Rapport TP6

Willian Ver Valen Paiva Alan Guitard

October 16, 2016

Contents

1 Introduction

L'objectif de ce travail consiste à utiliser et maîtriser les méthodes «setup» et «cleanup» qui sont impliquées par l'exercice proposé. La méthode «setup» est celle qui est appelée avant le «mapper» et où on initialise les objets qui vont être utilisés dans tous les «mappers». La méthode «cleanup» est celle qui est appelée après le «mapper».

2 Exercice 1

Pour faire cet exercice nous avons initialisé deux variables dans la méthode Setup; une qui s'appelle ktown et l'autre : topk dans lesquelles ktown est une TreeMap et topk dont la valeur est passée en paramètre par la ligne de commandes.

```
protected void setup(Context context)
{
  ktown = new TreeMap<Long,String>();
  topk = context.getConfiguration().getInt("topk", 1);
}
```

map Dans la liste ktown, nous entrons les valeurs des villes et populations grâce à la méthode add où la taille de la liste est définie par la valeur de topk . Lorsque nous appelons cette méthode, elle regarde si la liste est déjà remplie. Si non, elle ajoute la nouvelle valeur à la liste. Si elle est déjà remplie, elle recherche la plus petite valeur existante dans la liste et la remplace par la valeur que nous voulons ajouter, si cette dernière est plus grande.

```
private void add(long a,String town)
{
  if(ktown.size() < topk)
  {
    ktown.put(a,town);
  }
  else
  {
    long first = ktown.firstKey();
    if(first < a)
    {
      ktown.remove(first);
      ktown.put(a, town);
    }
  }
}
```

Puis, dans la méthode map, on appelle la méthode add avec toutes les villes.

```
public void map(Object key, Text value, Context context
```

```
    ) throws IOException, InterruptedException {

  String[] line = value.toString().split(",");

  if(!line[4].equals("Population"))
  {
    if(!line[4].equals(""))
    {
      String name = line[2];
      this.add(Long.parseLong(line[4]), name);
    }
  }
}
```

Ensuite, dans la méthode cleanup, on écrit toutes les valeurs de la liste ktown dans le contexte.

```
protected void cleanup(Context context) throws IOException,
    InterruptedException
{
  for(long d : ktown.keySet())
  {
    context.write(new Text(ktown.get(d)), new LongWritable(d));
  }
}
```

reducer Les méthodes setup et cleaner du reducer sont exactement les mêmes que celles du mapper. Et dans le reducer on appelle add pour toutes les valeurs reçues.

```
public void reduce(Text key, Iterable<LongWritable> values,
    Context context
    ) throws IOException, InterruptedException {
  for(LongWritable v : values)
  {
    this.add(v.get(),key.toString());
  }
}
```

3 Exercice 2

Il s'agit là d'ajouter la classe reducer en tant que combiner au job.

```
job.setCombinerClass(TP3Reducer.class);
```

4 Exercice 3

Les tests effectués ont tous été passés avec succès.