Game of Thrones

House of D2SI

1 Contexte

Aemon, maester de Castle Black, est l'un des plus proches conseillers de Lord Commander Jeor Mormont. Il est aussi le dirigeant de la bibliothèque de Castle Black qui contient des centaines de milliers de livres dont certains que l'on ne trouve nulle part ailleurs dans les sept royaumes, pas même dans la Citadelle.

Chercher de l'information dans cette bibliothèque immense est un véritable challenge, et au fur et à mesure que les années passent cette tâche devient de plus en plus fastidieuse pour Aemon. Il fait donc appel à Sam, son apprenti préféré, et lui demande de trouver une méthode qui va lui permettre de **trouver facilement une information à partir de mots clefs**. Sam demande à son tour de l'aide à son ami Jon Snow qui a toujours eu un faible pour la recherche d'information et le data engineering.
Jon Snow décide alors d'utiliser une représentation vectorielle de document.

Pour cet exercice, nous nous mettons dans la peau de Jon Snow pour implémenter de manière efficace la construction d'un **index inversé**. Souvenez-vous, la bibliothèque de Castle Black contient beaucoup de livres, le jeu de données fourni n'en est qu'une petite partie et votre implémentation doit tenir compte de cette contrainte.

2 Algorithme

Dans le domaine de la recherche d'information, les documents peuvent être représentés de plusieurs manière, et notamment de manière vectorielle. Une représentation vectorielle a pour avantage de donner un accès direct à des outils mathématiques tels que la distance, la similarité, la réduction de dimensions, etc...

Notre exercice ne concerne pas directement ces outils mathématiques mais il s'agit de créer un index inversé des documents afin d'accélérer les calculs. En effet ceux-ci s'appuient souvent sur des produits scalaires qui sont coûteux en calcul et en mémoire. Le calcul des produits scalaire est simplifié lorsque l'on dispose d'un index inversé.

2.1 Index inversé

Hypothèse Nous avons une collection de documents de taille N. Dans notre cas nous avons un fichier par document et le nom du fichier est simplement l'indice du document (1,2,3,...)

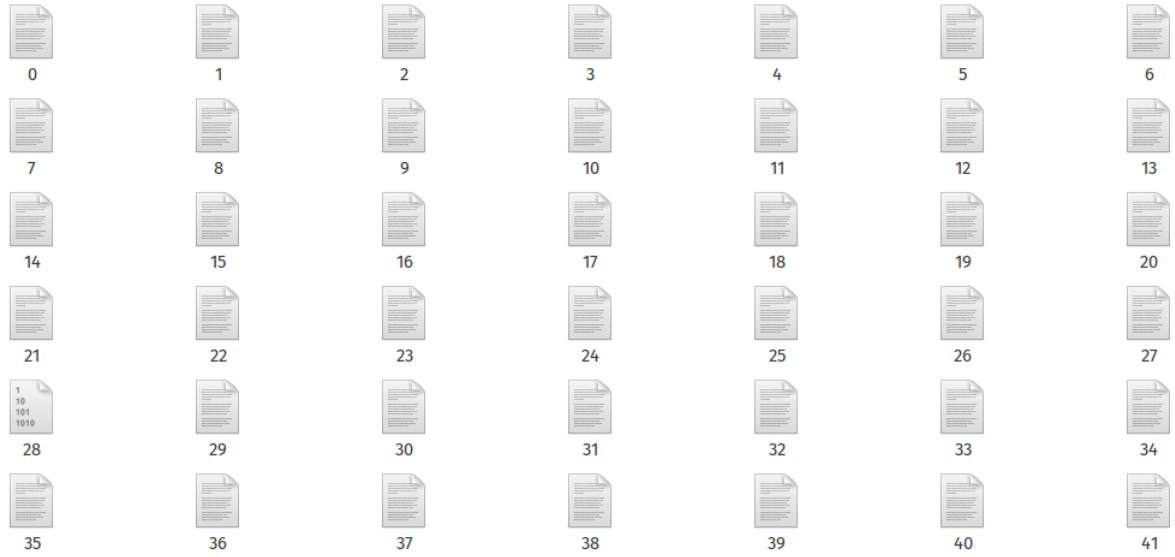

Figure 1: Collection de documents

Nous avons également un dictionnaire de taille M associant chacun des mots du corpus mot à un indentifiant unique.

```
Project         0
This            1
Gutenberg's     2
is              3
of              4
Copyright       5
Welcome         6
See             7
most            8
Shakespeare's   9
...
```

Figure 2: Exemple de dictionnaire

Cela va nous permettre de construire un index inversé qui pour chaque mot fournit l'ensemble des documents dans lesquels il apparaît.

```
(0, [2,5,13,24,30])

(1, [1,2,4,23,44])

(3, [5,9,24,44])

(4, [2,3])

(5, [3,6,9,10,33])

(6, [5,44])

(7, [30,40])

(8, [1,4,7,35])

(9, [16,22])

(10, [13,16,17,28,34])

(11, [1,9])
...
```

Figure 3: Exemple d'index inversé

2.1.1 Construction de l'index inversé

Notre solution doit fonctionner dans le cas d'un très grand corpus de documents. Notre algorithme doit pouvoir tourner sur un système distribué de plusieurs noeuds afin de ne pas être limité par le stockage, la puissance et la mémoire d'une seule machine.

Notre algorithme se fera en 4 étapes.

1. Collecter l'ensemble des paires (idMot, idDoc)
2. Trier ces paires par ordre idMot puis idDoc
3. Grouper ces paires pour chaque idMot, afin d'avoir la liste des docs pour chaque mot
4. Fusionner les index inversés intermédiares pour obtenir l'index inversé final

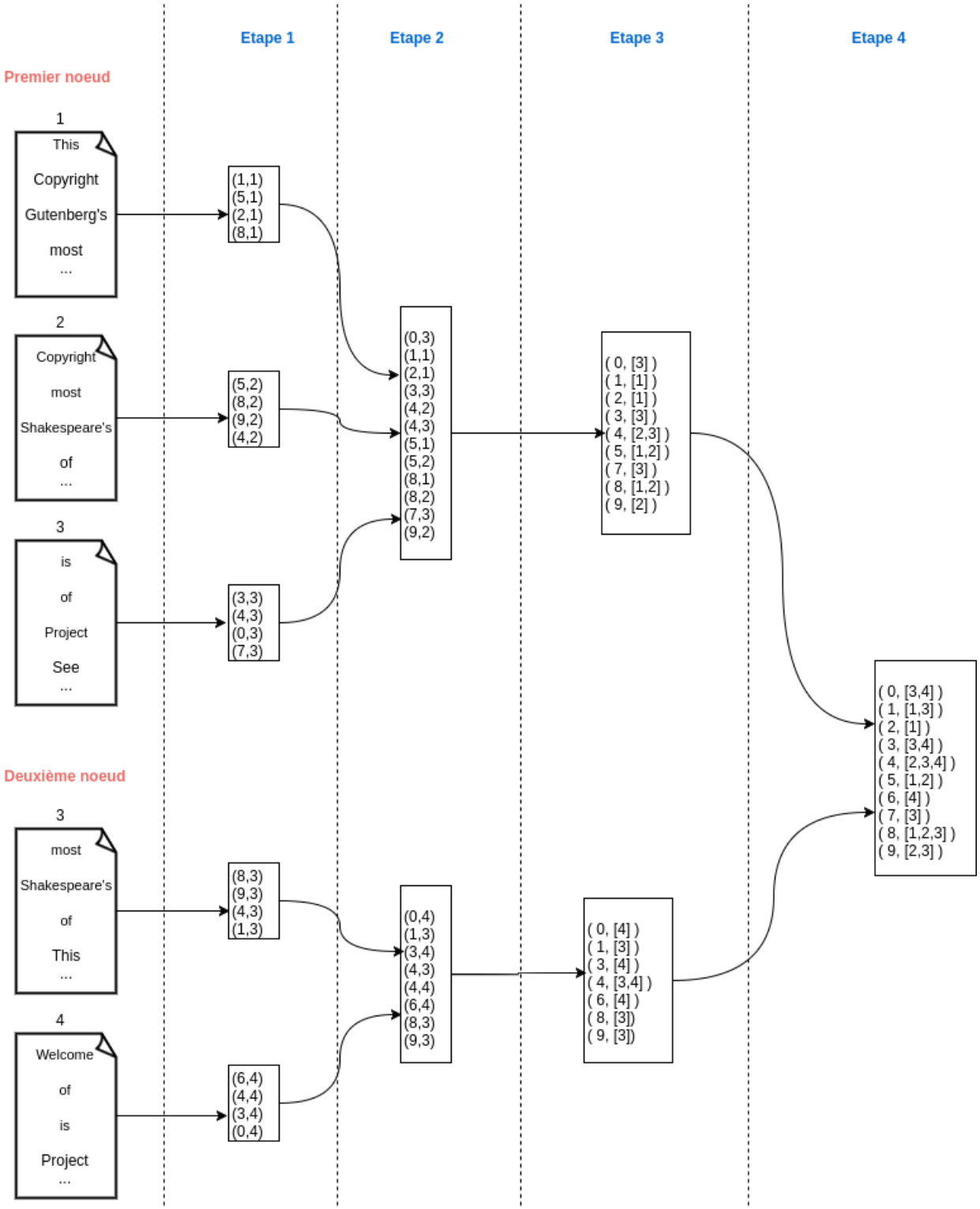

Figure 4: Illustration de l'algorithme

N.B: L'index inversé doit être trié par ordre des identifiants des mots et chaque liste d'une entrée doit être triée par ordre des identifiants des documents.

Algorithme associé: Blocked sort-based indexing

3 Questions

Nous fournissons un corpus de documents disponible dans le dossier corpus de dépôt.

1. Implémenter un job pour construire le dictionnaire. Dans les documents, les mots sont séparés par un ou plusieurs espaces. Une ligne de ce dictionnaire contiendra le mot et son identifiant.

2. Implémentez la construction de l'index inversé en un ou plusieurs jobs.

Vous avez le choix entre les outils Hadoop et Spark pour l'écriture de ces jobs, dans le langage de votre choix.