Big Data Analytics Project Spring 2015, Option A
Finding most top 100 universities in Wikipedia Dataset

Due Sunday May 3rd 11:59 pm.

1. Introduction

"One of the biggest changes in our lives in the decade following the turn of the century was the availability of efficient and accurate Web search, through search engines such as Google. While Google was not the first search engine, it was the first able to defeat the spammers who had made search almost useless. Moreover, the innovation provided by Google was a nontrivial technological advance, called "PageRank." (Mining of Massive Data Sets—Chapter 5).

In this project you are to use Amazon Elastic Map Reduce (EMR) to implement the pageRank algorithm on the provided Wikipedia dataset. The goal is to find the top 100 universities with the highest page rank in Wikipedia dateset.

2. What is PageRank and how it is computed

Algorithm

PageRank is a function that assigns a real number to each page in the Web. The intent is that the higher the PageRank of a page, the more important it is. The page rank PR of a page p_i is calculated as follows:

$$PR(p_i) = \frac{1-d}{N} + d \sum_{p_j \in in(P_i)} \frac{PR(p_j)}{L(p_j)}$$

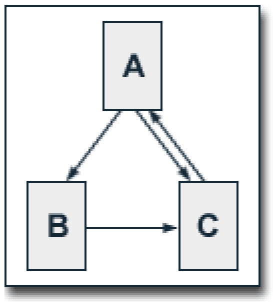
Where

- d: is a constant (it is typically set to 0.85)
- N: is total number of pages (In this case Wikipedia articles)
- $in(p_i)$: The set of all incoming links to p_i
- $L(p_j)$: The number of outgoing links form p_j

At the initial point, each page is initialized with PageRank 1/N and the sum of the PageRank is 1. But the sum will gradually decrease with the iterations. **For this project, you only need to do 10 iterations** and print the page ranks for each Wikipedia page after the 10th iteration.

Example

We regard a small web consisting of three pages A, B and C, whereby page A links to the pages B and C, page B links to page C and page C links to page A. The following figure illustrates the link graph of this simple problem.



1) Initialization
   d=0.85, N=3, PR(A)=1/3, PR(B)=1/3, PR(C)=1/3
2) Iteration =1

$$PR(A) = \frac{1-0.85}{3} + 0.85 \times \left(\frac{PR(C)}{1}\right) = 0.05 + 0.85 \times \left(\frac{1/3}{1}\right) = 0.333333$$

❖ In the computation of PR(A), page C is the only incoming link to page A, and the number of outgoing link from page C is 1.

$$PR(B) = \frac{1-0.85}{3} + 0.85 \times \left(\frac{PR(A)}{2}\right) = 0.05 + 0.85 \times \left(\frac{1/3}{2}\right) = 0.191667$$

❖ In the computation of PR(B), page A is the only incoming link to page B, and the number of outgoing link from page A is 2.

$$PR(C) = \frac{1-0.85}{3} + 0.85 \times \left(\frac{PR(A)}{2} + \frac{PR(B)}{1}\right) = 0.05 + 0.85 \times \left(\frac{1/3}{2} + \frac{1/3}{1}\right) = 0.475$$

❖ In the computation of PR(C), page A and page B are the incoming links to page C, and the number of outgoing link from page A is 2 and the number of outgoing link from page B is 1.

3) Iteration=2

$$PR(A) = \frac{1-0.85}{3} + 0.85 \times \left(\frac{PR(C)}{1}\right) = 0.05 + 0.85 \times \left(\frac{0.475}{1}\right) = 0.45375$$

$$PR(B) = \frac{1-0.85}{3} + 0.85 \times \left(\frac{PR(A)}{2}\right) = 0.05 + 0.85 \times \left(\frac{0.333333}{2}\right) = 0.191666667$$

$$PR(C) = \frac{1-0.85}{3} + 0.85 \times \left(\frac{PR(A)}{2} + \frac{PR(B)}{1}\right) = 0.05 + 0.85 \times \left(\frac{0.333333}{2} + \frac{0.191667}{1}\right) = 0.354583333$$

4) We can repeat this process until the number of Iterations is 8

3. Wikipedia Dataset

Wikipedia offers a free copy of all of its available content to the interested users. For this project you should use the dump of the English –language Wikipedia articles which is available in XML format at http://dumps.wikimedia.org/enwiki/latest/enwiki-latest-pages-articles.xml.bz2 . The size of the data is approximately 10GB compressed and about 45GB uncompressed).

Attention: You don't need to download the dataset and upload it to your s3 bucket. You can access it directly from my s3 bucket using the following path:

s3://class-data-set/ProjectOptionA/enwiki-latest-pages-articles.xml

The XML file contains some meta information about each page as well as the title, and the actual text of the page. A single page is represented as follows (some attributes and meta information are omitted)

```
<page>
<title>Title</title>
...
<text ...>
Body of page
[[link text]]
</text>
</page>
```

If you are interested to see a smaller sample of the dataset please go to this website: http://en.wikipedia.org/w/index.php?title=Special:Export type a category (e.g., "computer Science") then hit "add" and then click on "export" to download the xml file containing the information regarding the Wikipedia articles that fall within this category. You should first test and debug your program locally with a smaller sample of the dataset and then run it on Amazon EMR.

Wikipedia articles can and do link to one another. These links ("Wikilinks") are what we are interested in. Our goal in this project is to use these wikilinks to determine the importance (page rank) of each Wikipedia article. A **wikilink** (or **internal link**) links a page to another page within English Wikipedia. Within the text of a page, a wikilink is indicated by **[[link text]]** where link text is the title of another Wikipedia page There are two formats of wikilinks:

- **[[link text]]** is seen as "link text" and links to a page with title "link text". For example, [[C programming language]] is seen as "c programming language" in text and links to the Wikipedia page "C programming language".

- **[[link title | link text]]** is seen as "link text" but links to a page with title "link title". For example, [[Java programming language | java]] is labelled "java" on this page but links to page "java programming language".

To read more about wiki links, please refer to http://en.wikipedia.org/wiki/Help:Link#Wikilinks

4. What you need to do

Step 1 : Extracting the link dataset

The first thing you need to do is to extract the wikilinks from the xml file. This will be a dataset that contains records of the following form:

Page_title 1, Page_title 2

Where there is a wikilink from page_title 1 to page_title 2. This dataset is typically called a "link graph".
How can you extract the link graph from the xml document? Well, there are multiple ways to do it. One way is to use Mapreduce along with the XMLInputFormat in Apache Mahout library to parse xml files. Mapreduce does not come with a built in InputFormat for xml but Apache Mahout does. You can use Mahout XMLInputFormat calss to extract everything between <page> </page> tags and send it to your map function. Then in your map function you can use java built-in library (StAx) as well as JDome library to extract the value of the <title> and the <text>.
Once you extract the text, then you can use java regular expressions to look for wikilinks. Your mapreduce job will be a map only task. You don't need a reducer to produce the link dataset.
Notes:

- You do not need to include the mahout library in your MapReduce project. You can just copy and paste the XMLInputFormat.java file in your project folder. I have posted the file XMLInputFormat.java on blackboard.
- To use Jdome library, you can download the jar file from here: (http://www.java2s.com/Code/Jar/j/Downloadjdomjar.htm) and add it as an external library to your project.

Here are some helpful links:

- http://xmlandhadoop.blogspot.com/ . This page shows you an example of using Apache Mahout and Mapreduce to parse xml files in hadoop. **Note:** you do not need to install mahout, just copy the source code provided in the link for Mahout XMLInoutFormat and you'll be able to use it in your MapReduce project.
- http://ocpsoft.org/opensource/guide-to-regular-expressions-in-java-part-1/ this is a short quick start guide to java regular expression. Once you extracted the text of a page in your map function, you need to use regular expression to extract all the wikilinks, i.e., patterns that match [[link page]] or [[link page | link text]]. You can look at this forum for information on how to do this.
http://stackoverflow.com/questions/8138153/matching-wikilinks-with-a-regex

Step 2 : Writing a Pig or Hive program to compute the top 10 universities with the highest page rank

Once you have the link dataset you can write a Pig or Hive program to compute the top 10 universities with the highest page rank:

- The first thing you need to do is to remove the red links from the data set. The red links are the links that point to a page that does not exist. For example, after extracting the links if you have the following record in your link dataset:
  Page_title 1, Page_title 2
  But there is no other record in your dataset with page_title2 in its first column, then page_title 2 is a red link and you need to remove this record from your link dataset. You can easily do this in Pig or Hive.
  Note: it might be the case that after removing the red links for the first time, you might still end up having some pages in your right column without having them anywhere in the left column. But that's ok. You don't need to do another round of red link removal.

- Now your dataset it ready for page rank computation. You can write a pig/hive program to compute the page rank of all pages in your link dataset.

- You need to write a pig /hive script to do the page rank calculation and repeat it for 10 iterations to update the page ranks in each iteration.

- Once you calculated the page rank for all pages in your link dataset, then select the pages whose title starts with "university of" or ends with "university".

- Produce in the output only the top 10 universities with the highest page rank.

5. What you need to turn in:

Here are the items that you need to submit:

1. **"Links.txt ". The link data set after removing the red links**. The records in this dataset should be in the following format.
   Page_title1, page_title2
   Where there is a link from page_title 1 to page_title 2.

2. **"university_rank.txt". The top 10 universities with the highest page rank.** The records of this data set should be in the following format (fields are tab separated):
   ```
   Page_title	page_rank
   ```
   The format of your output should be as follows (This is just an example and the numbers are made up)

   ```
   Harward University	0.1
   ```

Stanford University 0.08
……………………………………….

3. **“Source.zip”.** All your source files in a zip folder along with a readme.txt describing the submission
4. **Project Report (Doc or PDF File) with the following format:**
   - Project title & your name
   - Introduction: A short paragraph explaining the problem
   - Methodology: explaining the algorithms that you used (your mapreduce and hive or pig algorithms)
   - Results and Conclusion
   - References: if you used any website, book, etc. please list it here and cite it in your text.
5. **Presentation:** You need to prepare a set of power point (or openOffice) slides to present your work. This should include 20-25 slides. The first slide should include the title of your project and your name. The next slides should include a quick overview of the presentation. The rest of the slides should explain your work and the results.
   Your presentation should include your voice narrating the slides. If you are using PowerPoint you can simply go to slideshow tab and choose "record narration".