Web Graph Similarity for Anomaly Detection (Poster)

Panagiotis Papadimitriou
Stanford University
Stanford, CA 94305, USA
ppapadim@stanford.edu

Ali Dasdan
Yahoo! Inc.
Sunnyvale, CA 94089, USA
dasdan@yahoo-inc.com

Hector Garcia-Molina
Stanford University
Stanford, CA 94305, USA
hector@cs.stanford.edu

ABSTRACT

Web graphs are approximate snapshots of the web, created by search engines. Their creation is an error-prone procedure that relies on the availability of Internet nodes and the faultless operation of multiple software and hardware units. Checking the validity of a web graph requires a notion of graph similarity. Web graph similarity helps measure the amount and significance of changes in consecutive web graphs. These measurements validate how well search engines acquire content from the web. In this paper we study five similarity schemes: three of them adapted from existing graph similarity measures and two adapted from well-known document and vector similarity methods. We compare and evaluate all five schemes using a sequence of web graphs for Yahoo! and study if the schemes can identify anomalies that may occur due to hardware or other problems.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Search Process

General Terms

Algorithms, Design, Experimentation

Keywords

anomaly detection, graph similarity, web graph LSH

1. INTRODUCTION

Web graphs represent the graph structure of the web and constitute a significant offline component of a search engine. Web graphs are useful in many ways but their main purpose is to compute properties that need a global view of the web. PageRank is one such property, but there may be hundreds of other properties that need a global view. These properties are used during crawling, indexing, and ranking of web pages.

Search engines crawl the Web on a regular basis and recreate web graphs according to the data of the latest crawl. The generated web graphs consist of tens of billions of vertices and hundreds of billion of edges. Since the graph data is massive, it is spread over multiple machines and files. The quality of the web graphs that we obtain from a crawl can be affected by a variety of what we call *anomalies*. These anomalies 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. The detection of such anomalies is critical since they can have a significant impact on the search results returned to user queries, e.g., they can affect ranking.

We address the anomaly detection problem for the web graph component through the calculation of similarities or differences between consecutive snapshots. To make this possible, we propose different web graph similarity metrics and we check experimentally which of them yield similarity values that differentiate a web graph from its version with injected anomalies.

2. POTENTIAL ANOMALIES

In this section we give some examples of the types of anomalies that we are interested in detecting. We can classify these anomalies into the following categories:

Missing Connected Subgraph: A web graph may miss a connected subgraph of the web because of public infrastructure failures or network problems. For example, a web host that is unavailable at crawl time may cause us to miss its content, as well as the content reachable from that host. In this case the resulting web graph misses a connected subgraph that may consist of all the hosts of a specific domain.

Missing Vertices: A web graph may also miss a subgraph that consists of vertices that are not necessarily connected. We can have such corrupted web graphs either because of failures during crawl time or because of machine failures in the cluster where we store the web graph. In the latter case, although we are usually aware of the anomaly we would like to be able to estimate its extent.

Random Topological Changes: The anomalies of this category are results of data corruption or software bugs in the crawler or the graph data management code. For example, if we use column orientation to store the graph vertex attributes, buggy software may map outlinks to source nodes incorrectly. Anomalies of this type are usually the most difficult to detect.

In all the anomalies above, the problems get more serious if the anomalies affect important vertices that have very high PageRank (our quality score) or are major authorities or hubs. If graph similarity is sensitive to the importance of vertices, we hope to detect the problem if we were unaware of it, or we hope to quantify the loss if we were already aware of it.

3. ANOMALY DETECTION

It is very hard to detect certain problems of a web graph simply by examining a single snapshot or instance. For example, how can one tell that an important part of the web is missing? Or that IP addresses were not properly grouped into hosts?

Because of the difficulty of identifying anomalies in a single data snapshot, it is more practical to identify anomalies based on "differences" with previous snapshots. Our approach to detecting anomalies in web graphs can be described as follows. We have a sequence of web graphs, $G_1, \ldots, G_n$ built consecutively. We want to quantify the changes from one web graph to the next. We do this by computing one or more similarity scores between two consecutive web graphs, G_i and G_{i+1}. Similarity scores, when viewed along the time axis, create a time series. Anomalies can be detected by comparing the similarity score of two graphs against some threshold, or by looking for unusual patterns in the time series.

4. SIMILARITY COMPUTATION

The challenge in this approach to anomaly detection is in developing similarity metrics that (a) can be computed in a reasonable time and in an automated way, and (b) are useful for detecting the types of anomalies experienced in practice. A metric that is too sensitive or sensitive to differences that do not impact the quality of search results, will yield too many false positives. Similarly, a metric that is not sensitive enough will yield too many false negatives. There is of course also the challenge of selecting the proper thresholds that tell us when similarities are "too high" or "too low."

In the extended version of this paper [3], we present some candidate metrics and discuss how they can be implemented efficiently. Here we only give a brief description of the main similarity or dissimilarity measures that we use in the definition of each metric. These measures are:

1. The edit distance between graphs, which is equal to the minimum number of edit operations required to transform one graph to another (used in *Vertex/Edge Overlap*).

2. The Euclidean distance between the principal eigenvectors of the graph adjacency matrices (*Vector Similarity*).

3. The Spearman's ρ applied to calculate the rank correlation between sorted lists of vertices of the two graphs. Vertices are sorted based on the quality or some other properties (used in *Vertex Ranking*).

4. The similarity of vertex sequences [1] of the graphs that are obtained through a graph serialization algorithm [3]. Such an algorithm creates a serial ordering of graph vertices which is maximally edge connected. That means that it maximizes the number of vertices that are consecutive in the ordering and are edge connected in the graph (used in *Sequence Similarity*).

5. The Hamming distance between appropriate fingerprints of two graphs. To get such fingerprints we apply a similarity hash function to the compared graphs. We developed an effective hash function with the required properties by converting a graph to a set of weighted features and then applying *SimHash* [2]. The set of features for each graph consists of its vertices and edges, which are appropriately weighted using properties like PageRank (used in *Signature Similarity*).

5. EXPERIMENTAL RESULTS

We performed experiments to evaluate our approach to anomaly detection using the similarity metrics that are briefly presented in Section 4. For our experiments we used real web graphs provided the Yahoo! search engine over a month long period in 2007. We give here a summary of our experiments and their results, while a detailed description is available in the extended version of the paper [3].

We found experimentally how similarity varies over time for the different similarity metrics. This analysis helped us determine the thresholds that identify "significant" changes in a graph and indicate possible anomalies. Then we simulated anomalies like the ones presented in Section 2 to certain graphs. For each graph we checked whether the simulated anomaly affected its similarity score relative to its predecessor graphs so that the anomaly can be detected.

The experimental analysis on real web graphs showed that it is feasible to detect anomalies through similarity computations among consecutive web graphs. The effectiveness of the different similarity metrics varies for anomalies of different types. For example, the use of the Vertex/Edge Overlap allowed us to detect a missing connected graph, but was unsuccessful in detecting arbitrary topological changes in a web graph.

Of the five schemes explored, Signature Similarity was the best at detecting what we consider significant anomalies, while not yielding false positives when the changes were insignificant. Vector Similarity also gave promising results in all the studied cases, while the other three schemes proved to be unsuccessful at detecting anomalies of certain types.

The common feature of the two successful schemes is the similarity calculation based on appropriately weighted web graph features. The proposed algorithms are independent from the weighting schema that is used and, hence, we believe that they can be effective in anomalies that are not studied here.

6. CONCLUSIONS

We defined the web graph similarity problem and proposed various metrics to its solution. We have experimentally shown that they work well on real web graphs in detecting anomalies. Our future work will include research on feature selection and weighting schemas.

7. REFERENCES

[1] A. Broder, S. Glassman, M. Manasse, and G. Zweig. Syntactic clustering of the web. In *Proc. of Int. World Wide Web Conf. (WWW)*, pages 393–404, Apr 1997.

[2] M. Charikar. Similarity estimation techniques from rounding algorithms. In *Proc. of Symp. on Theory of Comput. (STOC)*, pages 380–388. ACM, May 2002.

[3] P. Papadimitriou, A. Dasdan, and H. Garcia-Molina. Web graph similarity for anomaly detection. Technical Report 2008-1, Stanford University, 2008. URL: http://dbpubs.stanford.edu/pub/2008-1.