International Journal of Electrical and Computer Engineering (IJECE)
Vol.1, No.2, December 2011, pp. 187~194
ISSN: 2088-8708

An Efficient Approach for Finding Near Duplicate Web pages using Minimum Weight Overlapping Method

Shine N. Das[1], Midhun Mathew[2], Pramod K.Vijayaraghavan[3]
[1]Dept. of Computer Applications, CUSAT, INDIA
[2]Dept. of Comp.Science & Engg., *MBITS, Kothamangalam*, INDIA
[3]Dept. of Computer Applications, CUSAT, INDIA

Abstract

The existence of billions of web data has severely affected the performance and reliability of web search. The presence of near duplicate web pages plays an important role in this performance degradation while integrating data from heterogeneous sources. Web mining faces huge problems due to the existence of such documents. These pages increase the index storage space and thereby increase the serving cost. By introducing efficient methods to detect and remove such documents from the Web not only decreases the computation time but also increases the relevancy of search results. We aim a novel idea for finding near duplicate web pages which can be incorporated in the field of plagiarism detection, spam detection and focused web crawling scenarios. Here we propose an efficient method for finding near duplicates of an input web page, from a huge repository. A TDW matrix based algorithm is proposed with three phases, rendering, filtering and verification, which receives an input web page and a threshold in its first phase, prefix filtering and positional filtering to reduce the size of record set in the second phase and returns an optimal set of near duplicate web pages in the verification phase by using Minimum Weight Overlapping (MWO) method. The experimental results show that our algorithm outperforms in terms of two benchmark measures, precision and recall, and a reduction in the size of competing record set.

Keywords: *near duplicate detection, term document weight matrix, prefix filtering, positional filtering, minimum weight overlapping, web page classification*

1. Introduction

Holocene epoch has detected the enormous emergence of Internet document in the World Wide Web. Internet is now a vital factor for day today life in gathering information and extracting useful information from web pages thus becomes an important task. The performance and reliability of web search engines face huge problems due to the presence of extraordinarily large amount of web data. The voluminous amount of web documents has resulted in problems for search engines leading to the fact that the search results are of less relevance to the user. In addition to this, the presence of duplicate and near duplicate web documents has created an additional overhead for the search engines critically affecting their performance [1]. The demand for integrating data from heterogeneous sources leads to the problem of near duplicate web pages. Near duplicate data bear high similarity to each other, yet they are not bitwise identical [2][3]. "Near duplicates" are documents (web pages) that differ only slightly in content. The difference between these documents can be due to elements of the document that are included but not inherent to the main content of the page. For example, advertisements on web pages, or timestamps of when a page was updated, are both information that is not important to a user when searching for the page, and thus not informative for the search engine when crawling and indexing the page.

The existences of near duplicate web page are due to exact replica of the original site, mirrored site, versioned site, and multiple representations of the same physical object and plagiarized documents. In many situations, two documents that are not exactly indistinguishable may still contain the same content and should be treated as near duplicates. For example, web pages from different mirrored sites may only differ in the header or footnote zones that denote the site URL and update time. Two such documents are identical in terms of content but differ in a small portion of the document such as advertisements, counters and timestamps. These differences are irrelevant for web search [4]. Near duplicate detection has been recognized as an important research problem in recent years. Several applications are benefited by the identification of the near duplicates in the field of plagiarism detection, spam detection and in focused web crawling scenarios. In plagiarism detection, a portion of one document, such as a sentence or a paragraph, is contained in another document, and then these two documents could be seen as near duplicates. Spam messages that belong to the same campaign may look very different because spammers often need to randomize the messages by modifying it with obscure terms or adding unrelated paragraphs to pass the filters [5][3]. However, these spams could be easily discovered by near duplicate detection methods. The determination of near duplicate web pages aids in focused crawling, and ensures enhanced quality and diversity of query results.

In this paper, we propose a novel idea for finding near duplicates of an input web page, from a huge repository. This approach explores the semantic structure, content and context, of a web page rather than the content only approach. The weighting scheme suggested in [6] is considered for creating a term document weight matrix (TDW) which plays an important role in the proposed algorithm. We present a three-stage algorithm which receives

Received Sep 10th, 2011; Revised Nov 13th, 2011; Accepted Nov 18th, 2011

an input record and a threshold value and returns an optimal set of near duplicates. In first phase, rendering phase, all pre-processing are done and a weighting scheme is applied. Then a global ordering is performed along with some standard normalization techniques to form a term document weight matrix. In second phase, filtering phase, two well-known filtering mechanisms, prefix filtering and positional filtering [2], are applied to reduce the size of competing record set and hence to reduce the number of comparisons. In third phase, verification phase, a similarity checking is done by introducing a new technique known as Minimum Weight Overlapping (MWO) based on the threshold value and finally we get an optimal number of near duplicate records.

The rest of this paper is organized as follows. Section 2 describes related works. Section 3 gives the details of proposed work. Section 4 analyses the experimental results in terms of precision and recall. In the last section, we give the conclusion and future works.

2. Related works

Very few research papers have suggested methodologies for near duplicate detection both in general documents and the web documents obtained by web crawling. Technique for the estimation of the degree of similarity among pairs of documents is known as *shingling*. Broder et al. [7] have suggested a technique on this, in which all sequences of adjacent words are extracted. If two documents contain the same shingles set they are treated as equivalent and if the shingles set overlaps, they are considered as exact similar. In this method the authors noted that it does not work well on small documents. Fetterly et al. [8] use five-gram as a shingle and sample 84 shingles for each document. Then the 84 shingles are built into six *super shingles*. The documents having two *super shingles* in common are considered as nearly duplicate documents. Andrei Z. Broder et al. [7] have developed an efficient way to determine the syntactic similarity of files and have applied it to every document on World Wide Web. Using their mechanism, they have built a clustering of all the documents that are syntactically similar.

Another method to detect and delete near duplicated web pages; priority-based on text information, is proposed in [9]. By this method, an algorithm to extract text information of web pages by DOM tree and a priority-based algorithm for detecting near duplicated text information are implemented, so as to reduce the noise of web pages and hence to improve the efficiency of detecting near duplicated text information.

Narayana et al. [10] presented an approach for the detection of near duplicate web pages in web crawling. Near duplicate web pages are detected followed by the storage of crawled web pages in to repositories. The keywords are extracted from crawled pages and on the basis of these keywords; the similarity score between two pages is calculated. The documents are considered as near duplicates if its similarity score satisfies a threshold value. But they haven't exploited the structural information of web page which is more important for getting the semantic information about the page.

This work is an extension of work [11] which offered a novel idea for the detection of near duplicate web pages. It also uses a three stage algorithm in which the similarity verification is based on Singular Value Decomposition (SVD) [12] using angle threshold θ. But SVD requires more complicated Mathematical operations on TDW matrix along with the conversion of Jaccard threshold t into angle threshold θ. It increases the algorithm complexity as well as the practical difficulties in measuring the angle. Here we introduce a new technique called MWO for similarity verification which directly works on Jaccard threshold t and it reduces the complexity of the algorithm.

3. Proposed Work

Here we propose an innovative idea for finding near duplicate web pages of an input record *r*. Similarity verification is done on a huge record set having *n* number of records $\{r_1,r_2,\ldots,r_n\}$ and an optimal set of near duplicate records are returned. The similarity verification is based on a Jaccard threshold value *t*, $0 \le t \le 1$, such that our algorithm returns a set of records having a minimum similarity *t*. $t=0$ returns full record set which shares at least a single word with input record *r*, while $t=1$ return only exactly similar records of *r*. Our objective is to find how to select similar records from the entire record set with a reduced number of comparisons. A three-stage selection algorithm is proposed by combining a different term weighting approach [6], term filtering approach and a new similarity verification technique, *MWO*, for this purpose.

In the first phase, rendering phase, standard web page pre-processing methods like *html* tag removal, tokenization, removal of stop words and stemming are applied on the input record *r* and a feature set *F* of *m* tokens $\{x_1,x_2,\ldots,x_m\}$ are retrieved. Then a standard weighting scheme *W* is proposed, based on the term field where the term *x* is present in a record. Even though a common term is present in two web pages, it not only depends on the number of occurrences in each page, but also depends on the field in which it is present, since a web page is entirely different from a normal text file. For example, if a word is present in the title of one web page and in the content block of another web page, they differ by significance. In the first document, it may be a main feature while in the second one it has got a less importance. Number of occurrences of each token *x* is multiplied with the weight of respective term fields w_i wherever the term is present and added together for total weight of a term W_x.

Table 1. Proposed Weighting Scheme

Term Field	Weight
URL	2
Heading	2
Title	2
Anchor Text – To the same web site	1
Anchor Text – To a different web site	0.5
Keywords	3
Description	3
Main content block	1

Further this weight is normalized based on the total weight of the record, W_r and column sum of TDW matrix using some standard approaches. This is done in perspective of the global ordering to be used in the upcoming stages. Obtained feature set along with its weights is compared with the features of other records present in the repository for measuring the similarity.

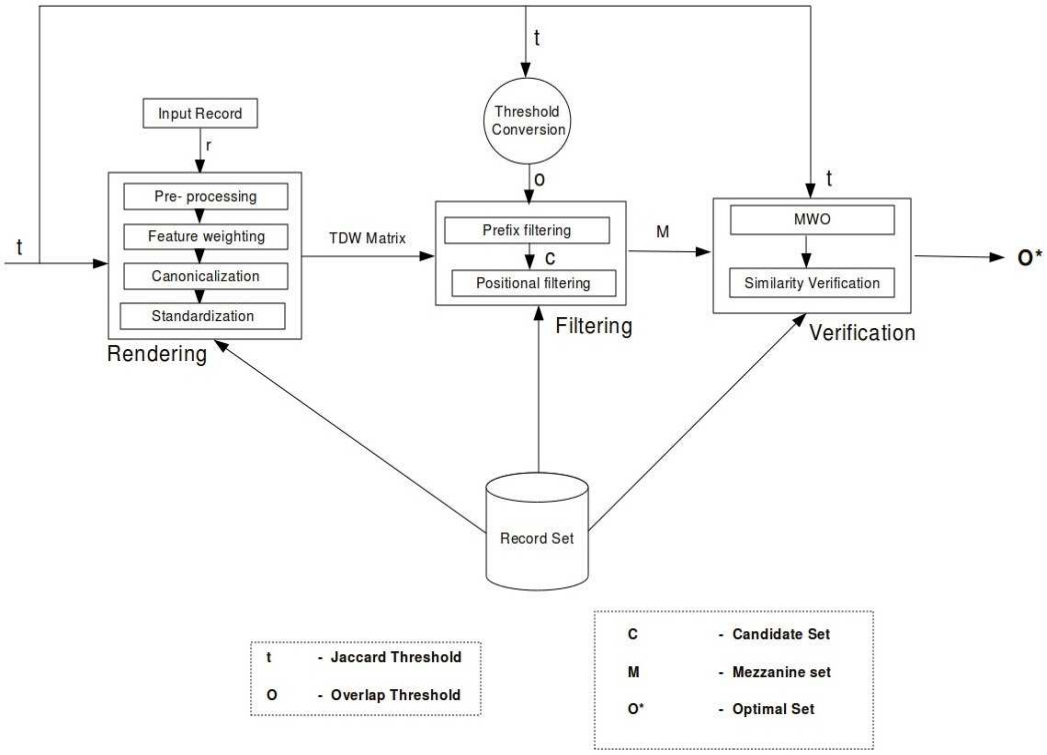

Figure 1. General Architecture

If a term x is present in a record r, its total weight is represented as W_{xr}. Here we apply some filtering mechanisms which further reduce the size of candidate set and improve the efficiency. Prior to this, we have to perform a global ordering based on an inverted index. The document frequency of a token is the number of records that contain the token. We can canonicalize a record by sorting its tokens based on a global ordering. A document frequency ordering arranges the entire tokens according to the increasing order of its document frequency. Instead of simply considering the document frequency, we may use the feature weights for this ordering. The next step is to create a term document weight matrix (TDW) that maps a token x to a list of records that contain x and the total weight percentage of the token in each record. If a particular token is not present in a record, its weight is considered as zero. Each record is represented as per the global ordering.

Let r_1, r_2, r_3 be three canonicalized records. $r_1=\{x_2, x_1, x_3\}$ $r_2=\{x_4, x1, x_3\}$ $r3=\{x_2, x_4, x_1, x_3\}$ The TDW matrix is given in Table 2 where $1 \leq i \leq 4$. In filtering phase, two filtering mechanisms, prefix filtering and positional filtering, are performed to reduce the number of candidate records. Prefix length of r is calculated as**:**

$$Prefix\ length = |r| - \lceil t.|r| \rceil + 1 \qquad (1)$$

by assuming the threshold value t as Jaccard similarity threshold. Each token in prefix set of r is compared with prefix set of all records in the repository and if any record r_i is sharing a token with r in its prefix set, it is added to

candidate set *C*. The basic idea behind prefix filtering principle is that if two web pages share rare tokens, there is a chance that it might be similar. Since we are doing a global ordering based on document frequencies or feature weights, prefix set of a record contain rare tokens or irrelevant tokens. If no tokens are shared in prefix set, that record can be avoided from further processing. Once prefix filtering is over, positional filtering principle is applied in order to prune unwanted records from candidate set *C*.

Table 2. Sample TDW Matrix

records / terms	r_1	r_2	r_3
x_2	$\frac{Wx2r1}{\sum Wxir1}$	0	$\frac{Wx2r3}{\sum Wxir3}$
x_4	0	$\frac{Wx4r2}{\sum Wxir2}$	$\frac{Wx4r3}{\sum Wxir3}$
x_1	$\frac{Wx1r1}{\sum Wxir1}$	$\frac{Wx1r2}{\sum Wxir2}$	$\frac{Wx1r3}{\sum Wxir3}$
x_3	$\frac{Wx3r1}{\sum Wxir1}$	$\frac{Wx3r2}{\sum Wxir2}$	$\frac{Wx3r3}{\sum Wxir3}$

Chuan Xiao et. al.[2] observed that positional information can be exploited in several ways to further reduce the candidate size. Position of each token in a record can be counted, starting from one, which gives information about the upper bounds of overlapping in which Jaccard threshold t can be stated in terms of overlap threshold *O* as

$$J(r,r_i) \geq t \Leftrightarrow O(r,r_i) \geq \frac{t}{1+t}(|r|+|r_i|) \qquad (2)$$

and upper bounds of *O* can be calculated as

$$ubound = 1 + min(|r|-p, |r_i|-q) \qquad (3)$$

where record *r* shares a term at p^{th} position with another record r_i at position *q*. If *ubound* satisfies overlap threshold *O*, record r_i can be added into the *mezzanine set M* from where an *optimal set* is extracted.

Table 3. Calculating MWO

records / terms	r_1	r_2	*Min (W)*
x_2	$\frac{Wx2r1}{\sum Wxir1}$	0	0
x_4	0	$\frac{Wx4r2}{\sum Wxir2}$	0
x_1	$\frac{Wx1r1}{\sum Wxir1}$	$\frac{Wx1r2}{\sum Wxir2}$	$\min(\frac{Wx1r1}{\sum Wxir1}, \frac{Wx1r2}{\sum Wxir2})$
x_3	$\frac{Wx3r1}{\sum Wxir1}$	$\frac{Wx3r2}{\sum Wxir2}$	$\min(\frac{Wx3r1}{\sum Wxir1}, \frac{Wx3r2}{\sum Wxir2})$
Total	**1**	**1**	***MWO* = ∑Min(*W*)**

Based on the records from mezzanine set *M*, we create a weight matrix *A* such that columns represent documents and rows represent terms. An element a_{ij} represents the weight percentage of the global feature x_i in

record r_{j-1} since the first column represents the input record r. In verification phase, each record in M is considered for similarity verification with r. Let r and r_j be the records under consideration. Then its MWO is calculated as the sum of minimum weights of each token in those records. An example calculation of MWO between r_1 and r_2 are shown in Table 3 where $1 \leq i \leq 4$. If MWO satisfies the threshold t, it can be marked as a near duplicate of r and ranked on the basis of MWO.

3.1 Proposed Algorithm

3.1.1 Algorithm: Near_Duplicate_Detection

Input: A Web page (*Web_Document*), records in repository (*Record_Set*) and Jaccard threshold (*t*)
Output: Optimal near duplicate record set, *O**
Remarks: *M* → Mezzanine record set
TDW_Matrix → term document weight matrix

```
Near_Duplicate_Detection (Web_Document, Record_Set, t)
    TDW_Matrix ← Rendering (Web_Document, Record_Set);
    M ← Filtering(TDW_Matrix, Record_Set, t);
    O*← Verification(M, Record_Set, t);
    return O*;
```

3.1.2 Algorithm: Rendering

Input: *Web_Document, Record_Set*
Output: *TDW_Matrix*
Remarks: W_x→ total weight of the term *x*

```
Rendering (Web_Document, Record_Set)
        Input_Record← Pre_Processing(Web_Document);
        F←Full feature set(Input_Record);
        for all x_i ∈ F
                W_x←Weight_Scheme(x_i);
        W_r←∑W_x;
        for all i, 1 ≤ i ≤ |F|
                W_x←Normalize(W_x, W_r);
        T←Dynamic_Thresholding(W_r);
        r← φ;
        for all x_i ∈ F
                if (W_x ≥ T)
                        r ← r ∪ x_i;
        TDW_Matrix← Canonicalize(r, Record_Set);
        TDW_Matrix← Column_Normalize(TDW_Matrix);
        return TDW_Matrix;
```

3.1.3 Algorithm: Filtering

Input: *TDW_Matrix,Record_Set,t*
Output: *M*
Remarks: Assume that *Input_Record* is represented as the first column in *TDW_Matrix*

```
Filtering (TDW_Matrix, Record_Set, t)
    r←TDW_Matrix[1];
    //prefix filtering
    C← φ;
    Prefix_Length← |r|- ⌈t.|r|⌉+1;
    for all r_i ∈ Record_Set
        Prefix_i←|r_i|- ⌈t.|r_i|⌉+1;
        for all j,k; 1≤ j ≤ Prefix_Length, 1≤ k ≤ Prefix_i
                if (r[j] == r_i[k])
```

$C \leftarrow C \cup r_i$;
//positional filtering
$M \leftarrow \varphi$;
for all $r_i \in C$
$O \leftarrow \frac{t}{1+t}(|r|+|r_i|)$;
for all p,q; $1 \le p \le Prefix\_Length$, $1 \le q \le Prefix_i$
if ($r[p]==r_i[q]$)
$ubound \leftarrow 1+ \min(|r|-p, |r_i|-q)$;
if ($ubound \ge O$)
$M \leftarrow M \cup r_i$;
return M;

3.1.4 Algorithm: Verification

Input: M, *Record_Set*, t
Output: O^*

Verification(M, *Record_Set*, t)
$O^* \leftarrow \varphi$;
for each $r_i \in M$
for each feature $x \in Record\_Set$; $1 \le j \le |r|$, $1 \le k \le |r_i|$
$min\_Wx \leftarrow \min\left(\frac{Wxr}{\sum_j Wxjr}, \frac{Wxri}{\sum_k Wxkri}\right)$;
$MWO \leftarrow \sum min\_Wx$;
if ($MWO \ge t$)
$O^* \leftarrow O^* \cup r_i$;
return O^*;

4. Experimental Results

4.1 Data Collection

We have generated a huge repository of web page records from Google search engine. It was done for some specific query words given to Google and collected all similar pages with respect to the first ranked result of Google. Some of these pages were removed due to the lack of required contents retrieved and non-suitable file formats like PDF. Each page thus obtained is pre-processed, featured and weighted according to the weighting scheme and properly indexed to create a TDW matrix. This procedure was repeated for 10 different queries and experiments were conducted on 10 different repositories thus created. Each experiment is resulted an optimal set of near duplicate web pages with respect to the first ranked web page in the query result.

4.2 Experimental Set up

To conduct required experiments, we created an online tool which retrieves the contents of an input web page; perform all pre-processing steps and extract weighted feature set. With respect to the huge repository created earlier, a TDW matrix is formed in a global ordering and filtering principles are applied. The resultant matrix thus obtained is supplied for similarity verification by analyzing MWO of each record with the input record. If it satisfies the threshold value t, that record is marked as a near duplicate one. All these steps were implemented using PHP.

4.3 Result Analysis

To evaluate the degree of accuracy, efficiency and scalability of our algorithm, we have used two standard benchmark measures, precision and recall.

$$Precision = \frac{\text{Number of web pages detected correctly}}{\text{Total number of near\_duplicate web pages detected}}$$

$$Recall = \frac{\text{Number of web pages detected correclty}}{\text{Total number of near\_duplicate web pages}}$$

The measures obtained for 10 different repositories are given by Table 4. While creating a repository of 50 pages, a TDW matrix of almost size 3000 x 50 is created in rendering phase. After filtering phase, its size was reduced to 2500 x 6 when a threshold t=0.5 is applied. And finally, a set of three near duplicates $\{d_1, d_2, d_3\}$ are returned. We could improve recall value without compromising precision value.

Table 4: Performance Measures

Query word	No of near duplicates	Precision %	Recall %
Q_1	167	94.4	94.2
Q_2	98	94.9	94.9
Q_3	156	94.8	93.6
Q_4	123	93.0	93.7
Q_5	79	95.2	95.8
Q_6	129	94.6	94.5
Q_7	63	93.1	94.1
Q_8	112	94.3	95.6
Q_9	109	93.6	94.9
Q_{10}	59	94.7	95.8
Average		**94.3**	**94.7**

All the results obtained during the experiments were having a good level of similarity both in terms of syntax and semantics. Weighting scheme plays an important role in semantic analysis while majority of existing works depend only on simple term frequency based approaches or boolean approaches. Shingle based approaches check only the number of overlapped shingles in the pair which is purely a syntax based one. In our method, by measuring the *MWO* values, we could predict the optimal set of near duplicates instantly and precisely.

5. Conclusion and Future Work

This paper proposed a novel task for finding near duplicate web pages. The proposed technique aims at helping document classification in web content mining based on TDW matrix and MWO method. Instead of simply using the traditional cosine similarity or document frequency for finding similar pages, we proposed a three stage algorithm which reduces the size and enhances the relevancy further. In this paper, we focus a TDW matrix based MWO method that can be applied for generating an optimal near duplicate set. We evaluate the accuracy and scalability of our algorithm in terms of precision and recall and we could achieve a much better recall value without compromising the precision value. The experiments proved that our work has a better performance than other methods.

Further research works can extend this to a more efficient method for finding similarity joins which can be incorporated in focused web crawlers. Accuracy can be improved further by applying this method after identifying the main content blocks rather than the entire web page.

References

[1] Fetterly D, Manasse M, Najork M, On the evolution of clusters of near duplicate Web pages, In Proceedings of the First Latin American Web Congress, pp.37- 45 Nov. 2003.

[2] Chuan Xiao, Wei Wang, Xuemin Lin, Efficient Similarity Joins for Near Duplicate Detection, Proceeding of the 17th international conference on World Wide Web, pp 131 – 140. April 2008.

[3] Gurmeet Singh Manku, Arvind Jain and Anish Das Sarma, Detecting near duplicates for web crawling, In Proceedings of the 16th international conference on World Wide Web, pp. 141 - 150, Banff, Alberta, Canada, 2007.

[4] Dennis Fetterly, Mark Manasse and Marc Najork, Detecting phrase-level duplication on the World Wide Web, In Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, pp.170 - 177, Salvador, Brazil, 2005.

[5] D. Lowd and C. Meek, Good word attacks on statistical spam filters, Second Conference on Email and Anti-Spam, July 2005.

[6] Shine N Das, Midhun Mathew, Pramod K.Vijayaraghavan, An Approach for Optimal Feature Subset Selection using a New Term Weighting Scheme and Mutual Information, Proceeding of the International Conference on Advanced Science, Engineering and Information Technology, Malaysia, 2011, pp 273-278, January 2011.

[7] Broder, A., Glassman, S., Manasse, M., and Zweig G. Syntactic Clustering of the Web, In 6th International World Wide Web Conference, pp: 393-404, 1997.

[8] Fetterly, D., Manasse, M. and Najork, M. On the evolution of clusters of near duplicate web pages, In Proceedings of the first Latin American Web Congress (LAWeb), 37–45, 2003.

[9] Yun Ling, Xiaobo Tao Hexin Lv, A Priority-Based Method Of Near duplicated Text Information Of Web Pages Deletion, IEEE International Conference on Software Engineering and Service Sciences (ICSESS), August 2010.

[10] V.A. Narayana, P. Premchand and A. Govardhan, Effective Detection of Near Duplicate Web Documents in Web Crawling, International Journal of Computational Intelligence Research, Volume 5, Number 1, pp. 83–96, 2009.

[11] Midhun Mathew, Shine N Das, Pramod K.Vijayaraghavan "A Novel Approach for Near-Duplicate Detection of Web Pages using TDW Matrix." *International Journal of Computer Applications 19(7):16-21, April 2011.Published by Foundation of Computer Science*

[12] Shine N Das, K. V. Pramod, Relevancy based Re-ranking of Search Engine Result, Proceedings of International Conference on Mathematical Computing and Management, Kerala, India, June 2010.

Bibliography of authors

Shine N Das received his M.Tech degree from Cochin University of Science and Technology. He is working as an Associate professor in Department of Computer Science & Engineering, College of Engineering Munnar, India. He has more than 9 years of industrial experience and 12 years of teaching experience. He is doing his research in the Department of Computer Applications, Cochin University of Science and Technology under the guidance of Dr. Pramod K Vijayaraghavan. His research interests include web content mining, biologically inspired computing, artificial neural network and machine learning, and e-learning. He has published around 9 peer reviewed research papers in International Journals.

Midhun Mathew B.Tech. degree in Computer Science & Engineering from Cochin University of Science & Technology, India, in 2008 and M.Tech. Degree in Advanced Computing from SASTRA University, India, in 2011. He started his teaching career as a lecturer in College of Engineering Munnar, India, in 2008. Currently he is an assistant professor and a researcher with Mar Baselios Institute of Technology & Science [MBITS], Kerala, India. His research interests include Web mining, Algorithm design, Data structure design, Information Processing, Machine learning and Theoretical computing. He has published about 10 research papers in reputed journals and won several awards.

Dr. Pramod K Vijayaraghavan, presently working as Associate Professor and Head of the Department of Computer Applications, Cocshin University of Science and Technology. He has more than 20 years of research and teaching experience. His areas of Interest are Cryptography and Coding Theory, Simulations and Modelling, Mathematical Morphology and Data mining. He has got more than twenty five research publications in National and International Journal