Amestoy M.
Auffret A.

6 February 2015

Éléments logiciels pour le traitement des données massives

PageRank

Encadrants:

Matthieu DURUT et Xavier DUPRÉ

ENSAE ParisTech

Sommaire

1 Introduction

L'algorithme que nous avons choisi d'implémenter est celui du célèbre PageRank, utilisé par Google pour le référencement naturel des pages web. Il permet d'attribuer à chaque page un niveau de qualité (appelé *PageRank*, justement) afin de classer les pages web entre elles sur la base de cet indicateur. L'algorithme s'appuie sur la structure des réseaux formés par les liens hypertextes qui relient les pages web les unes aux autres. Attribuer des niveaux de qualité aux pages web revient alors à leur attribuer des poids en tant que noeuds de ce réseau. Le principe de base du PageRank est de considérer que plus un grand nombre de pages renvoie vers une page donnée (et plus ces pages ont un PageRank élevé), plus la page en question est digne d'intérêt, ce qui signifie qu'on lui attribuera un niveau de qualité plus élevé.

Nous allons dans un premier temps rappeler rapidement le principe de l'algorithme de façon plus formelle pour comprendre de quelle façon un modèle de programmation MapReduce peut lui être appliqué. Nous présenterons ensuite plus concrètement notre façon de procéder, les données utilisées, les difficultés rencontrées et enfin les résultats auxquels nous avons pu parvenir.

2 Algorithme

2.1 Méthode de calcul du PageRank

La méthode de calcul du niveau de qualité dans l'algorithme PageRank ne s'appuie en réalité pas uniquement sur les seules données du réseau formé par les pages web. La formule utilisée est la suivante :

$$P(n) = \alpha \frac{1}{|G|} + (1 - \alpha) \sum_{m \in L(n)} \frac{P(m)}{C(m)} \tag{1}$$

Avec :

- $P(n)$: le PageRank de la page n (niveau de qualité)
- $|G|$: nombre total de noeuds dans le réseau
- $L(n)$: ensemble des pages possédant un lien qui renvoie vers la page n
- $C(m)$: nombre total de lien sur la page m
- α : facteur de "téléportation"

Le terme de droite de la formule (1) correspond à l'explication faite en introduction. Plus les pages m qui renvoient vers n ont un PageRank élevé, plus celles-ci contribueront à renforcer le PageRank de n (en pondérant cela grâce au nombre de total de lien sur la page m). Le terme de gauche en revanche ne fait en rien référence à ce que nous avons pu dire plus haut. En réalité, pour comprendre cette formule, il faut imaginer qu'on se place dans le cadre théorique d'un internaute qui se déplace de façon aléatoire sur le réseau des pages web en ayant à chaque fois deux possibilités :

1. Cliquer sur un lien de la page sur laquelle il se trouve, avec une probabilité $(1 - \alpha)$.

2. Se "téléporter" de façon complètement aléatoire sur n'importe quelle page du réseau (avec donc une probabilité $\frac{1}{|G|}$ de tomber sur une page n donnée). Il choisit cette option avec une probabilité α, ce qui explique la méthode de calcul du PageRank et la dénomination "facteur de téléportation" pour α.

2.2 Algorithme et utilisation de la méthode MapReduce

Il convient tout d'abord de noter que l'algorithme va devoir prendre en compte deux étapes bien distinctes au moment du calcul du PageRank :

- L'étape de calcul des contributions aux PageRanks des pages entre elles.
- L'ajout du terme de "téléportation" et la gestion des noeuds isolés. Nous regroupons ces deux éléments car ils se gèrent de façons similaires.

Le premier calcul qui constitue le premier job de l'algorithme est obtenu de la façon suivante :

1. On initialise l'algorithme en distribuant de façon uniforme le même PageRank à toutes les pages.
2. A l'étape "Map", on détermine pour chaque page les contributions à "envoyer" sur les pages vers lesquelles elle renvoie, donc $\frac{P(m)}{C(m)}$. Puis on étiquette chaque contribution par le nom du noeud auquel elle est destinée.
3. L'étape "Reduce" va alors consister à regrouper toutes les contributions par noeud de destination et les sommer pour les attribuer aux noeuds en question. On note p cette somme. $p = \sum_{m \in L(n)} \frac{P(m)}{C(m)}$

La figure 1 permet d'illustrer l'enchaînement de ces étapes.

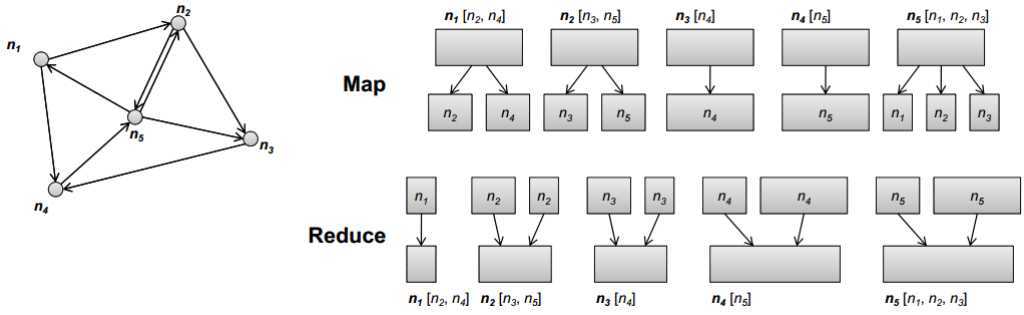

Figure 1: Illustration de l'algorithme
Source : Jimmy Lin and Chris Dyer - *Data-Intensive Text Processing with MapReduce*, 2010

Il faut ensuite gérer la distribution du terme de téléportation et aussi redistribuer la "masse" de PageRank qui n'est pas transférée au niveau des noeuds isolés (c'est-à-dire ceux ne renvoyant vers aucune autre page). Ceci correspond au deuxième job de

l'algorithme.

On sait que chaque noeud va recevoir une contribution de $\alpha \frac{1}{|G|}$ concernant la "téléportation". En ce qui concerne la masse de PageRank isolée, le principe est de redistribuer cette masse de façon uniforme sur tous les noeuds du réseau, du côté du terme "en $(1-\alpha)$" de l'équation. On obtient, pour chaque noeud, la valeur finale p' du PageRank attribué :

$$p' = \alpha(\frac{1}{|G|}) + (1-\alpha)(\frac{m}{|G|} + p) \tag{2}$$

où m est la masse perdue de PageRank, c'est-à-dire la masse totale de PageRank des noeuds isolés.

On réitère alors à partir de l'étape 2 du premier job.

Pour terminer, il est nécessaire de déterminer un critère d'arrêt pour l'algorithme, pour fixer les valeurs définitives des PageRank de nos pages. Une méthode pourrait consister à considérer que l'algorithme doit s'arrêter à partir du moment où les valeurs des PageRanks ne changent plus d'une étape à l'autre (à un intervalle de détail près, à déterminer). Cependant, ceci peut s'avérer plutôt long et la méthode suivante, plus rapide, serait préférable. L'objectif étant de classer les pages les unes par rapport aux autres, on peut très bien décider d'arrêter l'algorithme dès que le classement des pages n'est plus modifié d'une étape à l'autre. Ceci peut arriver quand bien même les valeurs des PageRanks bougent encore (mais elles ne viendront de toute façon plus modifier le classement des pages).

2.3 Données utilisées

Nous avons décidé d'utiliser les données "Google web graph" disponibles sur le site web *SNAP: Stanford Network Analysis Project* (http://snap.stanford.edu/). Cette base contient 875 713 noeuds et 5 105 039 liens.

Nous avons utilisé comme base test une base comportant cinq noeuds reproduisant celle présentée dans l'article.

2.4 Difficultés rencontrées

La première des difficultés que nous avons pu rencontrer a été de devoir écrire un algorithme en PIG sans avoir beaucoup d'expérience dans ce langage et avec une documentation relativement rare sur Internet. Cela a eu pour conséquence de nous faire passer un temps assez important pour mettre en place des éléments parfois "basiques" dans notre code.

2.5 Résultats obtenus

Nous avons mis en place l'algorithme et après l'avoir testé sur les données simulées nous l'avons lancé sur les données Google. Ne sachant pas combien d'itérations il était nécessaire de faire pour se rapprocher de la situation d'équilibre nous avons arbitrairement choisi d'en réaliser huit. Pour choisir correctement cette valeur nous aurions pu mesurer la distance

entre deux itérations et continuer jusqu'à ce que celle-ci soit inférieur à un certain seuil. Voici la répartition des poids obtenue après huit itérations:

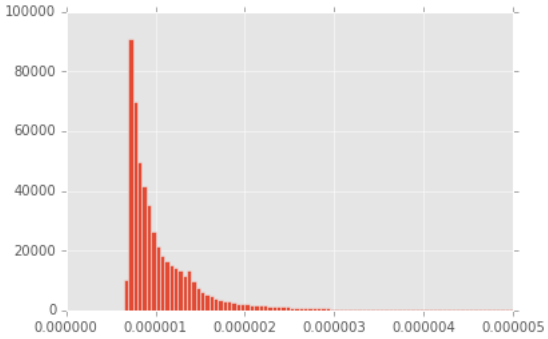

Figure 2: Répartition des scores PageRank après huit itérations

Comme on peut le voir on est déjà très loin de la répartition uniforme initiale et comme on pouvait s'y attendre, les pages avec un poids très faible sont nombreuses et leur nombre décroît exponentiellement avec le poids.

3 Conclusion

La mise en oeuvre du PageRank sur les données Google a donc été pour nous une première occasion d'appliquer un algorithme sur des données massives. Ceci nous a permis de nous familiariser davantage avec PIG et avec les différentes problématiques liées à la gestion de grosses bases de données.

Cependant, nous sommes bien conscients que le PageRank ne représente en réalité qu'une petite partie de ce que Google utilise pour déterminer le classement de ses pages web. En effet, ne considérer que cette méthode de calcul pour le niveau de qualité d'une page peut entraîner des comportements pervers de la part des propriétaires des pages web, qui peuvent par exemple créer un grand nombre de "fausses" pages web contenant chacune un lien vers leur page principale, ce qui permet de gonfler artificiellement le PageRank de la page en question.