An Analysis of the Web Graph Utilizing Apache Flink

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

This project aims to achieve a comprehensive analysis of the structure of the World-Wide-Web. Therefore, we used the data sets provided by the Web Data Commons project which extracted the web graph from the Common Crawl project. A small example data set was used while implementing algorithms that compute degree, connectivity, PageRank, closeness and betweenness of the nodes of the graph. Graph analyzing tools were used to evaluate the algorithms' validity. The computations then were run on a four node cluster on larger data sets to retrieve results of the whole web graph.

1 Problem Statement

In the last years a vast amount of data has been collected and has been made available to the public in order for everyone to analyze this data and gain knowledge about its structure. One example for such data set is the Hyperlink Graph of the University of Mannheim which has been extracted from the Common Crawl. There are currently two versions of the graph namely the 2012 and 2014 version each covering billions of pages and hyperlinks between those pages. Analyzing this data set may be beneficially for multiple research fields like search algorithms, SPAM detection or graph analysis algorithms.

Our target is to implement the algorithms to compute the statistics of importance such as indegree and outdegree distribution, the PageRank, the closeness and the betweenness centrality. Those statistics rank the nodes from different aspects. Further, we want to analyze the data set with the help of Apache Flink to process huge parts of the data set. Based on the results statements can be made about the structure of the Hyperlink Graph and therefore statements can be formulated about the structure of the World-Wide-Web.

2 Methodology

Centrality is a measure which indicates the importance of a node. The degree of a node is the most common and basic measure in the field of network analysis. The degree centrality counts the number of paths of length 1 that emanate from a node. Another degree centrality measure instead is the PageRank, which simulates a random walk on the graph. The rank gives the probability a surfer reaches the respective page.

The second group of centrality measures is the group of diameter related measures, which count the length between nodes. The closeness for example takes the diameter into account in order to identify a nodes importance. A general definition of closeness is based on the length of the average shortest path between a node and all other nodes in the network.

The third measure is the betweenness which considers the flow. A general definition of the betweenness is the number of node pairs that need to go through a node in order to reach each other taking the minimum number of hops. Figure **??** shows the three groups of measures.

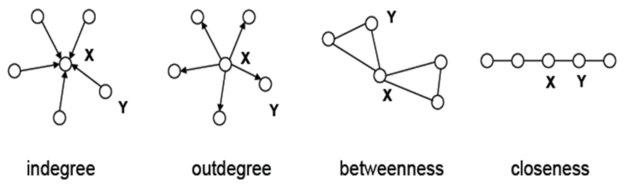


Figure 1: Measures overview

2.1 Degree, indegree and outdegree

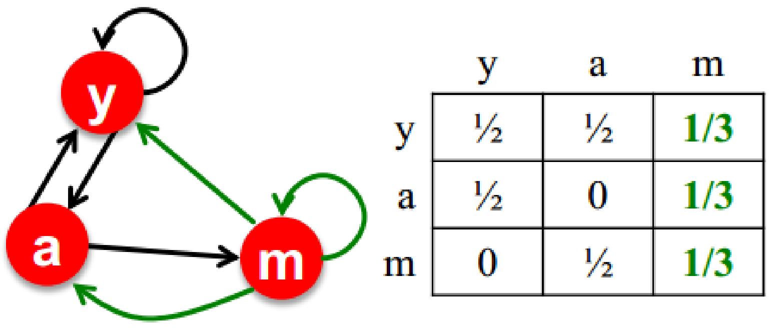
The degree is a fundamental measure within the analysis of networks. It gives some indication of a nodes position in the network. The indegree of a vertex in a directed graph is given by the number of edges pointing to the respective vertex. The outdegree on the contrary gives the number of edges coming from the corresponding node. In order to efficiently compute the both degrees, we build the adjacency matrix of the network and count the neighbors of node with MapReduce.

2.2 Degree distribution

The degree distribution is the probability distribution of the degrees over the whole network. This projects scalable computes them by using the MapReduce model using the degree as key in a key-value pair. The subsequent reduce job builds the sum of these key-value pairs and so builds the degree distribution.

2.3 PageRank

The PageRank algorithm was developed by Google founders Larry Page and Sergej Brin at Stanford University in 1996. The algorithm is used to rank pages in order to measure the importance of websites. Therefore, the algorithm computes a probability distribution representing the likelihood that a user reaches a specific web page by randomly clicking on links. In order to compute the PageRank, we run the algorithm in iterations. To handle circles in the graph, which might prevent the algorithm from converging, we implemented the random teleport behaviour. Nodes without outgoing edges have been modified in a way that these point to all other nodes in the network visualized as green arcs in Figure **??**.



	y	a	m
y	½	½	1/3
a	½	0	1/3
m	0	½	1/3

Figure 2: PageRank sinks

2.4 Closeness centrality

Closeness measures the importance of a node based on the distance to other nodes in the network. The intuition of the large-scale algorithm is to count approximately the number of neighbors that a node connects to at each step of the iterative computation. This approach is proposed by Kang et al.

in their paper Centralities in Large Networks: Algorithms and Observations. Effective Closeness is a large-scale centrality algorithm.

Following Figure **??** shows a toy example showing the intuition. For node 1, at step 1, it connects to two nodes, at step 2, it connects to two nodes and at the final step, it connects one node. Therefore, node 1s count is given by 1 * 2 + 2 * 2 + 3 * 1 = 9. For node 3, the count is 3 * 1 + 2 * 2 = 7. It is observable that node 3 is more central than node 1. The closeness supports this since node 3s count is less than node 1s count. Note that the closeness does not consider the direction of edges.

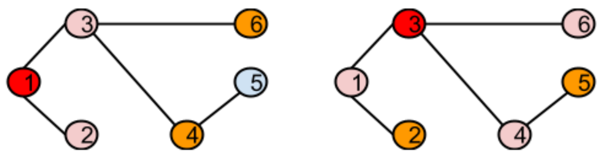


Figure 3: Closeness

The other scalable skill applied is the count distinct elements skill in data stream which is called Flajolet-Martin (FM). When observing a stream of random integers, see an integer which binary representation starts with limited buckets, there is a higher chance that the cardinality of the stream is 2ˆ(size of the limited buckets). For instance, 25% of bits start with ”01”, 12,5% starts with ”001” and if the limited bucket show ”001”, the estimated cardinality is 8. Thus, effective closeness algorithm uses bitstring to represent nodes and update the next step with bitwise OR.

Originally, the algorithm is designed for undirected graphs, thus, this project firstly transforms the directed Hyperlink Graph into an undirected graph, and then applies the described algorithm on it.

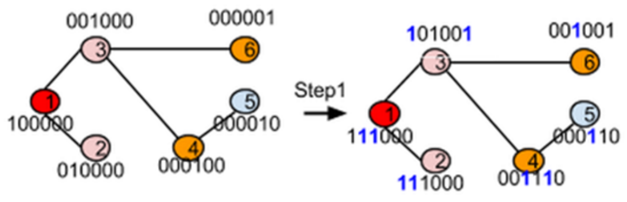


Figure 4: One step of closeness computation

2.5 Betweenness centrality

The betweenness centrality of a node in a network is given by the number of shortest paths from any other node to another node that passes the respective node. It is therefore an indicator of a nodes importance or centrality. To calculate it, the intuition is to score each edge by power iteration and then calculate the importance of nodes by aggregating the edges score. This approach is as well proposed by Kang et al. in the paper Centralities in Large Networks: Algorithms and Observations. For the Hyperlink Graph in this project, the implementation pre-processes the graph by removing dead-ends and transform the directed graph into undirected graph.

The intuition of the betweenness, which is also called LinkRank, is that some edges are important since many nodes have to go through them to reach other nodes. This flow-based measure is similar to the main road and branch approach. Nodes that connect to the main roads are more important. Thus, the scalable calculation of the betweenness uses similar measures to the PageRank way in order to identify more important edges in the network.

Based on the description above, the first step is to transform the graph G into a line graph L(G) where nodes become arcs and edges become nodes, respectively. The scalable idea here is instead of materializing L(G), decomposing L(G) by S(G) x T(G) which are in-edges and out-edges.

The next step in the computation is the calculation of the eigenvector S(G) multiplied by T(G) which is the same as the PageRank which is given by the stationary probability. After finishing the

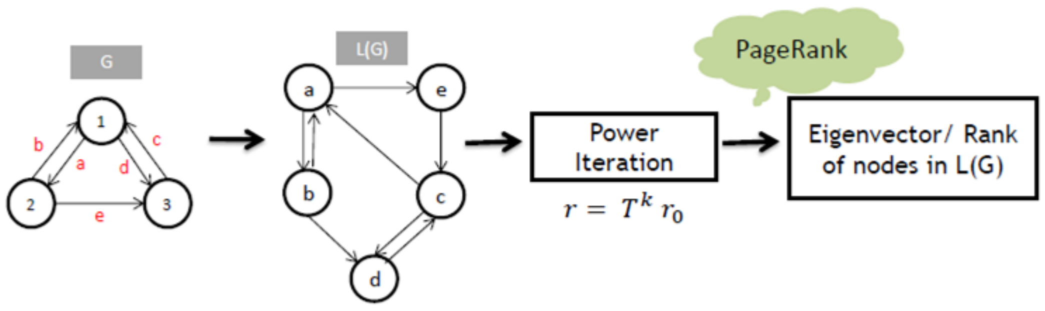


Figure 5: Betweenness

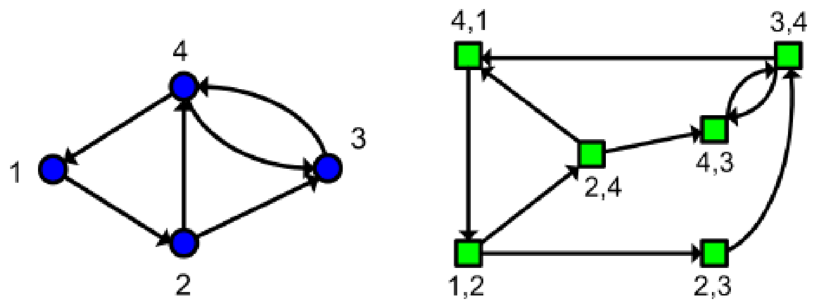


Figure 6: Graph G and its line graph L(G)

computation of the arcs importance score, the scores are summed up to retrieve the final LineRank score for each node T.

3 Experiments

3.1 Apache Flink

Apache Flink is a large-scale data processing engine optimal to process large amounts of data. It offers APIs for Java, Scala and hadoop MapReduce as well as various APIs to access data. Flink programs can be run locally on a single machine or on a cluster of multiple nodes. When ran on cluster, load distribution and fault tolerance are handled by Apache Flink independently with few configuration effort. Flink enables processing huge amount of data while offering an easy to use API for programmers to implement algorithms.

3.2 Web Data Commons

The Web Data Commons is project of the University of Mannheim which is supported by the European Union, Amazon Web Services in Education Grant Award and by the German Research Foundation (DFG). The project offers data sets of the web graph to the public. The data sets were extracted from the Common Crawl Foundation which provides a web corpus to the public.

3.3 Data sets

There are multiple data sets offered by the Web Data Commons project differed by year and aggregation level. They offer the Hyperlink Graph 2012 and Hyperlink Graph 2014. Due to different crawling strategies which were used to gather the web corpora, the authors suggest to use the Hyperlink Graph 2012 for comprehensive network analysis of the web graph.

Table 1: S(G) and T(G)

S(G)			T(G)		
	source			target	
arc1	1	1	arc1	2	1
arc2	2	1	arc2	3	1
arc3	2	1	arc3	4	1
arc4	3	1	arc4	4	1
arc5	4	1	arc5	1	1

Table 2: Number of Nodes and Arcs

Data Set	#Nodes	#Arcs
Page Graph	1,727M	64,422M
Subdomain/ Host Graph	22M	123M
PLD Graph	13M	56M

For the Hyperlink Graph 2012 there are the following four different aggregation levels. Page Graph, Subdomain/Host Graph, 1st Subdomain Graph and Pay-Level Domain Graph (PLD).

The Page-Level Graph represents every web page with all details as single node in the graph. An example for a node in this graph would be: dima.tu-berlin.de/menue/database_ systems_ and_ information_ management_ group/

The Host Graph aggregates the Page Graph by the subdomains and hosts. Therefore, each subdomain is represented as node within the Host Graph. The two pages tu-berlin.de and dima.tu-berlin.de are two different nodes within this graph.

The PLD reduces the Host Graph by merging the subdomains with their host. The two nodes tu-berlin.de and dima.tu-berlin.de are represented in the PLD as a single node tu-berlin.de.

It is obvious that the size of the graphs decreases with the increase of the granularity level. The graphs are separated into two different files, namely an index file and an arc file. The index file consists of tuples which hold an identifier and the node name. The arc file gives tuples of two identifiers representing a link from one node to another. An overview of the different sizes is given by the following table.

4 Computation

We initially planned to run the computations of the algorithms on a ten node cluster. Therefore, we initially intended to use the Host Graph as data set to achieve a comprehensive analysis of the web graph. Unfortunately, due to organizational problems we were not able to run the computation on the cluster.

As a fallback plan, we decided to run the computation locally on our machines. Very first steps however, revealed that computing these amount of data on a single machine is not feasible, since our machines do not have enough memory. Hence, we reduced the data set to the PLD. This reduction allowed us to compute the indegree distribution and the Top-K outdegree of the PLD. These results were achieved on a machine with 3GB RAM and 2 processors with 2 cores. The computation took around 40 minutes.

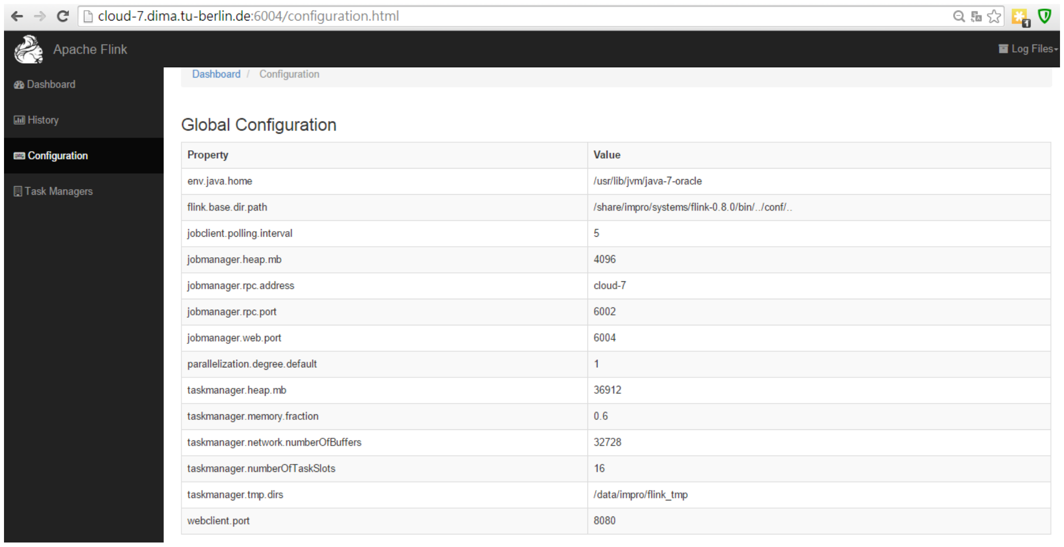
Other implemented algorithms such as the PageRank, closeness and betweenness could not be executed since our machines ran out of memory during the computation.

During the course of the project we were able to run the algorithms on a four node cluster. A detailed view on the configuration of the four node cluster can be seen in Figure **??**. This time we used the Host Graph as well as the PLD Graph as fallback. The computation of the indegree and outdegree was successful on the four node cluster. For both graph levels we were able to retrieve

Table 3: Computation overviews

	Example graph (locally)	PLD graph	Subdomain-Host graph
Degree	Correct	Correct	Correct
Connectivity	Correct	Memory issue	Memory issue
PageRank	Correct	Memory issue	Memory issue
Closeness	Correct	Memory issue	Memory issue
Betweenness	Correlation is 0	Not run in cluster	Not run in cluster

results. Nevertheless, the other algorithms to measure the connectivity, PageRank and closeness of the graph faced memory issues on the four node cluster.


cloud-7.dima.tu-berlin.de:6004/configuration.html
Apache Flink
Log Files
Dashboard
History
Configuration
Task Managers
Dashboard / Configuration

Global Configuration

Property	Value
env.java.home	/usr/lib/jvm/java-7-oracle
flink.base.dir.path	/share/impro/systems/flink-0.8.0/bin/../conf/..
jobclient.polling.interval	5
jobmanager.heap.mb	4096
jobmanager.rpc.address	cloud-7
jobmanager.rpc.port	6002
jobmanager.web.port	6004
parallelization.degree.default	1
taskmanager.heap.mb	36912
taskmanager.memory.fraction	0.6
taskmanager.network.numberOfBuffers	32728
taskmanager.numberOfTaskSlots	16
taskmanager.tmp.dirs	/data/impro/flink_tmp
webclient.port	8080

Figure 7: Configuration of the 4-node cluster

4.1 Evaluation

To rapidly test our implementations of the different algorithms we used an example data set provided by the Web Data Commons project. This data set contains 106 nodes and 141 arcs. The Web Data Commons project provides results for the indegree and outdegree (see Figure 7 and 8). A comparison to these results show that our implementation in respect to the indegree and outdegree is correct and computes the expected results.

Further, we used the graph visualization tool Gephi which also provides computation functionalities for the measures degree, PageRank and closeness of graph. We compared our results to the results of Gephi and computed the Pearson correlation between the two result sets. The correlation coefficient of the two different PageRank results is 1 (see Figure **??**). The correlation coefficient for the Closeness results is 0.985 (see Figure **??**). This shows that our implementation of the algorithms computes correct results.

4.2 Results

The results of our computation are very limited due to the lack of computation power. We retrieved results from the computation of the indegree and outdegree which can be seen in Figure **??** and Figure **??**. Since these are our only results, we try to gain as much information from it as we can. Nevertheless, the results of our indegree distribution show the power-law. Yet, tt is observable that the, whether indegree distribution nor the outdegree distribution do not follow a strict linear

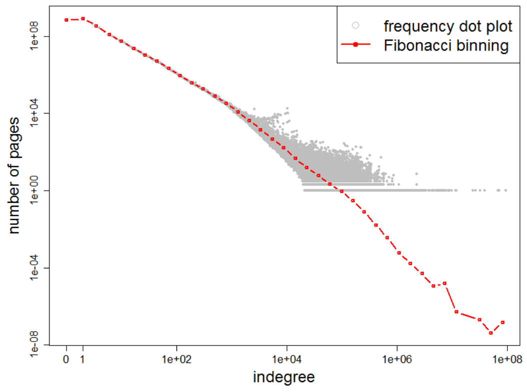


Figure 8: Indegree Distribution

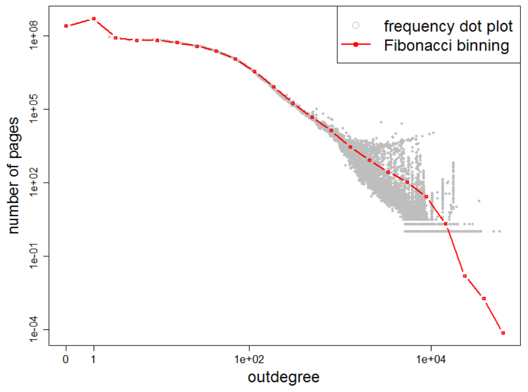


Figure 9: Outdegree Distribution

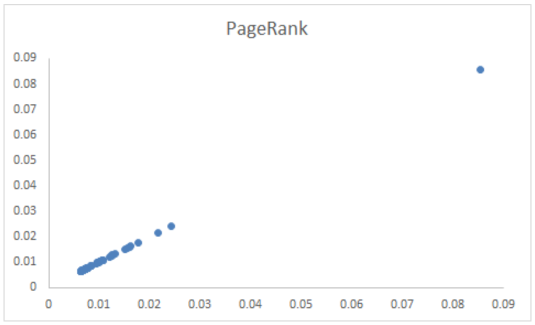


Figure 10: Correlation our results Gephis results (PageRank)

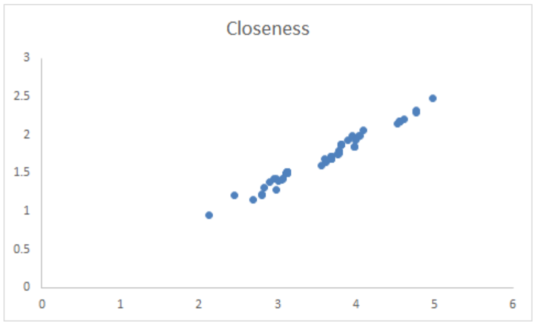


Figure 11: Correlation our results Gephis results (Closeness)

regression, therefore, we applied a regression ANOVA to confirm or reject the significance of a power-law.

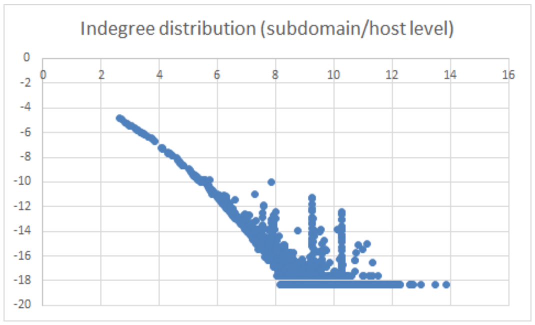


Figure 12: Indegree distribution

The regression ANOVA (see Figure **??**) reveals that the p-value is significant which is 0. However, one of the residual plots, which is the relationship between fitted value (predicted value) and residual, indicates the line is not a regression line (see Figure **??**). The reason there is that the residuals have patterns, which violates the assumption of regression that the residuals must have no patterns. Besides, the data also does not follow the assumption that should be normal distribution based on the probability graph. In conclusion, the power-law is not significant. We can therefore confirm the results of other research [4].

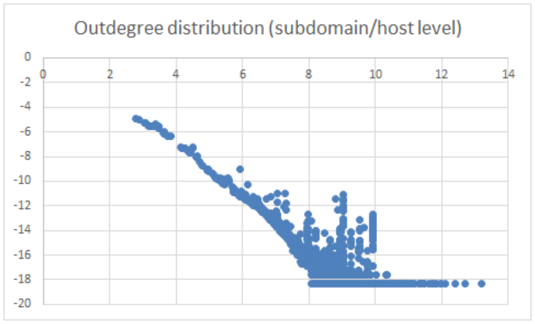


Figure 13: Outdegree distribution

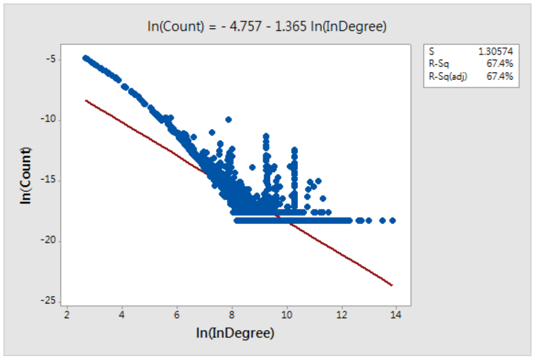


Figure 14: Regression ANOVA

The results of our Top-10 indegree computation can be seen in Table 4. Further, see Table 5 for the results of our Top-10 outdegree computation. These results are also confirmed by the Web Data Commons project [1].

5 Conclusion

The goals of this project have been partly achieved. The implementation of the algorithms in order to compute the different measures have been fully completed. An evaluation of these algorithms showed that the implementations produce correct results. For this evaluation an example data set was used to rapidly test the algorithms. Due to lack of computation power, we reduced the granularity from the Host Graph to the PLD. A comprehensive network analysis of the Web Graph was only limited feasible. However, a few results could be obtained from the computation. These information were gained previously in other projects. Those results could be partly confirmed by our results.

The developed algorithms implemented with the help of the Java API of Apache Flink can be used to retrieve the measures in other networks. Additional optimizations of the implementations can be achieved. The computations can be also run on a multiple node cluster to gain more results.

References

[1] Kang, U., Papadimitriou, S., Sun, J., Watson, T., & Tong, H. (2011). Centralities in large networks: Algorithms and observations. [2] WDC - Download the 2012 Hyperlink Graph. URL: http://webdatacommons.org/hyperlinkgraph/2012-08/download.html Last accessed: 10th of February 2015 [3] Chakkaradhari, J., Schelter S., Markl, V. (2014). Large Scale Centrality Measures in Apache Flink and Apache Giraph. [4] Meusel, R., Vigna, S., Bizer, C. & Lehmberg, O. (2014). Graph Structure in the Web Revisited

[1]Topology of the 2012 WDC Hyperlink Graph. URL: http://webdatacommons.org/hyperlinkgraph/2012-08/topology.html#toc9 Last accessed: 10th of February 2015

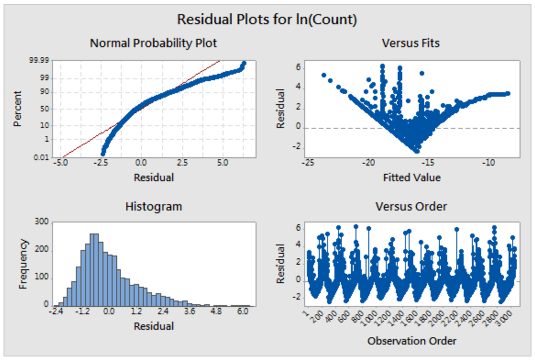


Figure 15: Residual Plots

Table 4: Top-10 Indegree (Host Graph)

Website	Indegree
wordpress.org	2,335,856
youtube.com	2,073,535
gmpg.org	1,784,793
en.wikipedia.org	1,545,864
twitter.com	1,036,611
google.com	798,348
rtalabel.org	657,414
wordpress.com	646,766
mp3shake.com	549,122
w3schools.com	507,184

Table 5: Outdegree and Page views

Website	Outdegree	Page views
serebella.com	699609	3
tumblr.com	496045	7.18
blogspot.com	3898561	3.21
wordpress.com	2249553	4.71
refertus.info	668271	1
typepad.com	551360	1.89
botw.org	496645	2.82
top20directory.com	650884	1.3
wikipedia.org	862705	3.53
youtube.com	1078938	6.08