Bonjour tout le monde :

Je vous présente mes résultats finaux de mon stage pendant 5 mois au soins de chez Ifp énergie. Je vais vous introduire d’abord le contexte et l’objectifs de ce projet, et ensuite je vous montre les informations des bases de données et les jeux de données que j’ai testé avec. Et puis je vais vous expliquer aussi l’architecture global du projet la mécanisme appliqué à chaque bloc. Pour conclure, je vais donner les données de la performance et les phénomènes intéressantes.

Objectifs

En parlant à la gestion des réseaux d’énergie, normalement il concerne plusieurs difficultés techniques : le stockage des gros volume de données qui sont envoyées par les capteurs, le pré-traitement des données avant stockage, le retard pour rechercher les données désirées dans la base, l’outil de la visualisation pour vérifier les données, etc. Donc j’ai effectué des tests pour trouver un meilleur choix pour les cas d’utilisations et comment accélérer le traitement des données.

Voici les informations de test que j’ai fait :

J’ai testé 4 types de base de données qui sont populaires pour stocker séries temporelles avec 3 jeux de données.

J’ai aussi simulé les données en streaming pour les tests de l’ingestion et la validation des données avant stockage.

Après j’ai passé les requêtes des opérations plus utilisées pour l'exploration et la visualisation des données dans chaque base de données et évaluer la performance des requêtes.

Les dernières parties donnent les évaluations pour préparer les données afin d’appliquer les différents modules d’analyse ou machine learning. Pour accélérer la vitesse de cette étape, je vais aussi avancer sur spark et dask et comparer ses performances.

Présentation des bases de données

Dans cette partie là je vais vous montrer les caractéristiques des bases de données à étudier et les introduire rapidement.

* MongoDB est la base NonSQL plus utilisée avec beaucoup outils bien dévelopés comme mongo-compass. Il est base de données orientée document, sans la nécessité de schéma prédéfinit, la vitesse de requête est très haut et le stockage des données est libre. En comparant les scripts pour insérer les données dans ces 4 bases, la préparation des données avant l’insertion pour MongoDB est la plus simple. Mais ça peut être aussi un désavantage : sa flexibilité conduit à une haute consommations de mémoire. Et en plus, lorsqu’il n’est pas base de données spécialement orientée les séries temporelles, il faut la gestion manuel du temps.
* KairosDB est base de données orientée séries temporelle qui support le stockage en utilisant cassandra, Hbase et H2. Cet étude utilise H2 comme le stockage sous-jacent. Les données dans KairosDB sont stockées au format de métrique donc on peut ajouter les tags pour faciliter les requêtes. Il a aussi des règles pour la gestion automatique des timestamps, mais à cause de ce système, on ne peut pas avoir plusieurs champs avec même tag et timestamp dans un point, donc il faut bien réfléchir la structure de données.
* InfluxDB est une base open source orientée séries temporelles donc il est aussi capable de gérer du temps. Il a une politique spéciale qui peut définir le temps de rétention des métrique. Ça peut aider d’économiser l’espace de disque si la politique est bien configurée. Mais quand la date est expirée, tout le fragment de données sera perdu. A cause de ligne protocole qui est très strict, il est aussi difficile à écrire les requêtes et préparer les données pour l’insertion.
* Warp10 n’est pas encore très populaire, il comprend une base de données Geo Time Series et un moteur d'analyse complémentaire. Il est intéressant pour le cas de test parce que normalement les données envoyées par capteur contiennent les informations de temps et géographie. Le moteur d’analyse dans Warp10 possède un langage très efficace qui d’appelle WarpScript pour manipuler les données, il propose plus de 1000 fonctions à choisir. Mais pour évaluer warp10 j’ai rencontré beaucoup de difficultés parce qu’il est difficile à installer et étudier WarpScript.

Présentation des jeux de données

On a fini l’introduction pour les bases de données à tester, maintenant on va passer à la partie des jeux de données. Dans ce partie là je vais vous introduire les structures des jeux de données.

* Le jeu de données smartGrid s’agit les informations d’énergie transmises par des capteurs du bâtiment Cryolite. La structure des données est simple, il contient que 4 colonnes pour le temps, les tags, la valeur et la qualité. Pour les tests de performance avec grande volume de données, j’ai pris le fichier correspondant à un mois de collecte qui a plus de 760,000 points.
* Les données éolienne sont collectées par les capteurs installés sur les fuites des éoliennes, ils sont consistés par 51 colonnes. J’utilise ce jeu de données afin de tester la performance de traitement pour les données complexes.
* Les données de pollution sont collectées par les capteurs installés dans les véhicules pour détecter les éléments de pollution sur des profils de trajet. La source de données est un fichier json binaire exporté par MongoDB. Vu que sa structure est proche que les données éolienne, et sa volume est inférieure aux données smartGrid, à cause de la limite du temps je vous montre que les résultats pour smartGrid et éolienne.

Présentation de l’architecture du framework

Après je vais vous présenter l’architecture de ce projet. Selon la figure, le framework du projet est composé par 3 blocs, les bases de données et les modules d’analyse et machine learning. Dans un premier temps, c’est le bloc Kafka pour l’ingestion des flux de données envoyés par les capteurs, et puis les données vont passer le bloc des traitements avant stockage pour les corriger et les préparer avant l’alimentation des base de données. Ensuite on va tester la performance des insertions dans la base et envoyer les requêtes pour l’exploration et la visualisation des données. Bien sur dans cette étape là on va aussi tester la performance pour récupérer les données. Et en plus, on peut encore appliquer les algorithmes d’analyse et les modules de machine learning aux flux de données pour évaluer la traitement des données. En bref, l’objectif de ce framework est d’accélérer la vitesse de traitement des données en recherchant un meilleur résultat de test.

Collecte & préparation des données

On a finalisé l’introduction de l’architecture global du projet et ensuite je vais vous présenter les détails de chaque bloc.

Dans un premier temps on doit récupérer les données avec Appache Kafka. Kafka est un plateforme de streaming distribuée, il est capable de publier les messages, les stocker d’une manière durable et les traiter. J’utilise les Producers pour simuler les capteurs, ce Producer continue d’envoyer les flux de données qui vient des fichier csv.

Voici un exemple de Producer : on va d’abord lire les fichiers csv et combiner les colonnes dans une chaine charactère avant les envoyer. C’est le cas de données de smartGrid et voici les messages finaux envoyer par Kafka Producer. … . Mais il existe un souci parce que les messages qu’on a envoyés ont perdu les entêtes de données, c-à-d on ne sait pas les structures originales des données.

Ensuite c’est un exemple de Kafka Consumer. On connecte aux bases de données à tester en mode local et commencer à recevoir les messages. Dans cette fonction on peut définir combien de ligne de données on va récupérer à la fois et la collection de base de données à alimenter.

Les schémas spéciaux des données sont enregistrés dans la base de MongoDB grâce à sa flexibilité. Il est composé par les noms de colonnes, les types de données et les règles de validation pour la validation et la reconstruction des données.

Ensuite on va convertir la structure de données pour évaluer l’insertion dans chaque base. J’ai mis des exemples d’insertion dans le notebook et normalement c’est fait avec les scripts python et Jaeger. Et vous voyez les résultats au-dessous des cellules.

Exploration et visualisation

Dans la partie d’exploration et visualisation des données, l’intérêt est le choix d’outil afin d’avoir une solution globale pour tous les bases de données.

Il y a des outils très populaire est bien développé comme Graphana, malheureusement comme j’ai présenté des écosystèmes des bases, il ne support pas toutes les 4 bases.

Il y a aussi les outils spécifiques à chaque base, soit on vérifie les logs, soit on installe les logiciel ou utilise les web UI, mais il est difficile d’avoir une comparaison pour toutes les bases et aussi, ce n’est pas pratique d’installer tellement d’extensions sur le cluster de l’IFP.

Donc finalement j’ai choisi visdom comme l’outil de visualisation, il est capable d’envoyer les figures produit par code vers les requêtes http et mettre les figures dans web UI. Aussi, visdom support python et son packages est léger qui est facile à installer.

Voici exemple de la visualisation. On fait exactement les mêmes choses pour récupérer toutes les données ou choisir les données en fonction de tag (cellule17-19). Ici j’ai mis un exemple pour nettoyer les données, normalement c’est fait avant le stockage. Ensuite on va convertir les données au DataFrame pour observer les courbes (celulle19). Ce sont les données de smartGrid dans un mois. …. Visdom….

Analyse et Machine Learning

L’objectif de ce bloc est de traiter et convertir les données afin d’appliquer les modules d’analyse et de machine learning aux données de test.

Voici un exemple simple pour illustrer comment préparer les données dans les bases de données et l’algorithmes d’analyse.

Les étapes ici sont les mêmes choses pour récupérer et convertir les données (cellule 28). Ici j’ai mis un exemple d’algorithme est pour calculer la valeur moyenne de courbes pour les données smartGrid en fonction de jours. Voici la courbe pour un seul tag dans une journée entre 8h du matin jusqu’à 17h. Et on peut aussi visualiser toutes les courbes pour le même tag en fonction de jour. C’est un exemple pour montrer que c’est pratique d’appliquer les modules d’analyse si les données sont déjà bien préparées.

Et voici un autre exemple pour le cas d’éolienne. Ce sont les courbes de la vitesse du vent (cellule 267). Dans cette figure là on peut observer qu’il y a un phénomène avec CavityTemp. Donc pour comprendre ce phénomène on va analyser la vitesse du vent. Pour obtenir un signal sinusoïde, on va d’abord établir une régression linéaire et supprimer la partie linéaire dans les données à étudier. Voilà le signal après traitement (cellule 281).

Ensuite on va calculer le transfert Fourier et la densité spectrale de puissance en DB, c’est pour filtrer la partie plus intéressant afin d’observer les énergies du signal. Et voilà un exemple simple pour la filtrage du brut, la courbe orange est le signal original et la courbe bleu est le signal débruité.

Ce partie là montre la possibilité d’appliquer facilement les modules d’analyse et ml à n’emport quelle base après le traitement des données.

Performance

J’utilise Jaeger comme outil de visualisation pour tester la performance des opérations.

Ici je vous montre aussi quelques tableaux de performance. Insert s’agit le temps d’insertion, …. BucketSize s’agit que l’on va insérer combien de ligne à la fois pour tester.

Après les tests des bases de données, on a aussi effectué des expérimentations pour accélérer le traitement des données avec dash et spark. Ici je vous montre directement les résultats obtenus sur cluster LAB. Vous voyez que le temps de traitement des données a réduit en utilisant spark et Dask.

Conclusion

Dans ce projet j’ai réalisé la simulation des capteurs dans les réseaux d’énergie avec kafka, j’ai évalué les performance de 4 bases de données en fonction de source de données et la taille du segmentation des données. J’ai aussi testé la performance pour la validation, la nettoyage et le traitement de données.

Dans le mois qui reste, je vais encore avancer sur quelques taches : faut je finaliser les tests sur cluster pour comparer avec les résultats sur pc fix.

Je veux aussi faire les tests avec plus de source de données si c’est possible.

Et je vais passer les requêtes plus compliqués pour les selects.

On peut aussi comparer la performance de fonction statistique avec pandas, spark, dash et les fonctions directement fournit par les bases données comme WarpScript.

Aussi, on peut encore tester les topic en parallèle pour voir si c’est possible d’accélérer l’alimentation des bases de données.