SearchResultFinder: Federated Search Made Easy

Dolf Trieschnigg, Kien Tjin-Kam-Jet and Djoerd Hiemstra
University of Twente
Enschede, The Netherlands
{trieschn,tjinkamj,hiemstra}@cs.utwente.nl

ABSTRACT

Building a federated search engine based on a large number existing web search engines is a challenge: implementing the programming interface (API) for each search engine is an exacting and time-consuming job. In this demonstration we present SearchResultFinder, a browser plugin which speeds up determining reusable XPaths for extracting search result items from HTML search result pages. Based on a single search result page, the tool presents a ranked list of candidate extraction XPaths and allows highlighting to view the extraction result. An evaluation with 148 web search engines shows that in 90% of the cases a correct XPath is suggested.

Categories and Subject Descriptors

H.2.8 [**Database applications**]: Data mining; I.5.4 [**Pattern Recognition**]: Applications—*Text processing*; H.3.5 [**Information Storage and Retrieval**]: Online Information Services—*Web-based services*

Keywords

Web extraction, Scraper, Wrapper, Search result extraction

1. INTRODUCTION

Federated search engines combine the power of multiple engines in one: based on the user query the most appropriate resources (search engines) are queried and their results are merged into a single view. The search results from these resources have to be obtained in a machine readable way so they can be combined in a uniform view. Some resources provide an application programming interface (API) to achieve this, or provide their search results in commonly used syndication formats such as Atom and RSS. The OpenSearch[1] project has proposed standardized formats to exchange search results, but has seen only limited adoption. As a result, the only way to obtain machine readable results from these search engines is to extract or *scrape* them from the generated HTML.

[1]`http://www.opensearch.org`

SIGIR'13, July 28–August 1, 2013, Dublin, Ireland.
ACM 978-1-4503-2034-4/13/07.

In this demonstration we present a browser plugin which aids search result scraping. Based on a single (currently viewed) search result page, SearchResultFinder returns a ranked list of XPath expressions which can be used to extract results from pages based on the same template. XPath is a query language to select nodes from XML documents, or in this case, select nodes from the document object tree parsed from the HTML page.

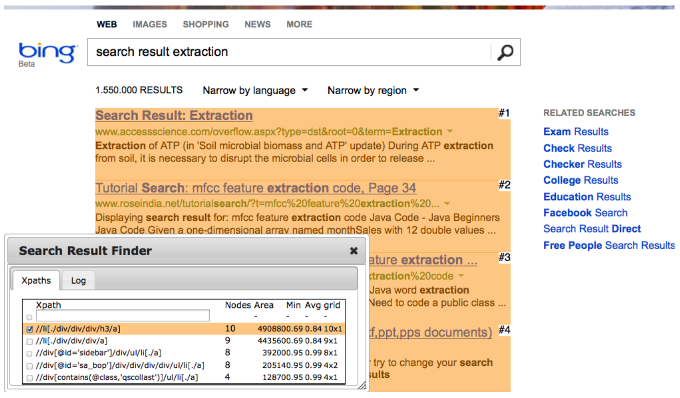
We first describe related systems and their shortcomings. Then we briefly describe the algorithm to extract and rank XPaths and the operation of the plugin. We describe an evaluation on 148 web search engines, and round up with a conclusion.

2. RELATED SYSTEMS

Related work can be found in the area of web information extraction and wrapper induction. Where early work focused on manual wrapper generation (e.g. [2]), later work focusses on interactive and fully automatic wrapper extraction (e.g. [1, 3–5, 7, 8]). For a more detailed review, we refer the reader to [6].

Systems vary in 1) the number of required, sometimes manually labeled, example pages; 2) the type of features used for extraction. Some methods treat a page as a sequence of tags, others exploit the tree structure and take into account rendering features; 3) the techniques used for extraction, varying from grammars and grammar learning to patricia trees, similarity learning and clustering techniques. Heuristics are frequently employed to reduce the complexity.

Typically a *scraper* or *wrapper* is constructed for each search engine: based on one or more search result pages from the same search engine, a program is generated or configured which can extract machine readable search results. The currently available wrapper generators have serious drawbacks making them unattractive to use. Firstly, they require a lot of effort to operate: multiple search result pages have to be saved to file, the generator has to be run and the output has to be inspected seperately and manually. Secondly, they are not robust or the software has been outdated. Existing wrapper generators frequently rely on buggy HTML parsers which are not easy to upgrade. Thirdly and finally, they create wrappers which are hard to integrate in a federated system. The wrapper frequently is a standalone program which requires additional programming effort to integrate.



#1 Search Result: Extraction
www.accessscience.com/overflow.aspx?type=dst&root=0&term=Extraction
Extraction of ATP (in 'Soil microbial biomass and ATP' update) During ATP extraction from soil, it is necessary to disrupt the microbial cells in order to release ...

#2 Tutorial Search: mfcc feature extraction code, Page 34
www.roseindia.net/tutorialsearch/?t=mfcc%20feature%20extraction%20...
Displaying search result for: mfcc feature extraction code Java Code - Java Beginners Java Code Given a one-dimensional array named monthSales with 12 double values ...

#3 ature extraction ...
xtraction%20code
Java word extraction
Need to code a public class ...

#4 f,ppt,pps documents)
try to change your search
ults

Xpath	Nodes	Area	Min	Avg	grid
☐		-	-	-	-
☑ //li[./div/div/div/h3/a]	10	490880	0.69	0.84	10x1
☐ //li[./div/div/div/a]	9	443560	0.69	0.84	9x1
☐ //div[@id='sidebar']/div/ul/li[./a]	8	39200	0.95	0.99	8x1
☐ //div[@id='sa_bop']/div/div/div/div/ul/li[./a]	8	20514	0.95	0.99	4x2
☐ //div[contains(@class,'qscollast')]/ul/li[./a]	4	12870	0.95	0.99	4x1

Figure 1: The SearchResultFinder plugin in action

In this demonstration we present a system which: 1) is integrated in the browser and easy to operate; 2) requires only a single search result page for operation; 3) outputs reusable XPath expressions to extract search results using a programming language of your choice; 4) allows the user to visually inspect the extracted result.

3. THE ALGORITHM AND PLUGIN

Due to space limitations we can only give a brief description of the algorithm for extracting XPaths. The full algorithm is described in Trieschnigg et al. [6].

The overall approach is as follows. The browser is used to fetch the webpage, construct a DOM tree and render the search results. Both the constructed DOM tree and information about its rendered components are used in the algorithm.

First a set of candidate XPaths is generated. Repeating anchor (`<a>`) nodes are searched and grouped according to their generalized XPath, consisting of the node names encountered when traversing from the root to this anchor (for instance, `/html/body/div/div/a`). Based on attribute values of these anchor nodes and their ancestors, XPath predicates are generated and used to select subsets of (ancestor) nodes. These XPaths are simplified by adding predicates with unique attribute values. For instance the XPath `//html/body/div/a` might be generalized to `//div[id='results']/a`.

Second, this set of candidate XPaths is ranked based on a number of features including rendering and similarity of the nodes retrieved by this candidate XPath. The ranking is based on rules and manually set thresholds (based on a training collection described in [6]).

The algorithm is implemented in JavaScript and is available as a plugin for Firefox. The user navigates to a search result page, starts the plugin and a popup is presented with the ranked list of candidate XPaths found for the page. The nodes selected by these XPaths can be visually inspected by ticking them in the list. Fig. 1 shows a screenshot of the plugin active on a webpage.

4. EVALUATION

We evaluated the plugin on 148 web search engines in 23 diverse categories such as academia, audio, video, images, shopping, books, recipes, health and news. Table 1 lists the results. In more than 81% of the pages, the first suggested XPath correctly extracts the results. In 9% of the pages a correct XPath is found at the second or higher rank. For 10% of the pages, no correct XPath is suggested. The average running time per page is less than 1 second (on a basic pc).

Evaluated search engines	148
Correct XPath at rank 1	120
Correct XPath at rank 2	8
Correct XPath at rank > 2	5
Correct XPath not found	15
Mean reciprocal rank	0.84

Table 1: Performance in extracting XPaths

5. CONCLUSION

In this demonstation we presented SearchResultFinder, a browser plugin for quickly and easily determining XPaths to scrape search results from web search engines. The plugin is available online[2]. In future work we plan to extend the plugin with automatic detection and labeling of attributes, such as primary anchor, title, and thumbnail.

Acknowledgements

This research was supported by the Netherlands Organization for Scientific Research, NWO, grants 639.022.809, 612.066.513 and CATCH project FACT.

References

[1] M. Álvarez, A. Pan, J. Raposo, F. Bellas, and F. Cacheda. Extracting lists of data records from semi-structured web pages. *Data & Knowledge Engineering*, 64(2):491–509, 2008.

[2] V. Crescenzi and G. Mecca. Grammars have exceptions. *Information Systems*, 23(8):539–565, 1998.

[3] D. Freitag. Multistrategy learning for information extraction. In *Fifteenth International Conference on Machine Learning*, pages 161–169, 1998.

[4] B. Liu, R. Grossman, and Y. Zhai. Mining data records in Web pages. In *SIGKDD '03*, pages 601–606, 2003.

[5] K. Simon and G. Lausen. Viper: augmenting automatic information extraction with visual perceptions. In *CIKM '05*, pages 381–388, 2005.

[6] D. Trieschnigg, K. Tjin-Kam-Jet, and D. Hiemstra. Ranking XPaths for extracting search result records. Technical Report TR-CTIT-12-08, CTIT, University of Twente, Enschede, 2012.

[7] H. Zhao, W. Meng, Z. Wu, V. Raghavan, and C. Yu. Fully automatic wrapper generation for search engines. In *WWW '05*, pages 66–75, 2005.

[8] S. Zheng, R. Song, J.-R. Wen, and C. L. Giles. Efficient record-level wrapper induction. In *CIKM '09*, pages 47–56, 2009.

[2] `http://snipdex.org/srf/`