Annals of the University of Craiova, Mathematics and Computer Science Series
Volume 37(3), 2010, Pages 92–100
ISSN: 1223-6934

Toward semantic annotation of Web page's segmentation blocks

Mirel Cosulschi

Abstract. The traditional search engines like *Google*, *Yahoo*, *Altavista*, etc. which gained a huge success during the last decade, were initially designed in order to easily locate and retrieve any information available on the Web. The computer engineers did not considered at that moment, the possibility that information meaning to be understood by the machines. Due to the huge amount of information currently available on the Web, the automatic intelligent retrieval of information becomes a tedious task: the user shall understand, combine, interpret, select and evaluate this information.

In most cases, a Web page contains multiple semantic topics and a natural approach would be not to consider it as an atomic element. It is more natural to reason and operate with the semantic blocks in which a Web page can be decomposed, each block corresponding to a different topics, as the smallest units of information.

2010 Mathematics Subject Classification. Primary 68U99; Secondary 68P20.
Key words and phrases. semantic annotation, Web page segmentation, Web page rendering, Web information retrieval, visual features.

1. Introduction

An important research aspect about Web information extraction relates to the inference of complex reasoning and correlation based on distributed information available in many different Web data sources. By defining the semantics of information and services available on the Web, the World Wide Web becomes a vast store of information that can be easily processed by computer applications. *Semantic Web* aims at creating an universal medium where data and knowledge can be exchanged between applications.

In most cases, a Web page contains multiple semantic topics and a natural approach would be not to consider it as an atomic element. In the perfect situation, a Web page is decomposed in many blocks, each block corresponding to different topics. It is more natural to reason and operate with these semantic blocks as the smallest units of information. The hyperlinks contained inside the blocks can also be used for enhanced topic distillation and discovering the semantic structure of the Web.

The goal of semantic annotation is to add formally defined semantics to the already existing Web documents (Web pages, images, audio files, pdf, Word files, etc). In this way an agent which is a piece of software, that works autonomously and proactively [30], will become more aware of the content and its meaning.

Received July 02, 2010. Revision received August 27, 2010.

2. Web page decomposition

The vast amount of information on World Wide Web cannot be fully exploited due to its main characteristics: Web pages are designed with respect to the human readers, who interact with the systems by browsing $HTML$ pages, rather than to be used by a computer program. A subsequent problem arisen here is the necessity to divide Web documents into different information chunks. The mere segmentation based on DOM tree representation [1] does not have enough power to semantically decompose a Web page.

A $HTML$ page may contain many types of information presented in different forms, such as text, images or applets (programs written in Java and executed, better said interpreted, inside a *virtual machine* - Java Virtual Machine - integrated into the browser). *Hyper Text Markup Language*($HTML$) is a language designed for data presentation, and was not intended as a mean of structuring information and facilitating the process of extracting structured data from Web documents. Another problem of $HTML$ pages is related to their bad construction, language standards frequently being broken (i.e. improper closed tags, wrong nested tags, bad parameters and incorrect parameters values).

Web pages from commercial Web sites are usually generated dynamically using various scripts and data stored in a back-end DBMS. A Web surfer can easily notice the usage of a few templates in the same site, with slight differences between them. Page-generation process can be interpreted as the result of two activities:

- the execution of some queries targeted at the support database;
- the source dataset intertwined with HTML tags and other strings in a process called *codification*. Eventually, URL links and banners or images can also be inserted.

A Web page must undergo some intermediary transformations in order to reach the proposed format [18], [19]:

- $HTML$ code cleaning (basically a cleaning-up process);
- Web page rendering;
- Web page segmentation.

2.1. HTML code cleaning. World Wide Web consortium has warmly recommended usage of stricter standard Markup Languages, such as $XHTML$ [2] and XML, in order to reduce the number of the errors resulted in the process of parsing various Web pages, created by disobeying the basic composition rules. Despite this recommendation, there still remains a huge amount of Web pages that do not respect the new standards, and with whom, the parser of a search engine must cope.

The code cleaning of each $HTML$ page into a *well-formed* page becomes a desiderate. From the applications that can be involved in this process, we advert to two of them:

- JTidy[1], a Java tool based on HTML Tidy [3] - that is an open source software released on W3C Website;
- Neko HTMLParser [4] started as personal efforts of Andy Clark.

Despite the fact that these are complex programs, with a great degree of generality, trying to satisfy overall demands, we encountered many situations during their usage when the result was not totally satisfactory [18], [19].

[1] `http://jtidy.sourceforge.net/`

The purpose of this step was to obtain a *DOM* tree representation of the input page, representation used in the next step of the process, created with minimal effort and as correct as possible.

2.2. Web page rendering. In order to efficiently obtain the *image* of a Web page, we have performed tests with many *rendering engines* from various browsers. A Web browser must fulfill a minimal list of conditions in order to be taken into consideration:

- to offer an access point to the internal representation of a Web page;
- to have a minimal documentation;
- to posses a free version or has an open-source licence;
- to be able to embed it in our Java project.

We have performed the experiments with the following Java Web browsers:

- *Grand-Rapid Browser*[2] provides excellent support for the general Web and comes with a variety of customization features, but it is like a black-box for a programmer who likes to use it inside an application (*development ceased*).
- *NetClue*[3] had very good results in rendering standard Web sites, almost identically to other more mature products, like *Netscape* and *IE*. The programmers have access to virtually every event and to directly manipulate DOM content (*development ceased*).
- *WebRenderer*[4] is a wrapper library which has support for the most encountered operating systems. The wrapper gives access to all internal events.
- *WebWindow* is a Web browser developed entirely in Java, with two versions, one for Swing and one for AWT. The renderer is quite fast, presenting nice features for zooming text and images (*development ceased*).
- *IceBrowser*[5] is one of the oldest Java Web browsers which reached its maturity. We can say that its performances are excellent from the hackability's point of view.
- *JRex*[6] acts like a wrapper for *Mozilla*, using the full power of its libraries. *JRex* has APIs to receive events, to access DOM and also supports *XUL*[7]. It can be used as an embedded browser into an application.
- *Flying Saucer*[8], does not provide support for *JavaScript*. development.
- The *Cobra* package[9] for visually displaying the Web pages during the rule creation process. *Cobra* is a pure Java HTML renderer and DOM parser that is being developed to support *HTML* 4, *Javascript* and *CSS* 2.

The problem with these software products is that the results are far from what can be seen in *IE* or *Firefox*. Either they do not have *Javascript* capabilities (e.g. *FlyingSaucer*), either are commercial (e.g. *WebWindow*) and do not provide access to bounding box coordinates, or do not simply render the page correctly (e.g. *LoboBrowser*).

---

[2] `http://www.meyou.com/grandrapid/`
[3] `http://www.netcluesoft.com/desktop/`
[4] `http://www.webrenderer.com/`
[5] `http://www.icesoft.com/products/icebrowser.html`
[6] `http://jrex.mozdev.org`
[7] `https://developer.mozilla.org/en/xul`
[8] `https://xhtmlrenderer.dev.java.net/`
[9] `http://lobobrowser.org/cobra.jsp`

As a direct consequence of the previous ideas, the authors are manipulating in [5] the *Internet Explorer* through their project *VIPS*[10], while in the project *ViPER*[11] [7] the authors extend some *JRex* C++ API stubs.

In another relevant project, *VENTex* [8], [9], it is used a *Firefox* extension that allows the access to the internal *Firefox*'s *CSS2* model, associated with a loaded Web page. They also use another plug-in –*webpagedump*[12]– in order to save the intermediate results.

2.2.1. *Extraction of bounding box.* An optional step related to segmentation of a Web page is extraction and usage, during the main process, of bounding box corresponding to each element from the *DOM* tree. *The bounding box* represents the smallest rectangle that fully covers an element of the *DOM* tree when it is rendered in a browser along with the whole page to which it belongs.

The coordinates of a bounding box are obtained from a software module specially written to catch *events* generated by *rendering engine*'s components.

A module developed by us [18] adds 4 new attributes to each element that has a visual representation (inside a *HTML* page could be elements without any visual representation, for example *comments*). For example, the next tag

```
<IMG    height="40" width="285" border="0"
    src="car08_file/ab41.gif" alt="Yahoo! Autos"/>
```

becomes

```
<IMG    dy="44" dx="285" ly="14" lx="127"
    height="40" width="285" border="0"
    src="car08_file/ab41.gif" alt="Yahoo! Autos"/>
```

The attributes *dx*, *dy*, *lx*, *ly* have the following semantics:

- *lx* and *ly* represent the coordinates of the upper-left corner of the bounding rectangle;
- *dx* and *dy* represent the *width* and *height* of the bounding rectangle.

In the figure 1, the reader can see the result of rendering a *HTML* Web page inside a Web browser (*IceBrowser* in this case).

In *VENTex*[13] from the rendered Web page it is obtained the coordinates of the bounding boxes of the element nodes (*VENs* = Visualized Element Nodes). To get the coordinates of words, first it is manipulated the internal DOM tree: the content is parsed and each word from the text nodes is placed inside a $< x >$ tag ("x-tagged" because $< span >$ changes layout) and then is treated separately ("Visualized Words"). After all those preparations (raw data is saved separately) it is possible to proceed with the next step and apply various algorithms.

In this moment, we are confident that the best outcome can be reached by using a rendering engine from a major browser (*IE*, *Firefox*) - *IE* can be easily manipulated from *.NET* or through Java using *JNI*[14], while *Firefox* through plug-in method [10]. *JRex* has not been updated for a while and has bugs. Another option referrs to the usage of *SWT* (The Standard Widget Toolkit)[15] component from *Eclipse*[16] for rendering a Web page, which is using the native *rendering engine*[17] from *IE* or *Mozilla*.

[10] http://www.cs.uiuc.edu/homes/dengcai2/VIPS/VIPS.html
[11] http://dbis.informatik.uni-freiburg.de/index.php?project=VIPER
[12] http://www.dbai.tuwien.ac.at/user/pollak/webpagedump/index.html
[13] http://education.dbai.tuwien.ac.at/ventex/index.php
[14] http://java.sun.com/docs/books/jni/
[15] http://www.eclipse.org/swt/
[16] http://www.eclipse.org/
[17] http://en.wikipedia.org/wiki/Web_browser_engine

FIGURE 1. A HTML page rendered inside a Web browser

Also we must take into consideration *Safari* and its core engine *WebKit*[18] - an open-source application framework, ported to various platforms and initiated by Apple from the *HTML* layout engine of *Konqueror*.

The usage of the bounding box inside the segmentation process increases the reliability of the results.

2.3. Web page segmentation. One of the motivation for decomposing a Web page into fine grained and also simpler parts, is that these elements can be used as inputs for automatic wrappers (by example [11]). In the paper [13], the authors demonstrate that the problem of schema inference from many Web pages in the presence of nullable attributes, belongs to the *NP*-complete class of problems, so the size reduction of input data for a program that constructs the schema during an inference process becomes a must.

Basically there are three main strategies that can be used for solving this problem: Web page segmentation using visual features, the segmentation based on textual characteristics and a combination of these methods.

In the paper [14], the authors motivate their attempt to decompose a Web page in small items called *pagelets* that simplify the page structure, by the fact that templates reduce precision, while navigation bar and paid advertisement contradict Hypertext IR Principles.

[18] `http://webkit.org/`

Xiaoli Li et al. [15] proposed a similar method for segmenting a Web page: they introduced the notion of *micro information units* (*MIU*). A *HTML* page is decomposed into many *MIU* elements by merging adjacent paragraphs, exploiting text font property and term set similarity. Another approach, implemented by Gu's [16], describes a top-down approach to segment a Web page and detects its content structure by dividing and merging blocks: the method consist of breaking out the *DOM* tree and comparing similarities among all the basic *DOM* nodes.

Other approaches that combine *DOM* structure and visual cues can be encountered in the papers [12],[5] – the work of Yang and Zhang [12] was continued by D. Cai et al. in [5], [6].

In [17], visual information are used to build up a *M-tree*, a concept similar to the *DOM* tree enhanced with screen coordinates, followed by recognition of common page areas such as header, left and right menu, footer and center of a page through different heuristics.

The authors have investigated the segmentation of a Web page in [18] , by comparing two similar pages, using bounding box coordinates and *spatial relations* [22]. Those *spatial relations* were also used for extracting information from Web tables [9].

WebVAT [24] is an open-source visualization tool aiming to facilitate Web page analysis. The application is built on top of the *Mozilla* Web browser facilitating the access to *Mozilla's* internal representation of Web pages (*a Frame Tree*). In [23] there are presented two algorithms, *BlockFinder* and *Partition finder*, inspired from traditional methods (e.g. the *X-Y cut* algorithm) used for detection of forms in images obtained from a scanner device.

Ramaswamy et al. [25] define a fragment of a Web page as a part shared among multiple documents or having different lifetime or personalization characteristics. The method proposed is based on the concept of *shingles* and is using the textual characteristics.

Two recent papers, [26] and [27], using the same features of a document, involve advanced mathematics: e.g. isotonic regression and graph specific methods. In [26] it is determined a *templateness score* for each *DOM* node, thus forming the input of training data for a classifier. Then, by using isotonic regression, each score associated to a node is recomputed in a global smoothing reevaluation of assigned values. Chakrabarti et al [27] developed a framework for automatically segmentation of webpages: the approaches consist of correlation clustering and energy-minimizing cuts in graphs. The drawback of the method is the learning phase where the weights of vertices from graph are computed from manually labeled data.

3. Web page semantic annotation

The traditional search engines like *Google*, *Yahoo*, *Altavista*, which gained a huge success during the last decade, were initially designed in order to easily locate and retrieve any information available on the Web. The computer engineers did not considered at that moment, the possibility that information meaning to be understood by the machines. Due to the huge amount of information currently available on the Web, the automatic intelligent retrieval of information becomes a tedious task: the user shall understand, combine, interpret, select and evaluate this information. The usage of metadata (*data about data*) is one acknowledged solution to this problem.

Semantic Web is expected to gain a great influence at obtaining better results for search engine queries, documents' ranking, etc. As the reader noticed, very often

a Web page is not related to single semantic topic containing description of various concepts. In order to support more accurate results for semantic Web searches, richer data integration, and better navigation experience (e.g. a mobile device has a smaller screen and, for the user best experience, it is not advisable to display an entire Web page) a Web page has to be decomposed into smaller semantic annotated fragments [21].

In case the Web page is first annotated and after that undergoes a separation process, the existing meta information will be used for the *page segmentation* together with the visual and structural information.

The other approach, when no annotation is available (this is the most frequent case in the current Web), is to first decompose a Web page and then annotate the outcome (*page–block*) of the segmentation. In case of automatic annotation, the latter approach could facilitate the process due to the reduction in the size of input data. One can label the extracted HTML blocks using RDFa (e.g. an HTML block with details about a person as *first name*, *last name* and *address*).

RDF (Resource Description Language), introduced by W3C as a family of specifications meant to be a metadata data model, uses the XML language for metadata representation and becomes an effective solution for conceptual description of information available on the Web. RDFa[19] (Resource Description Framework - in - attributes) is a set of extensions to XHTML and uses attributes from XHTML's elements allowing annotation of markup with semantics.

A small subproblem relates to automatic semantic labelling of the fields extracted by a wrapper [28], [29].

Information within a block might not refer only to named entities but also to generic concepts. After the identification of named entities, those may be automatically classified with respect to a generic ontological schema or a highly specialized ones. In the process of relieving the main concepts representing the information in the page–block, *Wordnet* would have a major role.

WordNet[20] is a lexical resource for the English language, structured as groups of synonyms called synsets, and linked by semantic and lexical relations. Using *WordNet* an application can extract the semantics of a Web page: discovery the candidate sense of words that compose it [31], followed by a disambiguation operation resulting the most probable sense of words. Using that computed set of words' senses, it is possible to find the most appropriate ontology from pools of Web ontologies and use that for the annotation process of the current Web page.

Usually the annotations are made with respect to an ontology, which describes formally a domain of discourse (it is composed of a set of terms and the relationships between the terms). The domain ontology associated to a Web page can be chosen manually by a user or automatically by a computer program.

The relevance of a set of resources with respect to a single piece of information in the extracted HTML block may be computed taking into account [21]:

(i) domain information from the selected (portion of the) ontology;
(ii) annotations related to all the other blocks extracted from the same $HTML$ page;
(iii) annotations either from similar pages or pages from the main Web site.

[19] `http://www.w3.org/TR/xhtml-rdfa-primer/`
[20] `http://wordnet.princeton.edu/`

4. Conclusion

In this paper, it is presented the details and problematic of Web page segmentation by using visual features and textual characteristics. On this subject various approaches and tools implementing the ideas of algorithms exposed in these methods, developed during the last decade, are mentioned in a survey that presents the timeline of theirs evolution and the connections between seminal ideas.

Semantic annotation proved to rise up many new complex problems with whom researchers has to deal. This topic is a current concern related to enhancements of search engines' capabilities aiming to reduce further the number of false positives returned as results of a query.

Acknowledgements The work reported was partly funded by the Romanian National Council of Academic Research (CNCSIS) through the grant CNCSIS 375/2008.

References

[1] Document Object Model (DOM) Level 1 specification. W3C Recommendation, October 1998, `http://www.w3.org/TR/REC-DOM-level-1`.

[2] *XHTML 1.0 The Extensible HyperText Markup Language (Second Edition)*, `http://www.w3.org/TR/2002/REC-xhtml1-20020801`.

[3] W3C, *HTML Tidy*, `http://www.w3.org/People/Raggett/tidy`.

[4] Neko HTML Parser, `http://www.apache.org/~andyc/neko/doc/html/index.html`.

[5] D. Cai, S. Yu, J.-R. Wen and M.-Y. Ma, Extracting content structure for web pages based on visual representation, *the Fifth Asia Pacific Web Conference (APWeb2003)*, Xián China, (2003).

[6] D. Cai, S. Yu, J.-R. Wen and M.-Y. Ma, VIPS–a vision based page segmentation algorithm, *Microsoft Technical Report*, MSR-TR-2003-79, (2003).

[7] K. Simon and G. Lausen, ViPER: Augmenting Automatic Information Extraction with Visual Perceptions, *in ACM International Conference on Information and Knowledge Management (CIKM '05)*, Bremen, Germany, (2005).

[8] W. Gatterbauer and P. Bohunsky, Table Extraction Using Spatial Reasoning on the CSS2 Visual Box Model, *in Proceedings of the 21st National Conference on Artificial Intelligence (AAAI 2006)*, Boston, Massachusetts, (2006), MIT Press.

[9] W. Gatterbauer, P. Bohunsky, M. Herzog, B. Krüpl and B. Pollak, Towards Domain-Independent Information Extraction from Web Tables, *in Proceedings of 16th International World Wide Web Conference (WWW'07)*, Banff, Alberta, Canada, ACM Press, (2007).

[10] K. C. Feldt, *Programming Firefox Building Rich Internet Applications with XUL*, O'Reilly, 2007.

[11] V. Crescenzi, G. Mecca and P. Merialdo, RoadRunner: Automatic Data Extraction from Data-Intensive Web Sites, *in Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data*, Madison, Wisconsin, USA, ACM Press, (2002).

[12] Y. Yang and H. Zhang, HTML page analysis based on visual cues, *Proceedings of the 6th International Conference on Document Analysis and Recognition (ICDAR 2001)*, (2001).

[13] G. Yang, I. V. Ramakrishnan and M. Kifer, On the complexity of schema inference from Web pages in the presence of nullable data attributes, *in 12th ACM Internation Conference on Information and Knowledge Management (CIKM'03)*, New Orleans, LA, USA, (2003).

[14] Ziv bar-Yossef and S. Rajagopalan, Template Detection via Data Mining and its Applications, *Proceedings of the 11th international conference on World Wide Web (WWW'02)*, Honolulu, Hawaii, USA, ACM Press, (2002).

[15] X. Li, T.-H. Phang, M. Hu and B. Liu, Using Micro Information Units for Internet Search, *Proceedings of the 11th International Conference on Information and Knowledge Management (CIKM'02)*, McLean, Virginia, USA, (2002).

[16] X. Gu, J. Chen, W.-Y. Ma and G. Chen, Visual Based Content Understanding towards Web Adaptation, *Proceedings of the 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH2002)*, Spain, (2002), Lecture Notes in Computer Science **2347/2006**, Springer–Verlag.

[17] M. Kovacevic, M. Diligenti, M. Gori and V. Milutinovic, Recognition of common areas in a web page using visual information: a possible application in a page classification, *Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02)*, IEEE Press, (2002).
[18] M. Cosulschi, N. Constantinescu and M. Gabroveanu, Classification and comparison of information structures from a web page, *The Annals of the University of Craiova, Mathematics and Computer Science series* **31** (2004), 109–121.
[19] M. Cosulschi, A. Giurca, B. Udrescu, N. Constantinescu and M. Gabroveanu, HTML Pattern Generator - Automatic Data Extraction from Web Pages, *in Proceedings of the 8th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC'06)*, Timişoara, Romania, IEEE Press, (2007).
[20] A. Iasinschi and M. Cosulschi, Semi-automated Wrappers using Rule Trees, *in Proceedings of the 10th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC'08)*, Timişoara, Romania, IEEE Press, (2009).
[21] M. Cosulschi, R. De Virgilio, T. Di Noia and R. Mirizzi, Web Information Extraction by Semantic Tagging, *The 28th International Conference on Conceptual Modeling (ER09)*, Gramado, Brazil, (2009).
[22] J.F. Allen, Maintaining Knowledge about Temporal Intervals, *Communications of the ACM*, 26:832-843, 1983.
[23] H. Guo, J. Mahmud, Y. Borodin, A. Stent and I.V. Ramakrishnan, A General Approach for Partitioning Web Page Content Based on Geometric and Style Information, *Proceedings of the 9th International Conference on Document Analysis and Recognition (ICDAR 2007)*, IEEE Press, (2007).
[24] Y. Borodin, J. Mahmud, A. Ahmad and I.V. Ramakrishnan, WEBVAT: Web Page Analysis and Visualization Tool, *Proceedings of the 7th International Conference on Web engineering (ICWE'07)*, Como, Italy, Lecture Notes in Computer Science **4607/2007**, Springer–Verlag.
[25] L. Ramaswamy, A. Iyengar, L. Liu and F. Douglis, Automatic detection of fragments in dynamically generated web pages, *Proceedings of the 13th international conference on World Wide Web (WWW'04)*, New York, NY, USA, ACM Press, (2004).
[26] D. Chakrabarti, R. Kumar and K. A. Punera, Page-Level Template Detection via Isotonic Smoothing, *in Proceedings of 16th International World Wide Web Conference (WWW'07)*, Banff, Alberta, Canada, ACM Press, (2007).
[27] D. Chakrabarti, R. Kumar and K. A. Punera, Graph-Theoretic Approach to Webpage Segmentation, *in Proceedings of 17th International World Wide Web Conference (WWW'08)*, Beijing, China, ACM Press, (2008).
[28] L. Arlotta, V. Crescenzi, G. Mecca and P. Merialdo, Automatic annotation of data extracted from large web sites, *in Proceedings of 6th Workshop on Web and Databases (WebDB03)*, (2003).
[29] K. Lerman, C. Gazen. S. Minton and C. Knoblock, Populating the Semantic Web, *in Proceedings of the Workshop on Advances in Text Extraction and Mining (AAAI-2004)*, (2004).
[30] G. Antoniou and Frank van Harmelen, *A Semantic Web Primer*, MIT Press, 2004.
[31] J. Gracia, M. dAquin and E. Mena, Large Scale Integration of Senses for the Semantic Web, *in Proceedings of the 18th International World Wide Web Conference (WWW 2009)*, Madrid, Spain, ACM Press, (2009).

(Mirel Cosulschi) Department of Computer Science, University of Craiova, 13 A.I. Cuza Street, Craiova, 200585, Romania
E-mail address: mirelc@inf.ucv.ro