Automatic Wrapper Generation for Metasearch Using Ordered Tree Structured Patterns

Kazuhide Aikou[1], Yusuke Suzuki[1],
Takayoshi Shoudai[1], and Tetsuhiro Miyahara[2]

[1] Department of Informatics, Kyushu University, Kasuga 816-8580, Japan
{k-aikou, y-suzuki, shoudai}@i.kyushu-u.ac.jp
[2] Faculty of Information Sciences,
Hiroshima City University, Hiroshima 731-3194, Japan
miyahara@its.hiroshima-cu.ac.jp

Abstract. A wrapper is a program which extracts data from a web site and reorganizes them in a database. Wrapper generation from web sites is a key technique in realizing such a metasearch system. We present a new method of automatic wrapper generation for metasearch using our efficient learning algorithm for *term trees*. Term trees are ordered tree structured patterns with structured variables, which represent structural features common to tree structured data such as HTML files.

1 Introduction

Due to the rapid growth of HTML files at the Web space, it is important to extract useful information from the vast Web space. Since general-purpose search engines are useful but not universal, many organizations have their own search engines on their web sites, which are called *search sites* [3]. To support unified access to multiple search sites, we have developed a metasearch system for search sites. Wrapper generation from web sites has been extensively studied [1–7,9] and is a key technique in realizing such a metasearch system. However only a few automatic technique is based on theoretical foundations of learning theory. In this paper, we present a new method of automatic wrapper generation for metasearch engines for search sites. Our learning algorithm from tree structured data, called MINL algorithm [10], plays important role in this method.

Our approach of wrapper generation from web sites has the following advantages. MINL algorithm is unsupervised and needs no labeled examples which are positive and negative examples. The algorithm needs only a small number of sample HTML files of a target web site which are considered to be positive examples. Our approach has a firm theoretical foundation based on Computational Learning Theory [10]. Term trees, our representation of ordered tree structured patterns, have rich representing power and are useful to Web mining and semistructured data mining [8]. According to Object Exchange Model, we treat semistructured data as tree structured data. Since tree based wrappers are shown to be more powerful than string based wrappers [4, 9], we use *term*

G.I. Webb and Xinghuo Yu (Eds.): AI 2004, LNAI 3339, pp. 1030–1035, 2004.

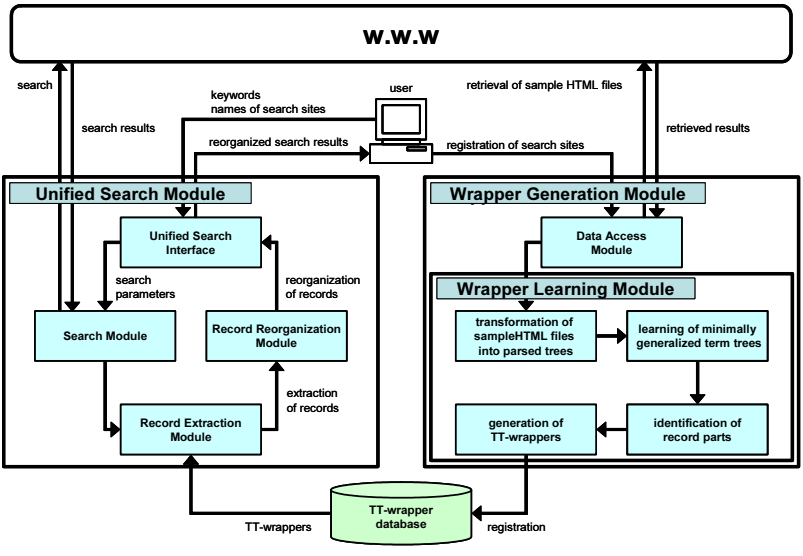

Fig. 1. Architecture of our system of metasearch from search sites

trees [10], which are ordered tree patterns with structured variables. A variable in a term tree can match an arbitrary subtree, which represents a field of a semistructured document. As a special case, a contractible variable can match an empty subtree, which represents a missing field in a semistructured document. Since semistructured documents have irregularities such as missing fields, a term tree with contractible variables is suited for representing tree structured patterns in such semistructured documents.

The key concept of our system is *minimally generalized term trees* obtained from tree structured data, which is briefly explained in Sec. 2. Let S be a set of trees each of which is transformed from an HTML file in a given HTML dataset. A *term tree wrapper generated by* S is a tuple (t, H) where t is one of minimally generalized term trees explaining S and H is a subset of variables of t. We call a term tree wrapper a **TT-wrapper**, for short. We note that search sites always output HTML files according to certain previously fixed rules. If we focus on one search site, the trees outputted by the site have no significant difference in the shapes. Thus, it is natural to guess that we can obtain a unique minimally generalized term tree for HTML files outputted by one search site. In this paper, we present a new method of automatic TT-wrapper generation for metasearch. The system provides unified access to multiple existing search sites (Fig. 1).

2 Term Trees with Contractible Variables

In this section, we give a rough definition and an example rather than give a full technical definition of a term tree. The reader is referred to [10] for details.

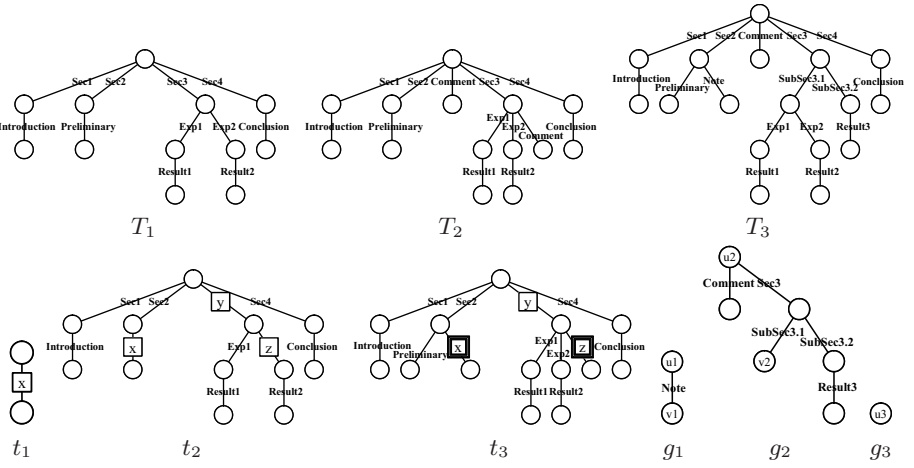

Fig. 2. Term trees t_1, t_2 and t_3 and trees T_1, T_2, T_3, g_1, g_2 and g_3. An uncontractible (resp. contractible) variable is represented by a single (resp. double) lined box with lines to its elements. The upper right tree T_3 is obtained from t_3 by replacing variables x, y, z with g_1, g_2, g_3, respectively (See [10])

Let $T = (V_T, E_T)$ be a rooted tree with ordered children, called an *ordered tree*, or a *tree* where V_T is a set of vertices and E_T is a set of edges. Let E_g and H_g be a partition of E_T, i.e., $E_g \cup H_g = E_T$ and $E_g \cap H_g = \emptyset$. And let $V_g = V_T$. A triplet $g = (V_g, E_g, H_g)$ is called an *ordered term tree*, or a *term tree* simply. And elements in V_g, E_g and H_g are called a *vertex*, an *edge* and a *variable*, respectively. We assume that every edge and variable of a term tree is labeled with some words from specified languages. There are two kind of variables, called *contractible variables* and *uncontractible variables*. A contractible variable may be considered to be an erasing variable, which must be adjacent to a leaf and can be replaced with any ordered tree including a singleton vertex. An uncontractible variable can appear anywhere in a term tree and be replaced with any ordered tree of at least 2 vertices. Variables with the same label must be replaced with the same tree. This rule often makes computational problems harder. Then we assume that all labels of variables in a term tree are mutually distinct.

Let t be a term tree. The *term tree language* of t, denoted by $L(t)$, is the set of all trees which are obtained from t by substituting trees for variables in t. We say that t *explains* a given set of trees S if $S \subseteq L(t)$. A *minimally generalized* term tree t explaining S is a term tree t which satisfies the following conditions: (i) t explains S, and (ii) $L(t)$ is minimal among all term tree languages which contain all trees in S. For example, the term tree t_3 in Fig. 2 is a minimally generalized term tree explaining T_1, T_2 and T_3. And t_2 is also minimally generalized term trees, with no contractible variable, explaining T_1, T_2 and T_3. A term tree t_1 is overgeneralized and meaningless, since t_1 explains any tree of at least 2 vertices. A term tree using contractible and uncontractible variables can express the structural feature of trees more correctly than a term tree us-

procedure GenTTwrapper(S, p, ϵ);
begin
Let t be a minimally generalized term tree explaining S;
foreach u which has at least p children **do**
begin
Let $c_1, \ldots, c_m$ all ordered children of u;
Let A^u be an $m \times m$ zero matrix;
for $i := 1$ **to** m **do**
for $j := i + 1$ **to** m **do**
if $t[c_i]$ and $t[c_j]$ are ϵ-equivalent **then**
$A^u[i, j] := 1$;
if A^u has a $q \times q$ nonzero submatrix which appears just $p - 1$ times horizontally **then** u is the parent of all records and **break**
end;
Let H be the set of all variables of $t[u]$;
return (t, H)
end;

$$A^u = \begin{pmatrix} 0\,0\,0\,0\,0\,0\,0\,0\,1\,0\cdots 0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,0\,0\cdots 0\,0\,0 \\ 0\,0\,0\,0\,\boxed{1\,0}\,\boxed{1\,0}\,\boxed{1\,0}\cdots\boxed{1\,0}\,0 \\ 0\,0\,0\,0\,\boxed{0\,0}\,\boxed{0\,0}\,\boxed{0\,0}\cdots\boxed{0\,0}\,0 \\ 0\,0\,0\,0\,0\,0\,1\,0\,1\,0\cdots 1\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,0\,0\cdots 0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,1\,0\cdots 1\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,0\,0\cdots 0\,0\,0 \\ \vdots \end{pmatrix}$$

Fig. 3. Procedure GenTTwrapper outputs a TT-wrapper for search results outputted by a fixed search site. The right matrix is an image of a boolean matrix A^u after procedure GenTTwrapper

ing only uncontractible variables. Then we consider that t_3 is a more precious term tree than t_2. We gave an algorithm, called **MINL** algorithm, for finding a minimally generalized term tree t explaining a given set of trees S which runs in $O(N_{min}^2 N_{max}|S|)$ time where $N_{\max}$ and $N_{\min}$ be the maximum and minimum numbers of vertices of trees in S, respectively [10].

3 Automatic Wrapper Generation for Metasearch

Our system of metasearch from search sites consists of two main modules, **Wrapper Generation Module** and **Unified Search Module**. The first module generates TT-wrappers from sample HTML files from search sites. When our system receives a user query, the second module collects and reorganizes the search results from the registered search sites by using corresponding TT-wrappers, and displays the unified search results to the user (Fig. 1) .

We describe the formal algorithm in Fig. 3. Let S be a set of trees converted from search results by a certain search site. Each tree in S corresponds to one search result and each result contains a fixed number of records. We assume that all trees have exactly $p \geq 2$ records. Let $t = (V_t, E_t, H_t)$ be a minimally generalized term tree explaining S. The purpose of a TT-wrapper in metasearch is to extract all and only records from trees obtained from newly outputted search results. Thus we need to specify p groups of subtrees corresponding to each of p records. We note that all roots of these subtrees must have the unique

parent. Let u be a vertex of t, which is a candidate of the parent of records, and $c_1, \ldots, c_m$ all ordered children of u. $t[v]$ denotes the term subtree of t which is induced by v and the descendants of v. First we find the candidates of the parent. Each record consists of a fixed but unknown number of subtrees. Let $q \geq 1$ be the number of subtrees corresponding to a record. For example, for some k, k' $(0 \leq k < k + q < k')$, q subtrees $t[c_{k+1}], \ldots, t[c_{k+q}]$ construct one record and other q subtrees $t[c_{k'+1}], \ldots, t[c_{k'+q}]$ construct another record and so on. In order to find such groups of subtrees, we use MINL algorithm again for testing whether or not given two term subtrees are approximately the same. For a term tree $t = (V_t, E_t, H_t)$, let $T\langle t\rangle = (V_T, E_T)$ be the tree obtained from t where $V_T = V_t$ and $E_T = E_t \cup \{\{u, v\} \mid [u, v] \in H_t\}$. All variable labels of t are thought of edge labels in $T\langle t\rangle$. For a fixed number $0 \leq \epsilon \leq 1$ and two term trees t and t', we say that t and t' are *ϵ-equivalent* if $||t| - |t'|| \leq \epsilon \cdot |t|$, $||\mathrm{MINL}(\{T\langle t\rangle, T\langle t'\rangle\})| - |t|| \leq \epsilon \cdot |t|$, and $||\mathrm{MINL}(\{T\langle t\rangle, T\langle t'\rangle\})| - |t'|| \leq \epsilon \cdot |t|$, where $\mathrm{MINL}(S)$ is an output term tree of our MINL algorithm. We find p groups $(c_{11}, \ldots, c_{1q}), \ldots, (c_{p1}, \ldots, c_{pq})$ of q children from all children $c_1, \ldots, c_m$ such that (i) $t[c_{ij}]$ and $t[c_{i'j}]$ are ϵ-equivalent for all $1 \leq i < i' \leq p$ and $1 \leq j \leq q$ and (ii) $c_{i,j+1}$ is the immediately right sibling of c_{ij} for all $1 \leq i \leq p$ and $1 \leq j < q$. The procedure GenTTwrapper (Fig. 3) generates a TT-wrapper which extracts all records of search results outputted by a fixed search site.

4 Experimental Results

We implemented our system by C on a PC with CPU Celeron 2.0 GHz and 512 MB memory. We chose total 25 search sites which output exactly 10 search results in English. The abbreviated names of these sites are given in the 1st column of Table 1. Firstly the system automatically gave two popular keywords to

Table 1. Total 17 TT-wrappers are obtained from 25 search sites by using our system

Search Sites	Page Size	Result
Pioneer	218	**OK**
GENERAL MILLS	241	**OK**
Yamanouchi	268	**OK**
VOS	303	*NG*
TOYOBO	324	**OK**
Golden Circle	329	*NG*
Oracle	341	**OK**
MAZDA	358	**OK**
Microsoft	359	**OK**
Seiko	378	**OK**
DAIHATSU	383	**OK**
RRD	404	**OK**
ISUZU	441	**OK**

Search Sites	Page Size	Result
TOSHIBA	454	**OK**
BMW	472	**OK**
GWF	482	**OK**
HONDA	485	*NG*
FUJITSU	488	**OK**
OMRON	491	*NG*
KDDI	527	*NG*
ASAHI_KASEI	547	*NG*
Yahoo	552	**OK**
SONY	578	*NG*
Teoma	608	*NG*
Heinz	608	**OK**

each search site, and retrieved 2 search results for each keyword. Next the system converted the first displayed pages of these 2 results into 2 trees. The entry of the 2nd column of Table 1 shows the size of one of the trees. The mark "OK" means that the system succeeded to get a TT-wrapper of the corresponding search site. The success rate decreases in proportion to the tree size. It is natural to consider that a larger tree can contain a lot of groups of similar subtrees which might become records. Since our current system uses only knowledge of structures of search results, it often failed to generate TT-wrappers for relatively large search sites. From these observations, we are now developing new similarity measures between two subtrees with text information in order to extract records exactly.

5 Conclusions

In order to provide unified access to multiple search sites, we have presented a new method of automatic wrapper generation for metasearch for search sites, by using our learning algorithm MINL for term trees. We have reported our metasearch system for search sites. Our method uses a new type of wrappers, called TT-wrappers, which are tree structured patterns with structured variables and useful to extract information from HTML files in search sites.

References

1. V. Crescenzi, G. Mecca, and P. Merialdo. ROADRUNNER: Towards automatic data extraction from large web sites. *Proc. VLDB-2001*, pages 109–118, 2001.
2. R. Dale, C. Paris, and M. Tilbrook. Information extraction via path merging. *Proc. AI-2003,Springer-Verlag, LNAI 2903*, pages 150–160, 2003.
3. S. Hirokawa and HumanTecnoSystem Co. Research and development of the next-generation search engine by dynamic integration of search sites (in Japanese). *http://daisen.cc.kyushu-u.ac.jp/thesis/thesis.pdf*, 2002.
4. D. Ikeda, Y. Yamada, and S. Hirokawa. Expressive power of tree and string based wrappers. *Proceedings of IJCAI-2003 Workshop on Information Integration on the Web (IIWeb-2003)*, pages 21–26, 2003.
5. N. Kushmerick. Wrapper induction: efficiency and expressiveness. *Artificial Intelligence*, 118:15–68, 2000.
6. A. H. F. Laender, B. A. Ribeiro-Neto, A. S. da Silva, and J. S. Teixeira. A brief survey of web data extraction tools. *SIGMOD Record*, 31(2):84–93, 2002.
7. B. Liu, R. L. Grossman, and Y. Zhai. Mining data records in web pages. *Proc. KDD-2003, AAAI Press*, pages 601–606, 2003.
8. T. Miyahara, Y. Suzuki, T. Shoudai, T. Uchida, K. Takahashi, and H. Ueda. Discovery of maximally frequent tag tree patterns with contractible variables from semistructured documents. *Proc. PAKDD-2004, Springer-Verlag, LNAI 3056*, pages 133–134, 2004.
9. H. Sakamoto, Y. Murakami, H. Arimura, and S. Arikawa. Extracting partial structures from html documents. *Proceedings of the Fourteenth International Florida Artificial Intelligence Research Society Conference,2001*, pages 264–268, 2001.
10. Y. Suzuki, T. Shoudai, S. Matsumoto, T. Uchida, and T. Miyahara. Efficient learning of ordered and unordered tree patterns with contractible variables. *Proc. ALT-2003, Springer-Verlag, LNAI 2842*, pages 114–128, 2003.