Programming Assignment 1

Hyperlink-Induced Topic Search (HITS) over Wikipedia Articles using Apache Spark

Due: Feb. 18 Tuesday 5:00PM
Submission: via Canvas, Individual submission
Instructor: Sangmi Lee Pallickara
Web page: http://www.cs.colostate.edu/~cs535/Assignments.html
Objectives

The goal of this programming assignment is to enable you to gain experience in:

- Installing and using analytics tools such as HDFS and Apache Spark
- Generating root set and base set based on link information
- Implementing iterative algorithms to estimate Hub and Authority scores of Wikipedia articles using Wikipedia dump data

1. Overview

In this assignment, you will design and implement a system that generates a subset of graphs using ***Hyperlink-Induced Topic Search (HITS)***[1][2] over currently available Wikipedia articles[3]. HITS is a link analysis algorithm designed to find the key pages for specific web communities. HITS is also known as ***hubs and authorities***.

HITS relies on two values associated with each page: one is called its hub score and the other its authority score. For any query, you compute two ranked lists of results. The ranking of one list is induced by the hub scores and that of the other by the authority scores. This approach stems from a particular insight into the very early creation of web pages --- that there are two primary kinds of web pages that are useful as results for broad-topic searches. For example, consider an information query such as "I wish to learn about the eclipse". There are authoritative sources of information on the topic; in this case, the National Aeronautics and Space Administration (NASA)'s page on eclipse would be such a page. We will call such pages as **authorities**.

[1] Christopher D. Manning, Prabhakar Raghavan and Hinrich Shutze, Chapter. 21, Link analysis, Introduction to Information Retrieval, https://nlp.stanford.edu/IR-book/pdf/21link.pdf

[2] U.S. Patant 6,112,202, https://www.google.com/patents/US6112202

[3] https://meta.wikimedia.org/wiki/Data_dumps

In contrast, there are pages containing lists of links (hand-compiled) to authoritative web pages on a particular topic. These **hub pages** are not in themselves authoritative sources of topic but they provide aggregated information. We use these hub pages to discover the authority pages.

1.1. Calculating scores

A good hub page is one that points to many good authorities; a good authority page is one that is pointed to by many good hub pages. Therefore, the definition of hubs and authorities is a circular concept and we will turn this into an iterative computation. Suppose that we have a subset of the web containing good hub and authority pages, with hyperlinks among them. We will iteratively compute a hub score and an authority score for every web page in this subset.

For a web page v in our subset of the web, we use $Hub(v)$ to denote its hub score and $Authority(v)$ to denote its authority score. Initially, we set $Hub(v) = Authority(v) = 1$ for all nodes v. We also denote by $v \rightarrow y$ the existence of a hyperlink from v to y. The core of the iterative algorithm is a pair of updates to the hub and authority scores of all pages. The following equations capture the intuitive notions that good hubs point to good authorities and that good authorities are pointed to by good hubs.

$$h(v) \leftarrow \sum_{v \mapsto y} a(y)$$
$$a(v) \leftarrow \sum_{y \mapsto v} h(y)$$

Thus, the first line of the above equation sets the hub score of page v to the sum of the authority scores of the pages it links to. In other words, if v links to pages with high authority scores, its hub score increases. The second line plays the reverse role; if page v is linked to by good hubs, its authority score increases.

1.2. Choosing the subset of the Web

In assembling a subset of web pages around a topic such as "eclipse", we must cope with the fact that good authority pages may not contain the specific query term "eclipse". For instance, many pages on the Apple website are authoritative sources of information on computer hardware, even though these pages may not contain the term "computer" or "hardware". However, a hub compiling computer hardware resources is likely to use these terms and also link to the relevant pages on the Apple website.

To compile the subset of web pages to compute hub and authority scores, HITS requires following steps. Note that we consider only internal-links that include only the links between Wikipedia pages.

Step 1. Given a query (e.g. "eclipse", there are more than 100 Wikipedia pages with title containing the text "eclipse"), get all pages containing "your-topic" in their **title**. Call this the ***root set*** of pages.

Step 2. Build a ***base set*** of pages, to include the root set as well as any page that either links to a page in the root set, or is linked to by a page in the root set.

Step 3. We then use the base set for computing hub and authority scores. To calculate your hub and authority scores, the initial hub and authority scores of pages in the base set should be set as 1 before the iterations. The iterations should be continued automatically until the values converge. You should normalize the set of values after every iteration. The authority values should be divided by the sum of all authority values after each step. Similarly, the hub values should be divided by the sum of all hub values after each step.

The hubs and authorities scores can be calculated using eigenvectors and Singular Value Decomposition (SVD). However, in programming assignment 1, **you must implement the algorithm with the iterative method.** The final hubs and authority scores are determined only after a large number of repetitions of the algorithm.

1.3. Example

Figure 1 shows an example of calculating hub scores for each page. In the diagram, each node depicts the Wikipedia page, and the edges represent the links included in the source page.

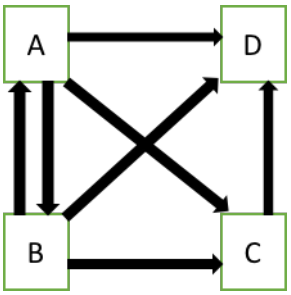

Figure 1. Example with 4 Wikipedia pages, A, B, C, and D

Initially the Hub and Authority scores for each page are set as 1.

$Authority(A) = Authority(B) = Authority(C) = Authority(D) = 1$

$Hub(A) = Hub(B) = Hub(C) = Hub(D) = 1$

In the first iteration, you will calculate authority scores:

$Authority_1(A) = Hub_0(B) = 1$

$Authority_1(B) = Hub_0(A) = 1$

$Authority_1(C) = Hub_0(A) + Hub_0(B) = 2$

$Authority_1(D) = Hub_0(A) + Hub_0(B) + Hub_0(C) = 3$

After normalizing these values with the sum of the new Authority scores,

$Authority\ (A)_1 = 1/(1+1+2+3) = 0.143$

$Authority\ (B)_1 = 1/(1+1+2+3) = 0.143$

$Authority\ (C)_1 = 2/(1+1+2+3) = 0.289$

$Authority\ (D)_1 = 3/(1+1+2+3) = 0.429$

Then, calculate hub scores:

$Hub(A)_1 = Authority\ (B)_1 + Authority\ (C)_1 + Authority\ (D)_1 = 0.858$

$Hub(B)_1 = Authority\ (A)_1 + Authority\ (C)_1 + Authority\ (D)_1 = 0.858$

$Hub(C)_1 = Authority\ (D)_1 = 0.429$

$Hub(D)_1 = 0$

After normalizing these values with the sum of new Hub scores

$Hub(A)_1 = 0.858/(0.858+0.858+0.429+0) = 0.4$

$Hub(B)_1 = 0.858/(0.858+0.858+0.429+0) = 0.4$

$Hub(C)_1 = 0.429/(0.859+0.859+0.429+0) = 0.2$

$Hub(D)_1 = 0$

You should repeat this iteration until the Hub and Authority scores have converged.

2. Requirements of Programing Assignment 1

Your implementation of this assignment should provide:

A. A selection of your root set based on a specified topic. This must be implemented with Apache Spark.

B. Base set for hubs and authorities. This must be implemented with Apache Spark.

C. A sorted list of Wikipedia pages based on the hub scores for the user specified topic (non-hardcoded) in descending order. Each should contain the title of the article and its hub value. This computation should be performed in your own Spark cluster with 5 machines. You should present results from at least 50 pages in the list. This must be implemented with Apache Spark.

D. A sorted list of Wikipedia pages based on the authority scores for the user specified topic (non-hardcoded) in descending order. Each should contain the title of the article and its authority value. This computation should be performed in your own Spark cluster with 5 machines. You should present results from at least 50 pages in the list. This must be implemented with Apache Spark.

You are not allowed to use existing HITS implementations. A 100% deduction will be assessed in that case.

Your submission should be via Canvas. Demonstration of your software should be on machines in CSB-120. Demonstration will include an interview discussing implementation and design details. Your submission should include your source codes and outputs generated for keyword "eclipse". **NOTE** - To generate root set, keyword should be passed as run time argument to your program.

3. Programming Environment

A. Requirements

All software demonstration of this assignment will be done in CSB120. You should make sure your software works on the machines in CSB120.

B. Setup your Hadoop cluster

This assignment includes a requirement of setting up your own HDFS and Spark cluster on every node. For writing your intermediate and output data, you should use your own cluster setup.

4. Dataset

The original dump dataset from Wikipedia is 12.1GB as a bz2 file, and about 50GB when it is decompressed. It follows an XML format and includes other information that is not relevant to this task. To optimize your work on this assignment, we will use Wikipedia dump[4] generated by Henry Haselgrove. Please find the instruction to download the data file from CS120 (instead of the external server) in the course assignment web page. If you have any problem with storage capacity, you should contact Prof. Pallickara immediately.

The dataset contains two files:

`links-simple-sorted.zip` (323MB)
`titles-sorted.zip` (28MB)

These files contain all links between proper Wikipedia pages. This includes disambiguation pages and redirect pages. It includes only the English language Wikipedia.

In `links-simple-sorted.txt`, there is one line for each page that has links from it. The format of the lines is:

$from_1$: to_{11} to_{12} to_{13} ...
$from_2$: to_{21} to_{22} to_{23} ...
...

[4] https://wayback.archive.org/web/20160516004046/http://haselgrove.id.au/wikipedia.htm

where $\texttt{from}_1$ is an integer labeling a page that has links from it, and $\texttt{to}_{11}$ $\texttt{to}_{12}$ $\texttt{to}_{13}$ ... are integers labeling all the pages that the page links to. To find the page title that corresponds to integer `n`, just look up the `n`-th line in the file `titles-sorted.txt`, a UTF-8 encoded text file.

5. Grading

Your submissions will be graded based on the demonstration to the GTA in CSB120. All of the items described in (2) will be evaluated. This assignment will account for 10% of your final course grade.

- 1 point: Generating the root set
- 1 point: Generating the base set
- 3 points: Generating a sorted list based on the hub scores
- 3 points: Generating a sorted list based on the authority scores
- 1 point: Set up the Hadoop (HDFS) cluster
- 1 point: Set up the Spark cluster

6. Late Policy

Please check the late policy posted on the course web page.