SG-WRAM: Schema Guided Wrapper Maintenance

Xiaofeng Meng, Haiyan Wang, Dongdong Hu, Mingzhe Gu
Information School
Renmin University of China, Beijing, 100872, China
xfmeng@mail.ruc.edu.cn

1. Introduction

The World Wide Web has become one of the most important connections to various sources of information. A large proportion of the data is embedded in HTML documents. This language serves the visual presentation of data in Internet browser, but does not provide semantic information for the data presented. This form of data presentation is, therefore, inappropriate for the demands of automated, computer assisted information management system. In particular, if data from different sources needs to be combined, it is necessary to develop special and often complex programs to automate the data extraction.

Wrappers are specialized program routines to fulfil such tasks. They automatically extract data from Internet web sites and convert the information into a structured format. As the manual coding of wrappers is time-consuming and error-prone process, different methods [1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12] have been proposed to automate the wrapper generation process.

As a rule, however, a specially developed wrapper is required for each individual data source, because of the different and unique structures of web sites. The WWW is also extremely dynamic and continually evolving, which results in frequent changes in the structures of web documents. Consequently, wrappers may stop working when the structures of the corresponding documents are changed no matter how they have been generated. It is often necessary to constantly update or even completely rewrite existing wrappers, in order to maintain the desired data extraction capabilities. The simplest way to maintain wrappers is to re-create wrappers using the new HTML documents. Obviously, this method is inefficient in that the maintenance depends mostly on the system developers.

In this demo, we propose a novel schema-guided approach for wrapper maintenance, called SG-WRAM, which is based on our previous work, a schema-guided wrapper generation system (SG-WRAP[8,9]). SG-WRAP can generate a wrapper to extract data from an HTML document to produce an XML document conforming to the user-defined Schema.

Although changes of HTML documents are extremely various, some features of desired information in previous document, e.g. syntactic features, data pattern, notation and underlying schemas are still preserved in the changed one. Syntactic features, data pattern and notation can be easily obtained from schemas, previous rules and extracting results. Therefore, it is feasible to recognize data items in the changed document using these features.

Based on these observations, we fulfill the maintenance following four sequential steps. At First, syntactic features, data pattern and notation are obtained from the schema, previous rule and extracted results, then they are used to recognize the data items. After that, they are grouped according to the given schema. Each group is an instance of the given schema. At last, the representative instances are selected to re-induce the extraction rule. We name these four steps as *features discovery*, *item recovery*, *block configuration* and *wrapper reparation* respectively.

Our schema guided method for wrapper maintenance has several unique features comparing to the related work.

- We make good use of schema, which is given by user during the process of wrapper generation, to assist the procedures of *item recovery* and *block configuration*;
- Our experience with real-life web documents shows that our method can deal with the changes from simple to complex including context shift, structural shift [12] and hybrid changes;
- In our system, we give different method for simple changes in which condition a part of the rule is disabled and the complex changes in which condition most of the rule is disabled. That makes the re-inducted rule more accurate and complete.

2. System Overview

Figure 1 depicts the system architecture of wrapper extraction and maintenance. The whole system consists of three major components: wrapper generator, wrapper executor and wrapper maintainer.

Wrapper generator offers a GUI, whereby the relevant data within an HTML document is highlighted with a mouse, and the program then generates a wrapper based on the specified information [8, 9].

Wrapper executor provides the executing environment for wrapper and manages the result extracted by wrapper. Once a wrapper fails to extract documents correctly, that means the corresponding web documents are changed, it throws out a message to wrapper maintainer.

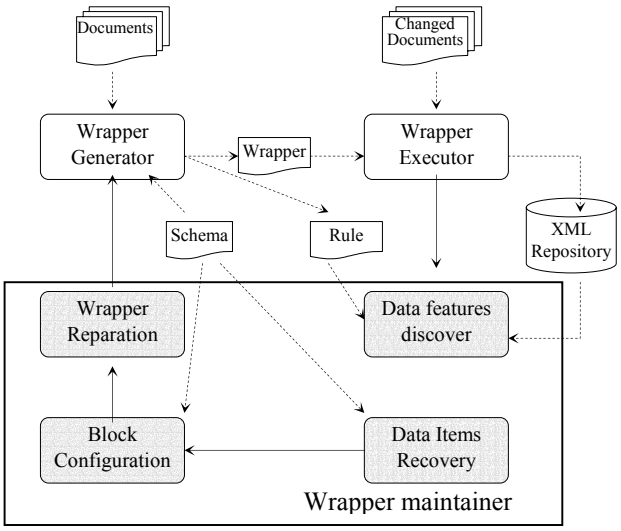

Figure 1. System Architecture

Wrapper maintainer is the core component we discussed in this demo. It completes the maintaining task based on four modules: *Data features discovery, data item recovery, Block Configuration* and *Wrapper Reparation*.

3. Maintenance Methods

Basically, we can generalize the changes into simple changes and complex changes.

Simple changes reflect small alterations on some nodes in the tree. For simple changes, extraction rules can be repaired by a few modifications.

Complex changes refer to changes that are relevant to the structure alterations. For complex changes, extraction rules can be re-induced after extraction instances can be re-obtained automatically.

First we give some concepts used in the maintenance.

3.1 Preliminaries

Semantic block: In our method, an HTML document can be viewed as a set of *semantic block*s, which are fragments of the HTML document conforming to the user-defined schema. In this demo paper, it's represented as *block* for short.

Candidate block: In the HTML tree, a sub-tree or sub-trees that are adjacent siblings are called candidate block `b′`.

3.2 Simple change maintenance

Because only small changes occur on the page, items without changes can be located using old extraction rules. Firstly, *Data features discovery* obtains data features of changed data items. Items unchanged can be easily identified according to their paths in the extraction rules. Then, with data features, *data item recovery* searches changed items within the same area of unchanged data items. At last, *Wrapper Reparation* repairs changed part in the original one. Thus, *Block Configuration* is not necessary in the simple change maintenance.

3.3 Complex Change Maintenance

Different from simple change maintenance, complex change maintenance use all four modules to fulfil the task.

Data features discovery finds the metadata for each data item within the extraction rule and learn the content features from schema and extracted results. Then, with these data features, *data item recovery* recognizes all possible data items by traveling the HTML tree of the changed document.

And in the real world, we find that data of the same topic are always putted together, which means that, in a tree view, the data are commonly in a sub-tree or several neighboring sub-trees. This kind of structure is much correlative with the structure of the user-defined schema. Based on such observation, we construct the module of *block configuration*. According to the user-defined schema, the *block configuration* module is used to group the recognized data items and construct candidate blocks to match candidate blocks with blocks defined in the schema. After that, the best candidate block is selected as an instance of the given schema, and from which the extraction rule is repaired. Meanwhile, when we group some date items, some noise of data items are removed for they can not be included by the candidate blocks.

At last, *Wrapper Reparation* picks up the representative instances to re-induce the extraction by *Wrapper generator.*

4. Related work

Recently, several methods are presented to address automatic repairing of web wrappers. Kushmerick [13, 14] defined a subproblem, the wrapper verification, and propose a solution that analyzes the page and extracted information and detects the page change with a given change. When the change is detected, the designer is notified; then s/he re-learns the wrapper from samples of the new format. Knoblock at el.[5] developed a method for wrapper repairing in the case of small mark-up change; it detects the most frequent patterns in the labeled strings; these patterns are searched in a page when the wrapper is broken. Recently, Chidlovskii [12] reported his automatic maintenance work. It repairs wrappers under the assumption of "small change". Although it has an advantage over other methods, it just repairs wrappers

when the sequence of data items is not changed in the new page. However, these approaches, any of which either did not achieve the ultimate goal or addressed a simplest fraction of the problem, have not resolved wrapper maintenance problem efficiently and effectively.

5. Conclusions

The system to be demonstrated is implemented in Java (Sun JDK1.3). During the demo, we will demonstrate the major algorithms used in SG-WRAM, i.e., the process of item recovery, block configuration and wrapper reparation.

Figure 2 shows some key scenes of the procedures of our wrapper maintenance. Figure 2.A shows the original page and the extracted results in the form of XML with the original wrapper. Figure 2.B shows the changed page and the results with the original wrapper, in which we will see that, since some parts of the page have changed, only parts of the Web document can be correctly extracted. Figure 2.C gives the comparison of original rule and the repaired rule. And the left part of Figure 2.D presents the procedures of item recovery, and the right part presents the procedure of block configuration in the process of wrapper maintenance.

Figure 2. Demonstration interface

Acknowledgements

This research was partially supported by the grants from 863 High Technology Foundation of China under grant number 2002AA116030 and from the Natural Science Foundation of China under grant number 60073014, 60273018.

References

[1] Ashish N, Knoblock C A. "Wrapper generation for semi-structured Internet sources". SIGMOD Record, 1997, 26(4): 8-15.

[2] Baumgartner R, Flesca S, Gottlob G.. "Visual Web Information Extraction with Lixto". In Proceedings of the Very Large Data Bases; 2001, 119-128.

[3] Doorenbos R, Etsionoi O, Weld D S. "A scalable comparison-shopping agent for the world-wide-web". In Proceedings of the First International Conference on Autonomous Agents, 1997, 39-48.

[4] Hammer J, Brenning M, Garcia-Molina H, Nestorov S, VassalosV, Yerneni R. "Template-based wrappers in the TSIMMIS system". In Proceedings of ACM SIGMOD Conference, 1997, 532-535.

[5] Knoblock C A, Lerman K, Minton S, Muslea I. "Accurately and Reliably Extracting Data from the Web: A Machine Learning Approach". Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 2000, 23(4): 33-41.

[6] Kushmerick N, Weil D, Doorenbos R. "Wrapper induction for information extraction". In Proceedings of International Joint Conference on Artificial Intelligence (IJCAI), 1997, 729-735.

[7] Liu L, Pu C, Han W. XWRAP: "An XML-enabled Wrapper Construction System for Web Information Sources". In Proceedings of ICDE, 2000, 611-621.

[8] Meng X F, Lu H J, Wang H Y, Gu M Z. "SG-WRAP: A Schema-Guided Wrapper Generator". In Proceedings of ICDE, 2002, 331-332.

[9] Meng X F, Lu H J, Wang H Y, Gu M Z. "Schema-Guided Data Extraction from the Web". Journal of Computer Science and Technology (JCST), 2002,17(4): 377-388.

[10] Muslea I, Minton S, Knoblock C A. STALKER: "Learning extraction rules for semistructured Web-based information sources". In Proceedings of AAAI: Workshop on AI and Information Integration, 1998, 74-81.

[11] Sahuguet A, Azavant F. "Building Light-Weight Wrappers for Legacy Web Data-Sources Using W4F". In Proceedings of VLDB, 1999, 738-741.

[12] Chidlovskii B. "Automatic repairing of Web Wrappers". In 3rd International Workshop on Web Information and Data Management, 2001, 24-30.

[13] Kushmerick N. "Wrapper verification". World Wide Web Journal, 2000, 3(2): 79-94.

[14] Kushmerick N. "Regression testing for wrapper maintenance". In Proceedings of AAAI, 1999,74-79