COSC 589 - Web Search and Sense-Making

Assignment 6
Due Wednesday , 4/18/2018, 11:59pm

Task: PageRank

Introduction:

In this assignment, we will implement the PageRank algorithm on the Wikipedia data.

Requirements:

>100GB free disk space in your machine.

Instructions:

Write a PageRank.scala file to calculate and sort the PageRank scores for the English Wikipedia pages, by taking the following steps:

1. Read in the output files of your last assignment, which contains the link graph in Wikipedia. The format should be:
    - One page per line
    - In each line, you have the title of a page and a list of the titles of the outlinks in the page
    - Each outline title is inside [[]], and separated by a tab "\t".
    - The title and the list of links is separated by a delimiter. We recommended "\t".
2. Create two pair RDDs from the input file. One pair RDD holds the page and its outlinks and another pair RDD holds the page and its PageRank score.
3. Implement the PageRank algorithm. Calculate PageRank scores for all the pages in the input file. Use ITERATION = 20.
4. Sort the PageRank scores for the pages in the descending order.
5. Save the title of a page and its PageRank scores into file. The format is like:

```
[[Serie A]]     4.798429823418915
[[United States dollar]]        4.796979625049858
[[Iraq]]        4.764439498823581
[[Columbia Pictures]]   4.762980165170773
[[mixed martial arts]]  4.756275511978413
[[Rio de Janeiro]]      4.7528282896086465
[[People (magazine)]]   4.7463648409870185
```

6. You are welcome to use the following code template:

```
import scala.util.matching.Regex
import org.apache.spark.SparkConf
import org.apache.spark.SparkContext
import org.apache.spark.SparkContext._
import org.apache.hadoop.io.compress.GzipCodec
```

```
object PageRank {
  def main(args: Array[String]) {
    val sparkConf = new SparkConf().setAppName("PageRank")
    val sc = new SparkContext(sparkConf)
    val input = sc.textFile("./linkgraph/*.gz")  // your output directory from the last
assignment
    val links =  // Load RDD of (page title, links) pairs
    val ranks = // Load RDD of (page title, rank) pairs

    val ITERATION = 20
    // Implement your PageRank algorithm according to the notes
    …

    // Sort pages by their PageRank scores
    ranks.sortBy …

    // save the page title and pagerank scores in compressed format (save your disk
space). Using "\t" as the delimiter.
    ranks.map(r => r._1 + "\t" + r._2).saveAsTextFile("./pageranks", classOf[GzipCodec])

  }
}
```

What to Submit:

- Your code
- Screen captures of the beginning of your pageranked pages, in descending order (e.g. the first 20 lines on the screen. Hint: Use 'gunzip part-00000.gz" to unzip, then view the documents and screen capture)

Bonus: (20% of the entire grade of this homework)

Output pages and their PageRank scores for only Persons.

What to Submit:

- Your code
- Screen captures of the beginning of your pageranked pages, in descending order (e.g. the first 20 lines on the screen. Hint: Use 'gunzip part-00000.gz" to unzip, then view the documents and screen capture)

What NOT to Submit:

- Your input or output files

Where to submit:

- Canvas

When:

- Due on 04/18/18, 11:59pm.