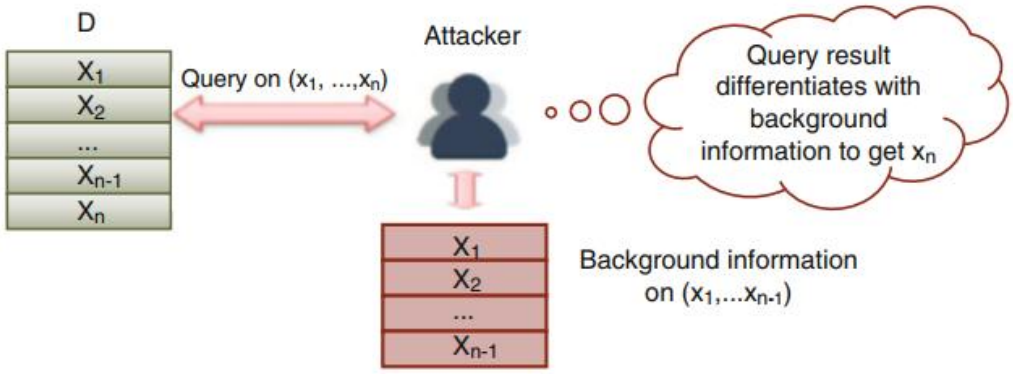


Figure3.1 Modèle d'attaquant [6]

3.3 Solution proposée

L'un des principaux objectifs de la solution proposée est de préserver la vie privée des individus. Les algorithmes de k-anonymisation sont susceptibles aux attaques qui peuvent utiliser l'arrière-plan individuel des «contributeurs» et les lier au référentiel, pour exposer les n-uplets appartenant à l'individu donné. Ils sont donc vulnérables aux attaques par couplage d'enregistrements et par attributs. Dans la littérature, on constate également que beaucoup de vie privée

Les modèles de préservation souffrent également de la liaison de table et des attaques probabilistes. Dans la solution proposée, la confidentialité différentielle (∈-différentiel privacy) sera utilisée pour protéger la date publiée à partir des attaques mentionnées ci-dessus.

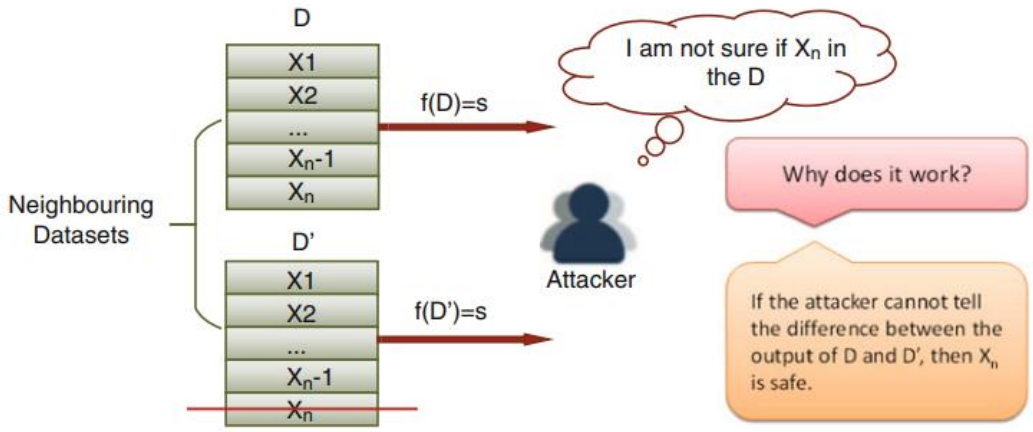


Figure 3.2 Protection de la vie privée différentielle [6]

3.4 Mécanismes de dé-identification

Notre système permet une anonymisation des données par deux processus principales: généralisation et suppression.

3.4.1 Généralisation

Définition: considérons une base de données $\mathbf{DB} = \mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_n$ être un ensemble d'enregistrements, où chaque enregistrement $\boldsymbol{r_i}$ représente l'information d'un individu avec des attributs

$$\mathbf{A} = \mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_d$$

On suppose que chaque attribut $\boldsymbol{A_i}$ a un domaine fini, noté $\boldsymbol{\Omega\ (A_i)}$. Le domaine de $\boldsymbol{DB}$ est défini comme

$$\boldsymbol{\Omega}(\mathbf{DB}) = \boldsymbol{\Omega}(\mathbf{A}_1) \times \boldsymbol{\Omega}(\mathbf{A}_2) \times \dots \times \boldsymbol{\Omega}(\mathbf{A}_d)$$

Pour anonymiser un $\boldsymbol{DB}$ d'ensemble de données, le processus de généralisation a lieu en substituant une valeur originale d'un attribut à une forme plus générale d'une valeur. La valeur générale exacte est choisie en fonction de la partition d'attribut.

Dans notre cas, on n'a pas tous les attributs qui ont un domaine fini. Par contre, quelques attributs contiennent une diversité des valeurs qui peuvent être dans la plupart du temps unique comme le nom, prénom...etc. Pour cela une étape de déterminais des attributs a domaine de valeur fini est nécessaire. A ce stade, notre approche consiste à détecter ce genre d'attribut en utilisant la distance de Levenshtein entre les différentes valeurs toute en détectant le nombre de répétition de chaque valeur

Par exemple: dans le cas d'un code ZIP, d'une adresse ou d'un identifiant AUTO_INCREMENT de la base de données, quelques caractères seront répètes dans plusieurs valeurs et même par fois on trouvera des distances Levenshtein égales a 0 ce qui veut dire que deux tuples partagent la même valeur. Dans ce cas, notre approche consiste a généraliser les valeurs des tuples partageant un nombre important de caractères en gardant ces caractères et en remplaçant le reste par des '*' (ZIP= 20000, 20002, 20005 seront remplace par la valeur 2000*)

Concernant les attributs dans les distances de Levenshtein montre une grande diversité des valeurs seront laisser pour l'étape de réduction est considères comme attributs sensibles nécessitant la suppression.

Le nombre de caractères maximum à garder pour qu'on peut généraliser un attribut est un paramètre que notre approche lit au début. Ce paramètre sera considère comme paramètre de sécurité qui doit rester secret.

3.4.2 Suppression

L'opération consiste à dégrader l'information initiale, en supprimant certaines données,

Une première étape, minimale consiste à supprimer les identifiants des fiches ou des bases de données concernées ; ces identifiants sont généralement :

- noms, prénoms (pseudonymisation du patient, du médecin, d'une fratrie).
- sexe.
- âge (éventuellement uniquement si supérieur à 90 ans).
- téléphone, télécopie, courriel, URLs, adresses IP.
- numéros identifiants.

Apres la suppression des valeurs sensibles, on aura des tuples de type <Key, value> qui ont le même Key. L'étape dernière consiste à regrouper ces tuples en enlevant la redondance et mettant à jour la valeur de tuple à la somme de la valeur de même clé.

3.4.3 Approche de dé-identification proposée

Nous proposant dans ce mémoire une approche qui consiste à utiliser la généralisation et la suppression sur des données structurées. Autrement dit, nous visant à utiliser la généralisation sur quelques attributs et la suppression sur d'autres attributs. Le choix des attributs à généraliser ou à supprimer est fait grâce à une métrique de distance entre les valeurs de l'attribut lui-même dont nous avons choisi la distance de Levenshtein comme métrique.

Plus précisément, les attributs ayant des valeurs dont elles partagent une partie de leurs contenus comme par exemple le ZIP code ou bien l'adresse domicile ou même l'adresse IP, peuvent être généralisés en remplaçant leurs valeurs par des valeurs communes plus générales. Exemple: dans le ZIP code où la plupart des valeurs partagent les mêmes 4 ou 5 premiers caractères, on peut remplacer les valeurs en gardant les caractères communs et remplaçant le reste par étoile. Un autre exemple concernant l'adresse domicile où l'attribut adresse généralement contient des valeurs de type "porte, rue, cité". Dans ce cas les valeurs seront modifiées en enlevant les numéros des portes et les rues et en gardant la cité. Les autres attributs d'où les valeurs présentent une grande diversité des valeurs seront considérés comme des attributs sensibles nécessitant la suppression. Cette opération, comme mentionné précédemment, est simple, elle consiste de remplacer les valeurs de l'attribut entièrement par une et une seule valeur unique qui peut être '*' ou '?' ou tout autre caractère.

Le pseudo-code suivant montre l'approche de dé-identification que nous proposons dans ce mémoire:

Algorithme de dé-identification

Entrées:

***BD**: une base de données claire*

***k:** paramètre de généralisation*

***d**: paramètre de distance minimale*

sorite:

***BD'**: base de données anonymisées*

début

pour tout attribut att_i de BD faire

k'=0;

compter val_i l'ensemble des valeurs sans répétition de att_i

pour tout valeur v_j de l'ensemble des valeurs de val_i faire

pour tout valeur v'_k de att_i

calculer distance $d'(v_j, v_k)$

si d' < d alors

k'++;

si k' < k alors

suppression (att_i)

si (non suppression (att_i)) alors

généraliser (att_i, min(d'));

ajouter att_i à BD'

retourner BD'

Fin

où 'k' représente un paramètre secret définis le taux de généralisation dont chaque valeur après dé-identification doit apparaître au moins k fois, ce paramètre va nous aider à comprendre l'apport de généralisation par rapport au métrique de perte d'information qu'on va expliquer par la suite.

En ce qui concerne le paramètre 'd' est un paramètre de distance de levenshtein dont il représente la valeur minimale pour que le système décide si deux valeurs peuvent être généralisées en une seule.

3.5 Conclusion

Dans ce chapitre, l'idée de préserver la confidentialité de la publication et l'analyse des données dans le contexte Big Data. Le but de ce travail est de mettre en œuvre un cadre pratique de préservation de la vie privée pour protéger la vie privée d'un individu tout en conservant les données anonymisées utiles au chercheur. Le principal avantage de ce travail est de promouvoir le partage de données pour l'exploration des connaissances. La confidentialité différentielle associée à la généralisation crée une forte garantie de confidentialité et un utilitaire de données pour les mineurs de données.

Chapitre 4

Chapitre4

Implémentation et démonstration

4.1 Introduction

L'objectif principale de ce chapitre est de présenter l'architecture générale de l'application développée et de déterminer son déploiement au sein de l'architecture générale d'un environnement Big Data tel que, Hadoop. A la fin le chapitre démontre l'impacte de notre approche sur la réponse aux requêtes des utilisateurs.

4.2 Implémentation

4.2.1 architecture de notre application

Notre application se fonctionne comme suit:

- Les données sont stockées dans hadoop dans le format <Key, value> où Key représente l'identifiant d'un tuple de données (identifiant d'un médecin par exemple) et value représente l'ensemble des données relatives à l'identifiant (par exemple l'ensemble des informations du médecin).

- Notre système consiste à appliquer une requête sur les données stockées. Par exemple on voulait savoir tous les médecins qui travail dans la spécialité "UROLOGY"

- Après l'application de la requête, le résultat en sortie consiste d'un ensemble de données contenant des informations sensibles. Pour cela, l'étape de de-identification arrive afin de cacher les informations.

La figure suivante montre l'architecture générale de notre système:

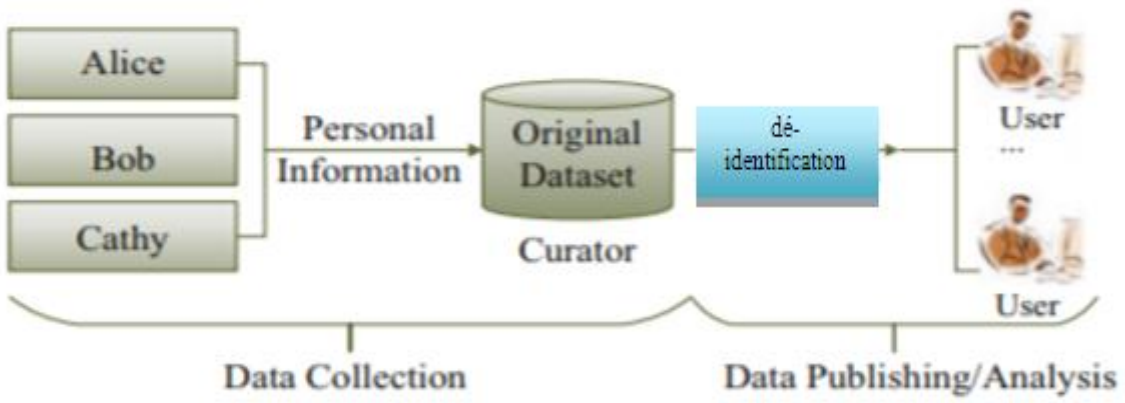


Figure 4.1 Confidentialité préservant la publication et l'analyse de données [6]

4.3 Evaluation

4.3.1 Dataset

Nous avons mené une série d'expériences en utilisant deux bases données: la première est construite a partir des ensembles de données de Comparaison des médecins fournis par ***Medicare & Medicaid Services (CMS)*** et la deuxième représente un fichier log de la NASA.

CMS

Le nom CMS, Content Management System, en français Système de Gestion de Contenu est un terme vague, puisqu'en fait tout logiciel gère un contenu. Plus précisément on donne ce nom à un logiciel qui gère la création et la publication de documents, éventuellement de façon collaborative. Sur ce site, on parlera plutôt des CMS en tant qu'applications web, donc de logiciels d'aide à la mise en ligne de documents sur Internet.

L'ensemble de données de comparaison des médecins **(CMS Physician Compare)** est dans son format principal un fichier csv qui contient 2.107.470 de lignes chacune représente un médecin ou un physicien professionnel de la santé actuellement inscrits dans l'assurance-maladie aux Etats-Unis. Le fichier de comparaison National Physician Compare comprend des informations générales distribuées en 47 attributs, telles que des informations démographiques et la participation au programme de qualité Medicare, pour les professionnels éligibles individuels (PE), qui peuvent être considérés comme des informations hautement sensibles.

NASA access_log

Le deuxième ensemble de données représente principalement une journalisation d'accès au serveur NASA. Cet ensemble comprend 1891714 lignes contenant des informations générales des accès au serveur web de la NASA par des différentes sites a travers le monde. Il comporte pas mal de données sensibles telles que les adresses des machines qui ont accéder au serveur et les chemins des données existant dans le serveur dont ces machines avaient accède. la structure générale de cette dataset est la suivante:

- Adresse IP distante ou nom de domaine: une adresse IP est une adresse hôte 32 bits définie par le protocole Internet; un nom de domaine est utilisé pour déterminer une adresse Internet unique pour n'importe quel hôte sur Internet. Une adresse IP est généralement définie pour un nom de domaine.
- Authuser: Nom d'utilisateur et mot de passe si le serveur requiert une authentification de l'utilisateur
- Entrer et sortir de la date et de l'heure.
- Modes de requête: méthode GET, POST ou HEAD de CGI (Common Gateway Interface).
- Statut: le code d'état HTTP renvoyé au client, par exemple, 200 est "ok" et 404 "non trouvé".
- Bytes: la longueur du contenu du document transféré.
- Journal distant et journal de l'agent.
- URL distante
- "request:" La ligne de requête exactement comme elle vient du client.
- URL demande

199.72.81.55 - - [01/Jul/1995:00:00:01 -0400] "GET /history/apollo/ HTTP/1.0" 200 6245

unicomp6.unicomp.net - - [01/Jul/1995:00:00:06 -0400] "GET /shuttle/countdown/ HTTP/1.0" 200 3985

Deux exemples de lignes fichier de NASA access_log

En générale, concernant le fichier log de NASA de Juilliet 1995 (la dernière version existe), il y a quelques informations enlevées de la version téléchargées comme l'Authuser.

4.3.2 Métrique de perte (Loss Metric)

Métrique de perte est défini en termes de perte normalisée pour Chaque attribut de chaque tuple. Pour un tuple t et un attribut catégoriel A, supposons que la valeur de $t\,[A]$ a été généralisée à x. Laisser $|A|$ représenter la taille du domaine de l'attribut Aet laissant Mreprésenter le nombre de valeurs dans ce domaine qui aurait pu être généralisé à x, alors la perte pour $t\,[A]$ est$(M-1)/(|A|-1)$. La perte pour l'attribut A est définie comme la moyenne de la pert e$t\,[A]$ pour tous les tuples t. Le LM pour l'ensemble des données est défini comme la somme des pertes pour chaque attribut**.**

4.4 Usage et Démonstration

Après description de l'architecture de notre application, cette section consiste à présenter quelques captures d'écran des interfaces de notre application.

En effet, puisque hadoop ne permet pas l'introduction des interfaces graphiques en java car l'exécution des programme java au sein de hadoop se fait à travers des processus en background ce qui rendre le suivi des traces difficile. Sur ce fait, que nous avons créé une copie local simulant notre approche dans nos machines afin de bien présenter et discuter les effets et les résultats de notre approche.

Dans la suite de cette section, nous allons présenter l'essentiel de résultats d'application de notre approche sur les deux dataset mentionnées précédemment.

La première interface graphique contient plusieurs boutons nous les mentionnons :

Boutons " choix " et de cela nous choisissons fichier dataset que ça soit **CMS Physician Compare** Ou **NASA.**

Boutons " Nombre d'éléments " c'est le bouton pour lequel nous sélectionnons un nombre, afin de sélectionner le nombre d'élément.

Boutons "Charger base d'information" cette partie charge l'information qui appartient à fichiers dataset Choisi et nombre d'éléments proposé.

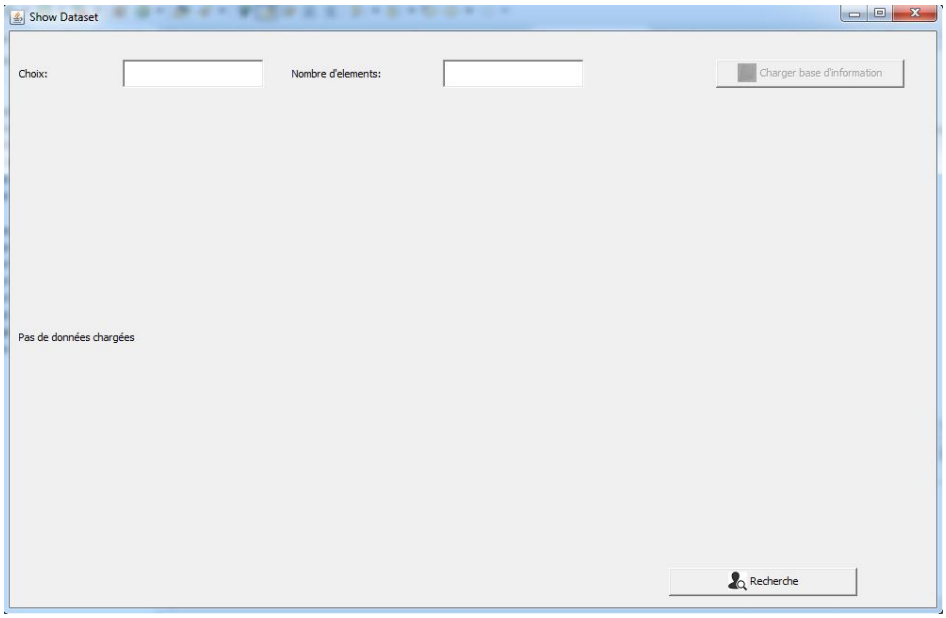
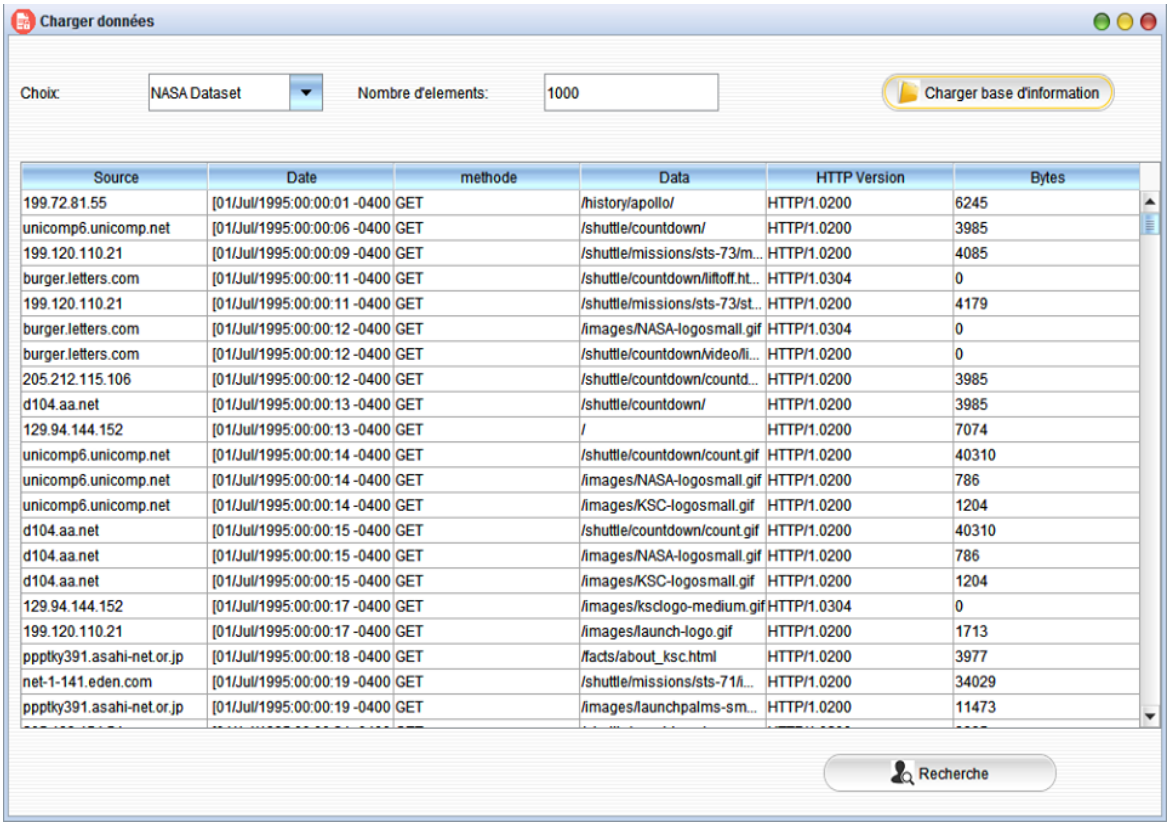


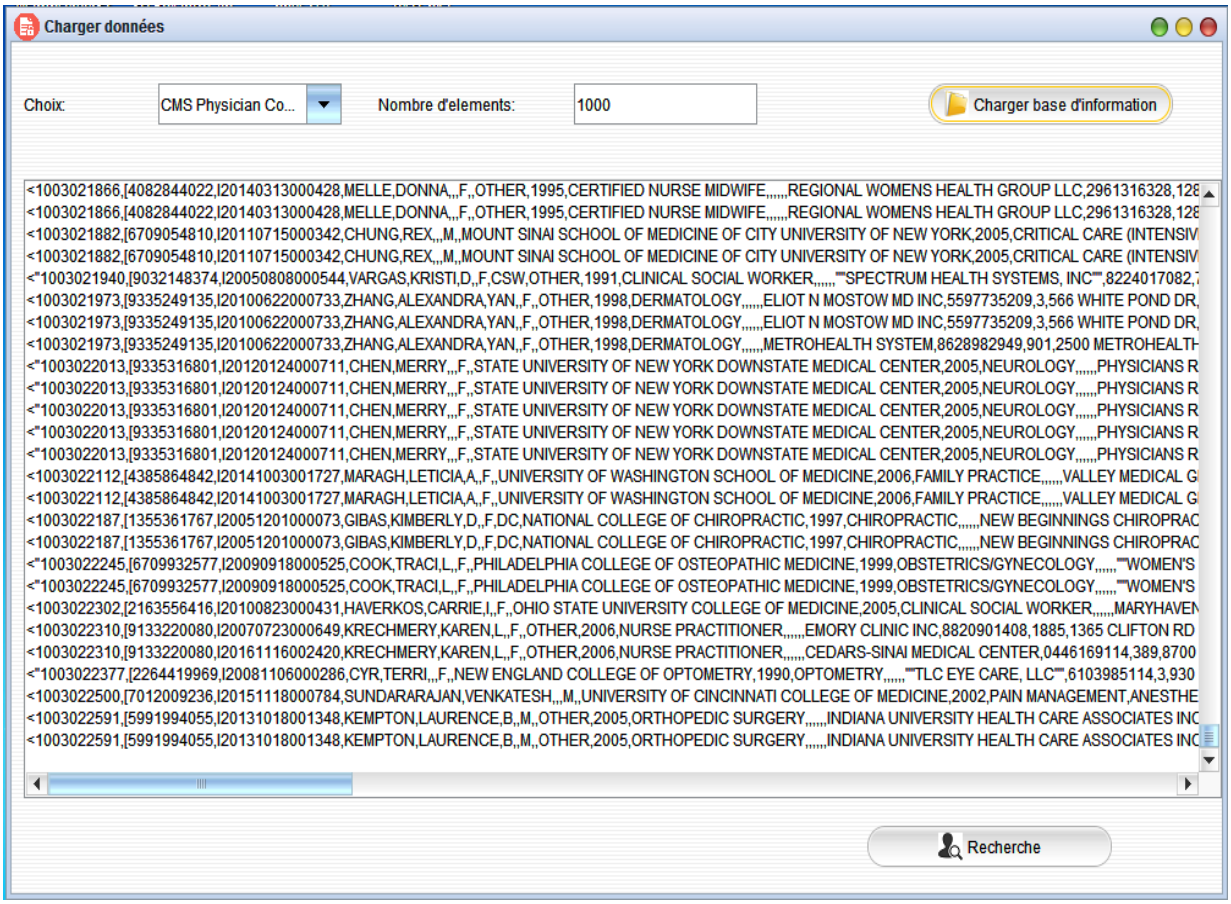
Figure 4.2 interface initiale

Cette interface présente l'interface de présentation des données où le bouton "charger" permet de charger et présenter les données de test selon le nombre définie dans l'espace de texte et la dataset choisi



Source	Date	methode	Data	HTTP Version	Bytes
199.72.81.55	[01/Jul/1995:00:00:01 -0400	GET	/history/apollo/	HTTP/1.0200	6245
unicomp6.unicomp.net	[01/Jul/1995:00:00:06 -0400	GET	/shuttle/countdown/	HTTP/1.0200	3985
199.120.110.21	[01/Jul/1995:00:00:09 -0400	GET	/shuttle/missions/sts-73/m...	HTTP/1.0200	4085
burger.letters.com	[01/Jul/1995:00:00:11 -0400	GET	/shuttle/countdown/liftoff.ht...	HTTP/1.0304	0
199.120.110.21	[01/Jul/1995:00:00:11 -0400	GET	/shuttle/missions/sts-73/st...	HTTP/1.0200	4179
burger.letters.com	[01/Jul/1995:00:00:12 -0400	GET	/images/NASA-logosmall.gif	HTTP/1.0304	0
burger.letters.com	[01/Jul/1995:00:00:12 -0400	GET	/shuttle/countdown/video/li...	HTTP/1.0200	0
205.212.115.106	[01/Jul/1995:00:00:12 -0400	GET	/shuttle/countdown/countd...	HTTP/1.0200	3985
d104.aa.net	[01/Jul/1995:00:00:13 -0400	GET	/shuttle/countdown/	HTTP/1.0200	3985
129.94.144.152	[01/Jul/1995:00:00:13 -0400	GET	/	HTTP/1.0200	7074
unicomp6.unicomp.net	[01/Jul/1995:00:00:14 -0400	GET	/shuttle/countdown/count.gif	HTTP/1.0200	40310
unicomp6.unicomp.net	[01/Jul/1995:00:00:14 -0400	GET	/images/NASA-logosmall.gif	HTTP/1.0200	786
unicomp6.unicomp.net	[01/Jul/1995:00:00:14 -0400	GET	/images/KSC-logosmall.gif	HTTP/1.0200	1204
d104.aa.net	[01/Jul/1995:00:00:15 -0400	GET	/shuttle/countdown/count.gif	HTTP/1.0200	40310
d104.aa.net	[01/Jul/1995:00:00:15 -0400	GET	/images/NASA-logosmall.gif	HTTP/1.0200	786
d104.aa.net	[01/Jul/1995:00:00:15 -0400	GET	/images/KSC-logosmall.gif	HTTP/1.0200	1204
129.94.144.152	[01/Jul/1995:00:00:17 -0400	GET	/images/ksclogo-medium.gif	HTTP/1.0304	0
199.120.110.21	[01/Jul/1995:00:00:17 -0400	GET	/images/launch-logo.gif	HTTP/1.0200	1713
ppptky391.asahi-net.or.jp	[01/Jul/1995:00:00:18 -0400	GET	/facts/about_ksc.html	HTTP/1.0200	3977
net-1-141.eden.com	[01/Jul/1995:00:00:19 -0400	GET	/shuttle/missions/sts-71/i...	HTTP/1.0200	34029
ppptky391.asahi-net.or.jp	[01/Jul/1995:00:00:19 -0400	GET	/images/launchpalms-sm...	HTTP/1.0200	11473

Données des NASA Log File



L'ensemble de données de CMS Physician Compare

Figure 4.3 Interfaces de présentation des données

L'interface suivante sert à permettre la recherche où le bouton "recherche clair" permet de faire une recherche d'un élément dans l'ensemble de données choisi précédemment avec un affichage détaillé concernant tous les lignes contenant la données recherchée et le nombre de répétition de chaque ligne.

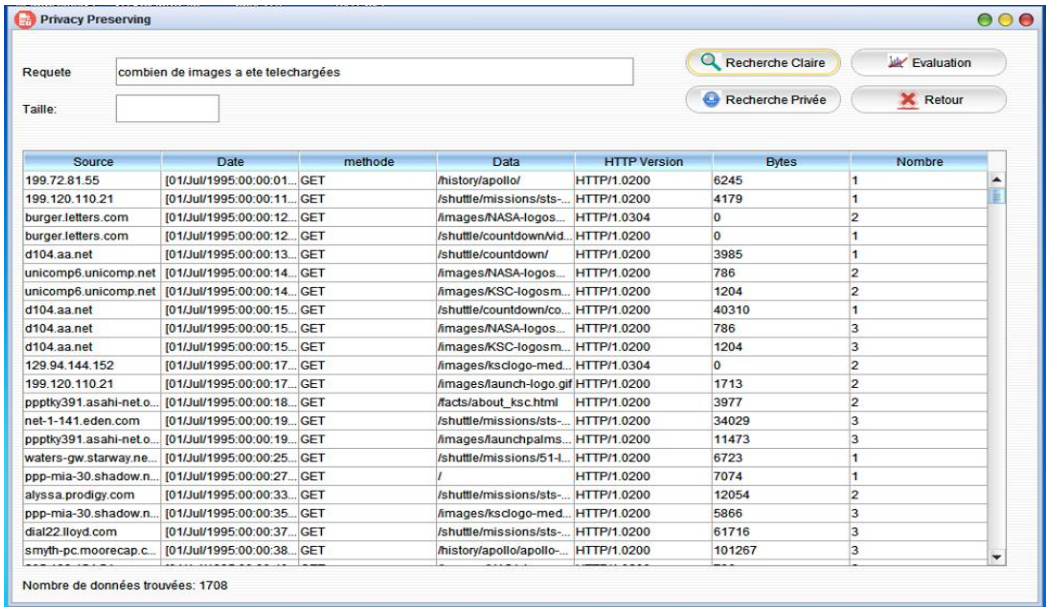


Figure 4.4 Exemple de démonstration de résultat de la recherche claire

Le bouton "recherche privée" présente l'objectif initial de notre projet d'où il permet une recherche et dé-identification des résultats avant les afficher. Comme décrit dans l'algorithme de chapitre précédent, une zone de texte permet le choix du paramètre k.

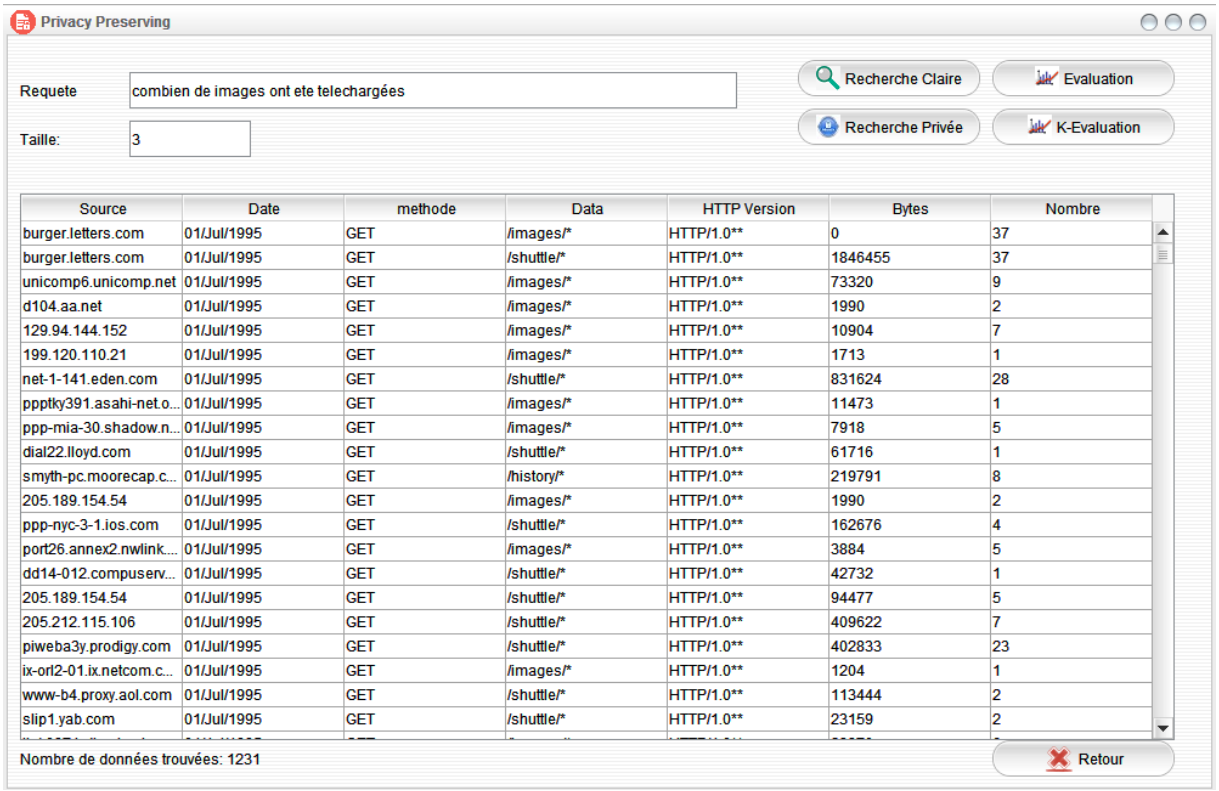


Figure 4.5 exemple de résultat de recherche privée sur les données de nasa avec k=3

De même, comme on remarque dans la figure au-dessus, en la comparant avec la figure 4.5 d'où il y a le label indiquant le nombre de données retournés qui a diminué de 1708 pour le clair à 1231 pour le privé en utilisant la même requête et cela dû au fait que la dé-identification des données permet de rendre plusieurs valeurs en une seule plus générale. Ainsi, on remarque la différence de nombre de répétition de chaque ligne à la fin.

Les boutons « évaluation » et « k-évaluation » sont ajouté afin d'évaluer notre approche. Leur fonctionnement sera décrit dans la section suivante lors de discussion des résultats :

4.5 Démonstration

Cette section est consacrée à la présentation des différents résultats qu'on a eu durant l'évaluation de l'approche proposée. Principalement l'évaluation a été faite en utilisant la métrique de perte de données décrite dans le chapitre 3.

Cette interface présente le Graphe de perte de données pour chaque attribut modifié par rapport à la requête utilisée.

Requête	Attribut date	Attribut data
Requête 1	0.02710761724044456	0.05513784461152882
Requête 2	0.027078256160303276	0.055
Requête 3	0.015306903413439462	0.024892703862660945
Requête 4	0.015306903413439462	0.024892703862660945
Requête 5	0.015306903413439462	0.024892703862660945
Requête 6	0.015306903413439462	0.024892703862660945

Table 4.1 résultats de perte de données par rapport à la requête utilisée avec k=3

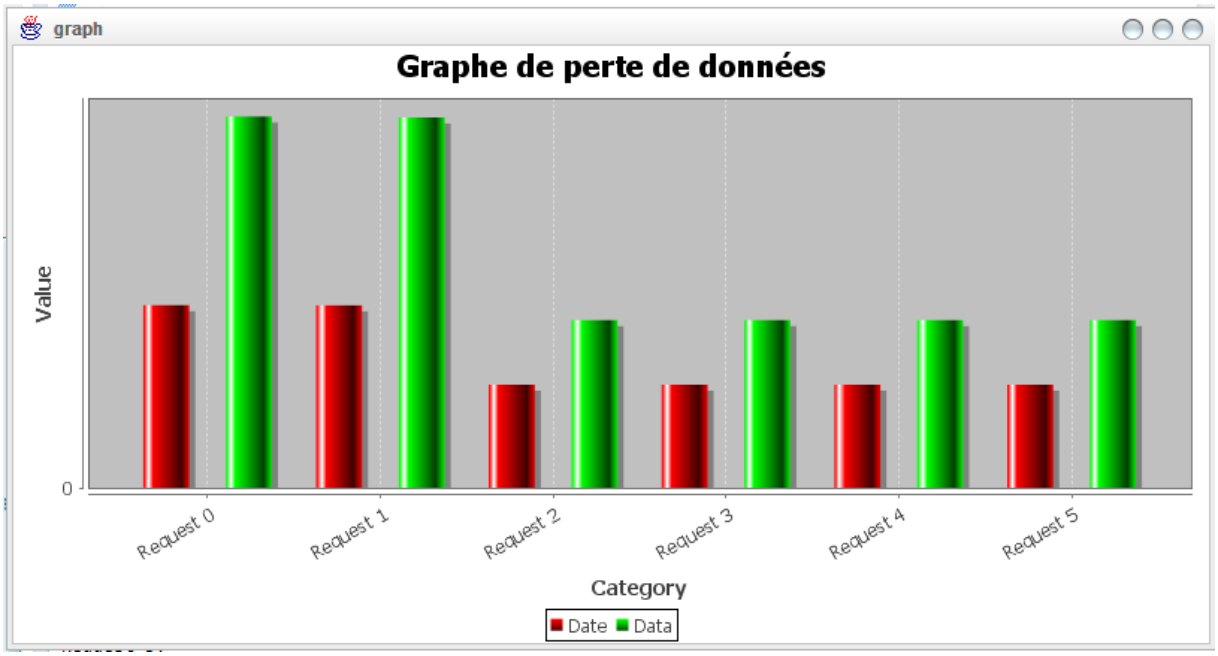


Figure 4.6graphe de perte de données par rapport à la requête utilisée avec k=3

Comme on remarque dans la figure4.7 et le tableau 4.1, les valeurs sont très proches et elles semblent les même dans la figure où l'attribut de la date n'a pas montrés une grande perte de données par rapport à l'attribut data a cause que la généralisation de ce dernier de fait qu'enlever l'heure de la requête sans modifier la date alors que la généralisation de l'attribut data présente une perte un peu plus large car la généralisation de ce dernier permet de supprimer plus de caractères des valeurs ce qu'il réduit le nombre des valeurs de domaine de l'attribut data. La cause que les valeurs de perte de données sont proches est due à la qualité de l'ensemble de données de NASA traité comme vous avez remarqué dans la figure 4.2 à gauche. Autrement dit, les valeurs des attributs sont courtes et présente des instances très similaires en terme distance de Levenshtein.

La deuxième évaluation nous visons à évaluer l'impact du paramétrage de k sur les résultats de perte de données. La figure 4.8 montre les résultats obtenus en variant le k de 1 à 3. La raison de choix de k sur des petites valeurs revient toujours sur la taille des valeurs des attributs de l'ensemble de données.

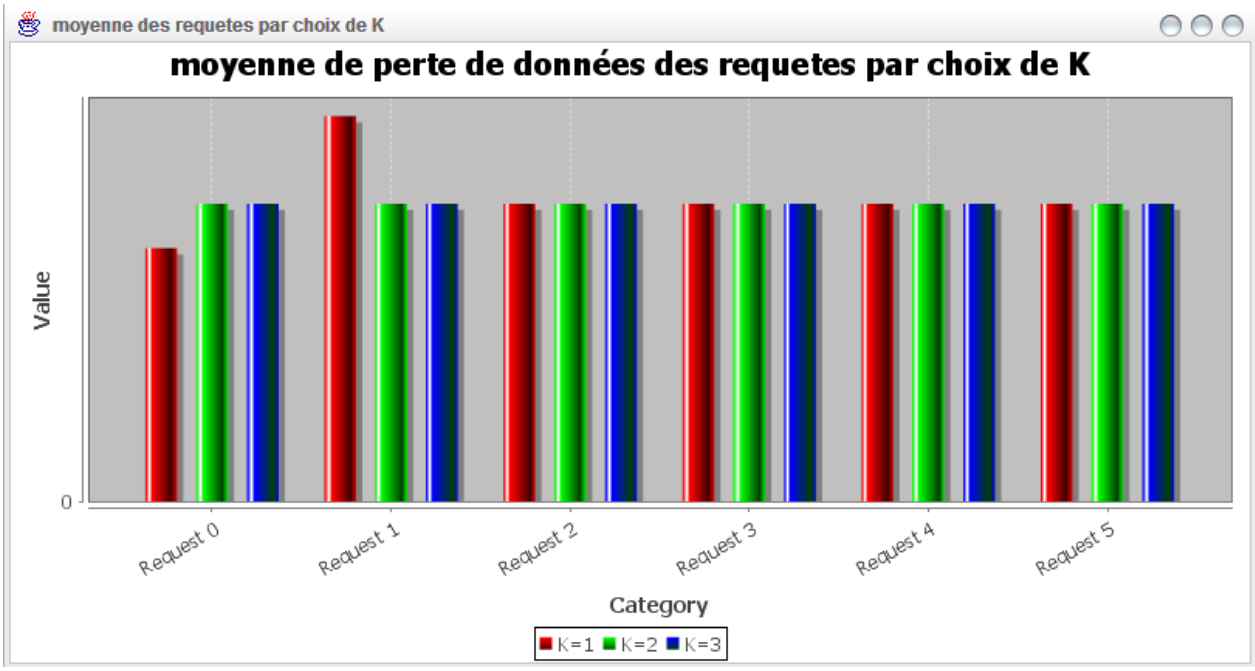


Figure 4.7 résultats de perte de données par rapport à la requête utilisée en variant le k

Comme montré dans la figure au-dessus, nous remarquons clairement que le paramètre k n'a pas vraiment un impact important sur la perte de données tant qu'on généralise les données vers des valeurs plus générales en changeant seulement quelques caractères (qui ne dépasse pas les 4 ou 5 caractères par valeur) ce qui presque ne présente pas un changement radical dans les données. En effet, en analysant les valeurs de cette figure, la perte de données n'a pas été importante (elle est au tour des 0,013) mais nous estimons pour des grosses bases de données contenant les centaines ou milliers d'attributs sensibles elle peut être importante.

La figure 4.8 montre les résultats de perte de données par rapport à la requête utilisée en fixant le k=5 appliqué sur un échantillon de 10 000 données (l'exécution de l'approche sur l'ensemble de plus de deux million est quasi-impossible vu au matériel disponible).

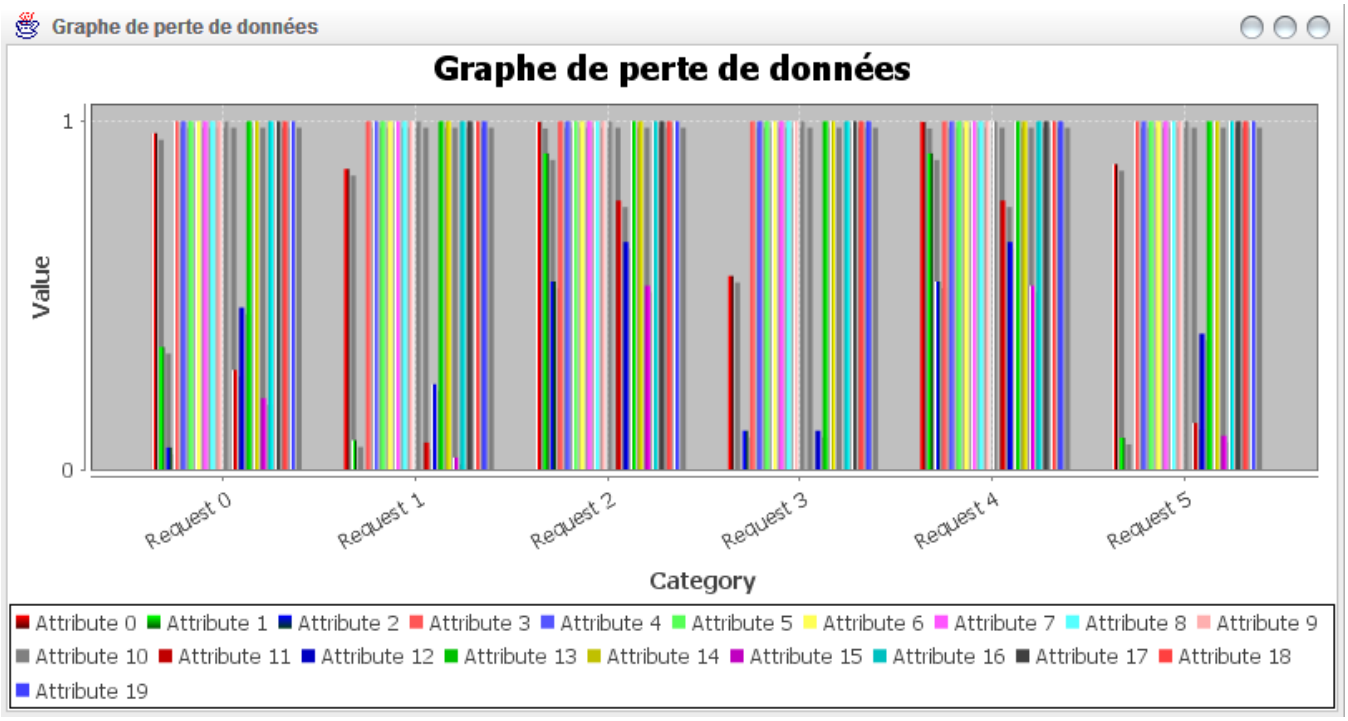


Figure 4.8 graphe de perte de données par rapport à la requête utilisée avec k=5

En observant la figure au-dessus, nous remarquons clairement que pour les attributs sensibles dont leurs valeurs ont été supprimées carrément ont montré une perte totale de leurs données ce qui est logique car ces derniers contiennent généralement des données uniques comme les noms, prénoms...etc. En revanche en ce qui concerne les attributs généralisées, nous remarquons l'effet de contenu de la requête sur les résultats dont la première et la quatrième requêtes ont montré une perte importante de quelques attributs que les autres requêtes ont fait.

La deuxième évaluation, comme prévu, sert à évaluer l'impact de paramétrage de k sur la perte des données. La figure 4.9 montre le résultat de la moyenne de perte des données par requête avec la variation de k de 1 à 7.

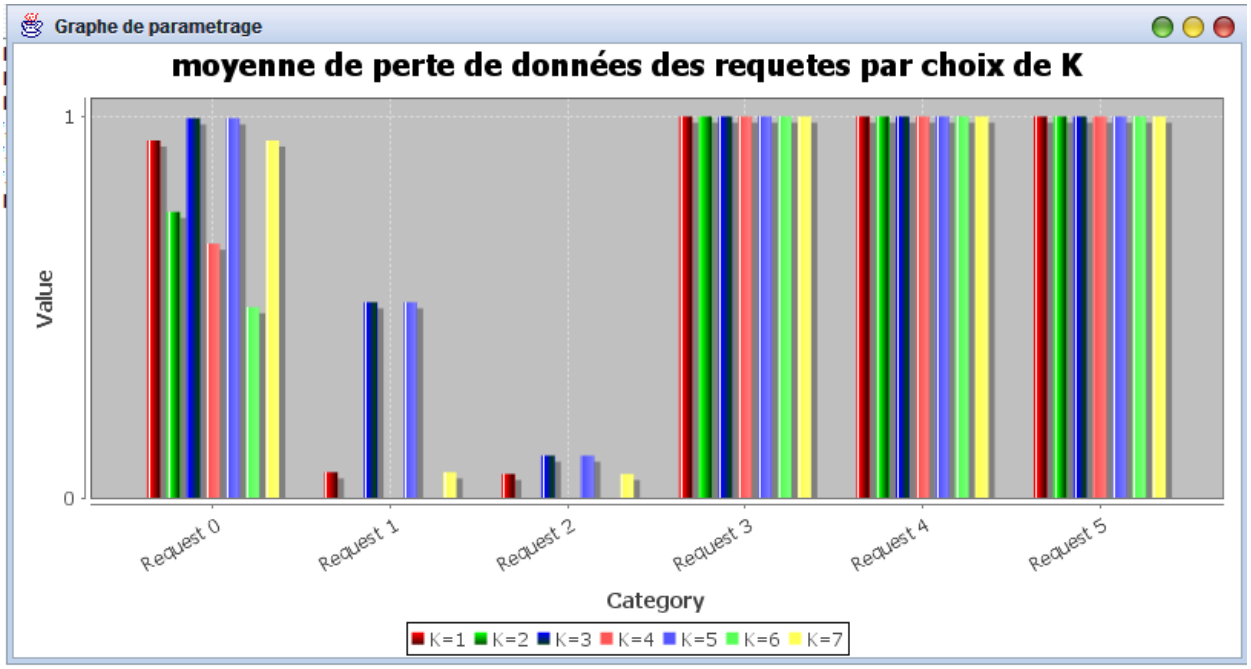


Figure 4.9 graphe de moyenne de perte de données par requête avec variation de k

Dans ce cas, contrairement aux résultats correspondants à la dataset de NASA, nous remarquons que la variation de k sur un nombre important de valeurs et attributs avait montrés un impact important dont l'augmentation de k avais dans la plupart des cas causé une perte importante des données et surtout concernant les données généralisées tout en dépend toujours par le contenu de la requête utilisée. Apparemment, au terme de moyenne de perte, nous remarquons que les requêtes 3, 4 et 5 ont montrés des pertes presque totale et cela dû à la perte totale des valeurs dans les cas de k=5, k=6, et k=7. Par contre les requêtes 1 et 2 ont montrés des résultats efficaces.

4.5.1 Discussion générale

En analysant les résultats présentés dans cette section, nous pouvons clairement déduire quelques points de conclusion sur l'effet de la dé-identification et plus précisément de l'approche proposée, ces points sont résumés comme suit :

- assurer une haute confidentialité tout en minimisant la perte de données peut être considéré comme un paradoxe dans le domaine de la recherche privée.

- le contenu de la requête joue un rôle important dans la qualité des résultats retournés, une requête demandant plus d'information peut provoquer un résultat plus précis et plus large selon le traitement associé ce qui affecte directement les résultats de la dé-identification

- le paramétrage de l'approche en choisissant la valeur de k affecte aussi le résultat. Un choix de k plus petit ne cause presque aucune perte de résultat tant que les données ne doit pas être répétées moins qu'un k petit ce qui veut dire que moins de perte de données plus de précision et moins de confidentialité alors que le choix des valeurs plus au moins élevées peut causer une très haute confidentialité en cachant le maximum d'information mais cela conduit aussi à une perte importante des données ce qui affecte la précision de la recherche.

4.5.2 Code mapreduce pour Hadoop

Concernant le code java implémenté dans Hadoop, nous avons utilisé le même principe que le fameux exemple WordCount.

L'image map code présente une partie de mapper d'où il sert a lire les fichiers et charger les données afin de les représenter sous forme de <key, value> dont key représente la clé et contient une ligne de l'ensemble de données identifiants une données spécifique. Et value représente le nombre de répétition qui est généralement égale à 1.

chargeData.java | mainClass.java | recherche.java | PrivacyMapper.java | PrivacyReducer.java | PrivacyDriver.java

```
public void map(WritableComparable key, Writable value,
    OutputCollector[] output, Reporter reporter) throws IOException {
    output=new OutputCollector[2];
    try {
        BufferedReader rd=new BufferedReader(new FileReader("dataset/phys.csv"));
          String s=rd.readLine();

          for(int i=0;i<1000;i++){
              s=rd.readLine();
              String[] obj=s.split(",");
              cmsdata.add(obj);
              //System.out.println("cms: "+tableToString(obj));
               output[0].collect(obj, one);
          }

        BufferedReader rd2=new BufferedReader(new FileReader("dataset/nasa"));
          String s2=rd2.readLine();
          String[] columns=s2.split("- -");
          for(int i=0;i<1000;i++){
              String[] raw=new String[6];
              String[] obj=s.split("- -");
              raw[0]=obj[0];
              String[] inf=obj[1].split("] ");
              raw[1]=inf[0];
              String[] dat=inf[1].replaceAll("\"", "").split(" ");
              raw[2]=dat[0];
              raw[3]=dat[1];
              if(dat[2].startsWith("H")){
                  raw[4]=dat[2]+dat[3];
                  raw[5]=dat[4];
              }
              else {
                  raw[4]=dat[2];
                  raw[5]=dat[3];
              }
              nasadata.add(raw);
               output[1].collect(raw, one);
              s2=rd.readLine();
          }
```

Figure 4.10 : code de mapper

Reduce code sert à faire une recherche privée sur les données chargées dans la phase de mapping. Cependant, c'est à cette phase où l'approche de dé-identification prend place afin de sécuriser les résultat de recherche. Le reducer consiste à regrouper les données sécurisées ayant la même clé (key) dans une seule en mettant à jour la valeur à la somme des nombres de répétitions correspondants.

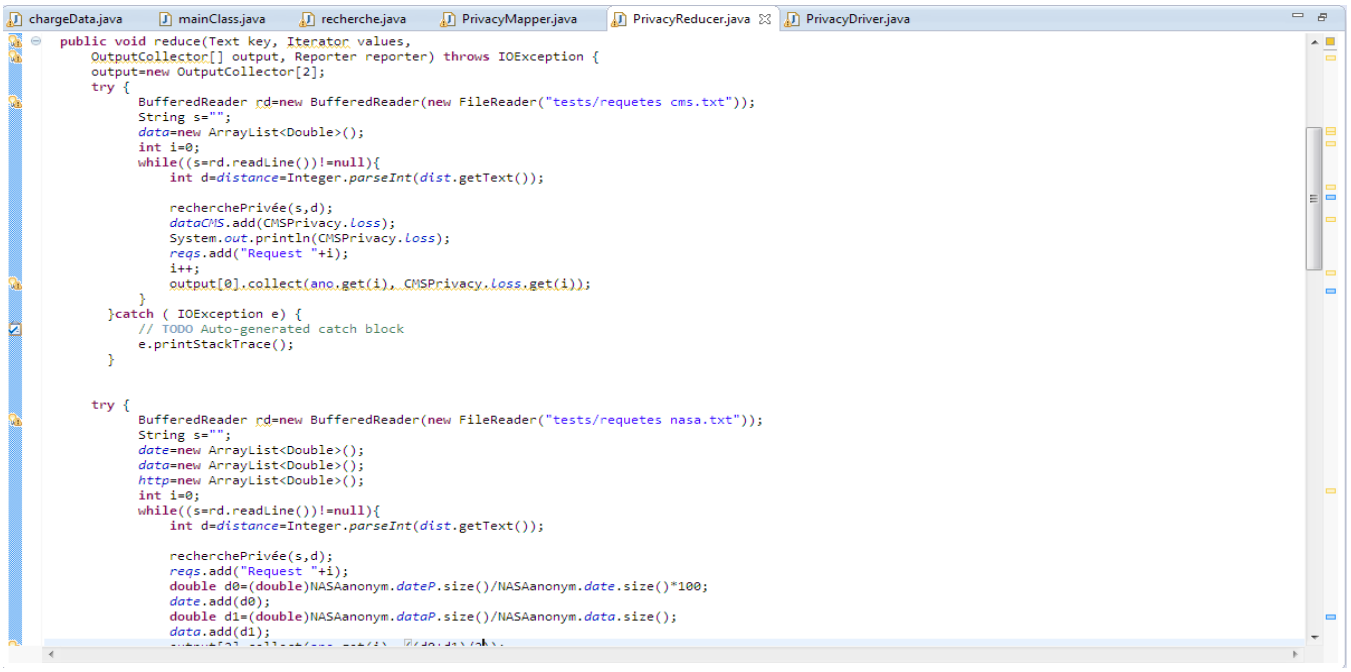


Figure 4.11 : code reducer

En ce qui concerne les requêtes, nous avons utilisé le même principe présenté dans les résultats au-dessus. Nous avons défini des fichiers contenant les mêmes requêtes dont le reducer sert à appliquer chacune des requêtes sur l'ensemble de données correspondant et enregistrer le code dans un fichier de sortie dans hadoop afin permettre une consultation futur.

mapreduce job c'est le job principale dont le traitement principale prend place. Dans cette classe, on fait appel au mapper au début pour charger les donner et puis au reducer pour la recherche.

chargeData.java | mainClass.java | recherche.java | PrivacyMapper.java | PrivacyReducer.java | PrivacyDriver.java

```
package hadoop;
import org.apache.hadoop.fs.Path;

public class PrivacyDriver {

  public static void main(String[] args) {
    JobClient client = new JobClient();
    JobConf conf = new JobConf(PrivacyDriver.class);

    // specify output types
    conf.setOutputKeyClass(Text.class);
    conf.setOutputValueClass(IntWritable.class);

    // specify input and output dirs
    //FileInputPath.addInputPath(conf, new Path("input"));
    conf.setInputPath(new Path("tests"));
    //FileOutputPath.addOutputPath(conf, new Path("output"));
    conf.setOutputPath(new Path("output"));

    // specify a mapper
    conf.setMapperClass(PrivacyMapper.class);

    // specify a reducer
    conf.setReducerClass(PrivacyReducer.class);
    conf.setCombinerClass(PrivacyReducer.class);

    client.setConf(conf);
    try {
      JobClient.runJob(conf);
    } catch (Exception e) {
      e.printStackTrace();
    }
  }
}
```

Figure 4.12 : code de job principale de mapreduce

4.6 Conclusion

Dans ce chapitre nous avons présenté la partie pratique de notre travail de ce fait nous avons présenté les détails de notre approche

On a présenté les différents résultats qu'on peut conclure dans les points suivants:

- assurer une haute confidentialité tout en minimisant la perte de données peut être considéré comme un paradoxe dans le domaine de la recherche privée.

- le contenu de la requête joue un rôle important dans la qualité des résultats retournés, une requête demandant plus d'information peut provoquer un résultat plus précis et plus large selon le traitement associé ce qui affecte directement les résultats de la dé-identification

- le paramétrage de l'approche en choisissant la valeur de k affecte aussi le résultat. Un choix de k plus petit ne cause presque aucune perte de résultat tant que les données ne doit pas être répétées moins qu'un k petit ce qui veut dire que moins de perte de données plus de précision et moins de confidentialité alors que le choix des valeurs plus au moins élevées peut causer une très haute confidentialité en cachant le maximum d'information mais cela conduit aussi à une perte importante des données ce qui affecte la précision de la recherche.

Conclusion générale

De nos jours, personne ne peut nier du rôle que l'internet joue souvent dans notre vie quotidienne. Différents réseaux de l'ADSL au 4G, ainsi que l'apparition du phénoméne des réseaux sociaux d'où les gens ont trouvé leur humour dans une sorte de vie virtuelle. Cependant, ces derniers avec les services de stockage et traitement dans les nuages informatique connu par leur nom anglais "CLOUD COMPUTING" ont causé une explosion en terme de volume et type des données créées et partagées sur internet tout en donnant naissance à un nouveau terme connu par "BIG DATA".

les BIG DATA occupent ces dernières années un intérêt important au sein de la communauté scientifique afin de tirer profit d'eux. Toutefois, cet intérêt croissant vise à exploiter les données réelles partagées afin de garantir des services beaucoup plus fiable que les approches testées au laboratoires. Autrement dit, la recherche se redirectionne afin de réduire l'écart entre la vie réelle et les laboratoires de recherche. Des technologies ont apparues comme le fameux "HADOOP" permettant de présenter des techniques plus avancées du traitement et représentation de la données, notamment le "MAPREDUCE"

En revanche, cela a permit aux services de BIG DATA d'être le "BIG BROTHER" de tout le monde ce qui a provoqué une résistance sévere contre eux a cause des problèmes de confidentialité des données. C'est dans ce stade que même les chercheurs en sécurité informatique ont commencé à donner plus d'intérêt au domaine de la préservation de confidentilaité tout en garantissant l'utilité des données. D'une autre façon, la sécurité des données dans les BIG DATA vise à garantir deux aspects paradoxials, d'un coté assurer un maximum de confidentialité en cachant les données personnelles et sensibles et d'un autre coté d'assurer la même utilité des données modifiées comme si elles ne sont pas.

Différentes techniques ont été élaborées dans le domine en citant la confidentialité différentielle, la perturbation des données et la cryptographie. C'est à cet objectif que notre mémoire s'inscrit afin de concevoir et réaliser une tel approche.

Dans ce mémoire, nous avons élaborées une technique dite de dé§identification des données dont l'objectif et de permettre une automatisation de détection et modification des attributs sensibles dans les données structurées. Nous avons utilisé deux techniques de modification l'une consiste a changer les valeurs de l'attribut en préservant une partie dite générale, cette technique est appelée généralisation. Et l'autre consiste à changer les valeurs entièrement par une valeur unique dont elle est appelée la suppression. Notre approche consiste à utiliser ces deux techniques sur différentes

attributs en choisissant pour chaque attribut la technique qui convient en basant sur la distance de Levenstein et le paramétrage d'anonymisation k décrit précédemment dans les chapitres.

Aprés une série d'expérimentation sur deux ensembles de données des physicien et médecins aux états unis et le fichier log de NASA. nous avons conclu ce mémoire par une suite de remarques concernant l'impact de contenu des données, leurs taille et le paramétrage de l'algorithme proposé sur la qualité de résultat. En résumé, nous avons conclu qu'un meilleur choix de paramétrage peut conduire à une haute protection de confidentialité des données tout en minimisant la perte d'information. Notre algorithme a présenté des résultats satisfesante sur ce terme.

Comme le domaine de la préservation de confidentialité est encore jeune et nouveau, nous visons au futur d'améliorer notre approche tout en permettant l'automatisation du paramétrage notamment dans le choix d'une valeur appropriée de k, ainsi l'introduction des techniques avancées de data mining et d'optimisation et l'utilisation des méta-heuristiques afin de permettre une amélioration des résultats et teste de validité de l'algorithme.

A la fin de ce rapport, il ne nous reste qu'à mentioner que la sécurité et la confidentialité dans les BIG DATA et l'internet en générale et malgré ses recommandations assez complexes, elle repose souvent sur le concept de confiance aux services et leurs fournisseurs.

Conclusion générale

De nos jours, personne ne peut nier du rôle que l'internet joue souvent dans notre vie quotidienne. Différents réseaux de l'ADSL au 4G, ainsi que l'apparition du phénomène des réseaux sociaux d'où les gens ont trouvé leur humour dans une sorte de vie virtuelle. Cependant, ces derniers avec les services de stockage et traitement dans les nuages informatique connu par leur nom anglais "CLOUD COMPUTING" ont causé une explosion en terme de volume et type des données créées et partagées sur internet tout en donnant naissance à un nouveau terme connu par "BIG DATA".

Les BIG DATA occupent ces dernières années un intérêt important au sein de la communauté scientifique afin de tirer profit d'eux. Toutefois, cet intérêt croissant vise à exploiter les données réelles partagées afin de garantir des services beaucoup plus fiables que les approches testées aux laboratoires. Autrement dit, la recherche se redirectionne afin de réduire l'écart entre la vie réelle et les laboratoires de recherche. Des technologies ont apparues comme le fameux "HADOOP" permettant de présenter des techniques plus avancées du traitement et représentation de la donnée, notamment le "MAPREDUCE"

En revanche, cela a permit aux services de BIG DATA d'être le "BIG BROTHER" de tout le monde ce qui a provoqué une résistance sévère contre eux a cause des problèmes de confidentialité des données. C'est dans ce stade que même les chercheurs en sécurité informatique ont commencé à donner plus d'intérêt au domaine de la préservation de confidentialité tout en garantissant l'utilité des données. D'une autre façon, la sécurité des données dans les BIG DATA vise à garantir deux aspects paradoxials, d'un coté assurer un maximum de confidentialité en cachant les données personnelles et sensibles et d'un autre coté d'assurer la même utilité des données modifiées comme si elles ne sont pas.

Différentes techniques ont été élaborées dans le domine en citant la confidentialité différentielle, la perturbation des données et la cryptographie. C'est à cet objectif que notre mémoire s'inscrit afin de concevoir et réaliser une tel approche.

Dans ce mémoire, nous avons élaborées une technique dite de dé§identification des données dont l'objectif et de permettre une automatisation de détection et modification des attributs sensibles dans les données structurées. Nous avons utilisé deux techniques de modification l'une consiste a changer les valeurs de l'attribut en préservant une partie dite générale, cette technique est appelée généralisation. Et l'autre consiste à changer les valeurs entièrement par une valeur unique dont elle est appelée la suppression. Notre approche consiste à utiliser ces deux techniques sur différentes

attributs en choisissant pour chaque attribut la technique qui convient en basant sur la distance de Levenshtein et le paramétrage d'anonymisation k décrit précédemment dans les chapitres.

Après une série d'expérimentation sur deux ensembles de données des physiciens et médecins aux états unis et le fichier log de NASA. Nous avons conclu ce mémoire par une suite de remarques concernant l'impact de contenu des données, leurs tailles et le paramétrage de l'algorithme proposé sur la qualité de résultat. En résumé, nous avons conclu qu'un meilleur choix de paramétrage peut conduire à une haute protection de confidentialité des données tout en minimisant la perte d'information. Notre algorithme a présenté des résultats satisfaisants sur ce terme.

Comme le domaine de la préservation de confidentialité est encore jeune et nouveau, nous visons au futur d'améliorer notre approche tout en permettant l'automatisation du paramétrage notamment dans le choix d'une valeur appropriée de k, ainsi l'introduction des techniques avancées de data mining et d'optimisation et l'utilisation des méta-heuristiques afin de permettre une amélioration des résultats et teste de validité de l'algorithme.

A la fin de ce rapport, il ne nous reste qu'à mentionner que la sécurité et la confidentialité dans les BIG DATA et l'internet en générale et malgré ses recommandations assez complexes, elle repose souvent sur le concept de confiance aux services et leurs fournisseurs.

Références bibliographiques

[1] Fabrice DEMARTHON, Denis DELBECQ, and Grégory FLECHET. La déferlante des octets. CNRS, (269), 2012.

[2] Abdesalam AMRANE. Big data : Concepts et cas d'utilisation. Rapport, CERIST, 2015.

[3] Abdeldjalil FELLAH, Abdelhamid BAACH. Une Approche Scalable pour le Traitement de Grande Quantité de Données. Année Universitaire : 2015 / 2016.

[4]http://dspace.univtlemcen.dz/bitstream/112/8090/1/implantation_du_modele_mapreduce_dansenvironnement_distribue%20_hadoop_distribution%20_cloudera.pdf
11/02/2018 18:50h
[5] Angeline KONE. Big data (rapport de stage). INSA LYON - Mastère spécialisé SI 2013.

[6] Cette traduction livre de l'anglais vers le français.
Book-Differential privacy And Applications1 4/11/2018 22:30h

[7] Security and Privacy Aspects in MapReduce on Clouds 02 /11/2018 23:25h