ISSN:0975-9646

# Automatic Template Extraction from Heterogeneous Web Pages

Rashmi D Thakare , Mrs. Manisha R Patil

*Dept. of computer science n engineering*
*SKNCOE,*
*Pune, India*

**Abstract-Many web sites contain large sets of pages generated using a common template or layout. For example, Amazon lays out the author, title, comments, etc. in the same way in all its book pages. The values used to generate the pages (e.g., the author, title,...) typically come from a database. In this paper, we study the problem of automatically extracting the database values from the web pages without any learning examples or other similar human input. We formally define the notion of a template, and propose a model that describes how values are encoded into pages using a template. We present an extraction algorithm that uses sets of words that have similar occurrence pattern in the input pages, to construct the template. The constructed template is then used to extract values from the pages. We show experimentally that the extracted values make semantic sense in most cases.**

**Keywords: - Webpage sectioning, webpage segmentation, template detection, isotonic regression. Information extraction; Wrapper induction; Clustering; Web modelling; Web mining.**

## 1. INTRODUCTION

The World Wide Web is a vast and rapidly growing source of information. Most of this information is in unstructured HTML pages that are targeted at a human audience. The unstructured nature of these pages makes it hard to do sophisticated querying over the information present in them. There are, however, many web sites that contain a large collection of pages that have more “structure.” These web pages encode data from an underlying structured source, like a relational database, and are typically generated dynamically.

An example of such a collection is the set of book pages in Amazon shows two example book pages from Amazon [1].

The classic analysis of hashing schemes often entails the assumption that the hash functions used are random. More precisely, the assumption is that keys belonging to a universe U are hashed into a table of size M by choosing a function h uniformly at random among all the functions U ! [M]. (The notation [M] stands for the set {0, . . . , M−1}. This is slightly non-standard, but convenient for our purposes.) This assumption is impractical since just specifying such a function requires |U| log(M) bits1, which usually far exceeds the available storage[2].The increased use of content-management systems to generate web pages has significantly enriched the browsing experience of end users; the multitude of site navigation links, sidebars, copyright notices, and timestamps provide easy to access and often useful information to the users. From an objective standpoint, however, these “template” structures pollute the content by digressing from the main topic of discourse of the webpage. Furthermore, they can cripple the performance of many modules of search engines, including the index, ranking function, summarization, duplicate detection, etc. With template content currently constituting more than half of all HTML on the web and growing steadily [3, 11], it is imperative that search engines develop scalable tools and techniques to reliably detect templates on a webpage. Most existing methods for template detection operate on a per website basis by analyzing several web pages from the site and identifying content and/or structure that repeats across many pages. While these “site-level” template detection methods offer a lot of promise, they are of limited use because of the following two reasons. First, site-level templates constitute only a small fraction of all templates on the web[3]. In a variety of applications ranging from optimizing queries on alphanumeric attributes to providing approximate counts of documents containing several query terms, there is an increasing need to quickly and reliably estimate the number of strings (tuples, documents, etc.) matching a Boolean query. Boolean queries in this context consist of substring predicates composed using Boolean operators. While there has been some work in estimating the selectivity of substring queries, the more general problem of estimating the selectivity of Boolean queries over substring predicates has not been studied[4]. As well as the paper investigates techniques for extracting data from HTML sites through the use of automatically generated wrappers. To automate the wrapper generation and the data extraction process, the paper develops a novel technique to compare HTML pages and generate a wrapper based on their similarities and differences. Experimental results on real-life data-intensive Web sites confirm the feasibility of the approach[5]. A large number of Web sites contain highly structured regions. The pages contained in these regions are generated automatically, either statically or dynamically, by programs that extract the data from a back-end database and embed them into an HTML template. As a consequence, pages generated by the same program exhibit common structure and layout, while differing in content[6].Clustering is a fundamental tool in unsupervised learning that is used to group together similar objects, and has practical importance in a wide variety of applications such as text, web-log and market-basket data analysis. Typically, the data that arises in these applications is arranged as a contingency or co-occurrence table, such as, word-document co-occurrence table or webpage-user browsing data. Most clustering algorithms focus on one-way clustering, i.e., cluster one dimension of the table based on similarities along the second dimension. For example, documents may be clustered based upon their word distributions or words may be clustered based upon their distribution amongst documents[7]. Template material is common content or formatting that appears on multiple pages of a site. Almost all pages on the web today contain template material to a greater or lesser extent. Common examples include navigation sidebars containing links along the left or right side of the page; corporate logos that appear in a uniform location on all pages; standard background colours or styles; headers or dropdown menus along the top with links to products, locations, and contact information; banner advertisements; and footers containing links to homepages or copyright information. The template mechanism is used to support many purposes, particularly navigation, presentation, and branding[8]. The clustering procedure arises in many disciplines and has a wide range of applications. In many applications, such as document clustering, collaborative filtering, and microarray analysis, the data can be formulated as a two dimensional matrix representing a set of dyadic data. Dyadic data refer to a domain with two finite sets of objects in which observations are made for *dyads*, i.e., pairs with one element from either set. For the dyadic data in these applications, co-clustering both dimensions of the data matrix simultaneously is often more desirable than traditional oneway clustering. This is due to the fact that co-clustering takes the benefit of exploiting the duality between rows and columns to effectively deal with the high dimensional and sparse data that is typical in many applications. Moreover, there is an additional benefit for co-clustering to provide both row clusters and column clusters at same time. For example, we may be interested in simultaneously clustering genes and experimental conditions in bioinformatics applications simultaneously clustering documents and words in text mining simultaneously clustering users and movies in collaborative filtering[9]. Clustering is a fundamental data mining problem with a wide variety of applications. It seeks good partitioning of the data points such that points in the same cluster are similar to each other and the points in different clusters are dissimilar. Many real-life applications involve large data matrices. For example, in text and web log analysis, the term-document data can be represented as contingency table. In biology domain, the gene expression data are organized in matrices with rows representing genes and columns representing experimental conditions. Recently there has been a growing research interest in developing co-clustering algorithms that simultaneously cluster both columns and rows of the data matrix. Co-clustering takes advantage of the duality between rows and columns to effectively deal with the high dimensional data[10]. The *MDL* (Minimum Description Length) principle for statistical model selection and statistical inference is based on the simple idea that the best way to capture regular features in data is to construct a model in a certain class which permits the shortest description of the data and the model itself. Here, a model is a probability measure, and the class is a parametric collection of such models; an example is the likelihood function. Despite its simplicity the idea represents a drastically different view of modelling. First, the model class has to be such that its members can be described or encoded in terms of a finite number of symbols, say the binary. We give a brief description of the elementary coding theory needed in the appendix. This requirement also means that the traditional nonparametric models as some sort of idealized and imagined data generating distributions cannot be used unless they can be ¯tted to data. In the *MDL* approach we just ¯t models to data, and no assumption that the data are a sample from a `true' random variable is needed. This in one stroke eliminates the difficulty in the other approaches to modelling that the more complex a model we ¯t the better estimate of the `truth' we get, a problem that has had only ad hoc solutions[11]. The availability of tools that simplify the design and implementation of data-intensive web sites has greatly contributed to the explosive growth of the Web. These tools involve "a combination of templates and design conventions" including templates for common types and classes of pages, as well as sets of templates for common pages in sub-sites. By automatically populating these templates with content, web site designers and content producers of large web portals achieve high levels of productivity and improve the usability of the sites by enforcing the uniformity of the pages[12].

## 2. Literature Survey :-

[1].Arasu and H. Garcia-Molina-

Arvind Arasu Hector Garcia-Molina They presented an algorithm, EXALG, for extracting structured data from a collection of web pages generated from a common template. EXALG first discovers the unknown template that generated the pages and uses the discovered template to extract the data from the input pages. EXALG uses two novel concepts,equivalence classes and differentiating roles, to discover the template. Our experiments on several collections of web pages, drawn from many well-known data rich sites, indicate that EXALG is extremely good in extracting the data from the web pages. Another desirable feature of EXALG is that it does not completely fail to extract any data even when some of the assumptions made by EXALG are not met by the input collection. In other words the impact of the failed assumptions is limited to a few attributes. There are several interesting directions for future work. The first direction is to develop techniques for crawling, indexing and providing querying support for the "structured" pages in the web. Clearly, a lot of information in these pages is lost when naive key word indexing, and searching is used. they indicate two specific problems in this direction. First, how do we automatically locate collections of pages that are · structured? Second, is it feasible to generate some large "database" from these pages? Any technique for solving the latter problem has to be much less sophisticated than the one discussed here, possibly by sacrificing accuracy for efficiency. Also when we work at the scale of the entire web we might be able to leverage the redundancy of the data on the web as in Brin The second direction of work is to develop techniques for automatically annotating the extracted data, possibly using the words that appear in the template.

[3]. D. Chakrabarti, R. Kumar, and K. Punera

They presented a framework for classifier based page-level template detection that constructs the training data and learns the notion of "templateness" automatically using the site-level template detection approach. We formulated the smoothing of classifier assigned templateness scores as a regularized isotonic regression problem on trees, and presented an efficient algorithm to solve it exactly; this may be of independent interest. Using human-labeled data we empirically validated our system's performance, and showed that template detection at the page-level, when used as a pre-processing step to web mining applications, such as duplicate detection and webpage classification, can boost accuracy significantly.

[4] Z. Chen,F. Korn,N. Koudas, and S. Muithukrishnan

They generalize the problem of substring selec- tivity estimation for Boolean predicates. Our novel idea is to capture correlations between Boolean query predicates in a space-e client but approximate manner. We employ a Monte Carlo technique called set hashing to succinctly represent the set of strings containing a given substring predicate as a signature vector of hash values. Correlations among substring predicates can then be generated by operating on these signatures. We present an algorithm to estimate the selectivity of any Boolean query and experimentally demonstrate the superiority of our approach.

[6] V. Crescenzi, P. Merialdo, and P. Missier

In this paper They have presented an algorithm to cluster pages from a data intensive Web site, based on the page structure. The structural similarity among pages is defined with respect to their DOM trees. The algorithm identifies the main classes of pages offered by the site by visiting a small yet representative number of pages. The resulting clustering can be used to build a model that describes the structure of the site in terms of classes of pages and links among them. The model can be used for several purposes. First, for each class of pages in the model we can generate a wrapper: the visited pages which have been grouped into one cluster can be used as input samples for automatic wrapper generator systems, overcoming the issue of the manual selection phase. Once a wrapper for each class has been built, the model can be used also for classifying pages, with the objective of determining which wrapper has to be applied against a given page.

[7]. I.S. Dhillon, S. Mallela, and D.S. Modha

They have provided an information-theoretic formulation for co-clustering, and presented a simple-to-implement, top-down, computationally efficient, principled algorithm that intertwines row and column clusterings at all stages and is guaranteed to reach a local minimum in a finite number of steps. We have presented examples to motivate the new concepts and to illustrate the efficacy of our algorithm. In particular, word-document matrices that arise in information retrieval are known to be highly sparse [7]. For such sparse high dimensional data, even if one is only interested in document clustering, our results show that co-clustering is more effective than a plain clustering of just documents. The reason is that when co-clustering is employed, we effectively use word clusters as underlying features and not individual words. This amounts to implicit and adaptive dimensionality reduction and noise removal leading to better clusters. As a side benefit, co-clustering can be used to annotate the document clusters.

## 3. PROPOSED APPROACH FRAMEWORK AND DESIGN

### 3.1 Problem Definition

we generalize the problem of substring selectivity estimation for Boolean predicates. Our novel idea is to capture correlations between Boolean query predicates in a space-e_ client but approximate manner. We employ a Monte Carlo technique called set hashing to succinctly represent the set of strings containing a given substring predicate as a signature vector of hash values. Correlations among substring predicates can then be generated by operating on these signatures. We present an algorithm to estimate the selectivity of any Boolean query and experimentally demonstrate the superiority of our approach. While there has been some work in estimating the selectivity of substring queries, the more general problem of estimating the selectivity of Boolean queries over substring predicates has not been studied.

### 3.2 Proposed Architecture and Design

EXALG to solve the EXTRACT problem. Figure 2 shows the different sub-modules of EXALG. Broadly, EXALG works in two stages. In the first stage (ECGM), it discovers sets of tokens associated with the same type constructor in the (unknown) template used to create the input pages.

In the second stage (Analysis), it uses the above sets to deduce the template. The deduced template is then used to extract the values encoded in the pages. This section outlines EXALG for our running example.

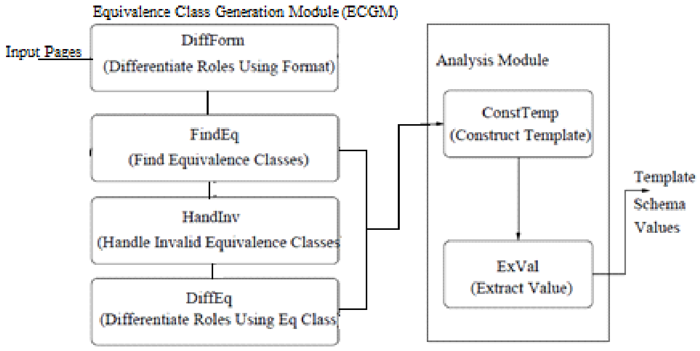
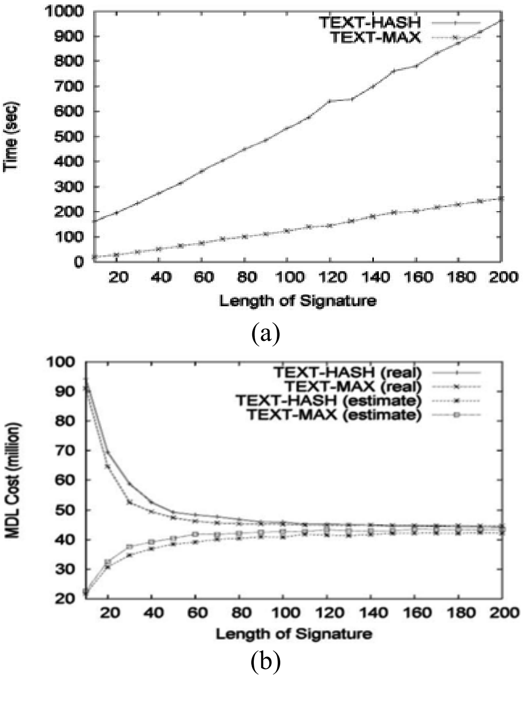

Equivalence Class Generation Module (ECGM)
Input Pages
DiffForm
(Differentiate Roles Using Format)
FindEq
(Find Equivalence Classes)
HandInv
(Handle Invalid Equivalence Classes
DiffEq
(Differentiate Roles Using Eq Class
Analysis Module
ConstTemp
(Construct Template)
ExVal
(Extract Value)
Template
Schema
Values

Figure 1: Modules of EXALG

In the first stage, EXALG (within Sub-module FINDEQ) computes "equivalence classes" — sets of tokens having the same frequency of occurrence in every page in Pe. An example of an equivalence class(call εe1) is the set of 8 tokens{<html>,<body>,Book.....,</html>} where each token occurs exactly once in every input page. There are other equivalence classes. EXALG retains only the equivalence classes that are large and whose tokens occur in a large number of input pages. We call such equivalence classes _ LFEQs (for Large and Frequently occurring Equivalence classes). For the running example there are two LFEQs. The first is εe1 shown above. The second, which we call εe3 consists of the 5 tokens. {<li>Review,Rating, Text,</li> Each token of εe3 occurs once in Pe1 twice in Pe2 and so on. *The basic intuition behind* LFEQ*s is that it is very unlikely for* LFEQ*s to be formed by "chance". Almost always,* LFEQ*s are formed by tokens associated with the same type constructor in the unknown) template used to create the input pages.* This intuition is easily verified for the running example where all tokens of εe1(Spec. εe3) are associated with Te1(Res εe3) of Se in Te2.For this simple example, Sub-module HANDINV does not play any role, but for real pages HANDINV detects and removes "invalid" LFEQs — those that are not formed by tokens associated with a type constructor.

For this simple example, Sub-module HANDINV does not play any role, but for real pages HANDINV detects and removes "invalid" LFEQs — those that are not formed by tokens associated with a type constructor.

## 4. Work Done



(a)

(b)

Fig. 2. Performance study varying length of signature and essential path threshold. (a) Execution times. (b) MDL costs.

The result is provided in Fig. 13c. In x-axis, 1 means itself and 0.5 means 0:5 _ t_. When the threshold is zero, nothing is pruned by the threshold but, with a small threshold such as 0:1 _ t_, the number of essential paths evidently decreases. Between 0:1 _ t_ and t_, the number of essential paths is almost the same but that with 1:1 suddenly decreases by about 90 percent. It shows that the paths from contents are eliminated by a small threshold such as 0:1 _ t_ and almost all paths from templates survive until the threshold becomes t_. If the threshold is too large, only generally common paths such as "Document nhhtmlinhbodyi" remain. Thus, we can conclude that t_ is very effective to identify templates.

Evaluation of clustering results. We report in detail the clustering results of TEXT-MAX with 1,000 documents. We manually opened all the documents and checked the correctness of each cluster. TEXT-MAX partitioned 1,000 documents into 77 clusters. Among them, 32 clusters cover 833 documents and the rest of clusters have a single document or very small number of documents. If a cluster has too few instances of its template, the template from the cluster is not reliable. Since Rank Mass crawled documents without considering the template extraction, some clusters have only few instances.

## 5. Conclusion and Future Work

This paper presented an algorithm, EXALG, for extracting structured data from a collection of web pages generated from a common template. EXALG first discovers the unknown template that generated the pages and uses the discovered template to extract the data from the input pages. EXALG uses two novel concepts, equivalence classes and differentiating roles, to discover the template. Our experiments on several collections of web pages, drawn from many well-known data rich sites, indicate that EXALG is extremely good in extracting the data from the web pages. Another desirable feature of EXALG is that it does not completely fail to extract any data even when some of the assumptions made by EXALG are not met by the input collection. In other words the impact of the failed assumptions is limited to a few attributes.

## References

[1]. Arasu and H. Garcia-Molina, "Extracting Structured Data from Web Pages," Proc. ACM SIGMOD, 2003.

[2]. 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.

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

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

[5]. 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), 2001.

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

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

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

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

[10]. F. Pan, X. Zhang, and W. Wang, "Crd: Fast Co-Clustering on Large Data Sets Utilizing Sampling-Based Matrix Decomposition," Proc. ACM SIGMOD, 2008.