International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE)
ISSN: 0976-1353 Volume 22 Issue 1 – MAY 2016.

REMOVING DUPLICATE URLS USING DUSTER

Tadepalli Sarada Kiranmayee[#1], S. Sai Vimal Prasath[*2], J. Daniel Samson[*3], K. Mukesh Kumar[*4], I. Monish Niveth[*5]

[#1]Asst Professor,Department Of Computer Science And Engineering, Srm University, Ramapuram, Chennai
*[*2]Student, Department Of Computer Science And Engineering, Srm University, Ramapuram, Chennai*
*[*3]Student, Department Of Computer Science And Engineering, Srm University, Ramapuram, Chennai*
*[*4]Student, Department Of Computer Science And Engineering, Srm University, Ramapuram, Chennai*
*[*5]Student, Department Of Computer Science And Engineering, Srm University, Ramapuram, Chennai*

Tskiranmayee@Gmail.Com
S.Vimalsai@Gmail.Com
Danie.Samson@Gmail.Com
Kmukeshkumar@Yahoo.Co.In
Monish.Niveth@Yahoo.In

***Abstract*— World Wide Web is a commonly used medium to search information using Web crawlers. Some of the information collected by the web crawlers include pages with duplicate contents. Different URLs with Similar Text are known as DUST. To improve the performance of search engines, a new method called DUSTER is proposed. The proposed method converts all the URL into multiple sequence of alignments and removes the duplicate URLs. The proposed method uses normalization rules to convert the duplicate URLs into a single canonical form. Using this method reduction of large number of duplicate URLs is achieved.**

Keywords—Crawlers, Dust, Duster

I. INTRODUCTION

There are different URLs that have similar content on the web. These similar URLs are known as DUST. Since Crawling this duplicate URLs is a waste of resources, the task of detecting DUST is important for a search engine. DUST results in poor user experience. The existing system focused on document content to remove Duplicate URLs. Generation of Dynamic webpages leads to Duplication of contents. In the proposed method, these duplicate URLs are converted into same canonical form which can be used by web crawlers to avoid DUST. The existing system uses random sampling instead of processing all URLs. In the proposed system, multiple sequence alignment is used to obtain a smaller and general set of rules to avoid duplicate URLs. Multiple sequence alignment is used in scientific technologies for identifying identical patterns. This Multiple sequence alignment can be used in identifying similar strings, which can be used for deriving normalization rules. More general rules can be generated using this multiple sequence alignment algorithm to remove the duplicate URLs with similar text.

II. PROBLEM DEFINITION

The goal of the proposed system is to detect dust and remove duplicate URLs. It is achieved by using two algorithms, the algorithm that detects the dust rules from a list and the algorithm that is used to convert these URLs into same canonical form. The contents of a valid URL can be fetched from its web server. If the documents of two URLs are similar, they are called DUST. Similarly identification of near duplicate documents is complex. The content of the near duplicate documents will be similar but there will be small differences in the content. The web pages with same content, but different URLs is the source of multiple problems. A list consists of an URL and a HTTP code. The list type can be obtained from web server logs. The algorithm to detect DUST rules is used to create an ordered list of DUST rules from a website. Canonization is the process of converting every URL into a single canonical form. There is a possibility of efficient canonization of DUST rules detected. Algorithm for optimal canonization cannot be designed. But the algorithm that can convert all the URLs to a small set of canonical form can be used. In the proposed system the URL normalization process identifies DUST without fetching the content of the URLs. Metrics of the proposed system includes the fraction of valid rules, amount of eliminated redundant URLs and the amount of duplicate URLs in the list.

III. PROPOSED SYSTEM

DUST is defined as Duplicate URLs having Similar Text. In this section, we describe a detailed method to avoid the presence of DUST in search engines. The earlier methods used were susceptible to error as they use methods such as random sampling, and generalized rules to detect duplicate URLs. This leads to lots of wasted resources and requirements for large storages for indexing. Needless to say, these methods also produce diluted results for a query.

Here we introduce the method of 'Sequence Alignment' to help overcome these undesired results explained above. Sequence alignment is defined as a method of arranging multiple sequences in order to identify similar parts among them. Thus the alignment of any two sequences in pairs to identify the similarities is explained to as 'Pair-wise Sequence alignment'. Here dynamic programming is used to calculate all the sub-problems involved in the process. The scoring function is frequently used in pair-wise sequence alignment as the scoring function defines the similarities between a pair of symbols.

MULTIPLE SEQUENCE ALIGNMENT:

For any given set of sequences with more than two sequences, multiple sequence alignment is considered as a natural generalization of the pairwise alignment problem. This problem requires that all sequences to be of the same length and thus spaces are inserted at the appropriate places. This method is taken into consideration when more than two big sequences are to be normalized. Here we use progressive alignment strategy that aligns most similar sequences at each stage and infers a new token set of rules from them. This is repeated until all sequences have been aligned and the result is the final multiple sequence alignment.

Definition (Multiple Sequence Alignment):

Let f{S1,S2,S....., Skg} be sequences of symbols, and let |Si| represent the size of Si. The Multiple Sequence Alignment from S1 to Sk is a mapping of f{S1, S2, ..., Skg} to other sequences f{S0, S02,...., S0kg} such that S0 i has the same symbols of Si in the same order with the possibly of the addition of spaces (also known as gaps) and |S01| = |S02| .... = |S0k|

Progressive alignment method is used to align clusters of duplicate URLs as this problem is defined as NP-Hard. Thus various Heuristic methods have been developed to solve this NP-Hard problem. This method starts with the alignment of two sequences at first. It then considers a new sequence which it uses to align with the already aligned sequence. Thus at each iteration, new sequences are added and the result in the end is the alignment of all the sequences. This process has various variations according to the similarities in the sequences. the most similar sequences are aligned first and as the iterations progress, the dissimilar sequences are aligned. Thus determining the best order of alignment sequencing is crucial and highly affects the time complexity.

This method of ideally aligning the most similar sequences first and the most divergent sequences until the end in order to reduce the error introduced by the heuristic function, we introduce the new step of URL alignment.

Algorithm Multiple URL Alignment (C)

```
Input: C = {u1 ...un}
Output: π = (consensus, domains, support)
 1:  σ = (x, y, consensus xy, scoring)
 2:  Domains = Φ; Sequences = Φ; Support = Φ;
     Aligned = Φ;
 3:  for all pairs of distinct URLs u1, u2 in C do
 4:      Support = Support U {(u1, u2)}
 5:      Domains = Domains U {domain (u1) U
          domain (u2)}
 6:      x = tokenize (u1)
 7:      y = tokenize (u2)
 8:      Sequences = Sequences U {x} U {y}
 9:      σ = PairURLAlignment(x, y)
10:      add σ to Q
11:  end for
12:  while Q is not empty do
13:      pop the first tuple σ from Q
14:      if σ.x ∉ Aligned and σ.y ∉ Aligned then
15:          Aligned = Aligned U {σ.x} U {σ.y}
16:          Sequences = Sequences – Aligned
17:          for all sequences s in Sequences do
18:              ℰ = PairURLAlignment (σ.consensus, s)
19:              add ℰ to Q
20:          end for
21:          Sequences = Sequences U {σ.consensus}
22:      end if
23:  end while
24:  Let s be unique consensus sequence in Sequences
25:  return π = (s, Domains, Support)
```

URL MULTI ALIGNMENT ALGORITHM

URL Multi Alignment algorithm is used to align all the URLs in the dup-clusters and produces consensus sequences for every dup-cluster. Then the candidate rules will be generated from the sequences. Heuristics are used to ensure the efficiency of the method for large clusters. It is a tool to identify similarities and differences among strings/sequences. These similarities and differences can be explored to determine fixed and mutable substrings in URLs, which helps to derive normalization rules. Multiple sequence alignment methods detects various patterns involving various strings. This method is able to detect more general rules and avoids pair-wise rule generation related problems, and finding rules across sites problems. Multi-sequence alignment of DUSTs, which is performed before rule generations, can reduce noise and the process remains more stable.

URL ALIGNMENT:

URL alignment is followed as a separate step in this process as it plays an important role in determining the time complexity of the whole process. Here, we create a *consensus sequence* for every training set in the dup cluster and introduce exact rules for them. The consensus sequence is generated only after aligning the URLs in each cluster. This method requires a series of steps that are explained below.

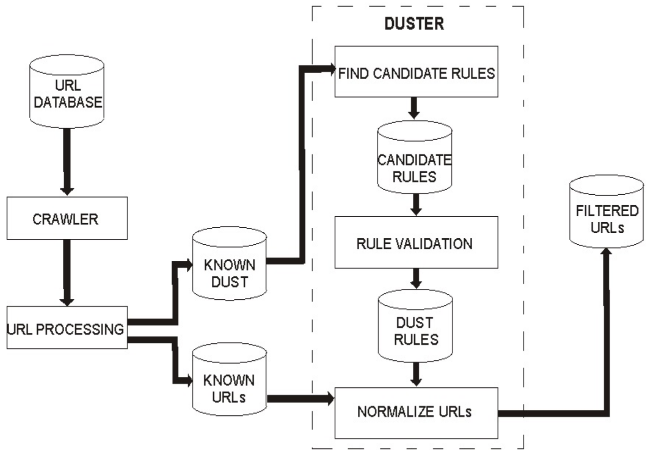
The first step is the URL tokenization. This step, as its name suggests involves the decomposition of the URL into a sequence of URL tokens. Each URL is re broken down and is represented as a singleton set of individual elements. For example the given URL, *s=http://www.google.com/news.html* is represented by the following sequences of 13 token set.
S=<{http},{:},{/},{/},{www},{.},{google},{.},{com},{/},{news},{.},{html}>.

The logical next step is to analyze the consensus that is formed and to group the pairs of the individual elements. This step is called as the pair-wise URL alignment. These individual elements are aligned in pairs and the union of these sets is taken into consideration. Now, when there are multiple URLs to arrange, we use the consensus sequence on entire clusters that are formed by the tokenization of these URLs.

ARCHITECTURE:

The architecture of the system consists of three main modules which are diagrammatically given and explained below. The modules are:

- Internal URLs
- External URLs
- Bad URLs



IN ORDER CRAWLED:

The pages were found within the site. The size is calculated by getting value of the Length of the text of the response text. This is the order in which they were crawled. This module contains the page size, view state size and list of internal URLs.

IN ORDER OF SIZE:

The pages are found within the site. The size is calculated by getting value of the Length of the text of the response text. This is the order in terms of total page size. This module also contains the page size, view state size and list of internal URLs.

EXTERNAL URL:

The pages are link to the outside of the site, a hyperlink on a Web page that points to the web page on a different Web site. On a blog, a link is typically considered external if it points to another blog, even though both blogs are hosted on the same blog site. It will search the link in different pages in a different web site.

BAD URL:

The crawler to find the URL in the overall site and to display all corresponding equaling lists in a web site. Non proper links will be displayed in this Bad URL list. This contains the irresponsive links and proper connectionless links. This links will not be used in our web sites.

IV. CONCLUSION

In this system, a new method called DUSTER is used to detect and eliminate duplicate URLs using multiple sequence alignment. It uses normalization rules to convert various URLs which includes DUST into a same canonical form, from which more general rules can be derived. Using this Multiple sequence alignment algorithm all the URLs are converted into a small set of canonical form from which more general rules can be derived to eliminate duplicate URLs. The DUST detection and canonization are measured using the average of valid rules, the amount of eliminated duplicate URLs and the percentage of Duplicate URLs detected. Studies on DUST rules is necessary for Canonizing URLs. Using this proposed system much greater reduction in Duplicate URLs is achieved. In future, the scalability of the system will be increased by using large dup-clusters. In this system the source and the target are generalized separately. In future generalization of both will be implemented. In this method Clusters include false positives which will be handled in future using new methods. More advanced method will be implemented in future to overcome tail traffic.

REFERENCES

[1] Kaio Rodrigues, Marco Cristo, Edleno S. de Moura, Altigran da Silva, “Removing DUST using Multiple alignment of sequences”, IEEE Transactions on Knowledge and Data Engineering, Vol. 27, NO. 8, August 2015.
[2] Bassma S. Alsulami, Maysoon F. Abulkhair, Fathy E. Eassa, “Near Duplicate Document Detection Survey”, International Journal of Computer Science and Communications Networks, Vol 2(2):147–151.
[3] Kanchan S. Khedkar, P.L. Ramteke, “A Survey on Design and Implementation of Clever Crawler based on DUST Removal”, International Journal for Scientific Research & Development, Vol. 3, No. 10, 2015.
[4] Prof. Sandhya Shinde, Ms. Rutuja Bidkar, Ms. Nisha Deore, Ms. Nikita Salunke, Ms. Neelay Shivsharan, “Detection of Distinct URL and Removing DUST Using Multiple Alignments of Sequences, International Research Journal of Engineering and Technology, Vol. 03, NO. 01, Jan 2016.
[5] Amit Agarwal, Hema Swetha Koppula, Krishna P. Leela, Krishna Prasad Chitrapura, Sachin Garg, Pavan Kumar GM, “URL Normalization for De-duplication of Web Pages”, In proceedings of the 18th ACM conference on Information and knowledge management, CIKM ’09, pages 1987–1990, New York, NY, USA, 2009. ACM.
[6] Jayashri Waman, Prof. Pankaj Agarkar, “Eliminate Duplicate URLs using Multiple Alignment of Sequences”, Multidisciplinary Journal of Research in Engineering and Technology.
[7] Hema Swetha Koppula, Krishna P. Leela, Amit Agarwal, Krishna Prasad, Sachin Garg, Amit Sasturkar, “Learning URL Patterns for Webpage De-duplication”, In proceedings of the third ACM

international conference on Web search and data mining, WSDM ’10, pages 381–390, NewYork, NY, USA, 2010. ACM.

[8] Ziv Bar-Yossef, Idit Keidar, Uri Schonfeld, “Do Not Crawl in the DUST: Different URLs with Similar Text”, ACM Trans. Web, 3(1):3:1–3:31, Jan 2009.

[9] C. L. A. Clarke, N.Craswell, and I. Soboro. Overview of the TREC 2004 Terabyte Track. In Proceedings of the Thirteenth Text Retrieval Conference, Gaithersburg, MD, November 2004. NIST Special Publication 500-261.

[10] T. Lei, R. Cai, J.M. Yang, Y. Ke, X. Fan, and L. Zhang, “A pattern tree – based approach to learning URL normalization rules”, In proceedings of the 19th international conference on World wide web, WWW ’10, pages 611–620, New York, NY, USA, 2010, ACM.

[11] Fetterly, D., Manasse, M., Najork, M., 2004. “Spam, damn spam, and statistics: Using statistical analysis to locate spam web pages”, in: Proceedings of the 7th International Workshop on the Web and Databases.

[12] M. Theobald, J. Siddharth, and A. Paepcke. Spotsigs, “Robust and efficient near duplicate detection in large web collections”, International ACM SIGIR conference on Research and development in information retrieval.