Web Graph Similarity for Anomaly Detection

Aritra Ghosh[*]
Indian Institute of Technology, Madras

Pallavi Gudipati[†]
Indian Institute of Technology, Madras

ABSTRACT

With massive increase in the data being generated each day, we need important automated tools to oversee the evolution of the web and to quantify global effects like PageRank of webpages. Central to this problem is the notion of Graph Similarity, which validates how well search engines secure content from web and the quality of their search results. In this paper, we propose extensions to the Signature Similarity method, which has limitations when anomalies occur due to wrong DNS records etc . Empirical evaluation and comparison with present techniques show significant improvement on real time webgraph.

Keywords
Anomaly Detection, Graph Similarity, SimHash, PageRank

1. INTRODUCTION

Web graphs are approximate snapshots of the web. Search engines crawl the Web on daily basis to create web graphs which are typically very large($\sim$100B vertices, $\sim$1T edges). Anomalies in such graphs are defined as *"factors that may result in web graphs with poor Web represenation"*. These anomalies in general refer either to failures of web hosts that do not allow the crawler to access their content, or to hardware/software problems in the search engine infrastructure that can corrupt parts of the crawled data.

Such anomalies have a very noteworthy impact on the search results returned to user queries. For example, they can affect ranking and are therefore are extremely crucial to detect. Checking the validity of a web graph(used in anomaly detection) requires a notion of web graph similarity which helps measure the amount and significance of changes in consecutive web graphs, typically modelled in time series analysis. Any similarity metric proposed brings about the problems of Scalability, Sensitivity and Coverage given the huge amount of data in Webgraphs. Another challenge in this approach of anomaly detection is to define similarity metrics that can be computed in reasonable time and in an automated way and be able to detect anomalies that can be realised in practice.

[*]**Arita Ghosh**
CS11B062
E-Mail: aritrag94@gmail.com
[†]**Pallavi Gudipati**
CS11B044
E-Mail: pallavigudipati@gmail.com

2. BACKGROUND

2.1 Webgraph

A (host level) webgraph is a directed weighted graph whose vertices correspond to active hosts of the web and whose weighted edges aggregate the hyperlinks of webpages in these hosts [5]. We represent the Webgraph as $G = \{V, E\}$ where V is the set of hosts and E is the set of aggregated hyperlinks. In our case, the weight $w(u, v)$ is equal to the number of hyperlinks from pages of u to pages of v.

2.2 Anomaly

An anomaly occurs when the stored graph representation does not reflect the topology and the properties of the actual graph at crawl time [5]. It could be caused by either the search engine infrastructure, like hardware failures, crawler bugs and webgraph management code bugs, or from issues in public internet infrastructure like host failures. It would be infeasible to deal with each of these problems in isolation as it would be very expensive and has no guaranteed overall solution. Thus we focus our efforts primarily on the effects of the anomalies rather than their causes, although we do characterize and differentiate between commonly occurring anomalies and analyze their causes.

2.3 Challenges in the Approach

The challenges in this approach are manifold. The first challenge of this approach lies in the designing of similarity metrics that are computable in reasonable time and scalable to large graphs. They should also be useful in detecting the types of anomalies experienced in practice.For example a metric too sensitive to differences would result in too many false positives. Similarly, a metric which is not sensitive enough would result in too many false negatives. An added challenge is the problem of selecting thresholds which helps us in characterizing very high and very low similarities.

Although there can be anomalies arising from various sources,

Table 1: Statistics of *stanford.edu* Webgraph

Nodes	281903
Edges	2312497
Average Clustering Coefficient	0.5976
Diameter	674

as a part of this project we focus only on anomalies generated from the Webgraph component.

3. PROBLEM STATEMENT

Given a sequence of web graphs $G_1, \dots, G_n$ built consecutively, we quantify the changes from one web graph to the next by computing multiple similarity scores between two consecutive web graphs. We can detect anomalies by comparing these similarity scores against their respective thresholds. We can also detect them by looking for unusual patterns in the time series created by the similarity scores as viewed along the time axis.

We propose extension to the set of similarity measures and to identify other types of anomalies. We also analyse some of the pros and cons of the existing methods.

We should note that the similarity measures we define should satisfy certain criteria.

- **Sensitive:** As discussed earlier, the function defined should be more sensitive to changes in high-quality vertices and edges.
- **Coverage:** The similarity function defined should be sensitive to changes in topology and different properties of the graph.

4. DATASETS

4.1 The *.uk* DELIS series

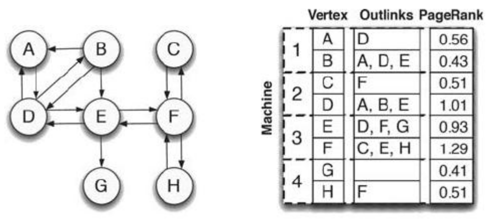
We were planning on using the snapshots from the *.uk* DELIS dataset. However the compressed versions of these graphs were very hard to work with. The graphs were static and introducing anomalies proved difficult in terms of time as well as coding effort. We tried using *WebGraph* (webgraph.di.unimi.it) library, however it did not simplify our effort. Thus, we switched to a new dataset.

4.2 Stanford Web

For this project, we used a dataset from the Stanford's SNAP Library. The dataset corresponds to pages from Stanford University (www.stanford.edu). Some of the statistics of the dataset are given in Table-1.

5. ANOMALIES AND DATASET GENERATION

To check the baseline techniques, we had to simulate various kinds of anomalies that can occur in Webgraphs. In this section, we describe the different anomalies that we are considering and the techniques used to simulate them. For each type of anomaly, we have a generated an anomalous Webgraphs that differ from the original Webgraph in varying degrees. We have generated Webgraphs where the vertices



Machine	Vertex	Outlinks	PageRank
1	A	D	0.56
	B	A, D, E	0.43
2	C	F	0.51
	D	A, B, E	1.01
3	E	D, F, G	0.93
	F	C, E, H	1.29
4	G		0.41
	H	F	0.51

(a) Tiny Web Graph. (b) Representation.

Figure 1: A tiny web graph and its column-wise representation

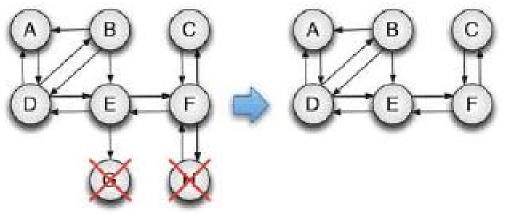


Figure 2: Missing Vertices

are affected with the following probabilities: 0.01, 0.05, 0.1, 0.2, 0.5 and 0.8.

5.1 Missing Random Vertices

- **Cause:** This anomaly usually occurs when one of the machines on which the data is stored fails. Suppose in Fig.1, if machine 4 fails, we will lose data of nodes G and H.
- **Anomaly:** In this anomaly, random vertices dissappear from the graph. The nodes that are lost will depend on the pattern according to which data is stored in the machines. Here we assume that the data is stored randomly.
- **Simulation:** By taking the above stated assumption into account, we generated datasets by randomly knocking off vertices and their corresponding edges. The dataset generated using this technique will henceforth be referred to as ***MRVD***.

5.2 Connectivity Change

- **Cause:** This anomaly occurs due to bugs in code, crawler errors etc. An example of a crawler error is when the crawler misinterprets the hyperlinks of some web pages because of error encoding assumptions. Suppose in Fig.3, the crawler might fail to discover the edges DA and EG and ends up inventing two non-existent edges FG and GH .
- **Anomaly:** In this anomaly, the scalar properties of the graph (total number of nodes, total number of edges etc.) remain constant, but the neighbors of some vertices change.
- **Simulation:** To simulate this anomaly, random nodes were picked, and the edge to one of its neighbor is

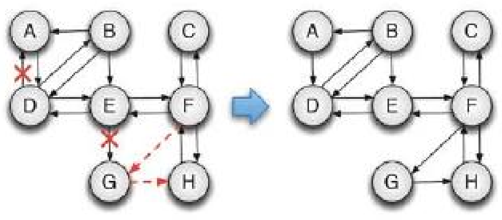


Figure 3: Connectivity Change

transferred to a pair of random nodes. The dataset generated using this technique will henceforth be referred to as ***CCD***.

5.3 Vertex Label Exchange

- **Cause:** This anomaly usually occurs due to wrong DNS records [1]. For example, at crawling time, the DNS record for *Ycorp.com* pointed to the Website of *Xcorp.com* and vice-versa. This DNS misconfiguration can be accidental, or malicious.
- **Anomaly:** In this anomaly, the scalar properties of the graph remain the same, but the labels of the vertices are exchanged. Here we assume that the exchange takes place between vertices with similar PageRank score. This is assumed to bring the weaknesses in *Signature Similarity*(which will be explained in further sections) to light.
- **Simulation:** To simulate this anomaly, nodes with similar PageRank scores were chosen and their vertex labels were swapped. The dataset generated using this technique will henceforth be referred to as ***VLED***.

5.4 Missing Random Connections

- **Cause:** This anomaly usually occurs when a web crawler receives responses with incomplete headers [1]. For example, the response can indicate a redirection, but can lack the destination URL. This can occur due to misconfigured software or misconfigured firewalls.
- **Anomaly:** In this anomaly, some of the edges dissappear from the graph. Here we assume that the edges that dissapear belong to the randomly selected nodes. This is assumed to bring out the worst case and to highlight the flaws in the existing similarity measures.
- **Simulation:** To simulate this anomaly, edges between randomly chosen edges are chosen and deleted. The dataset generated using this technique will henceforth be referred to as ***MRCD***.

6. BASELINE TECHNIQUES

We are using two baseline techniques to evaluate our work.

6.1 Vertex/Edge Overlap

It is based on the idea that *"Two graphs are similar if they share many vertices and/or edges"*. As is evident, this is not in line with sensitivity and coverage requirements. Inspite of this fact, it is seen that this works well in some situations. Here we derive ideas from Edit distance and Jaccard distance. We define the similarity of two graphs G and G' as following:

$$sim_{VEO}(G,G') = 2\frac{|V \cap V'| + |E \cap E'|}{|V + |V'| + |E| + |E'|} \quad (1)$$

where the notation is as has been defined.

This can be computed by scanning V vertices and checking if they occur in V' . Similarly, if the edge list can be scanned for both the graphs the time taken is linear. On the whole, the similarity computation takes $O(|V| + |V'| + |E| + |E'|)$ time.

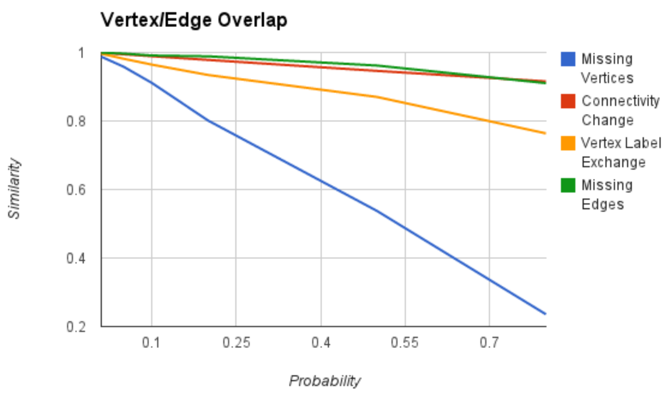


Figure 4: Vertex/Edge Overlap

Table 2: Vertex/Edge Overlap

prob	MRVD	CCD	VLED	MRCD
0.01	0.987	0.998	0.996	0.999
0.05	0.958	0.994	0.981	0.996
0.1	0.910	0.989	0.964	0.991
0.2	0.801	0.978	0.934	0.988
0.5	0.537	0.946	0.870	0.961
0.8	0.235	0.915	0.763	0.910

6.2 Vertex Ranking

This is based on the premise that *"Two graphs are similar if the rankings of their vertices are similar"*. In this method, the vertices are ranked using their quality, the similarity of rankings is computed using a rank-correlation method.

We use a version similar to the Spearman's ρ used for comparing two ranked lists that are permutations of each other i.e

$$\rho = 1 - \frac{2\sum_i d_i^2}{n(n^2-1)/3} \in [-1, 1] \quad (2)$$

This does not involve any weights. We would like the rank correlation to be sensitive to quality. Additionally, we need to modify this so that it is applicable when the lists are not of the same size. Thus we first define the Vertex Ranking

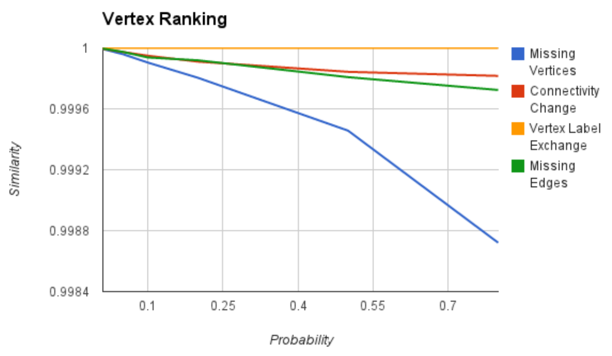


Figure 5: Vertex Ranking

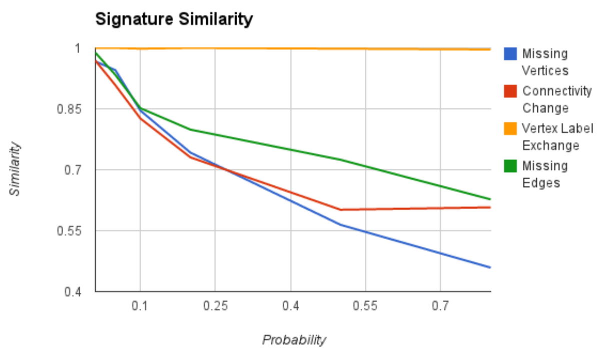


Figure 6: Signature Similarity

Table 3: Vertex Ranking

prob	MRVD	CCD	VLED	MRCD
0.01	0.999	0.999	0.999	0.999
0.05	0.999	0.999	0.999	0.999
0.1	0.999	0.999	0.999	0.999
0.2	0.999	0.999	0.999	0.999
0.5	0.999	0.999	0.999	0.999
0.8	0.998	0.999	0.999	0.999

Table 4: Signature Similarity

prob	MRVD	CCD	VLED	MRCD
0.01	0.966	0.968	1.000	0.988
0.05	0.945	0.908	1.00	0.933
0.1	0.845	0.826	0.998	0.851
0.2	0.742	0.730	1.00	0.798
0.5	0.564	0.601	0.998	0.724
0.8	0.458	0.607	0.996	0.626

Similarity measure as following:

$$sim_V R(G, G') = 1 - \frac{2\sum_{u \in V \cup V'} w_v(\pi_v - \pi_{v'})^2}{D} \quad (3)$$

where π_v and π_v' are the ranks of v in sorted list for G and G' respectively, w_v is the quality of v, and D is a normalization factor which limits the value. Firstly, we note that the similarity values lie between 0 and 1. Next we deal with the more important problem, i.e handling of the case when v does not exist in one of the graphs. The resolution strategy is as follows: If the vertex exists in both the graphs, we use the average quality. Otherwise we use the quality in whichever graph the node exists. Further, if a vertex does not exist in one graph ($G = (V, E)$), we take its rank to be $|V| + 1$. The other case is handled similarly.

Assuming the PageRanks(quality) have been pre-computed, the Vertex Ranking similarity can be computed with one scan of the vertex files and the expected running time is $O(|V| + |V'|)$. This can be done in linear time even if the vector of vertices for one of the graphs does not fit in memory.

6.3 Signature Similarity

This similarity measure is based on the premise that *"Two object are similar of their signatures are similar"*. In this measure, each graph is represented by a set of features. This set is then reduced in dimension to obtain a signature. These signatures are then compared to gain a measure of their similarity.

The method *SimHash* was applied to comparison of high-dimensional vectors in [2] and to document comparison in [4]. In this algorithm, a document is first tranformed into a set of weighted features, which can be viewed as a muli-dimensional vector. A hash function is used to obatain a digest for each feature. These individual digests are then combined to a single signature of the document. The similarity between two signatures is the percentage of agreeing bits in them. The similarity between two documents L and L' is as follows:

$$sim_{SimHash}(L, L') = 1 - \frac{Hamming(h, h')}{b} \quad (4)$$

where h and h' are the b-bit signatures of L and L' respectively.

To convert a graph into a set of features, all the vertices and edges are taken as features. The weight of a vertex is its PageRank and the weight of an edge (u, v) is the PageRank of u normalized by the out-degree of u. Let the function used to perform this transformation be ϕ. Then, the similarity between graphs G and G' is defined as follows:

$$sim_{SS}(G.G') = sim_{SimHash}(\phi(G), \phi(G')) \quad (5)$$

We are using SHA-512 as the hash function [3].

7. IMPROVEMENTS

We made improvements to [5] by introducing more anomalies that can occur in during webgraph crawling. We have also introduced a couple of similarity measures that improve upon Signature Similarity in some aspects.

7.1 Anomalies

We have defined and taken into consideration the following anomalies. They have been described in detail in Section-5.

- Vertex Label Exchange [5.3]

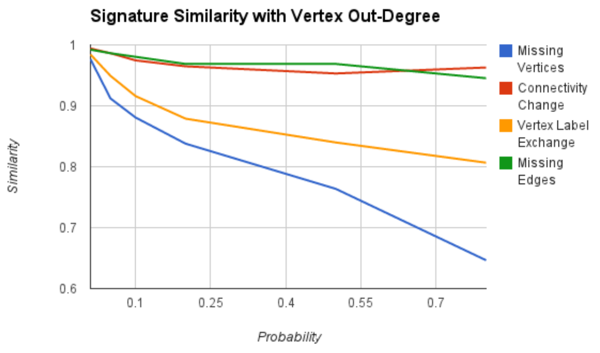


Figure 7: Signature Similarity with Vertex Out-Degree

- Missing Random Connections [5.4]

7.2 Similarity Measures

- **Signature Similarity with Vertex Out-Degree:** While Signature Similarity works well for Missing Vertices Anomaly and Exchanged Connections Anomaly, it fails in detecting Vertex Label Exchange Anomaly. One of the reasons for this failure is the heavy dependence of Signature Similarity on the quality of the vertex(PageRank). Thus, if similarly ranked vertex labels are swapped, the effect is not fully visible. To offset this, this similarity measure also includes some information on the vertex neighborhood structure. One of the simplest representations of neighborhood information is the vertex out-degree.

  To include this new information, we modify the process of converting the given graph to a set of features. Keeping the edge-based features same, we modify the vertex-based features. Previously, weight of a vertex was defined as follows:

  $$Weight_{SS}(u) = PageRank(u)$$

  In our modified version, the weight of a vertex is defined as follows:

  $$Weight_{SSV}(u) = \alpha PageRank(u) + (1-\alpha) OutDegree(u)$$

  where α lies between 0 and 1. α controls the trade-off between vertex quality and vertex neighborhood structure.

  This modified version of Signature Similarity performs better than Signature Similarity for the Vertex Label Exchange Dataset.

- **Signature Similarity with Average Neighbor Out-Degree:** While the above mentioned modified version of Signature Similarity performs well for Vertex Label Exchange Anomaly, it fails for both Connection Change Anomaly as well as Missing Edges Anomaly. But as we can observe, Signature Similarity works well for both of these anomalies. This shows that while PageRank is getting sufficiently affected by the missing edges, the out-degree term is not getting affected and thus bringing up the similarity score. Thus, we need a new representation for neighborhood structure to counter this. A missing edge should bring about more effect in this new representation.

  We take the new representation to be the average out-degree of a vertex's out-neighbors. Thus, now a missing edge effects all its neighbors too. In this new modification, weight of a vertec is defined as follows:

  $$Weight_{SSA}(u) = \alpha PageRank + (1-\alpha) \frac{\sum_{v \in N_u} OutDegree(v)}{|N_u|}$$

  where α lies between 0 and 1, and N_u is the set of all the out-neighbors of u. α controls the trade-off between vertex quality and vertex neighborhood structure.

Table 5: Signature Similarity with Vertex Out-Degree

prob	MRVD	CCD	VLED	MRCD
0.01	0.976	0.994	0.984	0.992
0.05	0.912	0.986	0.949	0.986
0.1	0.880	0.974	0.916	0.980
0.2	0.837	0.964	0.878	0.968
0.5	0.763	0.953	0.839	0.968
0.8	0.646	0.962	0.806	0.945

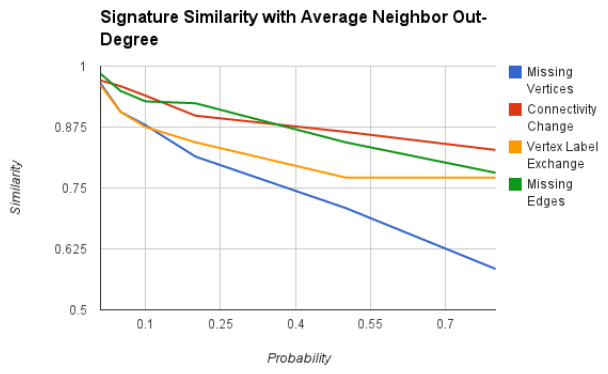


Figure 8: Signature Similarity with Average Neighbor Out-Degree

8. RESULTS

The *prob* field denotes the probability that a given vertex is affected in the anomalous graph.

Vertex/Edge Overlap: This measure works well for MRVD

Table 6: Signature Similarity with Average Neighbor Out-Degree

prob	MRVD	CCD	VLED	MRCD
0.01	0.964	0.970	0.960	0.984
0.05	0.906	0.958	0.906	0.949
0.1	0.878	0.939	0.875	0.927
0.2	0.814	0.898	0.843	0.923
0.5	0.708	0.865	0.771	0.843
0.8	0.583	0.828	0.771	0.781

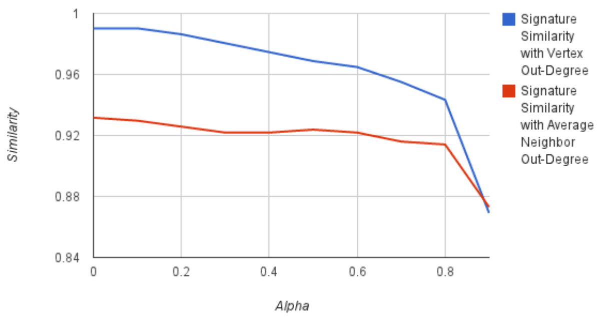


Figure 9: Variation in similarity with α (0.2 dataset)

and reasonably well for VLED, but fails for CCD as well as MRCD.

Vertex Ranking: This measure fails for all the anomalous datasets.

Signature Similarity: This measure works well for MRVD, CCD and MRCD, but fails for VLED.

Signature Similarity with Vertex Out-Degree: This measure works well for MRVD, reasonably well for VLED and fails for CCD and MRCD.

Signature Similarity with Average Neighbor Out-Degree: This measure works well for MRVD, VLED and MRCD, and reasonably well for CCD.

9. CONCLUSIONS

In this paper, we studied the problem of detecting anomalous Webgraphs. Two new types of anomalies: Vertex Label Exchange and Missing Random Connections, which have real life applications were proposed. The present techniques suggested, like Signature Similarity, fail to work for such types of anomalies. Thus, we proposed two modifications to Signature Similarity, which take into account structural properties of the vertex neighbourhood. Experimental results on a real life Webgraph show the effectiveness of the proposed methods.

In this work, it is only possible to label the whole graph as anomalous. In future we intend to extract the anomalous subgraphs, which is a more challenging task. Another aspect that can be pursued is that of detecting the type and extent of different anomalies present.

10. REFERENCES

[1] C. Castillo. Effective web crawling. *SIGIR Forum*, 39(1):55–56, June 2005.

[2] M. S. Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the Thiry-fourth Annual ACM Symposium on Theory of Computing*, STOC '02, pages 380–388, New York, NY, USA, 2002. ACM.

[3] A. Dasdan and P. Papadimitriou. Methods and apparatus for computing graph similarity via signature similarity, June 2009. US Patent App. 11/951,172.

[4] M. Henzinger. Finding near-duplicate web pages: A large-scale evaluation of algorithms. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, pages 284–291, New York, NY, USA, 2006. ACM.

[5] P. Papadimitriou, A. Dasdan, and H. Garcia-Molina. Web graph similarity for anomaly detection. *Journal of Internet Services and Applications*, 1(1):19–30, 2010.