Wrapping HTML Tables into XML

Shijun Li[1], Mengchi Liu[2], and Zhiyong Peng[3]

[1] School of Computer, Wuhan University, Wuhan 430072, China
shjli@public.wh.hb.cn
[2] School of Computer Science, Carleton University,
1125 Colonel By Drive, Ottawa, ON, Canada K1S 5B6
mengchi@scs.carleton.ca
[3] State Key Lab of Software Engineering, Wuhan University,
Wuhan 430072, China
peng@whu.edu.cn

Abstract. HTML tables are information rich and are used frequently in HTML documents, but they are mainly presentation-oriented and are not really suited for database applications. To wrap HTML tables, in this paper, we introduce a conceptual model for HTML tables, and based on it, we present a new approach to wrap HTML tables into XML documents. It can automatically convert basic HTML tables, nested tables, composite HTML table and the tables without marked headings, into XML documents.

1 Introduction

Tables are used frequently in HTML documents and are information rich. They are intended for human users, rather than for computer applications. Being able to capture the meanings of HTML tables automatically has many potential applications. In HTML documents, the cells in HTML tables may span multiple rows or columns. So HTML tables may have nested heading; and each row or column may have different number of data cells that may not aline. Also the cells may be not atomic as the cells may contain nested table, list etc. It is not trivial to automatically wrap them into XML document, which escribes contents of data and can be processed by database applications.

As the number of rows or columns spanned by a cell is set by *rowspan* or *colspan* attributes, we can normalize HTML tables by inserting redundant cells into them. Based on their normalized tables, we introduce a conceptual model for HTML tables, in which an HTML table consists of seven elements: captions, ULC (upper-left corner), row headings, column headings, the affiliation of ULC (affiliating to row headings or column headings), data, and attribute-value pairs. Based on the conceptual model, we introduce a mapping rule to map the normalized HTML tables to XML documents. Since the mapping rule uses recursive function, it is well-suited for capturing the nested table, list and the nested structure marked by formatting information, in cells of HTML tables.

For some complex HTML tables that consist of sub-tables, we introduce the notion of *composite table* and wrap them into XML by decomposing them into sub-tables, and then wrap each sub-table by the mapping rule.

X. Zhou et al. (Eds.): WISE 2004, LNCS 3306, pp. 147–152, 2004.

Table 1. The normalized HTML table

ULC			$h_{1,q+1}$	...	$h_{1,n}$
			⋮		⋮
			$h_{p,q+1}$	...	$h_{p,n}$
$v_{p+1,1}$	...	$v_{p+1,q}$	$d_{p+1,q+1}$	...	$d_{p+1,n}$
⋮		⋮	⋮		⋮
$v_{m,1}$	...	$v_{m,q}$	$d_{m,q+1}$	...	$d_{m,n}$

The paper is organized as follows. Section 2 introduces a conceptual model for HTML tables. Section 3 presents the approach to wrapping HTML tables. Section 4 introduces the approach to wrapping composite HTML tables. Section 5 compares our work with other related work. Finally we conclude in Section 6.

2 Conceptual Model for HTML Tables

In HTML, table cells generally contain heading information via the *th* element and data via the *td* element. Table cells may span multiple rows and columns, which cause complex nested heading structure. To capture the attribute-value pairs in an HTML table, we need to normalize the HTML table. In an HTML table, if a *th* or *td* element contains $colspan = k$ or $rowspan = k$, it means that the particular cell of the *th* or *td* is to be expanded to $k-1$ more columns or $k-1$ more columns, starting from the current cell in the current row or collumn. By using the attributes colspan and rowspan in *th* or *td* elements, that is, by inserting redundant cells into an HTML table, we can normalize the HTML table in the form of Table 1.

Based on the general form of the normalized HTML tables as show in Table 1, we introduce a conceptual model for HTML tables as follows.

Definition 1. An HTML table T is denoted by
$T_{m\times n\times p\times q}(C, ULC, RH, CH, F, D, AD)$
where C is the content of the caption of T; ULC is the content of the upper-left corner of T; $ULC = \phi$, if $p = 0$ or $q = 0$; RH is the row headings of T; if $q = 0$, then $RH = \phi$, otherwise $RH = \{V_{p+1} = (v_{p+1,1}, \ldots, v_{p+1,q}), \ldots, V_m = (v_{m,1}, \ldots, v_{m,q})\}$; CH is the column headings of T; If $p = 0$, then $CH = \phi$, otherwise $CH = \{H_{q+1} = (h_{1,q+1}, \ldots, h_{p,q+1}), \ldots, H_n = (h_{1,n}, \ldots, h_{p,n})\}$; F is the affiliation of ULC; $F \in \{F_{RH}, F_{CH}, \phi\}$; if $ULC = \phi$, then $F = \phi$; if ULC is affiliated to row headings, then $F = F_{RH}$, otherwise $F = F_{CH}$; D is the data part of T; $D = \{d_{ij} \mid i = p+1, \ldots, m;\ j = q+1, \ldots, n\}$; AD is the attribute-value pairs of the table; if $p \neq 0$ and $q \neq 0$, then $AD = \{((V_i, H_j), d_{ij}) \mid V_i \in RH, H_j \in CH, d_{ij} \in D\}$; if $p = 0$ and $q \neq 0$ then$AD = \{(V_i, d_{ij}) \mid V_i \in RH, d_{ij} \in D\}$; if $q = 0$ and $p \neq 0$, then$AD = \{(H_j, d_{ij}) \mid H_j \in CH, d_{ij} \in D\}$.

If $p \neq 0$ and $q \neq 0$, then the table is called *two dimensional* tables. Otherwise it is called *one dimensional* tables. In the case of one dimensional table, if $p = 0$, the table is called column-wise table; if $q = 0$, the table is called row-wise table. If $p = 0$ and $q = 0$, the table is called *no-attribute* table.

3 Mapping Nested Structure of HTML Tables to XML

In the above conceptual model for HTML tables, if $F = F_{RH}$ (i.e. the ULC is affiliated to the row headings), then the nested structure of an HTML table can be expressed as (denoted in path expressions in XPath):
captions/ULC/row headings/column headings/data cell;
if $F = F_{CH}$ (i.e. the ULC is affiliated to the column heading), then the nested structure of an HTML table can be expressed as:
captions/ULC/column headings/row headings/data cell.
So we can map this nested structure of an HTML table to the corresponding XML document as follows.

Rule 1. Let an HTML table T be $T_{m\times n\times p\times q}(C, ULC, RH, CH, F, D, AD)$, let the value of C, ULC be c, ulc, respectively, and RH, CH, F, D, AD take the form in Definition 1. Let the rows that contain data be $r_{p+1}, r_{p+2}, \ldots, r_m$, and the columns that contain data be $c_{q+1}, c_{q+2}, \ldots, c_n$, where $0 \leq p \leq m$, $0 \leq q \leq n$. If $F = F_{RH}$ or $F = \phi$ and T is a two dimensional tables or T is a column-wise table (i.e. $p = 0$), then we use row-priority conversion as follows:
$\psi(T) =< c >< ulc > \psi(r_{p+1}) \oplus \psi(r_{p+2}) \oplus \ldots \oplus \psi(r_m) < /ulc >< /c >$.
Otherwise we use column-priority conversion as follows:
$\psi(T) =< c >< ulc > \psi(c_{q+1}) \oplus \psi(c_{q+2}) \oplus \ldots \oplus \psi(c_n) < /ulc >< /c >$,
where ψ is the converting function which converts an HTML document into XML one, and $\oplus$ is the operation that concatenates two character strings and

$$\begin{aligned}
\psi(r_i) = &< v_{i,1} > \ \ldots \ < v_{i,q} > \\
&< h_{1,q+1} > \ldots < h_{p,q+1} > \psi(d_{i,q+1}) < /h_{p,q+1} > \ldots < /h_{1,q+1} > \\
&\ldots \\
&< h_{1,n} > \ \ldots \ < h_{p,n} > \psi(d_{i,n}) < /h_{p,n} > \ldots < /h_{1,n} > \\
&< /v_{i,q} > \ \ldots \ < /v_{i,1} > \\
\psi(c_j) = &< h_{1,j} > \ \ldots \ < h_{p,j} > \\
&< v_{p+1,1} > \ldots < v_{p+1,q} > \psi(d_{p+1,j}) < /v_{p+1,q} > \ldots < /v_{p+1,1} > \\
&\ldots \\
&< v_{m,1} > \ \ldots \ < v_{m,q} > \psi(d_{m,j}) < /v_{m,q} > \ldots < /v_{m,1} > \\
&< /h_{p,j} > \ \ldots \ < /h_{1,j} >
\end{aligned}$$

where $i = p+1, \ldots, m;\ j = q+1, \ldots, n;\ 0 \leq p \leq m;\ 0 \leq q \leq n$.

After an HTML table is converted into an XML document by using above rule, we need to refine the XML document by merging the content of the same XML element, i.e. by using the following equation:
$< t1 >< t2 > s2 < /t2 >< /t1 >< t1 >< t3 > s3 < /t3 >< /t1 >$
$=< t1 >< t2 > s2 < /t2 >< t3 > s3 < /t3 >< /t1 >$.

Rule 1 covers not only two dimensional tables but also one dimensional tables. In Rule 1, we use F (i.e. the affiliation of ULC) to determine if using row-priority or column-priority conversion. Generally, the ULC is affiliated to the

row headings except that it ends with a colon. So we use the following heuristic approach: If the ULC ends with a colon, then the ULC is affiliated to the row headings. Otherwise it is affiliated to the column headings.

There are some HTML tables without marked headings via *th* element in HTML documents. To convert them, the key is to recognize their headings. For HTML tables without marked headings, the authors generally use different formatting information, which are mainly font (size, bold and italic), to mark headings for visual easy recognition by uses. So we can recognize their headings by formatting information implied by HTML tags [9].

The mapping rule Rule 1 use recursive function so that it is particularly well-suited for converting the nested structure in cells of HTML tables. Given the conversion rules for text-level elements and list elements, which are introduced in [9], by using Rule 1, we can wrap nested tables, nested lists and the nested structure marked by formatting information in cells of HTML tables into XML.

4 HTML Composite Tables

For the HTML tables that have separate headings and more than one caption, we introduce the notion of *composite table*.

Definition 2. Let T be an HTML table, and T have n rows: $r_1, r_2, \ldots, r_n$, where r_i is the ith row of table T, $i = 1, 2, \ldots, n$. If T has at most one caption and no separate headings, we call T a basic table. If the rows of T can be grouped as: $r_1, \ldots, r_{k_1}$; $r_{k_1+1}, \ldots, r_{k_2}$; $\ldots$; $r_{k_{p-1}+1}, \ldots, r_{k_p} = r_n$; where $0 \leq k_1 < k_2 < \ldots < k_p = n$, and the row set of each group composes a basic table called sub-table, then we call T a row-wise composite table.

We can similarly give the definition of column-wise composite tables. Table 2 is a sample of row-wise composite tables. For the rows of a table T that contain only one cell, if it is the first row, then we can take it as the caption of T; otherwise we can take it as the *sub-caption* of T. For example, in Table 2, the first row is the caption of the Table 2, and the fourth row and the seventh row are the caption of the second and third sub-table of Table 2, respectively. It is much more difficult to capture the nested structure in the row-wise composite tables than the basic tables discussed in Section 3, since the column heading of the first sub-table is shared by the other sub-tables, and the caption and ULC of the first sub-table are the caption and ULC of the composite table. For example, in Table 2, the second row is the column heading, which is shared by the second and third sub-table. To convert row-wise composite table, we need add column heading of the first sub-table to the other sub-tables.

Rule 2. Let T be a row-wise composite table which consists of n sub-tables: $T_1, T_2, \ldots, T_n$, and the column heading of T_1 be h. Then dismiss the caption (let its value be c) and ULC (let its value be ulc) from the first sub-table, and add h to each of $T_2, \ldots, T_n$ as headings. After that, let we turn $T_1, T_2, \ldots, T_n$ into $T_1^{'}, T_2^{'}, \ldots, T_n^{'}$, respectively.
Then $\psi(T) =< c >< ulc > \psi(T_1^{'}) \oplus \ldots \oplus \psi(T_n^{'}) < /ulc >< /c >$,
where $\psi(T_1^{'}), \ldots, \psi(T_n^{'})$ can be computed by Rule 1 using row priority conversion.

Table 2. A sample composite table

Percentage of children enrolled				
Characteristic	1991	1995	1999	2001
Total students	52.8	55.1	59.7	56.4
Poverty status				
Below poverty	44.2	45.1	51.4	46.7
At or above poverty	55.7	58.8	62.3	59.1
Race/ethnicity				
White	54.0	56.9	69.0	59.0
Black	58.3	59.5	73.2	63.7
Hispanic	38.8	37.4	44.2	38.8

Example 1. Consider the row-wise composite table T in Table 2, which consists three sub-tables. By using Rule 2, the heading of the first sub-table, that is the second row need be added into the second and the third sub-table. After that its three sub-tables are turned into the following three sub-table: $T_1{}'$, which consists of tw0 rows: the second and third row of Table 2; $T_2{}'$, which consists of four rows: the forth, the second, the fifth and the sixth row of Table 2; and $T_3{}'$, which consists of five rows: the seventh, the second, the eighth, the ninth and the tenth row of Table 2. By using Rule 2, as c = Percentage of children enrolled, ulc =Characteristic,

$\psi(T)$ =<Percentage-of-children-enrolled>
<Characteristic> $\psi(T_1^{'}) \oplus \psi(T_2^{'}) \oplus \psi(T_3^{'})$ </Characteristic>
</Percentage-of–enrolled>.

By using Rule 1, we can easily get the results of $\psi(T_1^{'})$, $\psi(T_2^{'})$, and $\psi(T_3^{'})$, and then the result of $\psi(T)$, that is, the wrapped XML document of Table 2.

5 Comparison with Other Work

The manual approaches w4f [6], Tismmis [3], XWrap [7], and DOM-based approach [5] have strong expressive capability, but they often involve the writing of complicated code. Lixto [10] manage to avoid the writing of code by providing a fully visual and interactive user interface. The induction approach [1] uses machine-learning techniques to generate wrappers automatically which need to be trained under human supervision. Ontology-based approach [2] also involves the writing of RDF code. [4] presents an automatic approach to converting HTML tables. However, it cannot convert the tables without marked heading and composite tables. [8] presents an automatic approach to extracting attribute-value pairs. But it gives no formal approach to capture the attribute-value pairs in HTML tables with nested headings, and cannot capture nested structures in table cells. In this paper, we present a new approach to wrapping HTML tables into XML, and have implemented it using java and performed experiment on HTML tables on the web related to car selling, national statistic information, etc., with the average precision being about 93% and the recall about 98%. We

must stress the fact that some of the presented rules are based on heuristics and might fail. Compared with the other related approaches, our approach introduces a conceptual model for HTML tables, which is the basis of our approach so that it is more formal and precise. And it uses recursive function, so that it can wrap nested HTML tables. In addition, it introduces the notion of HTML composite tables and the approach to wrap them. While the other approaches cannot apply the HTML composite tables or just cover part of them.

6 Conclusion

We have introduced a conceptual model for HTML tables, and based on it, we present a new approach that automatically captures the nested structure in HTML tables and converts them into XML documents, and a system implementing this new approach, which can automatically wrap the selected tables, including nested tables, composite HTML table into XML documents with a good performance. It can be used as an efficient auxiliary tool in recycling HTML tables as XML documents.

References

1. Nicholas Kushmerick, D. Weld, and R. Doorenbos. Wrapper Induction for Information Extraction. In proceedings of IJCAI (1997), 729–737
2. David W. Embley, Cui Tao, and Stephen W. Liddle. Automatically Extracting Ontologically Specified Data from HTML Table of Unknown Structure. In proceedings of ER (2002), 322–337
3. Joachim Hammer, Hector Garcia-Molina, Svetlozar Nestorov, Ramana Yerneni, Markus M. Breunig, and Vasilis Vassalos. Template-Based Wrappers in the TSIMMIS System. In proceedings of SIGMOD Conference (1997), 532–535
4. Seung Jin Lim, Yiu-Kai Ng, and Xiaochun Yang. Integrating HTML Tables Using Semantic Hierarchies And Meta-Data Sets. In proceedings of IDEAS (2002), 160–169
5. Suhit Gupta, Gail E. Kaiser, David Neistadt, and Peter Grimm. DOM-based content extraction of HTML documents. In proceedings of WWW (2003), 207–214
6. Arnaud Sahuguet and Fabien Azavant. Building Intelligent Web Applications Using Lightweight Wrappers. Data and Knowledge Engineering (2001), 36(3): 283–316
7. L. Liu, C. Pu, and W. Han. XWrap: An Extensible Wrapper Construction System for Internet Information. In proceedings of ICDE (2000) 611–621
8. Yingchen Yang and Wo-Shun Luk. A Framework for Web Table Mining. In proceedings of WIDM'02 (2002), 36–42
9. Shijun Li, Mengchi Liu, Tok Wang Ling, and Zhiyong Peng. Automatic HTML to XML Conversion. In proceedings of WAIM (2004), 714–719
10. R. Baumgartner, S. Flesca and G. Gottlob. Visual Web Information Extraction with Lixto. In proceedings of VLDB (2001), 119–128