Rapport TP4 : CouchDB - MapReduce

ABDOULAHI SALAHANNE Ahmed 12306430

Ing3 Informatique

Enseignant : Mr. Y.Samir

Table des matières

Introduction . . . 2
Installation de CouchDB via Docker . . . 2
- Prérequis . . . 2
- Création du Dockerfile . . . 2
- Construction de l'image Docker . . . 3
- Exécution du conteneur CouchDB . . . 3
- Vérification de l'installation . . . 3
- Conclusion de l'installation . . . 3

Fonctionnement de MapReduce . . . 4
Exemple 1 : Calcul du nombre de films par année . . . 4
- Contexte . . . 4
- Fonction Map : Explication et Code . . . 4
- Fonction Reduce : Explication et Code . . . 4

Exemple 2 : Calcul du nombre de films par réalisateur . . . 5
- Fonction Map : Explication et Code . . . 5
- Fonction Reduce : Explication et Code . . . 5

Exercice 1 : Représentation d'une matrice et calculs avec MapReduce . . . 6
- Modèle pour représenter la matrice M . . . 6
- Calcul de la norme d'un vecteur . . . 6
- Calcul du produit de la matrice M avec un vecteur W . . . 7
- Résumé . . . 8

Conclusion . . . 9

Introduction

Le MapReduce est un concept incontournable dans le domaine du Big Data, utilisé pour effectuer des calculs parallèles sur des volumes importants de données. CouchDB, base de données NoSQL open-source, propose un moteur MapReduce simple à utiliser, en particulier grâce à son interface RESTful. L'une des particularités de CouchDB est qu'il permet de sauvegarder les résultats intermédiaires de la fonction Map sans avoir besoin de définir la fonction Reduce, ce qui simplifie grandement l'analyse des données.

Ce rapport explore, à travers plusieurs exemples, la définition et l'utilisation des fonctions Map et Reduce dans CouchDB.

Installation de CouchDB via Docker

Cette section détaille l'installation de CouchDB en utilisant Docker, en expliquant chaque étape depuis la création du fichier Dockerfile jusqu'à l'exécution du conteneur.

Prérequis

Avant de commencer, assurez-vous que les outils suivants sont installés sur votre machine :
— Docker (dernière version recommandée).
— Un éditeur de texte comme `nano` ou `vim`.
— Une connexion internet pour télécharger l'image Docker de CouchDB.

Création du Dockerfile

Pour installer CouchDB, nous allons utiliser un fichier **Dockerfile** qui permet de construire une image Docker personnalisée.

1. **Créer le fichier Dockerfile** : Ouvrez un terminal et accédez au répertoire de travail.

```
(base) ahmed@ahmed-HP-ProBook-440-14-inch-G9-Notebook-PC:~/NOSQL$ nano
    Dockerfile
```

Code 1 – Création du fichier Dockerfile

2. **Contenu du Dockerfile** : Ajoutez les lignes suivantes dans le fichier.

```
#  tape  1 : Utiliser l'image officielle CouchDB
FROM apache/couchdb:3

#  tape  2 : Configuration des variables d'environnement
ENV COUCHDB_USER=admin
ENV COUCHDB_PASSWORD=password

#  tape  3 : D finir le volume pour la persistance des donn es
VOLUME ["/opt/couchdb/data"]

#  tape  4 : Exposer le port CouchDB
EXPOSE 5984

#  tape  5 : Commande pour d marrer CouchDB
CMD ["couchdb"]
```

Code 2 – Contenu du Dockerfile

Explications :
— `FROM` : Utilise l'image CouchDB officielle version 3.
— `ENV` : Définit les variables d'environnement pour l'utilisateur et le mot de passe par défaut.
— `VOLUME` : Assure la persistance des données dans le répertoire spécifié.
— `EXPOSE` : Expose le port 5984 pour accéder à CouchDB.
— `CMD` : Commande par défaut pour exécuter CouchDB.

Construction de l'image Docker

Après avoir créé le fichier Dockerfile, construisez l'image Docker en exécutant la commande suivante :

```
(base) ahmed@ahmed-HP-ProBook-440-14-inch-G9-Notebook-PC:~/NOSQL$ docker build -t my-couchdb .
```

Code 3 – Construction de l'image Docker

Explications :

- `docker build` : Construit une image Docker.
- `-t my-couchdb` : Attribue un nom à l'image (`my-couchdb`).
- `.` : Indique que le fichier Dockerfile se trouve dans le répertoire courant.

Une fois la construction terminée, le message suivant s'affiche :

```
Successfully built <IMAGE_ID>
Successfully tagged my-couchdb:latest
```

Exécution du conteneur CouchDB

Une fois l'image créée, lancez un conteneur CouchDB en utilisant la commande suivante :

```
(base) ahmed@ahmed-HP-ProBook-440-14-inch-G9-Notebook-PC:~/NOSQL$ docker run -d \
--name couchdb-container -p 5984:5984 -v couchdb-data:/opt/couchdb/data my-couchdb
```

Code 4 – Lancement du conteneur CouchDB

Explications :

- `docker run -d` : Exécute le conteneur en arrière-plan (mode détaché).
- `--name couchdb-container` : Attribue un nom au conteneur.
- `-p 5984:5984` : Lie le port 5984 de l'hôte au port 5984 du conteneur.
- `-v couchdb-data:/opt/couchdb/data` : Persiste les données dans un volume nommé `couchdb-data`.
- `my-couchdb` : Nom de l'image Docker construite précédemment.

Vérification de l'installation

Pour vérifier que CouchDB est correctement installé et en cours d'exécution :

1. **Liste des conteneurs en cours d'exécution** :

   ```
   docker ps
   ```

   Code 5 – Lister les conteneurs actifs

2. **Accéder à CouchDB via un navigateur** : Ouvrez un navigateur web et accédez à l'URL suivante :

   `http://localhost:5984`

   Vous devriez voir la page d'accueil de CouchDB avec le statut `Welcome to CouchDB`.

Conclusion de l'installation

L'installation de CouchDB via Docker est maintenant terminée. Grâce à Docker, la configuration est simplifiée, avec persistance des données et possibilité d'accéder à CouchDB via son interface web ou son API REST.

Fonctionnement de MapReduce

Le processus MapReduce se divise en deux étapes principales :

— **Fonction Map** :
La fonction Map est appliquée à chaque document de la base de données indépendamment. Elle extrait certaines informations pertinentes des documents et les émet sous forme de paires clé-valeur. Le choix de la clé est crucial car elle permet de regrouper les résultats par catégorie.

— **Fonction Reduce** :
La fonction Reduce est utilisée pour agréger les résultats intermédiaires. Elle prend en entrée les clés produites par la fonction Map et applique une opération d'agrégation sur les valeurs associées.

Exemple 1 : Calcul du nombre de films par année

Contexte

On veut calculer combien de films sont sortis chaque année. Pour ce faire :

— La fonction Map extrait l'année et le titre des films.

— La fonction Reduce compte le nombre de films par année.

Fonction Map : Explication et Code

La fonction Map parcourt chaque document et extrait deux informations :

— La clé : l'année de sortie du film (`doc.year`).

— La valeur : le titre du film (`doc.title`).

```
function (doc) {
    emit(doc.year, doc.title); //  met  l'ann e  comme  c l   et le titre comme valeur.
}
```

Code 6 – Code de la fonction Map

Explications :

— `doc` représente un document individuel dans la base CouchDB.

— `emit(clé, valeur)` est une instruction spéciale qui envoie les résultats intermédiaires.

Exemple d'entrée (document JSON) :

```
{
    "year": 1940,
    "title": "Le Dictateur"
}
```

Code 7 – Document JSON d'entrée

Sortie intermédiaire (après exécution de Map) :

Clé (Année)	Valeur (Titre)
1940	Le Dictateur
1940	Rebecca
1921	Le Kid

Fonction Reduce : Explication et Code

La fonction Reduce regroupe les résultats intermédiaires par clé (ici, l'année) et compte le nombre d'occurrences.

```
function (keys, values) {
    return values.length; // Retourne le nombre total de titres pour chaque ann e.
}
```

Code 8 – Code de la fonction Reduce

Explications :

— `keys` représente les clés produites par la fonction Map.

— `values` est un tableau contenant les valeurs associées aux clés.
— `values.length` retourne le nombre total d'éléments dans ce tableau.

Résultat final :

Année	Nombre de Films
1921	1
1940	2

Exemple 2 : Calcul du nombre de films par réalisateur

Contexte

On veut déterminer combien de films chaque réalisateur a réalisés.
La clé est le nom du réalisateur (`doc.director.last_name`). La valeur est le titre du film.

Fonction Map : Explication et Code

```
function (doc) {
    emit(doc.director.last_name, doc.title); //  met  le nom du r alisateur et le
        titre du film.
}
```

Code 9 – Code de la fonction Map

Sortie intermédiaire :

Clé (Réalisateur)	Valeur (Titre)
Chaplin	Le Kid
Chaplin	Les Temps Modernes

Fonction Reduce : Explication et Code

```
function (keys, values) {
    return values.length; // Retourne le nombre de films pour chaque r alisateur.
}
```

Code 10 – Code de la fonction Reduce

Résultat final :

Réalisateur	Nombre de Films
Chaplin	2
Hitchcock	3

Exercice 1 : Représentation d'une matrice et calculs avec MapReduce

Cet exercice s'intéresse à la représentation et au traitement d'une matrice M de dimension $N \times N$, représentant des liens entre un grand nombre de pages web, en utilisant le concept MapReduce. Chaque lien est étiqueté par un poids indiquant son importance.

Modèle pour représenter la matrice M

Pour représenter la matrice M, nous adoptons un modèle inspiré du PageRank de Google. Chaque ligne de la matrice correspond à une page, identifiée par un identifiant unique, avec les liens pondérés vers d'autres pages.

Structure du document JSON :

```
{
    "page_id": "P_i",
    "links": [
        {"target": "P_j1", "weight": 0.5},
        {"target": "P_j2", "weight": 0.3},
        {"target": "P_j3", "weight": 0.2}
    ]
}
```

Explications :

- **page_id** : représente l'identifiant de la page P_i.
- **links** : une liste d'objets contenant les cibles (**target**) et le poids (**weight**) des liens vers d'autres pages.

Ce modèle permet une organisation claire des documents pour appliquer efficacement MapReduce.

Calcul de la norme d'un vecteur

La ligne i de la matrice peut être vue comme un vecteur $V(v_1, v_2, ..., v_N)$. La norme d'un vecteur est donnée par :

$$||V|| = \sqrt{v_1^2 + v_2^2 + ... + v_N^2}$$

Traitement avec MapReduce :

1. **Fonction Map :** Calculer le carré de chaque élément du vecteur et émettre la somme des carrés.
2. **Fonction Reduce :** Additionner les résultats intermédiaires et appliquer la racine carrée pour obtenir la norme.

Code Map :

```
function (doc) {
    var sum = 0;
    for (var i = 0; i < doc.links.length; i++) {
        var weight = doc.links[i].weight;
        sum += weight * weight; // v_j^2
    }
    emit(doc.page_id, sum);
}
```

Code 11 – Fonction Map pour calculer les carrés

Code Reduce :

```
function (keys, values) {
    var total = 0;
    for (var i = 0; i < values.length; i++) {
        total += values[i];
    }
    return Math.sqrt(total);
}
```

Code 12 – Fonction Reduce pour calculer la norme

Explications :

— **Map :** Calcule le carré des poids pour chaque ligne.
— **Reduce :** Additionne les carrés et calcule la racine carrée pour obtenir la norme.

Calcul du produit de la matrice M avec un vecteur W

Le produit de la matrice M avec un vecteur $W(w_1, w_2, ..., w_N)$ est donné par :

$$\phi_i = \sum_{j=1}^{N} M_{ij} W_j$$

Hypothèse : Le vecteur W est accessible en mémoire RAM comme une variable statique pour toutes les fonctions Map et Reduce.

Traitement avec MapReduce :

1. **Fonction Map :** Multiplier chaque poids M_{ij} par W_j et émettre la somme pour chaque ligne.
2. **Fonction Reduce :** Additionner les résultats pour obtenir ϕ_i pour chaque page.

Code Map :

```
var W = [0.5, 0.8, 0.2, ...]; // Vecteur W

function (doc) {
    var result = 0;
    for (var i = 0; i < doc.links.length; i++) {
        var target = parseInt(doc.links[i].target.replace("P_", ""));
        result += doc.links[i].weight * W[target - 1];
    }
    emit(doc.page_id, result);
}
```

Code 13 – Fonction Map pour produit matriciel

Code Reduce :

```
function (keys, values) {
    var total = 0;
    for (var i = 0; i < values.length; i++) {
        total += values[i];
    }
    return total;
}
```

Code 14 – Fonction Reduce pour additionner les produits

Explications :

— **Map :** Multiplie les poids M_{ij} par les composantes W_j du vecteur W.
— **Reduce :** Additionne les résultats pour chaque ligne de la matrice afin d'obtenir ϕ_i.
—

Résumé

Dans cet exercice, nous avons :

— Proposé un modèle pour représenter une matrice sous forme de documents JSON.

— Spécifié le traitement MapReduce pour calculer la norme des vecteurs d'une ligne.

— Implémenté le produit matriciel entre la matrice M et un vecteur W en utilisant MapReduce.

Ces traitements montrent la capacité de MapReduce à paralléliser des calculs sur de grandes matrices.

Conclusion

Ce rapport a montré comment utiliser les fonctions **Map** et **Reduce** dans CouchDB pour analyser des documents JSON :

— Extraction des données via **Map**.

— Agrégation des résultats intermédiaires via **Reduce**.

CouchDB facilite ces opérations grâce à son architecture simple et la possibilité de visualiser les résultats intermédiaires.