Graph Grammar Based Web Data Extraction

Amin Roudaki
Computer Science Department
North Dakota State University
amin.roudaki@ndsu.edu

Jun Kong
Computer Science Department
North Dakota State University
Jun.kong@ndsu.edu

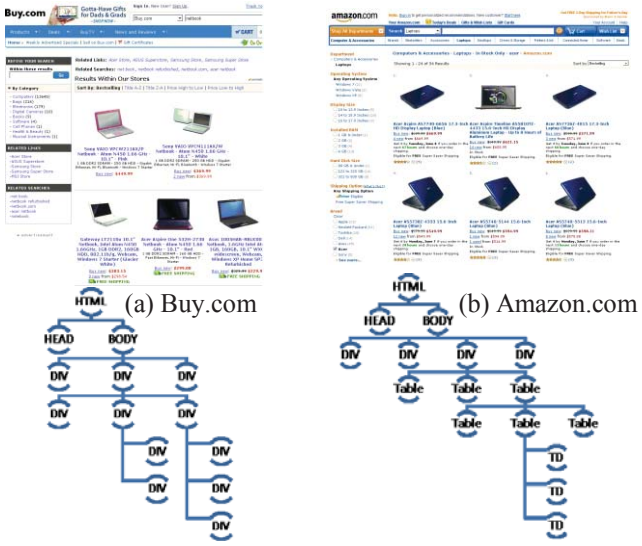
***Abstract*—Web data extraction becomes a hot topic after the invention of World Wide Web, because the large amount of information on the Web makes it challenging to retrieve useful information. Due to the diverse designs and presentations of information on different Web sites, it is hard to implement a general solution to extract data across different Web sites. This paper presents a novel method based on graph grammar to extract the same type of information from different Web sites without the need of training or adjustment. Our approach formalizes a common Web pattern as a graph grammar. Then, based on the visual layout and HTML DOM structure, a Web page is abstracted as a spatial graph that highlights the essential spatial relations between information objects. According to the defined graph grammar, a spatial parsing is performed on the spatial graph to extract structured records. We have evaluated our approach on twenty one different Web sites, and achieved the F1-score as 97.49% which shows promising performance.**

Keywords: Web Data Extraction; Graph Grammar; Wrapper

I. INTRODUCTION

Exploring useful information becomes increasingly difficult as the volume and diversity of available information rapidly grow. To efficiently discover knowledge from the vast amount of heterogeneous digital data on the Web, it is critical to extract meaningful contents from Web pages and organize extracted information in a structured format, i.e. *Web data extraction.*

HTML DOM structures could be very diverse among different Web sites. For example, some Web designers may use *table* to present tabular data while others use *table* to divide space into different grids for layout purpose. Therefore, even if two Web pages have similar layouts, their HTML source codes may be completely different. For example, Figure 1 presents two Web pages that have similar layouts but different DOM structures. Due to the diversity, it is challenging to make a wrapper applicable to different Web sites of the same category. In addition, a DOM structure is complex. For example, even the DOM structure of Google homepage includes 110 HTML tags. The complexity of DOM structures could further increase the diversity and reduce the accuracy. Recently, layout based analysis receives more and more attention [3, 14, 19]. In order to provide efficient Web browsing, Web pages that include similar information in general are presented with consistent layouts even though those pages may be implemented completely differently. Therefore, layout based analysis addresses the diversity issue to a certain degree. However, existing layout-based approaches have limited applicability. Some approaches [19] need training before they are applied to a Web site. Some approaches are optimized for a specific type of domains, and may not be easily adjusted to another domain. For example, the Visual Wrapper [3] is powerful to extract news stories while it is not applicable to other domains.



(a) Buy.com (b) Amazon.com

Figure 1. The same visual layout with different DOM structures

This paper presents a novel approach that combines layout and DOM structure analysis. Our approach extracts structured records by analyzing the screenshot of a Web page that is coherent with human's cognition of visual perception. In our approach, the screenshot of a Web page is first abstracted as a spatial graph, in which a node is an information object (such as text or image) and an edge indicates a close semantic relation between two information objects. Instead of using machine vision technique to recognize images and texts, our approach recognizes information objects based on the DOM structure. Though a DOM-structure based recognition is not as powerful as the machine vision technique, it is efficient and sufficient to recognize texts, links and images that are useful in information extraction. Based on the spatial graph, we are using the graph parsing technique to extract structured records.

The graph parsing assumes that Web designers usually follow some guidelines or patterns to present information on the Web. This is a valid assumption in practice since a consistent layout style can provide effective browsing for end users and ease the efforts of development and maintenance. This assumption has been validated by our evaluation on different Web sites and also by other researchers [15]. Those common guidelines are referred to as patterns that are visually specified through graph grammars [7] in our approach.

This work is supported in part by ND EPSCoR Grants #FAR0015845.

Since the input of data extraction is a Web page that is abstracted as a spatial graph and the output is a tree, the process of data extraction can be considered as transforming from one graph to another one that can be naturally specified through the graph grammar technology. Graph grammars provide a solid theoretic foundation to define computing in a two-dimensional space. In our approach, a graph grammar visually yet formally defines a Web pattern, and then the data extraction is implemented as a process of graph parsing that searches in a spatial graph the sub-structures consistent with the defined pattern. In order to minimize the manual effort of designing a graph grammar, we implemented a graphical interactive tool to facilitate the design of graph grammars. We have evaluated our approach on 21 Web sites to extract product information. The results are promising and the performance of our approach measured in terms of F1-Score (See Section V for further detail) is high.

II. RELATED WORK

With a clear structure to specify the layout of a Web page, the HTML source codes have been commonly analyzed to extract structured data records [1, 4, 5, 6, 8, 9, 10, 11, 12, 16, 18]. Several approaches [6, 8, 11] use the machine learning technique to automatically derive a wrapper based on a set of manually labeled training data. Though the above approaches apply different technologies to derive a wrapper, they all require a set of training data, which are manually labeled by human experts. Several approaches [1, 4, 5, 12] automatically derive a template from sample Web pages and use the extracted template to discover structured records. These approaches do not require manually labeled data, which greatly reduces the manual effort in the data extraction process. However, they require that Web pages being analyzed must follow the same template as the sample Web pages. MDR [10] and DEPTA [16] generate an HTML tag tree based on table and form related tags, e.g., *table*, *form*, *tr*, *td*, and etc. This HTML tag tree significantly reduces the complexity of the original Web page. Based on the HTML tag tree, they use the string comparison technique to divide a Web page into different regions. In each region, it identifies data records by calculating similarity between tag strings. Zhai *et. al.* [18] rendered a Web page and allowed users to select information objects in the screen shot to define a data pattern. Different from our approach, this data pattern is defined on the DOM structure, not on the visual layout. Laber *et. al.* [9] used some statistical analysis to analyze the DOM tree elements and identify the relevant information objects. All of the above methods use HTML DOM structure as the main source for data extraction. Instead, our approach, i.e. *Visual Grammar Based Extractor - VGE*, implements data extraction based on the information presentation. The visual analysis can address the issues of complexity and diverse usages of HTML DOM structures to a certain degree. By actually rendering a Web page, our approach supports dynamic information objects which are generated at run time.

Recently, the visual perception technique has been applied to extract structured data since it is independent from the detailed implementation underlying a Web page. These approaches [3, 14, 19] basically calculate the visual similarity among different Web pages to group semantically related information. [3, 19] are limited to extract news stories, and are not applicable to other domains. ViPER [14] is implemented on statistical models that emphasize on extracting repetitive data records.

The Hybrid method takes benefits from both DOM Structure and Visual Perception, and combines them together. ViNTs [17] automatically recognizes different content shapes based on the visual position of information objects. Afterward, a wrapper is generated based on an HTML structure which represents each shape. This approach still extracts information based on HTML DOM structures, though the wrapper is derived from visual analysis. Instead, our approach specifies extraction rules from both the layout and the DOM structure. ViNTs is limited to search results, while our approach is general to different domains.

III. A GRAMMAR BASED APPROACH

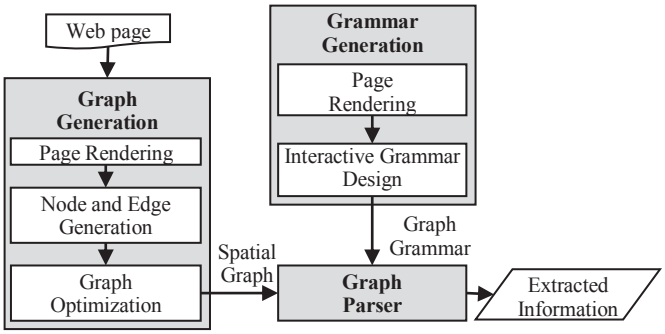


Figure 2. A Graph Grammar based Extractor

This paper presents a novel and robust approach, i.e. ***V**isual **G**rammar based **E**xtractor - VGE*, to extracting structured information. Our approach consists of three components as shown in Figure 2. The graph generation component abstracts a Web page as a spatial graph that simplifies the original Web page and highlights important semantic relations between recognized information objects. The graph generation proceeds in the following steps: (1) render a Web page on the screen, (2) recognize information objects and divide a Web page into different regions according to its DOM structure, (3) calculate semantic relations between recognized information objects based on the layout information, and finally (4) optimize the spatial graph. Based on the generated spatial graph, the data extraction is implemented as a graph parsing process that searches in the spatial graph sub-graphs satisfying certain spatial properties. Those spatial properties are visually defined through a graph grammar. The grammar generation component provides an interactive grammar design tool that allows end users to define a graph grammar by directly manipulating the screenshot of a Web page. This interactive grammar design tool eases the process of developing a graph grammar and improves the usability of our approach.

A. Graph Generation

HTML is a very flexible language. Information with the same presentation could be implemented in many different ways. Being coherent with the HCI principle that consistent presentations can improve the usability of an interface [13], Web designers across different Web sites commonly use similar layouts to present the same type of information.

Therefore, our approach extracts structured records by analyzing the layout of a Web page. The visual analysis can address the diversity of HTML usages and make our approach applicable to different Web sites. The process of graph generation is a critical step in our approach since it simplifies original Web pages and eliminates variations among different Web pages. The simplification only preserves essential information objects. Especially, the graph generation process removes (1) style and layout elements, which do not include any real content, (2) advertisements and (3) menus in the border areas. The simplification effectively reduces the complexity of HTML pages and removes potential noises in the data extraction. The graph generation process proceeds in three steps: Web page rendering, node and edge generation, and graph optimization.

The first step in the graph generation is to render a Web page. The visual layout of a Web page is determined by three variables, i.e. (1) the actual HTML source code that specifies the DOM structure of the page, (2) data items such as text and picture and (3) style sheets and client side scripts which are executed by a browser at run time. We can access all HTML elements, especially dynamic elements which are generated on the fly, only by actually rendering a Web page. Also, the page rendering determines the position and size of each element.

Based on the dynamic and static HTML elements and their spatial properties, the second step generates a spatial graph in which a node represents an information object for data extraction and an edge indicates a close semantic relation between the pair of connecting nodes. Contents are stored in three types of nodes, i.e. image, text and link. The contents enclosed in the <img> or <a> tags are recognized as an image node or a link node, respectively. However, it is challenging to identify a text node since one complete sentence may be separated by several HTML tags and it is necessary to consolidate those information pieces together. For example, inside the text block of a sentence, formatting and styling tags (such as <b>,
, <font>,) can divide the sentence into several pieces. In the graph generation, all those formatting and styling tags are removed and adjacent contents are consolidated as one single text node.

After identifying atomic information objects as nodes, it is critical to calculate semantic relations between information objects and use an edge to connect two nodes that are closely related in semantics. In a two dimensional space, an information object can have an arbitrary spatial relation with adjacent nodes. A complete spatial parsing that analyzes different spatial properties in a graph could be time consuming. Our approach first derives the semantic relation between adjacent nodes, and each close semantic relation is represented as an edge in the spatial graph. Based on the derived semantic relations, we can limit the spatial parsing to objects that have semantic relations and thus reduce the search space to speed up the parsing process. We have extensively investigated different Web sites and found that a small distance strongly indicates a close semantic relation between two objects. This observation is consistent with the Human Computer Interaction principle that closely related objects should be grouped together and placed in proximity [13]. Accordingly, we derive the semantic relation by calculating the distance between two objects. Also, an HTML DOM structure provides valuable hints for deriving semantic relations. Web designers group related objects together by using a container, such as *table* or *div*. In general, two objects belonging to two containers are not related. For example, in Figure 3, though *text* objects 4 and 5 are placed in proximity, they are not semantically related since they belong to two different containers. Our approach uses HTML DOM structures to recognize the containers, and semantic relations are limited to information objects that have one common (ancestor) container. In order to accommodate different image sizes and variations in Web pages, we propose a novel approach to calculating distance and deriving semantic relations. The size of an information object *a* is extended to a certain degree. If the extended object *a* is overlapping with at least two corners of another information object *b*, *a* has a semantic relation with *b*.

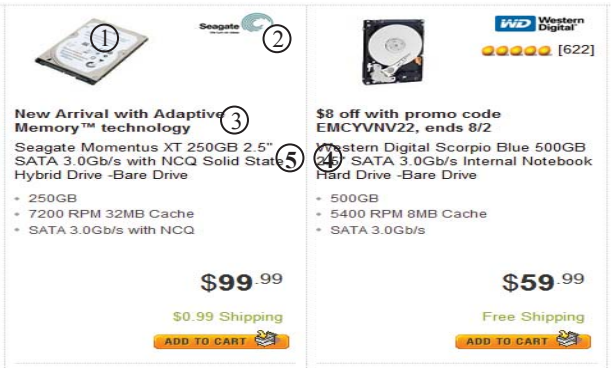


Figure 3. Different containers in a Web page

The last step in the graph generation is to optimize the generated spatial graph. In a spatial graph, some nodes may be considered as noises (such as advertisements and menus), which do not contribute to the data extraction process. Since those objects in general are placed in the border areas of a Web page, we can remove them according to their position. Another type of noise is small repetitive pictures, such as the "*Add to Cart"* icon in Figure 3.

B. Grammar Generation and Parser

The graph generation component generates a spatial graph, and the data extraction is performed on a spatial graph to search for information objects having certain spatial relations. In order to support efficient browsing, the same type of information in general is presented similarly across different Web sites. Those consistent spatial features among information objects are summarized as a Web pattern that is visually specified through a graph grammar. A graph grammar defines computation in a multi-dimensional fashion based on a set of rewriting rules, i.e. *productions*. Since the input of data extraction is a graph and the output is a tree structure that represents structured records, the data extraction is essentially a process of graph transformation that can be naturally specified through graph grammars. Furthermore, a graph grammar is powerful to handle the variations among instances of a Web pattern. This paper selects the Spatial Graph Grammar (SGG) [7] as the specifying formalism. With the capability of spatial specification in the abstract syntax, SGG provides the flexibility to define a pattern from both the edges (i.e. close semantic relations) and spatial features (e.g. directions).

Instead of designing a graph grammar from scratch, we designed an interactive design tool that allows users to design a graph grammar visually and intuitively. This interactive tool renders a sample Web page on the screen, and highlights recognized information objects in the Web page. Users can directly select one or more information objects in the Web page to make a production. This tool supports a direct manipulation interaction on the grammar design that reduces the gap between a concrete Web pattern and an abstract graph grammar. With the help of this tool, even users without much training in graph grammars may design a graph grammar.

IV. System Implementation

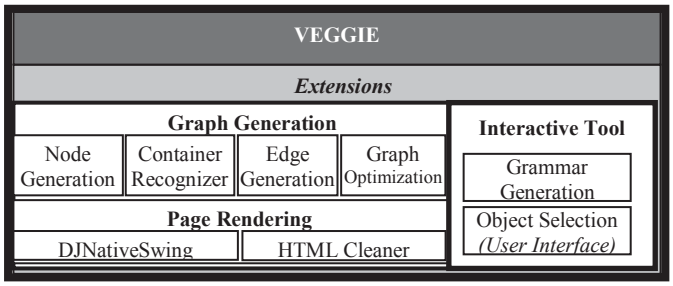


Figure 4. The VGE system architecture

We have implemented a prototype for our approach. Our prototype is built based on VEGGIE - **V**isual **E**nvironment for **G**raph **G**rammars: **I**nduction and **E**ngineering [2]. VEGGIE is a general visual programming environment, and supports the Spatial Graph Grammar specification and parsing. VEGGIE mainly consists of three independent editors (i.e., the Type Editor, the Grammar Editor, and the Graph Editor) and an SGG parser. The three editors provide GUIs for designers to visually design a graph grammar, and are closely related and seamlessly working together in VEGGIE. Grammar designers can visually create visual objects, i.e. node types, in the Type Editor, or import existing node types from a file in the form of GraphML. Then, based on these defined nodes, the designer can define productions in the Grammar Editor. The designer can visually draw or import a host graph to be analyzed by the SGG parser. As shown in Figure 4, we have extended VEGGIE with two subsystems, both implemented in Java. The first sub-system is responsible to generate a spatial graph from a Web page. The generated spatial graph is fed to the SGG parser for a spatial parsing. The second sub-system provides an interactive graphic tool to design a graph grammar.

The Graph Generation sub-system has several components. The *Page Rendering* component renders a Web page, extracts size/position information and passes the output to the *Node Generation* to generate nodes. Containment relations are identified by the *Containment Recognizer*. The *Edge Generation* derives semantic relations based on containers and spatial properties. The *Graph Optimization* component optimizes a generated spatial graph by removing noises.

The *Page Rendering* component renders a Web page based on the DJNativeSwing Browser (http://djproject.sourceforge.net/ns/). The HTML Cleaner component (http://htmlcleaner.sourceforge.net) solves syntactical problems (e.g., unclosed tags and markup errors). The HTML Cleaner returns a clean and well-structured HTML DOM tree that includes all static and dynamic elements. Based on this DOM tree, the graph generation component, including *node generation*, *container recognizer*, *edge generation* and *graph optimizer*, generates an optimized spatial graph for data extraction. The second sub-system eases the process of designing a graph grammar. It consists of two components, i.e. *object selection* and *grammar generation*.

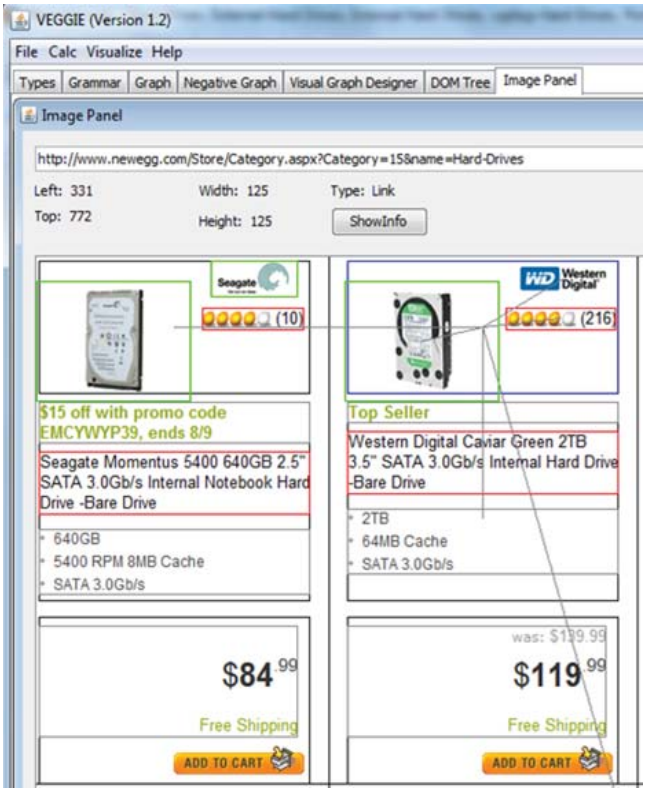


Figure 5. Browsing a Web page

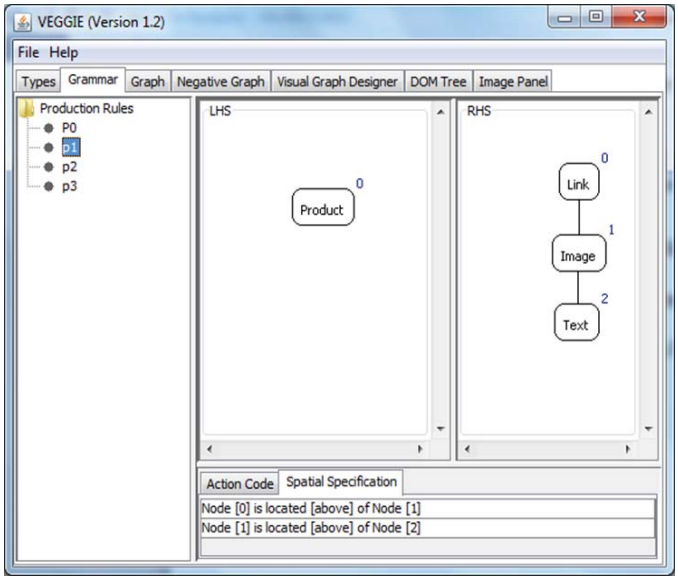


Figure 6. A grammar editor

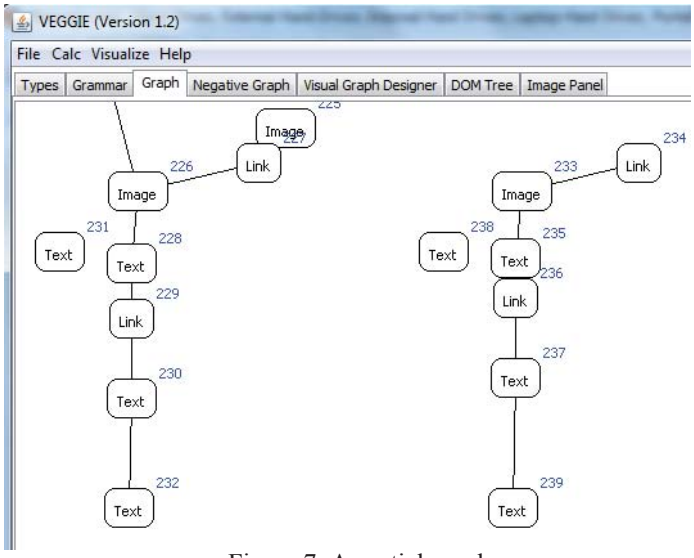


Figure 7. A spatial graph

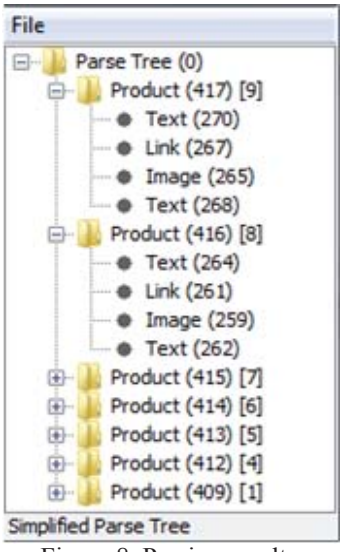


Figure 8. Parsing result

The prototype provides a complete system to support data extraction, from the specification of extraction knowledge (i.e. defining a graph grammar) to the visualization of extraction results. At first, a user uses the *Image Panel*, as presented in Figure 5, to display a Web page. In Figure 5, recognized information objects are highlighted with rectangles, and edges indicate close semantic relations. Based on the screenshot of a Web page, a user can select related information objects to intuitively design a graph grammar. After a graph grammar is defined, the user can go to the VEGGIE Type Editor or Grammar Editor (as presented in Figure 6) to elaborate the designed graph grammar. Once a graph grammar is finalized, a user can use the prototype to extract structured records that are consistent with the defined graph grammar. A user first inputs the URL of a Web page in the image panel. Then, the corresponding spatial graph is automatically generated and can be retrieved in the graph panel, as presented in Figure 7. By applying the defined graph grammar to the spatial graph, a small popup window shows up to present the extracted records, as shown in Figure 8.

V. EXPERIMENT

This section discusses the primary experiment on VGE. We first discuss the design of the experiment, and then present the results.

A. Setup

Experiment web pages: We have evaluated our approach on 21 ecommerce Web sites, which include well known Web sites, such as ebay.com, lycos.com, amazon.com, compusa.com, and etc.

Measurement: We measured the performance with the standard metrics: $recall = \frac{Ecorrect}{Ntotal}$; $precision = \frac{Ecorrect}{Etotal}$;

Where *Ntotal* is the number of data records contained in a Web page; *Ecorrect* indicates the total number of correctly extracted data records; and *Etotal* denotes the total number of data records extracted from a Web page. We also calculated the F1-Score, which is the harmonic mean of *precision* and *recall* and is defined as $\frac{2\times recall\times precision}{recall+precision}$. The F1-Score has been commonly used as a metric to evaluate the overall performance in many approaches [3, 9].

Execution Platform: We have evaluated our approach on a desktop with a Core 2 Duo CPU 2.26 GHz and 4 GB RAM, running Windows 7 Professional.

B. Evaluation

Precision/Recall/F1-Score: The evaluation results are presented in Table 1. The recall of our approach is 99.5. The high recall rate in our approach indicates that graph-grammar-based visual analysis is powerful to recognize structured records. The precision of our approach is 95.5%. Our approach may falsely recognize some records, which are mainly caused by noise. For example, if an advertisement is placed in the central area and its overall layout is similar to our selected pattern (e.g. including a link, a picture and several lines of textual description that are displayed vertically); this advertisement may be recognized as a product record. In order to improve the precision, it is critical to improve the graph generation process by removing potential noise. F1-Score shows the overall performance. Our approach has a high F1-Score of 97.49%. In summary, the results indicate our approach has a good performance in terms of both precision and recall.

Table 1. Evaluation Results

Domain Name	# of Structured Records	Our approach	
		Correct	Found
shopping.yahoo.com	15	14	14
scistore.cambridgesoft.com	13	13	14
shop.lycos.com	18	18	18
www.barnesandnoble.com	48	48	48
www.borders.com	27	28	29
www.circuitcity.com	5	5	7
www.compusa.com	18	18	21
www.drugstore.com	15	13	14
www.ebay.com	20	20	20
www.etoys.com	32	32	32
www.kidsfootlocker.com	29	29	29
www.kodak.com	20	20	20
www.newegg.com	20	20	26

www.nothingbutsoftware.com	24	24	24
www.overstock.com	18	18	18
www.powells.com	50	50	51
www.softwareoutlet.com	14	14	15
www.ubid.com	8	8	9
www.amazon.com	7	7	8
www.shopping.hp.com	5	5	5
www.qualityinks.com	24	24	26
Total	430	423	443
Recall/Precision		99.5%/95.5%	
F1-Score		97.49%	

VI. CONCLUSION

This paper presents a novel and general solution to extract data across different Web sites. Our method works based on graph grammars to extract the same type of information from different Web sites without the need of training and adjustment for different Web sites. Our approach utilizes both the visual features of a rendered Web page and the HTML DOM structure to extract structured records. We have implemented a prototype and tested the prototype on 21 Web sites. The evaluation shows promising results. Our approach has a high F1-Score as 97.49%. The evaluation results indicate our approach has a good performance in terms of both precision and recall. The main advantage of our approach lies in its ability to distinguish the most important contents from less important and noisy information and to convert the complex HTML DOM structure to a simple spatial graph. The generated spatial graph significantly reduces the complexity of the original Web page. Based on the simplified spatial graph, our approach is efficient to extract structured records through a graph parsing.

In the future work, we will identify more spatial relations between information objects, and optimize the graph generation algorithm. These optimizations may increase the quality of generated spatial graphs, which can affect both the precision and recall.

REFERENCES

[1] Arasu, A. and Garcia-Molina, H. 2003. Extracting structured data from Web pages. In *Proceedings of the 2003 ACM SIGMOD international Conference on Management of Data*. 337-348.

[2] Ates, K. and Zhang, K. 2007. Constructing VEGGIE: machine learning for context-sensitive graph grammars. In *Proceedings of the 19th IEEE international Conference on Tools with Artificial intelligence - Volume 02*. ICTAI. IEEE Computer Society, Washington, DC, 456-463.

[3] Chen, J. and Xiao, K. 2008. Perception-oriented online news extraction. In *Proceedings of the 8th ACM/IEEE-CS Joint Conference on Digital Libraries*. ACM, New York, NY, 363-366.

[4] Chuang, S. and Hsu, J. Y. 2004. Tree-structured template generation for Web pages. In *Proceedings of the 2004 IEEE/WIC/ACM international Conference on Web intelligence*. IEEE Computer Society, Washington, DC, 327-333.

[5] Crescenzi, V., Mecca, G., and Merialdo, P. 2001. RoadRunner: towards automatic data extraction from large Web sites. In *Proceedings of the 27th international Conference on Very Large Data Bases*. P. M. Apers, P. Atzeni, S. Ceri, S. Paraboschi, K. Ramamohanarao, and R. T. Snodgrass, Eds. Very Large Data Bases. Morgan Kaufmann Publishers, San Francisco, CA, 109-118.

[6] Hsu, C. and Dung, M. 1998. Generating finite-state transducers for semi-structured data extraction from the Web. *Inf. Syst.* 23, 9, 521-538.

[7] Kong, J., Zhang, K., and Zeng, X. 2006. Spatial graph grammars for graphical user interfaces. *ACM Trans. Comput.-Human Interact.* 13, 2, 268-307.

[8] Kushmerick, N., Weld, D., & Doorenbos, R. 1997. Wrapper induction for information extraction. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*. 729-737.

[9] Laber, E. S., de Souza, C. P., Jabour, I. V., de Amorim, E. C., Cardoso, E. T., Rentería, R. P., Tinoco, L. C., and Valentim, C. D. 2009. A fast and simple method for extracting relevant content from news webpages. In *Proceeding of the 18th ACM Conference on information and Knowledge Management*. ACM, New York, NY, 1685-1688.

[10] Liu, B., Grossman, R., and Zhai, Y. 2003. Mining data records in Web pages. In *Proceedings of the Ninth ACM SIGKDD international Conference on Knowledge Discovery and Data Mining*. ACM, New York, NY, 601-606.

[11] Muslea, I., Minton, S., and Knoblock, C. 1999. A hierarchical approach to wrapper induction. In *Proceedings of the Third Annual Conference on Autonomous Agents*. O. Etzioni, J. P. Müller, and J. M. Bradshaw, Eds. AGENTS '99. ACM, New York, NY, 190-197.

[12] Reis, D. C., Golgher, P. B., Silva, A. S., and Laender, A. F. 2004. Automatic web news extraction using tree edit distance. In *Proceedings of the 13th international Conference on World Wide Web*. ACM, New York, NY, 502-511.

[13] Shneiderman, B. 2009. Designing the User Interface: Strategies for Effective Human-Computer Interaction. Addison-Wesley Longman Publishing Co., Inc.

[14] Simon, K. and Lausen, G. 2005. ViPER: augmenting automatic information extraction with visual perceptions. In *Proceedings of the 14th ACM international Conference on information and Knowledge Management*. ACM, New York, NY, 381-388.

[15] Zhang, Z., He, B., and Chang, K. C.-C. 2004. Understanding Web query interfaces: best-effort parsing with hidden syntax. In *Proceedings of 2004 ACM SIGMOD International Conference on Management of Data*, 107-118.

[16] Zhao, H., Meng, W., Wu, Z., Raghavan, V., and Yu, C. 2005. Fully automatic wrapper generation for search engines. In *Proceedings of the 14th international Conference on World Wide Web*. ACM, New York, NY, 66-75.

[17] Zhai, Y. and Liu, B. 2005. Web data extraction based on partial tree alignment. In *Proceedings of the 14th international Conference on World Wide Web*. ACM, New York, NY, 76-85.

[18] Zhai, Y. and Liu, B. 2007. Extracting Web data using instance-based learning. *World Wide Web* 10, 2, 113-132.

[19] Zheng, S., Song, R., and Wen, J. 2007. Template-independent news extraction based on visual consistency. In *Proceedings of the 22nd National Conference on Artificial intelligence - Volume 2*. A. Cohn, Ed. Aaai Conference On Artificial Intelligence. AAAI Press, 1507-1512.