TME 2 - Hadoop, une plate-forme open-source de MapReduce. Calcul de Π en Map Reduce

Jonathan Lejeune, Julien Sopena

Exercice(s)

Exercice 1 – Calcul de Π en *MapReduce*

Nous allons maintenant calculer la valeur de π grace à une méthode de Monte Carlo. Cette approche probabiliste du calcul de π considère :

- un repère orthonormé $(O, \overrightarrow{i}, \overrightarrow{j})$;
- un cercle de diamètre d centré sur O l'origine du repère ;
- un carré de coté d centré sur O.

L'aire A_{cercle} du cercle peut s'exprimer :
$A_{cercle} = \pi * rayon^2 \Leftrightarrow A_{cercle} = \pi * \frac{d^2}{4} \Leftrightarrow \pi = 4 * \frac{A_{cercle}}{d^2} \Leftrightarrow \pi = 4 * \frac{A_{cercle}}{A_{carre}}$

En discrétisant par un nombre suffisamment grand de points l'aire du carré, on peut obtenir une bonne approximation du rapport des aires et donc de la valeur de π. Ainsi, la méthode consiste à tirer aléatoirement des points se situant dans l'espace $[\frac{-d}{2}, \frac{d}{2}] * [\frac{-d}{2}, \frac{d}{2}]$, puis à calculer la proportion des points appartenant au cercle de diamètre d. La valeur de π s'obtient alors en multipliant par 4 ce ratio. Pour savoir si un point P de coordonnées (x, y) appartient au cercle, il suffit de calculer sa distance par rapport à l'origine du repère $(0, 0)$ en utilisant le théorème de Pythagore. Ainsi $Distance(O, P) = \sqrt{x^2 + y^2}$. Si cette distance est supérieure au rayon du cercle (c'est à dire $\frac{d}{2}$), le point n'est pas dans le cercle.

Application au Hadoop

Chaque map va prendre un nombre fixe de points choisis aléatoirement et équitablement répartis dans le carré et déterminer pour chaque point s'il est dans le cercle ou pas. Les maps transmettent leur résultats aux reduces qui additionneront le nombre de points présents dans le cercle. Le programme appelant le job doit ensuite parcourir l'ensemble des résultats produits par les reduces, les additionner et calculer π selon la formule décrite ci-dessus. Les maps choisissent leur points aléatoirement en initialisant une séquence de Halton. Une séquence de Halton est une séquence de réels compris entre 0 et 1. Le diamètre du cercle ne peut dans ce cas pas dépasser 1. Pour éviter que les maps choisissent tous les mêmes points, la graine qui initialise la séquence de Halton doit être propre à chaque map.

Question 1

Comment peut-on faire pour passer une graine différente à chaque map ?

Question 2

Identifiez les types des clés et valeurs des entrées des maps.

Question 3

Sachant que les maps calculent si un point est dans le cercle ou pas, quelle sera le type de sortie des clés et valeurs des maps ?

Question 4
Sachant que les reduces comptent en fonction de la sortie des maps, le nombre total de points qui sont dans le cercle, quel sera le type de sortie des clés et valeurs des reduces ?

Question 5
Liez chaque type à une classe de l'API Hadoop.

Question 6
Écrivez en JAVA le code d'un map. Pour cette question, comme pour les suivantes, il est fortement déconseillé d'utiliser les classes déprèciées de l'API.

Question 7
Écrivez en JAVA le code d'un reduce.

Question 8
Écrivez un programme JAVA MapReduce qui prendra en paramètre le nombre de maps, le nombre de reduces ainsi que le nombre de points par map. Le programme devra configurer les fichiers d'entrée des maps, soumettre le job, récupérer et fusionner les résultats des reduces, calculer π et afficher sa valeur. Pour cette question, vous devez remplir le fichier MyPiEstimator.java qui vous a été fourni dans les ressources du TP.

Conseil : Désactiver l'option exécution spéculative des maps et des reduces qui est activée par défaut grâce aux lignes de code suivantes :

```
conf.setBoolean("mapred.map.tasks.speculative.execution", false);
conf.setBoolean("mapred.reduce.tasks.speculative.execution", false);
```

Question 9
Le programme vous donne la possibilité de paramètrer le nombre de reduces. Est-ce vraiment nécessaire ? En déduire comment le nombre maximum de reduces nécessaires peut être déterminé par l'utilisateur.

Exercice 2 – Hadoop en total distribution

Question 1
Vous testerez en dernier lieu les différents programmes Hadoop produits jusqu'à présent avec plusieurs machines en configurant le fichier conf/slaves. Arrangez-vous avec d'autres binomes pour utiliser leurs machines. N'oubliez pas alors de modifier les fichiers de configuration de la question 4 de l'exercice 1 :

- À la place de *localhost*, mettre le nom de la machine maître (Namenode + JobTracker) qui est dans votre cas le nom d'une machine de l'ARI (ari-31-201-XX) dans les fichiers *conf/masters*, *conf/core-site.xml* et *conf/mapred-site.xml*.
- Ajoutez le nom des machines esclaves (datanode + tasktracker) dans le fichier *conf/slaves*.