Shavadoop Documentation

Jonathan OHAYON

Introduction

Shavadoop est un projet INF727 de l'école Telecom ParisTech, qui vise à créer un programme distribuée de wordcount à travers le protocole SSH en Java. SSH + Java + Hadoop = Shavadoop.

Table des matières

Description des algorithms

1 Récupération des machines

La méthode getMachine va à l'aide d'une liste d'adresse de machine récupérer une liste de machines vivantes. Cette liste sera utilisée pour attribuer les roles de mapper, shuffler et reducer dans la suite du Shavadoop.

2 Split

Le split est éffectué avec le package FileUtils, qui génère une liste contenant toutes les lignes du fichier. Ensuite, le programme réécrit des fichiers S_i dans le dossier "/Split/" pour donner au Mapper.

3 Distribution des taches

Pour distribuer à travers le réseau, le Master crée une HashMap de commande pour lancer un processus lourd sur les machines distantes. Puis il va utiliser la classe "shepherd" qui se charge de communiquer les processus lourds en utilisant le protocole ssh à l'aide de Jcabi. Finalement, la classe shepherd lance les Threads et se charge de les joindre pour attendre la fin de l'execution des threads.

4 Mapper

Un Mapper va lire le split qui lui a été attribué et va splitter ligne par ligne pour trouver les mots et créer les "unsortedmap". Il va donc créer des ArrayList LineInput_j différentes pour chaque reducer et écrire dans les fichiers de la forme UM_i_R_j. Étape par étape, le mapper réalise :

1. Lecture du fichier S_i qui lui correspond à l'aide de la fonction FileUtils.readlines qui effectue donc un split sur les lignes.
2. Par ligne, le mapper split avec les espaces pour trouver les mots.

3. Pour chaque mot, le mapper stock un string contenant le mot, un espace et "1". le string est stocké dans l'ArrayList correspondant au résultat du hashcode du mot.
4. De plus le mapper crée une liste des mots qu'elle traite.
5. Finalement le mapper écrit tout dans des fichiers à l'aide la fonction StringUtils qui se charge de transformer l'ArrayList de String en un seul String.

5 Shuffler

Un Shuffler va lire tous les fichiers UM_i_R_j qui lui correspond pour créer les fichiers SM_j pour les reducers. Étape par étape, le shuffler réalise :

1. Lecture des fichiers UM_i_R_j stocké dans une HashMap, les clés étant les mots et les valeurs sont la concaténation des 1 séparés par des virgules.
2. Finalement réecriture de la Hashmap dans un fichier SM_j.

6 Reducer

Finalement le Reducer lui recupère les sortedmaps SM_j et cré les fichier Red_j, qui est une partie du résultat final. Étape par étape, le shuffler réalise :

1. Lecture du fichiers SM_j stocké dans une ArrayList de Chaine de Caractères contenant le mot et la somme des valeurs contenu dans le SM_j.
2. Finalement écriture de l'ArrayList dans un fichier Red_j.

Limitation de Shavadoop

1 Splitting

Les limitations se situent au niveau de l'utilisation du FileUtils.readlines qui ouvre le fichier dans son intégralité. Il y a donc un problème si le fichier ne rentre pas dans la RAM. Une méthode pour éviter ce probème, est d'utiliser les offset pour splitter le fichier au lieu d'utiliser FileUtils.readlines.

2 Communication master-worker

Une autre limitation est le manque de communication entre le master et les workers. Il faudrait rajouter des sockets. Cela permettrait aussi de gérer les potentielles erreurs dans les workers.

3 Système NFS

Une limitation importante de Shavadoop est l'utilisation du système de fichier partagé qui sert a simuler HDFS de HADOOP. De plus le système NFS de l'école est limité a 1.5 Go.

Installation

1. Extraire le fichier zip qui contient le projet java Shavadoop.
2. Créer un dossier avec l'input du Shavadoop et configurer le chemin dans le programme.
3. Créer les jar executable Mapper, Shuffler et Reducer.
4. Deplacer les jar executable dans le dossier contenant l'input.

Configuration

La configuration du programme se fait dans la classe Main du projet Shavadoop. Il y a 4 paramètres à configurer qui sont :

1. filepath correspond au chemin où les Inputs et les Outputs vont être écrites.
2. filename correspond au nom de l'Input qui doit se trouver a la racine du filepath.
3. Number_Worker correspond au nombre de job et de machine pour le Mapper, le Shuffler et le Reducer.
4. liste_ordi correspond à la liste d'ordinateur où les jobs vont s'éffectuer.

l'output est générée comme suit :

- le dossier Splits contient les résultats du split de l'Input.
- le dossier UnsortedMap contient les résultats des mappers.
- le dossier Keys contient les fichiers qui représentent la liste des mots traités par le mapper.
- le dossier SortedMap contient les résultats des shufflers i.e. l'entrée pour les reducers de la forme clé unique et valeurs séparées par une virgule.
- le dossier Reduced contient les résultats des reducers.
- le fichier résultat contient l'output final du Shavadoop.