**Rapport sur la partie I du TER “traitement du flux de données horodatées” Hadja maimouna BALDÉ**

**Partie I : Etat de l’art sur les systèmes (architectures data / logicielle) de traitement de données horodatées**

**A) Définition des données horodatées**

Le traitement de flux de données horodatées est conçu :

* pour le traitement instantané des données (c à d des données qui sont encore en mouvement) et leurs analyses en temps réel.
* pour enregistrer aussi l’instant auquel une opération a été effectuée.

L’objectif principal est de fournir des informations actualisées à la milliseconde sur les événements qui se produisent dans un système et de nous aider à réagir aux événements critiques dès qu’ils se produisent. De ce fait, ce traitement est très avantageux pour les entreprises, car ça nous permet d’avoir des informations en temps réel sur la santé d’un système.

Il existe différents domaines dans lesquels ces données sont utilisés telle que :

* La détection des fraudes : techniques d’analyse des données pour la détection des fraudes
* Intelligence artificielle : les données brutes sont traitées comme un ensemble d'entraînement/apprentissage et un ensemble de tests lors de la construction d'algorithmes d’IA et d'apprentissage automatique.

**B) Les solutions existantes pour traiter les données horodatées**

En général il y a deux types de systèmes qui permettent de traiter les données horodatées :

1. un autour de kafka et,
2. un autre autour des bases de données time series.

1. Il existe plusieurs alternatives de kafka qui permettent de recevoir les données en temps réel. Le tableau ci dessous nous expliques et différencie quelques alternatives

| Nom | Apache KAFKA | Apache flink | Apache Étincelle | Tempête Apache | Apache Samza |
| --- | --- | --- | --- | --- | --- |
| Description | Apache Kafka est un logiciel sur site et basé sur le cloud conçu pour aider les utilisateurs d'Apache Kafka à gérer et à gérer la configuration des clusters afin de faciliter les processus de création et de développement d'applications. Les superviseurs peuvent utiliser la console de gestion pour obtenir une vue d'ensemble des flux de données en temps réel à l'aide de statistiques de surveillance et d'analyses exploitables  Icône de validation par la communauté | Apache Flink est un framework et un moteur de traitement distribué pour les calculs avec état sur des flux de données illimités et limités. Flink a été conçu pour fonctionner dans tous les environnements de cluster courants et effectuer des calculs à une vitesse en mémoire et à n'importe quelle échelle. | Apache Spark est un moteur d'analyse unifié pour le traitement de données à grande échelle. Il se distingue par ses hautes performances pour les données par lots et en continu en utilisant un planificateur DAG, un optimiseur de requêtes et un moteur d'exécution physique | Apache Storm est un système de calcul en temps réel distribué gratuit et open-source. Apache Storm facilite le traitement fiable de flux de données illimités, faisant pour le traitement en temps réel ce que Hadoop a fait pour le traitement par lots. | Apache Samza vous permet de créer des applications avec état qui traitent les données en temps réel à partir de plusieurs sources, dont Apache Kafka. |
| Déploiement | Basé sur cloud | Basé sur cloud et sur site | Basé sur cloud et sur site |  |  |
| Première version | Janvier 2011 | 6 janvier 2015 (28 octobre 2022) |  |  |  |
| Version actuelle | 3.3.1 (29 septembre 2022) | 1.16.0 |  |  |  |

2. Les bases de données conçues pour traiter les séries temporaires sont multiples mais parmi les plus populaires on a KDB+ et influxDb.

**Comparaison entre influxDB, KDB+ et timescaleDB**

| **Informations editorials fournies par DB-Engines** | | | |
| --- | --- | --- | --- |
| Nom | InfluxDB | KDB+ | TimescaleDB |
| Description | SGBD pour stocker des séries chronologiques, des événements et des métriques. | SGBD de séries chronologiques hautes performances. | Un SGBD de séries chronologiques optimisé pour l’ingestion rapide et les requêtes complexes, basé sur PostgresQL. |
| Modèle de base de données principal | SGBD de séries temporelles | SGBD de séries temporelles | SGBD de séries temporelles |
| Modèle de base de données secondaire | SGBD spatial | SGBD relationnel | SGBD relationnel |
| Classement des moteurs DB | Score 30.25  Rang #28 Globalement  #1 SGBD de séries temporelles | Score 7.23  Rang #59 Globalement  #2 SGBD de séries temporelles | Score 4.76  Rang #82 Globalement  #5 SGBD de séries temporelles |
| Première version | 2013 | 2000 | 2017 |
| Version actuelle | 2.5.1, novembre 2022 | 3.6, mai 2018 | 2.6.0, février 2022 |
| Licence | Open source | commercial | open source |
| Basé sur le cloud uniquement | Non | Non | Non |
| Caractéristique spécifiques | InfluxData est le créateur de influxDB, spécialement conçu pour gérer les volumes massifs de données horodatées produites par les appareils et capteurs IoT, Les applications en temps réel pour l’IoT, l’analyse et les services cloud natifs. | Base de données en colonnes intégrée et système de programmation pour le streaming, les données en temps réel et historiques.  Système d’exploitation et matériel standard. | Toutes les données sont des données chronologiques.  Elle est la seule base de données relationnelle conçue avec les performances, l'évolutivité et la fiabilité nécessaires pour stocker ces flux de données incessants. |
| Avantage concurrentiels | **Temps de valeur**  Disponible dans tous les langages et frameworks populaires, permettant aux développeurs d’être opérationnels en quelques minutes, et non en quelques jours ou semaines. Avec des démarrages rapides et une documentation facile à utiliser, les développeurs peuvent créer des applications ouvertes, extensibles et faciles à déployer.  **Action réelle en temps réel**  InfluxDB offre une visibilité avec des analyses en temps réel afin que les développeurs puissent agir rapidement sur leurs données, par exemple en identifiant des modèles, en prédisant les résultats futurs et en transformant les information en action. | Offre une évolution transparente.  Fonctionne sur des plateformes de serveur standard de l'industrie.  S’adapte facilement au API pour la connectivité aux principaux systèmes et modules externes.  Nécessite moins de développeurs | Conçu pour mettre à l'échelle et gérer les charges de travail de données de séries chronologiques et est intentionnellement conçu comme une extension PostgesSQL |

**Conclusion :** pour notre cas on utilisera Kafka pour recevoir les données en temps réel ou la base de données influxDB qui est orientée pour le traitement des données temporaire.

**Architecture Kafka**

Kafka possède un ensemble de composant assurant son fonctionnement en tant que système de messagerie distribué :

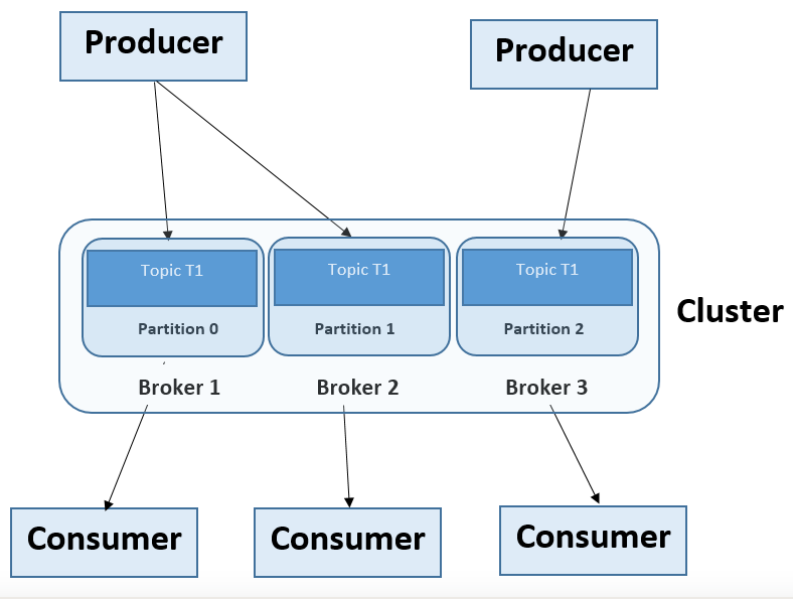


Figure 1 illustration du fonctionnement kafka

En claire :

* Topic : c’est ici où sont stockés ou enregistrés les données dans kafka
* Producteur : c’est le système qui va produire les données dans kafka, publie aussi les messages à un topic
* Broker : est un ensemble de serveur ou noeud formant un cluster kafka, permettant de stocker et gérer les messages publiés
* Consommateurs : peuvent s’abonner à un ou plusieurs topic et consomme les messages publiés à partir du cluster

Dans l’exemple de la figure 1, le topic T1 contenant N message est partitionné à travers 3 brokers du cluster kafka. Les consommateurs consomment toujours les messages à partir d’une partition spécifique, de façon séquentielle. Cela joue un rôle non négligeable dans les bonnes performances des traitements effectués par kafka. Avant toute consommation d’un message, chaque consommateur lance une requête asynchrone de type pull au broker afin que celui-ci lui alloue un tampon d’octets prêts à être consommé.

utiliser aussi influxDB ou KDB comme base de données les plus populaires pour traiter ces série temporelles (données horodaté). Ces bases de données sont conçues pour stocker et analyser des données de capteurs ou des journaux horodatés sur une période donnée. De même des outils de data visualisation comme prometheus ou graphite on vu le jour pour intégrer ces données horodaté au sein de tableau de bord, l’un des principaux objectifs est de détecter une panne sur un objet connecté

**Architecture influxDB**

InfluxDB fournit un agent de serveur piloté par plug-in appelé Telegraf qui collecte et rapporte les métriques de toutes les sources prises en charge et alimente InfluxDB avec ces données ou vice versa.

Telegraf peut être démarré en tant que service sysvinit ou systemd sous linux, ou l'exécuter en tant que commande de terminal.

À partir de la figure ci-dessous on peut voir que telegraf a des plug-ins d’entrée et de sortie, qui doivent être configurés à l’avance avant de le démarrer. Par exemple, il dispose d’un plug-in d’entrée consommateur apache kafka qui permet de lire les messages d’un sujet kafka et d’un plug-in de sortie qui permet d'écrire toutes les données d’entrée dans influxDB.

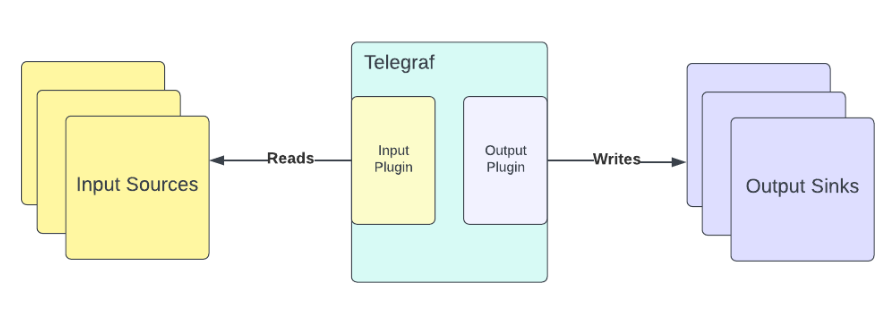


Figure 2 illustration du fonctionnement InfluDB

**Partie II : Etat de l’art sur les méthodes d’apprentissage (statistiques / machine learning) adaptées aux séries temporelles en flux**

**Les principaux axes d’études d'apprentissage autour des séries temporelles qui ont été proposés dans la littérature sont les suivants :**

**Prétraitement des données dynamiques**

1. **Données manquantes**

Pour traiter les données manquantes contrairement aux données statistiques on peut utiliser la **moyenne glissante**, la **moyenne mobile ou la moyenne de fenêtre**, qui consiste à prendre la moyenne des valeurs relatives à une fenêtre prédéfinie, par exemple une fenêtre de 7 jours ou d'un mois. Nous pouvons donc utiliser cette moyenne mobile pour combler les lacunes de nos séries temporelles. Une autre manière de les traiter consiste à faire **l’interpolation** qui est aussi la méthode la plus utilisée pour déterminer les parties manquantes des données de séries temporelles. Cette dernière utilise la valeur présente avant et après le point manquant pour calculer la donnée manquante. Par exemple, les **interpolations linéaires** fonctionnent en calculant une ligne droite entre les deux points, en en faisant la moyenne et en obtenant la donnée manquante. Il existe de nombreux types d'interpolations, comme **les interpolations linéaires, de Spline et de Stineman.** Dans certains cas où de nombreuses données sont manquantes, d’autres approches aussi basées sur des **modèles autorégressifs (AR)**, peuvent être utilisées.

On peut aussi utiliser d’autres méthodes à base d’heuristiques et parmi ces méthodes on peut citer :

* Moyenne : Cette méthode consiste à remplacer les données manquantes d’une série temporelle par la moyenne des valeurs observées sur la séquence, par une moyenne sur les valeurs qui précédent la valeur manquante ou par une moyenne des valeurs observées dans une certaine fenêtre autour d’une valeur manquante.
* Dernière observation : L’idée est de remplacer une valeur manquante par la dernière valeur observée sur la séquence. Dans le cas où les valeurs manquantes sont peu nombreuses, cette méthode peut donner des résultats surprenamment bons.
* Knn-substitution : utilisée dans la complétion de données manquantes dans le cas de séries temporelles multivariées, cette méthode consiste à rechercher la série la plus similaire (au sens des plus proches voisins) à une série donnée et de remplacer les valeurs manquantes de cette dernière par celles du plus proche voisin.

1. **Caractéristiques principale des serie temporaire**

Quant aux données de séries temporelles, elles ont obligatoirement des composantes de séries temporelles. L’extraction de ces caractéristiques peut s’effectuer via la décomposition STL (Seasonal and Trend decomposition using Loess) et certaines d’entre elles sont définies ci-dessous :

* **Saisonnalité :** La saisonnalité fait référence à une propriété des séries temporelles qui affiche des modèles périodiques se répétant à une fréquence constante.
* **Tendance :** Les données de séries chronologiques présentent une tendance lorsque leur valeur varie dans le temps. Une valeur croissante indique une tendance positive et une valeur décroissante, une tendance négative.
* **Reste :** Après avoir extrait la tendance et la saisonnalité des données, ce qui reste est ce que nous appelons le reste (erreur) ou le résidu. Cela permet de détecter les anomalies dans les séries chronologiques.
* **Cycle :** Les données de séries temporelles sont dites cycliques lorsqu'il existe des tendances sans répétitions fixes ou saisonnalité.
* **Stationnarité :** Les données de séries temporelles sont stationnaires lorsque leurs caractéristiques statistiques ne changent pas dans le temps, c'est-à-dire une moyenne et un écart-type constants.

Une fois extraites, ces composantes constituent la base pour l'analyse de l’évolution de la série temporelle afin de comprendre son comportement et de pouvoir choisir et adapter un modèle de série temporelle approprié.

Les données de séries temporelles peuvent avoir d'autres attributs que les caractéristiques temporelles. Si ces attributs sont temporels, la série temporelle résultante sera multivariée et si elle est statique, elle sera univariée avec des caractéristiques statiques. Les caractéristiques non temporelles peuvent utiliser des méthodes issues des techniques statiques de manière à ne pas nuire à l'intégrité des données.

1. **Approches algorithmiques pour la prévision des séries temporelles**

**3.1 Préparation des données dynamiques**

Il y a deux choses à établir avant de concevoir un modèle de prévision :

* Les informations disponibles au moment où la prévision est faite (caractéristiques) ;
* La période pendant laquelle l’utilisateur a besoin de valeurs prévisionnelles (objectif).

L'origine de la prévision est le moment auquel une prévision est effectuée. En pratique, l'origine de la prévision est le dernier moment pour lequel des données d'entraînement sont disponibles pour la période à prévoir. Tout ce qui va jusqu'à l'origine peut être utilisé pour créer des caractéristiques. L'horizon de prévision, qui décrit l’objectif, est la période pour laquelle une prévision va être effectuée.

**3.2 Modèles de prédiction de séries temporelles multivariées**

Les modèles de prédiction de séries temporelles ont tous été étendus au cas de la prédiction de séries temporelles multivariées, c’est-à-dire au cas où plusieurs valeurs évoluent simultanément. Parmi les modèles existants, le modèle VAR (vectorial autoregressiv) est l’un des plus répandus et les plus utilisés. Il s’agit d’une extension naturelle au modèle autorégressif univarié AR. Il part du principe que la valeur d’une série temporelle dépend des valeurs précédentes de la série pendant une fenêtre temporelle donnée, mais également des valeurs des autres séries considérées pendant ce même intervalle de temps. Ce modèle du fait de sa simplicité a été extrêmement utilisé et il s’est montré particulièrement efficace dans certaines tâches prédictives. Cependant les dépendances linéaires entre séries restent une hypothèse forte et limitant le pouvoir d’expression de ces modèles; de plus, la forte complexité en nombre de paramètres

**Autres modèles**

De nombreux autres modèles ont été proposés pour la prédiction de séries temporelles et il ne serait pas possible de faire ici un état de l’art exhaustif de ceux-ci. Plus précisément, un grand nombre de modèles existe dans des cadres restreints : on peut par exemple citer le modèle de Croston particulièrement adapté dans le cas où les séries temporelles valent souvent zéro. La méthode de lissage exponentielle est également souvent utilisée en pratique mais elle diffère peu des méthodes autorégressives.

Nous nous limitons dans ce travail aux modèles les plus familièrement utilisés dans la communauté de l’apprentissage automatique pour les prévisions de séries chronologiques comme :

* ARIMA : Cela signifie Autoregressive-Integrated-Moving Average. Il utilise la combinaison de moyennes autorégressives et mobiles pour prédire les valeurs futures.
* EWMA/Lissage exponentiel : la moyenne mobile pondérée de manière exponentielle ou le lissage exponentiel sert de mise à niveau des moyennes mobiles. Cela fonctionne en réduisant l'effet de décalage montré par les moyennes mobiles en mettant plus de poids sur les valeurs qui se sont produites plus récemment.
* Modèles de régression dynamique : cet algorithme prend également en compte d'autres informations diverses telles que les jours fériés, les modifications de la législation, etc.
* Prophet : Prophet , qui a été publié par l'équipe Core Data Science de Facebook, est une bibliothèque open source développée par Facebook et conçue pour la prévision automatique de données de séries chronologiques univariées.
* LSTM : Long Short-Term Memory (LSTM) est un type de réseau neuronal récurrent qui peut apprendre la dépendance d'ordre entre les éléments d'une séquence. Il est souvent utilisé pour résoudre des problèmes de prévision de séries chronologiques.

Sources utilisées :

<https://mcherif.wordpress.com/2015/08/07/big-data-kafka-concepts-generaux/>

<https://www.splunk.com/fr_fr/data-insider/what-is-stream-processing.html#what-is-stream-processing>

<https://www.journaldunet.fr/web-tech/guide-de-l-entreprise-digitale/1443846-influxdb-une-base-de-donnees-time-series-open-source-sur-optimisee/>

<https://www.ionos.fr/digitalguide/hebergement/aspects-techniques/quest-ce-quinfluxdb/>

<https://www.influxdata.com/blog/getting-started-apache-kafka-influxdb/>

<https://www.influxdata.com/blog/getting-started-apache-kafka-influxdb/>

[Tout savoir sur l’état de l’art (scribbr.fr)](https://www.scribbr.fr/methodologie/etat-de-lart/)

<https://db-engines.com/en/system/InfluxDB%3BKdb%2B%3BTimescaleDB>

<https://www.getapp.fr/alternatives/2049206/instaclustr-apache-kafka>

<https://solutionsreview.com/data-integration/the-best-open-source-data-streaming-software-and-tools/>

Etat de l’art 2

<https://datascientest.com/le-modele-var>

<https://eduscol.education.fr/sti/sites/eduscol.education.fr.sti/files/ressources/pedagogiques/14762/14762-series-temporelles-et-reseaux-de-neurones-recurrents-ensps.pdf>