UNIVERSITY OF ILLINOIS SPRINGFIELD

BIG DATA ANALYTICS
(CSC 570 U)
FINAL PROJECT
SPRING 2016

"FINDING MOST INFLUENTIAL PAPERS BASED ON DBLP CITATION NETWORK"

UNDER THE GUIDANCE:-

DR. ELHAM KHORASANI

SUBMITTED BY:-

MEGHA SOGANI(UIN: 659991045)

MANISH DUNUNG(UIN: 678127992)

INDEX

I. INTRODUCTION

The World Wide Web creates several challenges for retrieval of the data. It is substantial and diverse. Current assessments are that there are more than 150 million web sites with a multiplying life which is short than a year. Giving significant data to the users to meet their requirements is the essential objective of the website proprietors. In this manner, finding the substance of the Web and recovering the users interests and demands from their actions have become essentially important. Therefore Web Mining is utilized to classify the users and web pages by interpreting the users' conduct or actions, the information of the web pages and the order of the URLs that needs to be accessed in order. HITS and PageRank are the two page ranking algorithms that are mainly used in web structure mining.

II. PageRank Algorithm

In order to estimate the relative importance of Web pages, PageRank was developed at Stanford University by Larry page (cofounder of Google), a method for calculating a ranking for every web page taking into account the graph of the web. In other words, PageRank is nothing but a vote that is casted by all the other pages which are on the Web, to show how important a page is. A link to a page counts as a vote of support. Google utilizes this algorithm to order its search results in a way that imperative documents move up in the search results whereas moving the less critical pages down in its list. It has applications in ranking tweets in twitter, recommendation systems, suggesting friends over twitter, etc.

This algorithm states that **"If a page has some important incoming links to it, then its outgoing links to other pages also becomes important, thus it takes backlinks into account and propagates the ranking through the links."** When some query is given, Google consolidates precomputed PageRank scores with content matching scores to obtain an overall ranking score for each web page in response to the given query.

A simplified version of PageRank is defined as follows:-

$$PR(u) = c \sum_{v \in B(u)} \frac{PR(v)}{N_v}$$

Where u represents a web page. B(u) is the set of pages that point to u. PR(u) and PR(v) are rank scores of page u and v respectively. Nv denotes the number of outgoing links of page v. c is a factor used for normalization.

III. Weighted PageRank Algorithm

Wenpu Xing and Ali Ghorbani proposed an expansion to standard PageRank called Weighted PageRank (WPR). It presumes that more prominent the web pages are more linkages other website pages tend to have to them or are connected to by them. This algorithm allocates bigger rank values to more prominent pages rather than partitioning the rank value of a page uniformly among its outgoing linked pages. Each outlink page gets a value corresponding to its significance and this prominence is measured by its number of incoming and outgoing links. This significance is assigned in terms of weight values to the incoming and outgoing links which are represented as Win(v,u) and Wout(v,u).

Win(v,u) is the weight of link(v,u) that is estimated based on the number of inlinks of page *u* and the number of inlinks of all reference pages of page *v*.

$$W^{in}_{(v,u)} = \frac{I_u}{\sum_{p\in R(v)} I_p}$$

Wout(v,u) is the weight of *link*(*v*, *u*) that is estimated based on the number of outlinks of page *u* and the number of outlinks of all reference pages of page *v*.

$$W^{out}_{(v,u)} = \frac{O_u}{\sum_{p\in R(v)} O_p}$$

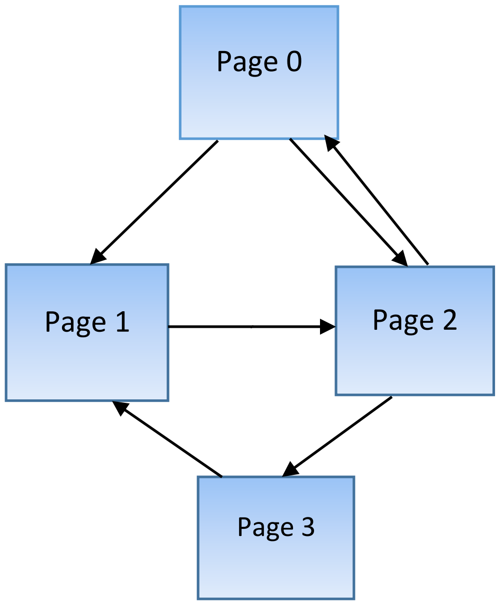
Considering the significance of pages, the original PageRank formula is altered as

$$PR(u) = \frac{1-d}{N} + d \sum_{v\in in(u)} PR(v) \times W^{in}_{(v,u)} \times W^{out}_{(v,u)}$$

Where

- d: is a constant (typically set to 0.85)
- N: is the total number of nodes (In this case total number of papers in the citation network)
- in(u): The set of all the nodes which link to node u

Consider a small example, suppose there is a network consisting of four papers i.e. Page 0, Page 1, Page 2 and Page 3, where Page 0 has references Page 1 and page 2, Page 1 has reference Page 2, Page 2 has reference Page 0 and Page 3, Page 3 has reference Page 1.



Step 1: We initialize d=0.85, N=4, PR(Page 0)=1/4, PR(Page 1)=1/4, PR(Page 2)=1/4, PR(Page 3)=1/4

Step 2: Calculate the number of inlinks and the number of outlinks for all of the above links:

Papers	inlinks	outlinks
Page 0	1	2
Page 1	2	1
Page 2	2	2
Page 3	1	1

Step 3: Calculating Win and Wout

Lets calculate for Page 1, Page 2

$$\text{Win(Page 1, Page 2)} = \frac{\text{\#inlinks(Page 2)}}{\text{\#inlinks(Page 2)}} = \frac{2}{2} = 1$$

$$\text{Wout(Page 1, Page 2)} = \frac{\text{\#outlinks(Page 2)}}{\text{\#outlinks(Page 2)}} = \frac{2}{2} = 1$$

Step 4: Now lets calculate PageRank i.e PR(Page1)

PR(Page1) = (1-d)/N + d* [PR(Page 0) * Win(Page 0, Page 1) * Wout(Page 0, Page 1) + PR(Page 3) * Win(Page 3, Page 1) * Wout(Page 3, Page 1)]

And we can get the PageRank of Page 1 by substituting the values after all the calculations are performed.

IV. AIM:

Our aim in this project is to obtain the top rated papers from the list of papers in the given set of Publications. To implement this, Weighted PageRank algorithm is used.

V. METHODOLOGY:

In this problem, to calculate the page ranks of the paper and produce top 10 papers with highest page rank, we must have to extract the required data from the dataset in very first place. For the simplicity, we have divided this program in two steps. In first place, we have used Map-Reduce framework to extract the index, citations and title with two different Map-Reduce jobs.

Map-Reduce produces the output which we consider it as link.txt and title.txt. These files are used as input to the Hive program to calculate the inlinks, outlinks and weights of papers. Using these attributes, we calculate page rank of each paper. After the 10 iterations, the final page rank will be generated and we can get top 10 papers which are having highest page rank amongst all.

Here is the approach to the program.

Step-1: Map-Reduce algorithm

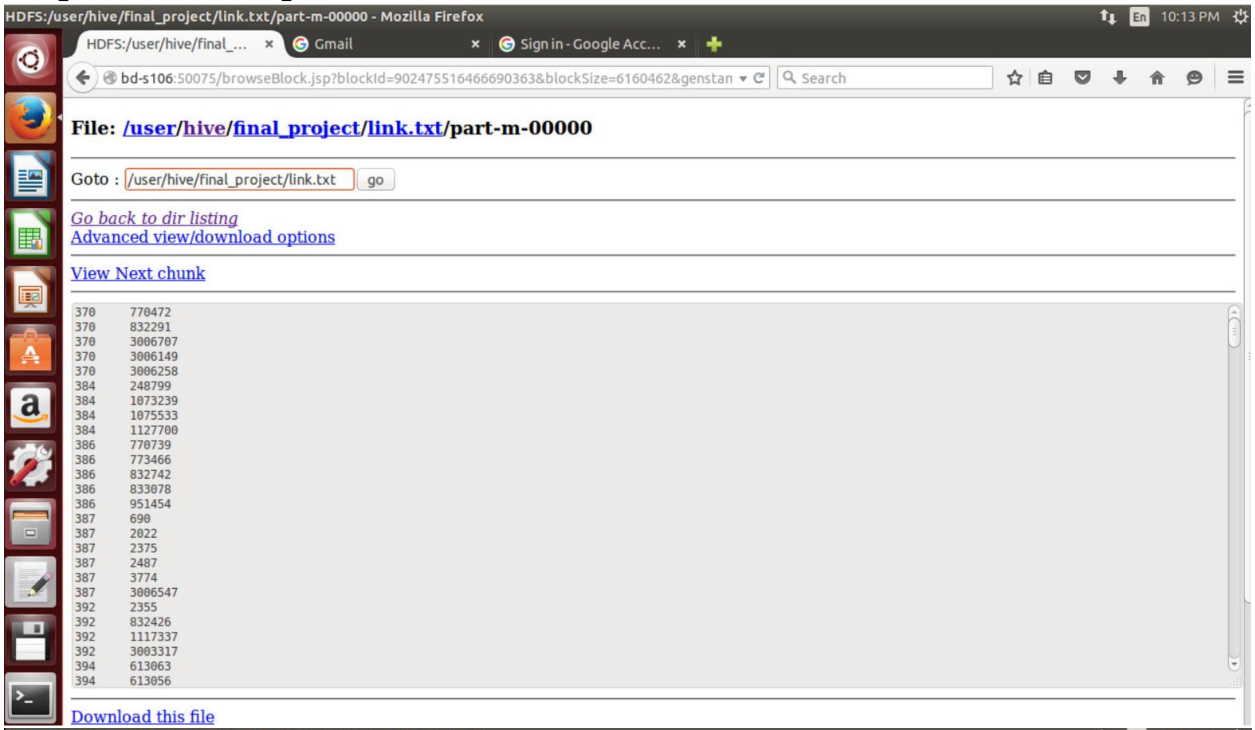
- In the first step, to distinguish between the required data and original data, we need to use DBLP input format. Include it as a Map-Reduce Job in the Driver class.
- As we are not going to use reducer, set number of reduce task to 0.
- Split the incoming string with the new line (\n).
- There might be inconsistency in the dataset. So to omit those records, we need to match the pattern starting with "#index" to get the index values.
- If it matches, then for the records after 5th column (i.e. citations), check for another pattern start with "#%" till the total citations are found for the particular index.
- If it matches, then emit the index and citations delimited by "\t". (linkgraph)
- To emit the titles, again find the pattern which contains the text. If found, then emit the title corresponding to index delimited by "\t". (titlegraph)
- Save the outputs as link.txt and title.txt.

Step-2: Hive algorithm

- Load the data from link.txt into the table linkgraph.
- Load the data from title.txt into the table titlegraph.
- Define total number of nodes in the citation graph and store it into the table nodes.
- Define scaling factor (1-d)/N and store it into table scale_factor.
- Find the inlinks and outlinks for the papers.
- Then calculate the total number of inlinks and total number of outlinks for the paper.
- Using the existing values of inlinks, outlinks, total inlinks and totlal outlinks, compute in-weights and out_weights.
- Assign the initial rankings as the base to compute future ranks within the iterations.
- Using the page rank formula (scaling factor+ d* summation of page ranks of inlinks*Win*Wout) to calculate the new page rank.
- Update the page rank after every iteration and iterate it for 10 times.
- After all the iterations, find out the top 10 papers with title, number of citations and highest page ranks.
- Store the result as paper-rank.txt.

VI. RESULTS

Map-Reduce Output



HDFS:/user/hive/final_project/link.txt/part-m-00000 - Mozilla Firefox

bd-s106:50075/browseBlock.jsp?blockId=9024755164666903638&blockSize=6160462&genstan

File: /user/hive/final_project/link.txt/part-m-00000

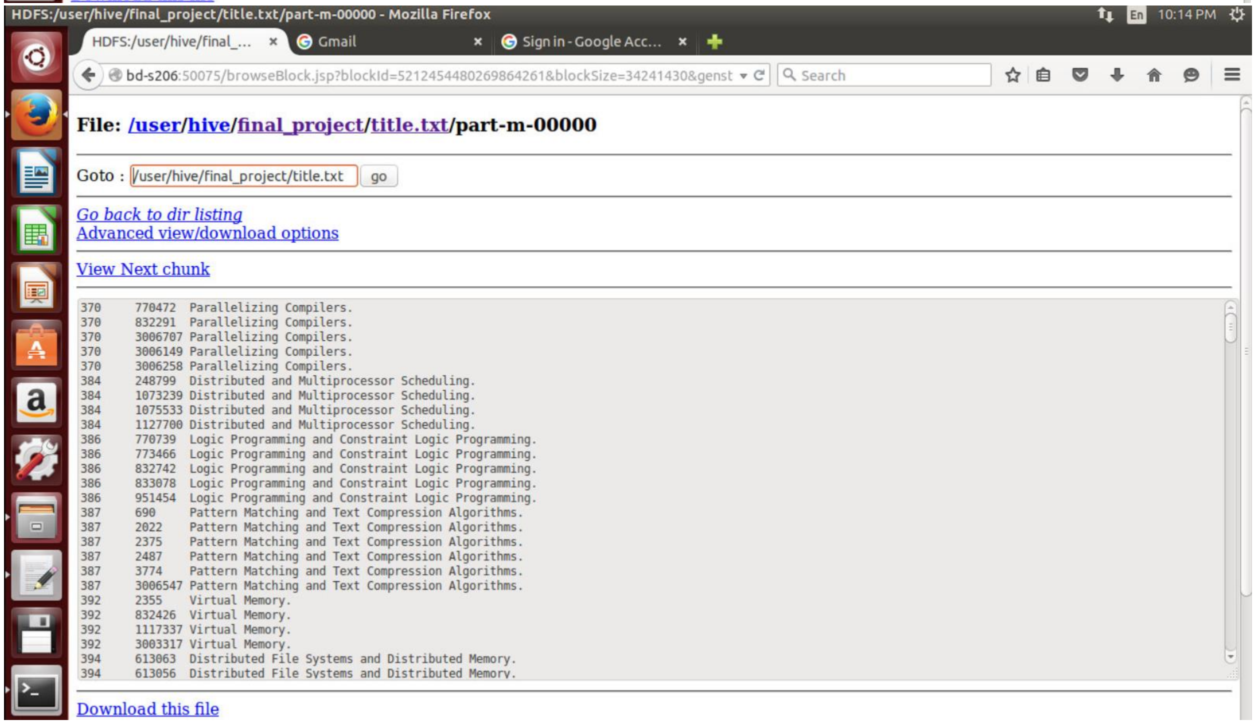
Goto : /user/hive/final_project/link.txt go

Go back to dir listing
Advanced view/download options

View Next chunk

```
370     770472
370     832291
370     3006707
370     3006149
370     3006258
384     248799
384     1073239
384     1075533
384     1127700
386     770739
386     773466
386     832742
386     833078
386     951454
387     690
387     2022
387     2375
387     2487
387     3774
387     3006547
392     2355
392     832426
392     1117337
392     3003317
394     613063
394     613056
```

Download this file



HDFS:/user/hive/final_project/title.txt/part-m-00000 - Mozilla Firefox

bd-s206:50075/browseBlock.jsp?blockId=5212454480269864261&blockSize=34241430&genst

File: /user/hive/final_project/title.txt/part-m-00000

Goto : /user/hive/final_project/title.txt go

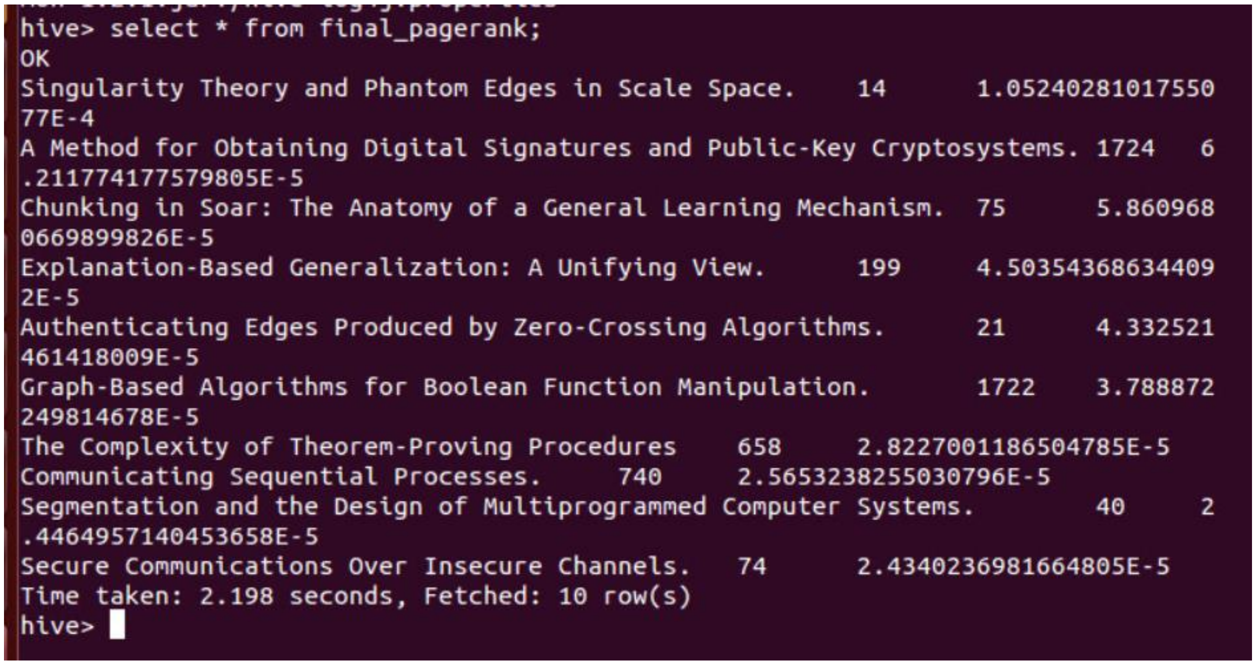
Go back to dir listing
Advanced view/download options

View Next chunk

```
370     770472  Parallelizing Compilers.
370     832291  Parallelizing Compilers.
370     3006707 Parallelizing Compilers.
370     3006149 Parallelizing Compilers.
370     3006258 Parallelizing Compilers.
384     248799  Distributed and Multiprocessor Scheduling.
384     1073239 Distributed and Multiprocessor Scheduling.
384     1075533 Distributed and Multiprocessor Scheduling.
384     1127700 Distributed and Multiprocessor Scheduling.
386     770739  Logic Programming and Constraint Logic Programming.
386     773466  Logic Programming and Constraint Logic Programming.
386     832742  Logic Programming and Constraint Logic Programming.
386     833078  Logic Programming and Constraint Logic Programming.
386     951454  Logic Programming and Constraint Logic Programming.
387     690     Pattern Matching and Text Compression Algorithms.
387     2022    Pattern Matching and Text Compression Algorithms.
387     2375    Pattern Matching and Text Compression Algorithms.
387     2487    Pattern Matching and Text Compression Algorithms.
387     3774    Pattern Matching and Text Compression Algorithms.
387     3006547 Pattern Matching and Text Compression Algorithms.
392     2355    Virtual Memory.
392     832426  Virtual Memory.
392     1117337 Virtual Memory.
392     3003317 Virtual Memory.
394     613063  Distributed File Systems and Distributed Memory.
394     613056  Distributed File Systems and Distributed Memory.
```

Download this file

Hive Output



```
hive> select * from final_pagerank;
OK
Singularity Theory and Phantom Edges in Scale Space.    14      1.05240281017550
77E-4
A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. 1724    6
.211774177579805E-5
Chunking in Soar: The Anatomy of a General Learning Mechanism.  75      5.860968
0669899826E-5
Explanation-Based Generalization: A Unifying View.      199     4.50354368634409
2E-5
Authenticating Edges Produced by Zero-Crossing Algorithms.      21      4.332521
461418009E-5
Graph-Based Algorithms for Boolean Function Manipulation.       1722    3.788872
249814678E-5
The Complexity of Theorem-Proving Procedures    658     2.8227001186504785E-5
Communicating Sequential Processes.     740     2.5653238255030796E-5
Segmentation and the Design of Multiprogrammed Computer Systems.        40      2
.4464957140453658E-5
Secure Communications Over Insecure Channels.   74      2.4340236981664805E-5
Time taken: 2.198 seconds, Fetched: 10 row(s)
hive> 
```

VII. CONCLUSION:

- Page Rank algorithm measures the importance of a page by considering importance of the other pages.
- Weighted PageRank algorithm is the extension to the standard page rank which considers the inlinks and outlinks of a page and distribute the rank score based on the popularity/importance of a page.
- This algorithm is scalable and can be applied to any size of data. But the condition is it must be a linked network.
- In this problem, the top 10 papers with highest page rank are successfully computed.

VIII. REFERENCES:

[1] Lecture Notes, Stanford University,

http://ilpubs.stanford.edu:8090/422/1/1999-66.pdf

[2] Wenpu Xing and Ali Ghorbani,” Weighted PageRank Algorithm” Faculty of Computer Science University of New Brunswick Fredericton, NB, E3B 5A3, Canada

[3] Rashmi Rani and Vinod Jain “Weighted PageRank using the Rank Improvement” Department of Computer Science and Engineering, B.S. Anangpuria. Institute of Technology and Management, Faridabad, India

[4] NeelamDuhan, A. K. Sharma, Komal Kumar Bhatia, “Page Ranking Algorithms: A Survey”. In proceedings of the IEEE International Advanced Computing Conference (IACC), 2009.

[5] Book - Programming Hive, Dean Wampler, Edward Capriolo, Jason Rutherglen