Template Extraction From Heterogeneous Web Pages

Prof. K. P. Chaudhari
Department of Computer Engineering
Marathwada Institute of Technology,
Aurangabad

Miss. Poonam Rangnath Dholi
Department of Computer Engineering
Marathwada Institute of Technology,
Aurangabad

Abstract

The World Wide Web is huge, rapidly growing and most valuable source of information. In which the most of the information is in the form of unstructured text stored in text database, which makes hard to query the information required. To extract template from these heterogeneous templates, the template detection techniques have received a lot of attention to improve the performance of search engine, clustering and classification of web documents.

In this paper introducing the novel approach to detect and extract the templates from heterogeneous web documents using MDL principle. After that, to provide optimal solution Minhash technique is in used. To provide high recital in terms of time TextMax algorithm is introduced. The template extraction depending on their resemblance of template structure so that the template for each cluster is extracted concurrently. This algorithm provides better performance compared to previous algorithm in terms of space and time.

Keywords: *Template extraction, clustering, Minimum Description Length (MDL) principle, MinHash*

1. Introduction

The World Wide Web (WWW) is getting a lot of attention as it is becoming huge repository of information. WWW is widely used to publish and access information on the internet. In order to achieve high productivity of publishing, the web pages in many websites are automatically populated by using common templates with contents. Web readers easily access to contents by their consistent structures. More and more people are interested to create their own websites for this reason, there have been several different ways to ease out the procedure of website making and its maintenance. The information is in the form of structure and unstructured data.

Structured data on the web are typically data records retrieved from underlying databases and displayed in web pages. Extracting such data records is useful because it enables us to obtain and integrate data from multiple sources (web sites and pages) to provide value-added services, e.g. customizable web information gathering, comparative shopping, meta-search, etc. with more and more companies and organizations disseminating information on the web, the ability to extract such data from web pages is becoming increasingly important. There are several companies working on extracting products sold online, product reviews, job postings, research publications, forum discussions, statistics data tables, news articles, search results, etc.

However, for machines unknown templates are considered harmful because they degrade the accuracy and performance due to irrelevant terms in templates. Template detection and
extraction techniques have received a lot of attention to improve the performance of web applications. The problem of extracting a template from the web document conforming to a common template has been studied in[18]. Due to the assumption of all documents being generated from a single common template only when, all documents are guaranteed to conform to a common template.

For example, biogene data are published on the internet by many organizations with different formats and scientists wants to integrate these data into unified database for price comparison purpose, the price information is gathered from various internet marketplaces. Good template extraction technologies can significantly improve the performance of these applications. If we group web documents by using URL, there may be different appearance of pages. Thus, we cannot group web documents by using URL.

To overcome the limitation of the techniques with assumption that the web documents are from a single template, the problem of extracting the template from heterogeneously web documents ,which are generated from multiple templates. This can be achieved by clustering of web documents by selecting a good separation method. The correctness of extracted templates depends on quality of clustering.

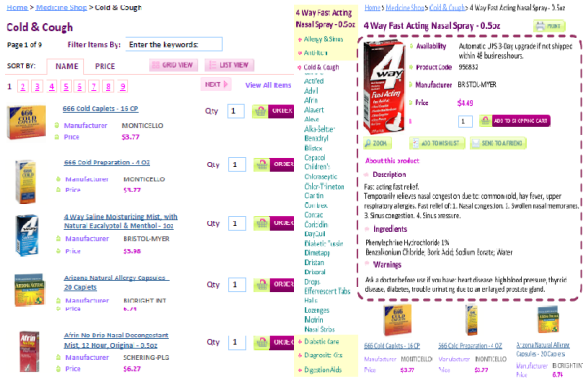


Figure 1: Different Templates of the Same URL.

In this paper, introduction of an approach to template extraction from heterogeneous web pages. In section 2, we present a survey of previous work done. In section 3, we present an approach for template extraction using proposed techniques. In section 4 gives a performance analysis and in section 5, we state our conclusion.

2. Previous Work Done

V. Crescenzi and G. Mecca [1], had presents MINERVA, a formalism for writing wrappers around web sites and other textual data sources. The key features of this system are to attempt to couple the benefits of a declarative, grammar based approach, with the flexibility of procedural programming. This is done by enriching regular grammars with explicit error handling mechanism. J. Hammer, H. Garcia-Mollina [2], had describes Jedi (Java based Extraction and Dissemination of Information), a light weight wrapping and mediation tool to reuse, combine, and reconcile information from several independent information sources. Jedi provides a fault-tolerant parser to extract data from external sources, an extensible object model to describe structure and semantics of heterogeneous sources uniformly, and a flexible query and manipulation language to realize integrated views on multiple sources. Observing a Web page and its source code, the human programmer finds some patterns and then writes a program to extract the target data. To make the process simpler for programmers, several pattern specification languages and user interfaces have been built.

Wrapper Induction the supervised learning approaches, and is semi-automatic. In this approach, a set of extraction rules is learned from a collection of manually labeled pages or data records. The rules are then employed to extract target data items from other similarly formatted pages. To achieve high accuracy, the task of extracting structured information from web pages is usually implemented by programs called wrappers. A wrapper induction system learns data extraction rules from a set of labeled training examples. Labeling is usually done manually, which simply involves marking the data items in the training pages/examples that the user wants to extract. The learned rules are then applied to extract target data from other pages with the same mark-up encoding or the same template. Shuyi Zheng [5], had describes a system to overcome the limitations of the separated template detection strategy related to effectiveness for large-scale websites and only one wrapper for complex template.

To solve the problems, propose detecting template solely based on the similarity among page representations that are also used in wrapper generation. In this system, tree structures are used as representations for pages and wrappers. Based on a distance metric between a page and a wrapper, a clustering algorithm is employed to cluster similar enough pages into a class and induces a central wrapper for the class at the same time..

Wrapper generation using supervised learning has two main shortcomings: 1. It is not suitable for a large number of sites due to the manual labeling effort. For example, if a shopping site wants to extract all the products sold on the Web, manual labeling becomes almost an impossible task. 2. Wrapper maintenance is very costly. The Web is a dynamic environment. Sites change constantly. Since wrapper learning systems mainly rely on HTML formatting tags, if a site changes its formatting templates, the existing wrapper for the site will become invalid.

As we discussed earlier, automatic verification and repair are still difficult. Doing them manually is very costly if the number of sites involved is large.

Due to these problems, automatic (or unsupervised) extraction has been studied by researchers. Automatic extraction is possible because data records (tuple instances) in a web site are usually encoded using a very small number of fixed templates. It is possible to find these templates by mining repeated patterns in multiple data records.

S. Panday [6], had consider a problem of extracting, with no manual intervention, the hidden structure behind the observed search queries in a domain: the origins of the constituent keywords as well as the manner the individual keyword assembled together. They formalize important properties of the problem and then give a principled solution based on generative models that satisfies these properties. M. Garofalakis et al. [7], had presents a novel approach for

XML documents, XTRACT solved the problem of DTD extraction from multiple XML documents. While HTML documents are semi structured and XML documents are well structured and all tags are always part of a template. The solution for XML documents fully utilize these properties. XTRACT is designed to learn regular expressions of length of magnitude of 1,0's of symbol. It uses heuristics to enumerate a small set of potential regular expressionand uses MDL principle to pic the best one.

D.C. Reis , P.B.Golgher [9], had presents a domain oriented approach to web data extraction and discuss its application to automatically extracting news from web sites. This approach is based on the concept of tree-edit distance and allows not only the extraction of relevant text passages from the web pages of a given web site, but also the fetching of entire web site content, the identification of the pages of interest and the extraction of the relevant text passages discarding non-useful material such as banner, menus and links.

Vieira et al.[10] , had described a fast and robust method for web page template detection and removal. RTDM-TD algorithm used to find optimal mapping between the Document Object Model (DOM) trees of web pages. This algorithm is based on a restricted formulation of top down mapping between two trees. Which particularly suitable for detecting structural similarities among web pages. But the operations related to trees are expensive.

Yanhong Zhai and Bing Liu [13], had describe technique to segment these data records, extract data items/ fields from them and put the data in database table. They provide a solution for problem of existing systems, 1. Method is based on machine learning requires human labeling of many examples from each web site that one is interested in extracting data from. The process is time consuming due to large no of sites and pages on the web. 2. Algorithm is based on automatic pattern discovery. These methods are either inaccurate or make many assumptions.

In these suggests a new technique to perform the task automatically. It consists two steps. 1) Identifying individual data records in a page using visual information to segment data records using MDR algorithm. 2) Aligning and extracting data items from the identified data records by using partial alignment technique based o tree matching.

A.Arsu et al. [18], presents in extracting structural data from web pages, extraction of data is done in two steps: 1. Formally define a template and propose a model that describes how values are encoded into pages using a template. 2. Presents an algorithm that takes as input a set of template generated web pages, deduce the unknown template used to generate pages and extract output.Chakrabarti et al. [21], had presents a framework for the page level template detection problem.1. Constructing a training data: for this purpose it uses a site level template detection algorithm. 2. Learning the classifier: It has several steps to be performed. Preprocessing – Each web page is proposed and parsed, so that features can be extracted from its DOM nodes and this step involves the cleaning HTML code. HYPAR2, annotating the DOM nodes with position\areas information using Mozilla, and parsing the HTML to obtain a DOM tree structure. The text in the HTML page is also processed to remove stop words. Feature extraction – from each Dom node, they extracts features that they believe are indicative of whether or not that DOM node is a template. Classifier training – Trained logistic regression classifiers over the set of features applied. These classifiers have traditional benefit that their classification output can be interpreted as the probability of belonging to the predicate class. Hence method have trained four logistic regression models for DOM nodes of different sizes. 3. Smoothing classifier scores – This algorithm allows arbitrary choices of penalty values for each tree node. Desiderate for penalties: There are three main desiderate for smoothing algorithm in the context of template detection. 1. Nodes that are too small in area should not form segment of their own. 2. Adding nodes as segments should be easier as we move up from leaves to root. 3. If a child node accounts for a large fraction of the area of its parent node.

Crescenzi et al. [23], had investigates the wrapper generation problem under a new perspective. In particular, we aim at automating the wrapper generation process to a larger extent and their approach clearly departs from the ones in several respects: 1) System does not rely on user-specified examples, and does not require any interaction with the user during the wrapper generation process; this means that wrappers are generated and data are extracted in a completely automatic way; 2) The wrapper generator has no a priori knowledge about the page contents, i.e., it does not know the schema according to which data are organized in the HTML pages: this schema will be inferred along with the wrapper; moreover, system is not restricted to flat records, but can handle arbitrarily nested structures.

3. Proposed Approach

1. The HTML document and Document Object Model.
2. Essential paths of document
3. Minimum description length for clustering

4. MDL cost estimation using Min Hash

System Architecture

In this system we are providing different web pages as input to the system. Each web page has different or may be same document structure. After parsing these web documents into HTML document using DOM model. Clustering documents based on the MDL principle, the model of each cluster is the template itself of the web documents belonging to the cluster. we do not need additional template extraction process after clustering. In order to improve efficiency and scalability to handle a large number of web documents for clustering, we extend MinHash. While the traditional MinHash is used to estimate the Jaccard Coefficient between sets, we propose an extended MinHash to estimate our MDL cost measure with partial information of documents.

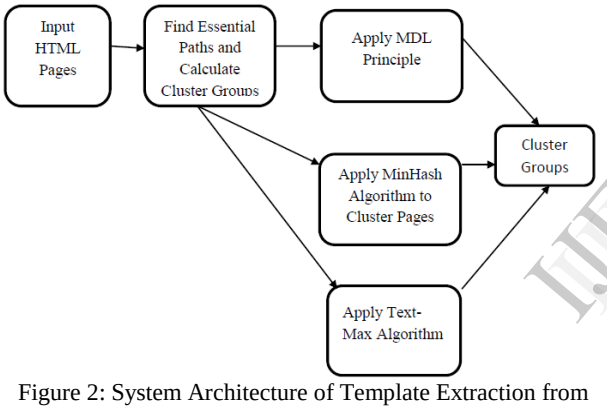


Figure 2: System Architecture of Template Extraction from Heterogeneous Web Pages

1. HTML Documents and Document Object Model

The Document Object Model (DOM) is an application programming interface (API) for valid HTML and well-formed XML documents. It defines the logical structure of documents and the way a document is accessed and manipulated. In the DOM specification, the term "document" is used in the broad sense increasingly, XML is being used as a way of representing many different kinds of information that may be stored in various systems, and much of this would traditionally be seen as data rather than as documents. Nevertheless, XML presents this data as documents, and the DOM may be used to manage this data.

With the document object model, programmers can build documents, navigate their structure, and add, modify, or delete elements and content. Anything found in an HTML or XML document can be accessed, changed, deleted, or added using the document object model, with a few exceptions in particular, the DOM interfaces for the HTML internal and external subsets have not yet been specified.

The DOM defines a standard for accessing documents like HTML. The DOM presents a tree structure for an HTML document. The complete document is a document node, every HTML tag element is an element node, the text in the HTML elements is text nodes, every HTML attribute is an attribute nodes. We denote the path of the node by listing nodes from root to the pendent node.

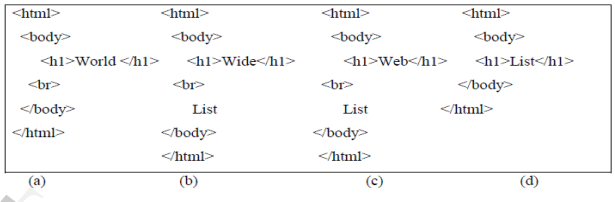


Figure 3: Simple Web Documents. (a) Document d_1. (b)Document d_2. (c) Document d_3. (d) Document d_4.

TABLE 1: Paths of Tokens and Their Support

ID	Path	Support
P_1	Document\<html>	4
P_2	Document\<html>\<body>	4
P_3	Document\<html>\<body>\<h1>	3
P_4	Document\<html>\<body>\ 	3
P_5	Document\<html>\<body>\List	3
P_6	Document\<html>\<body>\<h1>\World	1
P_7	Document\<html>\<body>\<h1>\Wide	1
P_8	Document\<html>\<body>\<h1>\web	1

From Fig.3 for a node in a DOM tree, denote the path of the node from the root to the node For example: The path of a node list is "Document <HTML>\<BODY>\List".

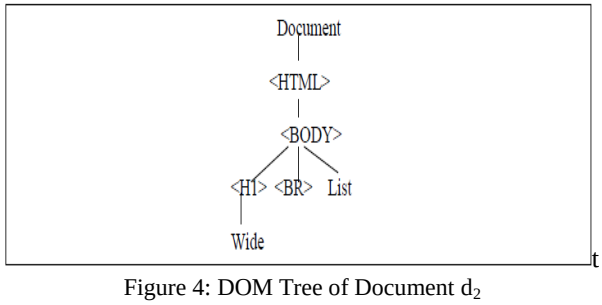


Figure 4: DOM Tree of Document d_2

2. Essential Paths and Templates

Given a web document collection $D = (d_1, d_2, d_3, \ldots, d_n)$, define a path set P_D as the set of all paths in D. Note that, since the document node is a virtual node shared by every document, is not consider the path of the document node in P_D. The support of a path is defined as the number of documents in D, which contain the path. For each document d_i, provide a minimum support threshold t_{d_i}. The thresholds t_{d_i} and t_{d_j} of two distinct documents d_i and d_j, respectively, may be different. If a path is contained by a document d_i and the support of the path is at least the given minimum support threshold t_{d_i}, the path is called an essential path of d_i. We denote the set of essential paths of an HTML document d_i by $E(d_i)$. For a web document set D with its path set P_D, a $|P_D| \times |D|$ matrix M_E with 0/1 values to represent the documents with their essential paths is use . The value at a cell (i, j) in the matrix M_E is 1 if a path P_i is an essential path of a document d_j. Otherwise, it is 0.

Example 1 .Consider the HTML documents $D = (d_1, d_2, d_3, \ldots, d_n)$, in Fig.3.2. All the paths and their frequencies in D are shown in Table 3.1 Assume that the minimum support thresholds t_{d_1}, t_{d_2}, t_{d_3} and t_{d_4} are 3, 3, 3, and 4,respectively. The essential path sets are $E(d_1)=(p_1, p_2, p_3, p_4)$, $E(d_2)=(p_1, p_2, p_3, p_4, p_5)$, $E(d_3)=(p_1, p_2, p_3, p_4, p_5)$, and $E(d_4)=(p_1, p_2)$. We have the path set $p_D = (p_i | 1 \le i \le 8)$ and the matrix M_E becomes as follows:

The goal of introducing essential paths is to prune the paths away in advance which cannot be a part of any template. It is kind of pre-processing to improve the correctness of clustering. If use the same threshold for all pages, it is not reasonable because the number of documents generated by each template is not the same.

$$M_E = \begin{array}{c} \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \\ p_7 \\ p_8 \end{array} \begin{array}{c} \begin{array}{cccc} d_1 & d_2 & d_3 & d_4 \end{array} \\ \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Thus there is need to use a different threshold for each page. The template of a document cluster is a set of paths which commonly appear in the documents of the cluster. If a path is contained in most pages of the cluster, we can assume that the occurrence of the path is not probably by chance, and thus, the path should be considered as a part of the template.

Contents are the paths which are not members of the template. If a document is generated by a template, the document contains two types of paths: the paths belonging to the template and the paths belonging to the contents. To separate the paths in contents from the paths in the template we assume that,1. The support of a path in a template is generally higher than that of a path in contents and 2. The number of the paths belonging to the template is generally greater than that of paths belonging to the contents.

For the first assumption, the paths in a template are shared by the documents generated by the template but those in contents are usually unique in each document. Thus, the support of the former is higher than that of the latter. For the second assumption, the paths from the template are typically dominant in a document. Based on our assumption, we found empirically that the mode of support values (i.e., the most frequent support value) of paths in each document is very effective to make templates survive, while contents are eliminated. Therefore, use the mode of support values of paths in each document as the minimum support threshold for each document. If there are several modes of support values, we will take the smallest mode.

Matrix Representation of Clustering: The representation of a clustering of web documents. Let us assume that we have m clusters such as $C = \{c_1, c_2, \ldots c_m\}$ for a web document set D. A cluster c_i is denoted by a pair (T_i, D_i), where T_i is a set of paths representing the template of c_i and D_i is a set of documents belonging to c_i. In our clustering model, we allow a document to be included in a single cluster only. That is $D_i \cap D_j = \emptyset$ for all distinct clusters c_i, c_j, and $U_{1 \le i \le m} D_i = D$. In addition, we define E_i for a cluster c_i as $\bigcup_{d_k \in D_i} E(d_k)$. To represent a clustering information $C = \{c_1, c_2, \ldots c_m\}$ for D, we use a pair of matrices M_T and M_D, where M_T represents the information of each cluster with its template paths and M_D denotes the information of each cluster with its member documents. If the value at a cell (i, j) in M_T is 1,it means that a path p_i is a template path of a cluster c_j(*i.e.*, $p_i \in T_j$). Otherwise, p_i does not belong to the template paths of c_j (*i.e.*, $p_i \notin T_j$). Similarly, the value at a cell (i,j)in M_D is 1 if a document d_j belongs to a cluster c_i (*i.e.*, $d_j \in D_i$).Regardless of the number of

clusters, we fix the dimension Of M_T as $|P_D| \times |D|$and that of M_D as $|D|\times|D|$. Columns and rows in M_T and M_D exceeding the number of clusters are filled with zeros. In other words, for a clustering with $C = \{c_1, c_2, \ldots c_m\}$, all values from $(m + 1)$ th to $|D|$th columns in M_T are zeros, and all values from $(m + 1)$th to $|D|$th rows in M_D are zeros. We will represent M_E by the product of M_T and M_D.However, the product of M_T and M_D does not always become M_E. Thus, we reconstruct M_E by adding a difference matrix M_Δ with 0/1/-1 values to $M_T \cdot M_D$, *i.e.*, $M_E = M_T \cdot M_D + M_\Delta$.

Example 2. Consider the web documents in Fig. 3.2 and M_E in Example 1 again. Assume that we have a clustering $C = \{c_1, c_2\}$, where $c_1 = \{(p_1, p_2, p_3, p_4, p_5), (d_1, d_2, d_3)\}$ and $c_2 = \{(p_1, p_2), (d_4)\}$. Then M_T, M_D, and M_Δ are as follows and we can see that $M_E = M_T \cdot M_D + M_\Delta$.

$$M_T = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad M_D = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad M_\Delta = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

3. Minimum Description Length Principle

In order to manage the unknown number of clusters and to select good partitioning from all possible partitions of HTML documents, Rissanen's MDL principle is used [20], [21]. The MDL principle states that the best model inferred from a given set of data is the one which minimizes the sum of, 1. The length of the model, in bits, and 2. The length of encoding of the data in bits, when described with the help of the model.

We refer to the above sum for a model as the MDL cost of the model. In this setting, the model is a clustering *C*, which is described by partitions of documents with their template paths (i.e., the matrices M_T and M_D), and the encoding of data is the matrix M_Δ. The MDL costs of a clustering model *C* and a matrix *M* are denoted as *L(C)* and *L(M)*, respectively. Considering the values in a matrix as a random variable *X*, *Pr*(1) and *Pr*(-1) are the probabilities of 1 s and -1 s in the matrix and *Pr*(0) is that of zeros. Then, the entropy is defined for random variable *X* is *H(X)*.

$$H(X) = \sum_{x \in \{1,0,-1\}} -\Pr(x) \log 2 \Pr(x)$$

$$L(M) = |M| \cdot H(X)$$

The MDL costs of M_T and M_Δ (i.e., $L(M_T)$ and $L(M_\Delta)$)are calculated by the above formula. For M_D, we use another method to calculate its MDL cost. The reason is that the random variable *X* in M_D is not mutually independent, since we allow a document to be included in a single cluster only (i.e., each column has only a single value of 1). Thus, we encode M_D by $|D|$ number of cluster IDs. Since the number of bits to represent a cluster ID is $log2\ |D|$, the total number of bits to encode M_D (i.e., $L(M_D)$) becomes $|D| \cdot log2\ |D|$. Then, the MDL cost of a clustering model *C* is defined as the sum of those of three matrices (i.e., $L(C) = L(M_T) + L(M_D) + L(M_\Delta)$). According to the MDL principle, for two clustering models $C = (M_T, M_D)$ and $C'' = (M'_T, M'_D)$, we say that *C* is a better clustering than *C*'' if *L(C)* is less than *L(C'')*.

Clustering with MDL Cost : Our clustering algorithm TEXT-MDL is presented in section 3. The input parameter is a set of documents $D = \{d_1, \ldots, d_n\}$, where d_i is the ith document. The output result is a set of clusters $C = \{c_1, \ldots, c_m\}$, where c_i is a cluster represented by the template paths T_i and the member documents D_i (i.e., $c_i = (T_i, D_i)$). A clustering model *C* is denoted by two matrices M_T and M_D and the goodness measure of the clustering *C* is the MDL cost $L(C)$, which is the sum of $L(M_T)$, $L(M_D)$, and $L(M_\Delta)$.

TEXT-MDL is an agglomerative hierarchical clustering algorithm which starts with each input document as an individual cluster. When a pair of clusters is merged, the MDL cost of the clustering model can be reduced or increased. The procedure GetBestPair finds a pair of clusters whose reduction of the MDL cost is maximal in each step of merging and the pair is repeatedly merged until any reduction is not possible. In order to calculate the MDL cost when each possible pair of clusters is merged, the procedure GetMDLCost(c_i , c_j , *C*) mentioned in nest section, where c_i and c_j are a pair to be merged and *C* is the current clustering, is called in GetBestPair and *C* is updated by merging the best pair of clusters.

The scale of the MDL cost reduction by merging a pair of clusters is affected by all the other clusters, GetBestPair should recalculate the MDL cost reduction of every pair at each iteration. Furthermore, the complexity of GetMDLCost is exponential on the size of the template of a cluster. Since it is not practical to use TEXT-MDL with a number of web documents, we will introduce use of MinHash to significantly reduce the time complexity.

Computation of Optimal MDL Cost: As MDL principle, a clustering model C by two matrices M_T and M_D and the MDL cost $L(C)$ is the sum of $L(M_T)$, $L(M_D)$, and $L(M_\Delta)$. By considering the independence of $L(M_D)$ from $L(C)$ and the sparsity of M_E. Independence of $L(M_D)$ from $L(C)$. Because $L(M_D)$ is constant for every M_D as $|D| \cdot \log_2 |D|$, the value of $L(C)$ is not affected by that of $L(M_D)$. Thus, minimizing $L(C)$ is the same as minimizing the sum of $L(M_T)$ and $L(M_\Delta)$ only.

Sparsity of M_E - since web documents are made by different templates of various sites, the web documents rarely have common paths. In templates, some paths can commonly occur in heterogeneous documents since the kinds of tags, which can be placed at the first or second depth are limited. However, as the depth is extended, the possibility that a path appears commonly in heterogeneous documents is decreased exponentially. Thus, we assume that the matrices, such as M_E, are sparse (i.e., zero is more frequent than other values in a matrix). If this assumption does not hold in an extreme case, we can add empty documents as many as the number of documents in D. Then, the empty documents are represented by only zeros in M_E and zeros in M_E become more than a half of M_E. Candidacy of template paths. For a cluster $c_i = (T_i, D_i)$ of a clustering model C, only the essential paths of documents in D_i can be included in the optimal template paths T_i to minimize the MDL cost of C.

$$|M_E| \cdot \frac{\beta}{\alpha} \cdot (\Pr(1) of M_T + (\Pr(1) + \Pr(-1)) of M_\Delta) + L(M_D)$$

Since the optimal template of a cluster is independent of those of the other clusters due to approximation, So reuse the MDL cost of merging each pair of clusters in the previous calls of GetBestPair mention in next section. These pairs of clusters with the MDL costs are maintained in a heap structure and the initial best pair is retrieved from the heap . Since the complexity of GetBestPair(c_k, C) is $O(ns)$, hence that of TEXT-MDL becomes $O(n^2 s)$.

4. Estimation of MDL Cost with MinHash

Although considering only essential paths, the dimension of Ei is still high and the number of documents is large. Thus, the $O(n^2 s)$ complexity of TEXT-MDL is still expensive. In order to ease this situation, introduces how to estimate the MDL cost of a clustering by MinHash to reduce the dimensions of documents as well as to find quickly the best pair to be merged in the MinHash signature space.

TEXT-HASH : Jaccard's coefficient between two sets S_1 and S_2 is defined as $\gamma(S_1, S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}$ and the Min-Wise independent permutation is a well-known Monte Carlo technique that estimates the Jaccard's coefficient by repeatedly assigning random ranks to the universal set and comparing the minimum values from the ranks of each set. Consider a set of random permutations $\Pi = \{\pi_1, \dots, \pi_L\}$ on a universal set $U = \{r_1, \dots, r_M\}$ and a set $S_1 \subset U$. Let $\pi_i(r_j)$ be the rank of r_j in a permutation π_i and $\min(\pi_i((S_1))$ denote $\min\{\pi_i(rj) | rj \in S_1\}$. Π is called minwise independent if we have $Pr\left(\min(\pi_i(S_1)) = \pi(x)\right) = \frac{1}{|S_1|}$ for every set $S_1 \subset$ U and every $x \in S_1$ for all $\pi i \in \Pi$. Then, for any sets S_1, $S_2 \subset$ U for all $\pi_i \in \Pi$, and $\Pr(\min(\pi_i(S_1)) = min(\pi_i(S_2)) = \gamma(S_1, S_2)$, where $\gamma(S_1, S_2)$ is the Jaccard's coefficient defined previously.

$sigs1 = [\min(\pi_1(S_1)) = min(\pi_2(S_1)), \dots, \min(\pi_L(S_1))]$

, and similarly, $sigs_2$ is produced for S_2. The ith entry of vector $sigs_1$ is denoted as $sigs_1[i]$. Matching the signatures $sigs_1$ and $sigs_2$ per permutation, $\gamma(S_1, S_2)$ can be estimated.

Extended MinHash: To compute the MDL cost of each clustering quickly, estimate the probability that a path appears in a certain number of documents in a cluster. However, the traditional MinHash was proposed to estimate the Jaccard's coefficient. Thus, given a collection of sets X={S_1,…, S_k}, we extend MinHash to estimate the probabilities needed to compute the MDL cost.To defining probability (denoted as $\xi(X, m)$) that $r_j \in U_{s\ell \epsilon x} S_\ell$ is included by m number of sets in X.Then, $\xi(X, m)$ is defined for 1≤ m ≤ |X| and $\xi(X, m)$ is the same as the Jaccard's coefficient of sets in X. The extended signature $sigx[i]$ for πi is a pair of the minimum $sigsj[i]$ for all S_j in X and the number of sets whose signature for π_i is the same as the minimum . The former is denoted as $r(sigx[i])$ and latter is denoted as $n(sigx[i])$ .By observing that $r(sigx[i])$ is the same as $\min(\pi_i(S_1 \cup \dots \cup S_k)$ and thus, $Pr\left(r(sigx[i]) = \pi_i(r_\ell)\right) = \frac{1}{|S_1 \cup \dots \cup S_k|}$ for every $r_\ell \in S_1 \cup \dots \cup S_k$ and every $\pi_i \epsilon \Pi$.

$$\xi(X,m) = \frac{|\{i|n(sigx[i]) = m\}|}{|\Pi|}$$

Clustering with MinHash

When we merge clusters hierarchically, select two clusters which maximize the reduction of the MDL cost by merging them. Given a clusterc_i, if a cluster c_j maximizes the reduction of the MDL cost, then c_j the nearest cluster of c_i. In order to efficiently find the nearest cluster of c_i, we use the heuristic. Then, given three clusters c_i, c_j, and c_k, if Jaccard's coefficient between c_i and c_j is greater than that between c_i and c_k, assume that the reduction of the MDL cost by merging c_i and $c_j$will be greater than that by c_i and c_k.

TEXT-MAX: By using Heuristic, we can reduce the search space to find the nearest cluster of a cluster c_i. The previous search space to find the nearest cluster of c_i was the same as the number of current clusters. But, using Heuristic, the search space becomes the number of clusters whose Jaccard's coefficient with c_i is maximal. The Jaccard's coefficient can be estimated with the signatures of MinHash and clusters whose Jaccard's coefficient with c_i is maximal can be directly accessed in the signature space.

TEXT MDL algorithm:

1. Begin
2. Consider C:={c_1,c_2,…..c_n} with c_i =(E(d_i),{d_i});
3. (c_i,c_j,c_k):= GetBestPair(C);
4. // Let c_i and c_j be the best pair of merging
5. // Let c_k be a new cluster made by merging c_i and c_j
6. While(c_i,c_j,c_k)is not empty do
7. begin
8. C:= C – {c_i, c_j} U {c_k};
9. (c_i,c_j,c_k):= GetBestPair(C);
10. end while_ do
11. return C
12. end

Procedure GetBestPair(C)

1. Begin
2. MDLcost$_{min}$:=∞;
3. For each pair (c_i, c_j)of clusters in C do{
4. (MDLcost, c_k):= GetMDLCost(c_i, c_j, C);
5. // GetMDLCost returns the optimal MDL cost
6. //When c_k is made by merging c_i and c_j
7. If MDLcost<MDLcost min then begin
8. MDLcost min :=(c_i,c_j,c_k);
9. end if
10. end for
11. Return(c_i^B,c_j^B,c_k^B);
12. End

4. Performance Analysis

This section is presenting the results of practical performance evaluation of TEXT-MDL, TEXT-HASH, TEXT-MAX approach, comparing their outputs with the RTDM. Although this algorithm do not necessarily locate the optimal solution, if is found to produce solutions that are sufficiently close to the optimal one, then a case can be made for using it in practice, given that they are simple and well established and understood.

Dataset

We use real life data set as follows:
Dataset (*D1*): The numbers of documents are from 10 templates and the number of documents from each template is 3.
Dataset (*D2*): The numbers of documents are from 20 templates and the number of documents from each template is 10.
Dataset (*D3*): It is the data set used in EXALG [3]. The numbers of documents are from nine templates and the number of documents from each template is from 10 to 50. The total no of documents are 142.

Experimental Results and Discussion

The Optimal Solution : The optimal solution with respect to RTDM gets its training set by sampling and MinHash used in TEXT-HASH is a probabilistic model. The idea is based on the MDL principle using metrics. This improves the performance of web applications in the form of accuracy and time.

Clustering and Template Accuracy: We compare the clustering results of RDTM and our proposed algorithms. In order to quantify the accuracy of a cluster, we use the precision and recall values between a cluster and the closest ground truth cluster. The number of clusters found by each algorithm with, P and R are the average precision and recall values of clusters, respectively and time duration in seconds.

- Precision (P):

Precision is defined as,

$$Precision = \frac{|\{relevent\ documents\} \cap \{retrived\ documents\}|}{|\{retrived\ document\}|}$$

Recall (R):

Recall is defined as,

$$Recall = \frac{|\{relevant\ document\} \cap \{retrived\ document\}|}{|\{relevant\ document\}|}$$

Empirical Results and Discussions

Performance with varying the number of documents:

- Consider Dataset *D1*:

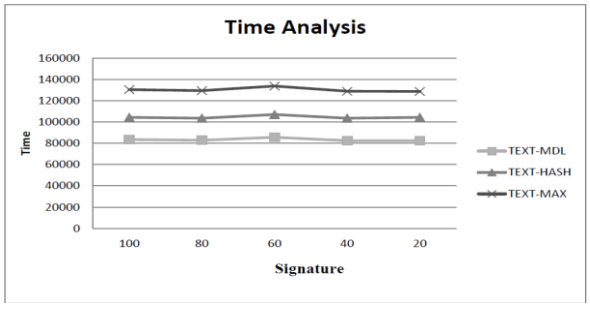
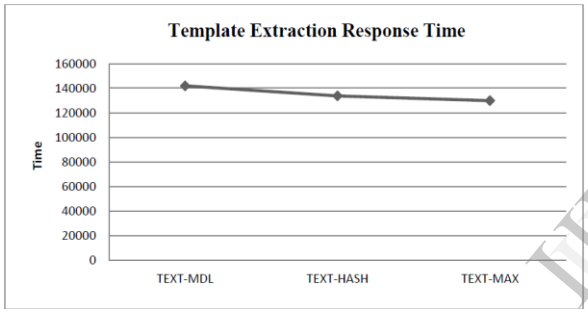


Figure 5(a): Template Extraction Response Time of Dataset *D1* Figure



5(b): Time Analysis with Signature Value of Dataset *D1*

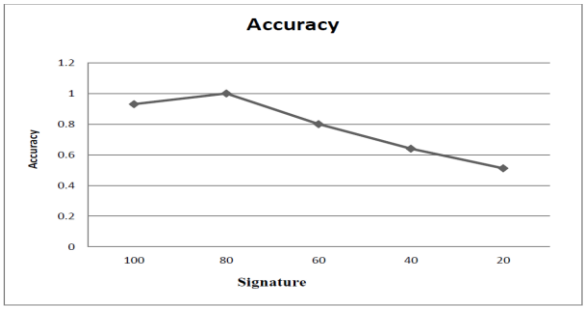


Figure 5(c): The Accuracy with Signature Value of Dataset *D1*

- Consider Dataset *D2*:

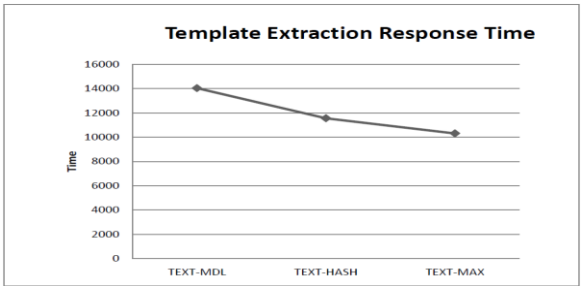


Figure 6(a): Template Extraction Response Time of Dataset *D2*

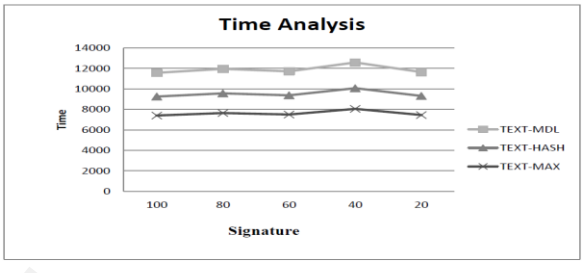


Figure 6(b): Time Analysis with Signature Value of Dataset *D2*

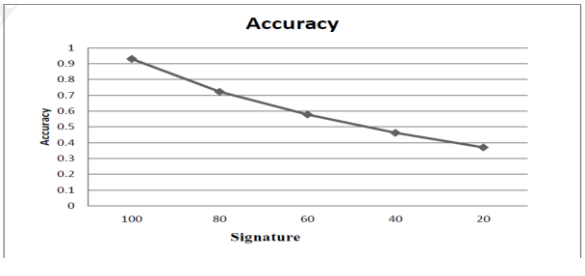


Figure 6(c): The Accuracy with Signature Value of Dataset *D2*

- Consider Dataset *D3*:

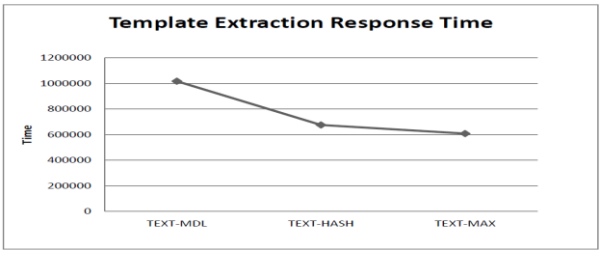


Figure 7(a): Template Extraction Response Time of Dataset *D3*

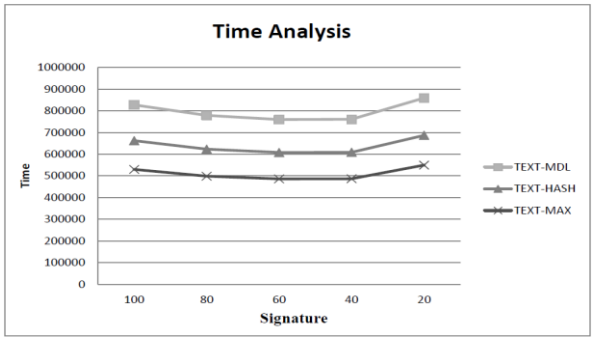


Figure 7(b): Time Analysis with Signature Value of Dataset *D3*

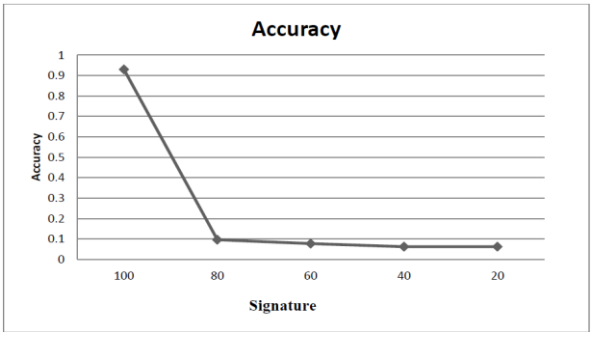


Figure 7(c): The Accuracy with Signature Value of Dataset *D3*

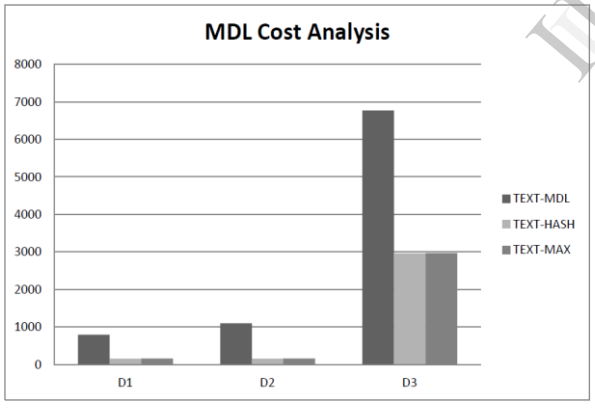


Figure 8: Comparison of MDL Cost of TEXT-MDL, TEXT-HASH, TEXT-MAX with Dataset *D1*, Dataset *D2* and Dataset *D3*

After comparing the execution times and the MDL costs of TEXT-MDL, TEXT-HASH, and TEXT-MAX with various numbers of documents from 10 to 20. Execution times are plotted for dataset *D2* in Fig.5(a) where y-axis is in a log-scale. The execution time of TEXT-MDL is quadratic to the number of documents. TEXT-HASH and TEXT-MAX are clearly much faster than TEXT-MDL at least by an order of magnitude. In Fig.5(a), we presented the scalability test of TEXT-MDL, TEXT-HASH and TEXT-MAX with dataset *D2*. TEXT-MDL took about 141,871 mile seconds with 1,000 documents. We next present the MDL cost at each execution in Fig.8, where y-axis is in a log-scale. The TEXT-HASH (estimate) and TEXT-MAX (estimate) are the estimated MDL costs with MinHash signatures. The MDL costs computed after the post processing step of the template path generation are denoted as TEXTHASH (real) and TEXT-MAX (real). Although TEXT-HASH and TEXT-MAX are much faster than TEXT-MDL, their MDL costs are very close. The differences between them are relatively small, and thus, we can confirm that the accuracy of clustering with the extended MinHash technique is still reliable, while the efficiency of algorithms is improved at least by an order of magnitude.

Performance with varying signature sizes:

The length of signature is the single parameter required by algorithms. Experiments with various lengths of signatures show that the algorithms are robust to the parameter and have good performance with a short length of signature. We used 1,000 document sets and changed the length of signature from 20 to 100. The execution times with various lengths of signatures are given in Fig.5(b) for dataset *D1*, Fig.6(b) for dataset *D2* and 7(b) for dataset *D3*. The execution times of TEXT-HASH as well as TEXT-MAX are linear on the length of signature but TEXT-MAX is much faster than TEXT-HASH, as shown in the previous performance study with various numbers of documents.MDL cost after the post processing step of the template path generation becomes very large. However, the estimated MDL costs and the real MDL costs are quickly converged as the length of signature becomes longer. If the length is longer than 50, the estimated and real MDL costs get stable and converge. Thus, we do not need to have very long signatures, which make algorithms slow without any significant improvement of accuracy.

Effectiveness of the threshold to generate essential paths:

As discussed in Section 3, each document has its own threshold to generate essential paths and we take the mode of supports, $t\theta$, in each document as the threshold of the document. In order to show the effectiveness of our threshold, we measured the number of essential paths generated from 5,000 document sets with various values of threshold. The result clearly shows that our threshold is very effective to make templates survive while contents are eliminated. When the threshold is zero, nothing is pruned by the threshold but, with a small threshold such as 0.1.$t\theta$, the number

of essential paths evidently decreases. Between 0.1. $t\theta$ and $t\theta$, the number of essential paths is almost the same but that with 1.1. $t\theta$ Suddenly decreases by about 90 percent. It shows that the paths from contents are eliminated by a small threshold such as 0.1. $t\theta$ and almost all paths from templates survive until the threshold becomes $t\theta$. If the threshold is too large, only generally common paths such as "Document\<html>\<body>" remain. Thus, we can conclude that $t\theta$ is very effective to identify templates.

Clustering and Template Accuracy:

The ground truth of clustering of data sets *D1* and *D2* is known and we compare the clustering results of RDTM and our proposed algorithms with the ground truth. In order to quantify the accuracy of a cluster, we use the precision and recall values between a cluster and the closest ground truth cluster. The results are given in Table 4.2, where # is the number of clusters found by each algorithm, P and R are the average precision and recall values of clusters, respectively, and Sec. is the execution time in seconds. In Table 4.1, the average precision value is always 1.0. It means that no documents from different templates were clustered in the same cluster. For the average recall values, RTDM with 50 percent sampling has very low recall values less than 0.5 for data sets 1 and 2. It shows that the ground truth clusters were segmented into many sub clusters by RTDM. Moreover, it takes a long time compared with our algorithms. For our algorithms, TEXT-HASH and TEXT-MAX are fastest and their precision and recall values are as good as those of TEXT-MDL. It shows that our extended MinHash technique allows TEXT-HASH and TEXT-MAX to have significantly faster execution times without sacrificing accuracy.

Table 2: Clustering Results

	D1				*D2*			
	#	P	R	ms	#	P	R	ms
TEXT-MDL	729.87	0.93	1.0	141.871	1097	1.0	1.0	14.042
TEXT-HASH	155.930	0.93	1.0	133.739	156.79	1.0	1.0	11.560
TEXT-MAX	158.57	0.93	1.0	129.870	158.47	1.0	1.0	10.312

Table 3: Accuracy of Extracted Contents

	D1		*D2*	
	P	R	P	R
RTDM	0.42	1.0	0.41	0.80
TEXT-MDL	0.93	1.0	1.0	1.0
TEXT-HASH	0.93	1.0	1.0	1.0
TEXT-MAX	0.93	1.0	1.0	1.0

5. Conclusions

Template Extraction from Heterogeneous Web Pages uses MDL principle, which is based on matrix representation. It is a kind of preprocessing to improve the correctness of clustering by managing unknown number of clusters.The number of clusters generated by selecting good partitioning from all possible partitions of documents. Extended MinHash technique is use to speed up the clustering process.Experimental results with real life data sets confirmed the effectiveness of algorithm.

This system provides approximately 93% precision and 97% recall where as existed system RTDM results 42% precision and 41% recall. Precision value for TEXT-MDL, TEXT-Hash and TEXT-Max is approximately 93% for dataset D1 and Recall value 99% , for dataset D2 P= 100% and R= 100%. From experimental results TEXT-MDL, TEXT-HASH, TEXT-MAX gives approximately same accuracy of cluster grouping but time required for TEXT-MDL is more. TEXT-HASH and TEXT-MAX are faster than TEXT-MDL, which improves the performance of web applications.

References

[1] V. Crescenzi, G.Mecca "Grammers Have Exceptions" Elsevier Science Ltd, pp 1- 25,1998.

[2] J. Hammer, H. Garcia-Molina,"Jedi:Java Based Extraction and Dissemination of Information", Proc. Academia Digital Libreries, pp 344-357,1997

[3] A.Z. Broder, M. Charikar, A.M. Frieze, and M. Mitzenmacher, "Min-Wise Independent Permutations," J. Computer and System Sciences, vol. 60, no. 3, pp. 630-659, 2000.

[4] K. Lerman, L. Getoor, S. Minton, and C. Knoblock, "Using the Structure of Web Sites for Automatic Segmentation of Tables," Proc. ACM SIGMOD, pp 119-130, 2004.

[5] S. Zheng, D. Wu, R. Song, and J.-R. Wen, "Joint Optimization of Wrapper Generation and Template Detection," Proc. ACM SIGKDD, pp 894-902, 2007

[6] S. Pandey, Kunal Punera," Unsupervised Extraction of Template Structure in Web Search Quries", ACM, Loyn France, pp 16-20, 2012.

[7] M.N. Garofalakis, A. Gionis, R. Rastogi, S. Seshadri, and K. Shim,"Xtract: A System for Extracting Document Type Descriptors from Xml Documents," Proc. ACM SIGMOD, pp 165-176, 2000.

[8] Z. Chen, F. Korn, N. Koudas, and S. Muithukrishnan, "Selectivity Estimation for Boolean Queries," Proc. ACM SIGMOD-SIGACTSIGART Symp. Principles of Database Systems (PODS), pp 216- 225, 2000.

[9] M. de Castro Reis, P.B. Golgher, A.S. da Silva, and A.H.F. Laender, "Automatic Web News Extraction Using Tree Edit Distance," Proc. 13th Int'l Conf. World Wide Web (WWW), pp 502-511, 2004.

[10] Document Object Model (dom) Level 1 Specification Version 1.0, http://www.w3.org/TR/ REC-DOM-Level-1, 2010.

[11] I.S. Dhillon, S. Mallela, and D.S. Modha, "Information-Theoretic Co-Clustering," Proc. ACM SIGKDD, pp 89-98, 2003.

[12] T.M. Cover and J.A. Thomas, Elements of Information Theory. Wiley Interscience, , pp 776 , 1991.

[13] Y. Zhai and B. Liu, "Web Data Extraction Based on Partial Tree Alignment," Proc. 14th Int'l Conf. World Wide Web (WWW), pp 76-85, 2005.

[14] J. Cho and U. Schonfeld, "Rankmass Crawler: A Crawler with High Personalized Pagerank Coverage Guarantee," Proc. Int'l Conf. Very Large Data Bases (VLDB), pp 375-386, 2007.

[15] B. Long, Z. Zhang, and P.S. Yu, "Co-Clustering by Block Value Decomposition," Proc. ACM SIGKDD, pp 635-640, 2005.

[16] M.D. Plumbley, "Clustering of Sparse Binary Data Using a Minimum Description Length Approach," *http://www.elec. qmul.ac.uk/staffinfo/markp/*, pp 1-18, 2002.

[17] K. Vieira, A.S. da Silva, N. Pinto, E.S. de Moura, J.M.B. Cavalcanti, and J. Freire, "A Fast and Robust Method for Web Page Template Detection and Removal," Proc. 15th ACM Int'l Conf. Information and Knowledge Management (CIKM), pp 258-267, 2006.

[18] A. Arasu and H. Garcia-Molina, "Extracting Structured Data from Web Pages," Proc.ACM SIGMOD, pp 337-348, 2003.

[19] H. Zhao, W. Meng, Z. Wu, V. Raghavan, and C. Yu, "Fully Automatic Wrapper Generation for Search Engines," Proc. 14th Int'l Conf. World Wide Web (WWW), 2005.

[20] H. Zhao, W. Meng, and C. Yu, "Automatic Extraction of Dynamic Record Sections from Search Engine Result Pages," Proc. 32nd Int'l Conf. Very Large Data Bases (VLDB), pp 989-1000, 2006.

[21] D. Chakrabarti, R. Kumar, and K. Punera, "Page-Level Template Detection via Isotonic Smoothing," Proc. 16th Int'l Conf. World Wide Web (WWW), pp 61-70, 2007.

[22] N.P.K. Ganesh Kumar, Dr. N.K. Sakthivel, "Development of an Efficient Vertex Based Template Extraction Technique for Web Pages", Proc. JATIT 2012.

[23] V. Crescenzi, G. Mecca, and P. Merialdo, "Roadrunner: Towards Automatic Data Extraction from Large Web Sites," Proc. 27th Int'l Conf. Very Large Data Bases (VLDB), pp 109-118, 2001.

[24] J. Rissanen, "An Introduction to the MDL Principle", *http://www.mdl-research.org/ jorma.rissanen/pub/Intro.pdf.*

[25] P.T. Santhiya, R.Tamizharasi, K. Sudharson,"A Min-Hash Algorithm for Clusteringeb Documents ", *http://www.ijsacs.org/vol2issue2/paper28.pdf* 2013.

[26] Chulyun Kim and Kyuseok Shim,"TEXT: Automatic Template Extraction from Hetrogeneous web Pages", IEEE Transaction on Knowledge and Data Engineering, vol . 23, no. 4 april 2011.

[27] Mohammed Kayed, Chia-Hui Chang, Khaled Shaalan, Moheb Ramzy Girgis,"FiVaTech : Page-Level Web Data Extraction from Template Pages", Proc. ACM SIGKDD, 2010.

[28] V.K. Raavi, P. K. Somayajula "Automatic Template Extraction from Heterogeneous Web Pages" ISSN: 2277128X, IJARCSSE, Volume 2, Issue 8, pp 408-418, 2012.

[29] D. Gibson, K. Punera, and A. Tomkins, "The Volume and Evolution of Web Page Templates," Proc. 14th Int'l Conf. World Wide Web (WWW), pp 165-176, 2005 .

[30] V. Crescenzi, P. Merialdo, and P. Missier, "Clustering Web Pages Based on Their Structure," Data and Knowledge Eng., vol. 54, pp. 279- 299, 2005.