Semantic Clustering of Website Based on Its Hypertext Structure

Vladimir Salin[1](✉), Maria Slastihina[1], Ivan Ermilov[2], René Speck[2], Sören Auer[3], and Sergey Papshev[1]

[1] Saratov State Technical University, 410054 Saratov, Russia
salinvs@gmail.com, masha-fat@yandex.ru, spapshev@list.ru
[2] Universität Leipzig, AKSW/BIS, PO BOX 100920, 04009 Leipzig, Germany
{ivan.ermilov,speck}@informatik.uni-leipzig.de
[3] Universität Bonn, CS/EIS, Römerstraße 164, 53117 Bonn, Germany
auer@cs.uni-bonn.de

Abstract. The volume of unstructured information presented on the Internet is constantly increasing, together with the total amount of websites and their contents. To process this vast amount of information it is important to distinguish different clusters of related webpages. Such clusters are used, for example, for knowledge extraction, named entity recognition, and recommendation algorithms. A variety of applications (such as semantic analysis systems, crawlers and search engines) utilizes semantic clustering algorithms to recognize thematically connected webpages. The majority of them relies on text analysis of the web documents content, and this leads to certain limitations, such as long processing time, need of representative text content, or vagueness of natural language. In this article, we present a framework for unsupervised domain and language independent semantic clustering of the website, which utilizes its internal hypertext structure and does not require text analysis. As a basis, we represent the hypertext structure as a graph and apply known flow simulation clustering algorithms to the graph to produce a set of webpage clusters. We assume these clusters contain thematically connected webpages. We evaluate our clustering approach with a corpus of real-world webpages and compare the approach with well-known text document clustering algorithms.

1 Introduction

The volume of unstructured information presented on the Internet in human-readable form is constantly increasing, together with the total amount of websites and their contents. Technologies for extracting, analyzing, automatic accessing and processing data become increasingly more important in Web with its continuous growth. However, finding and analyzing information relevant to a problem at hand from such a vast amount of information is still a major challenge in the Web. With processing such information volumes, it is important to distinguish collections of thematically connected webpages. Webpage clustering is used widely in a variety of web data extraction applications,

P. Klinov and D. Mouromtsev (Eds.): KESW 2015, CCIS 518, pp. 182–194, 2015.
DOI: 10.1007/978-3-319-24543-0_14

for example, for knowledge extraction, search results representation or recommendation algorithms (i.e. [5,10,11,13]).

The majority of such applications use clustering methods based on text analysis, treating webpages as common text documents, pushing aside their hypertext attributes. This leads to well-known limitations of text analysis techniques (varies for different algorithms), i.e. polysemy-capturing problem, limitations of bag of words model or at least requirement to have a representative amount of text content in the document. In addition, text clustering usually requires a preliminary step of documents indexing, which should be performed for every document in a target collection (thus, requiring full text scan of every webpage). As for treating webpages as text documents, this also implies extracting target text content from the HTML structure of the webpage, excluding template wrapping or other insufficient information. Overall, it becomes reasonable to take into consideration not only text contents of website pages, but also their hypertext features. Search engines usually do this for page ranking to improve search results representation, by analyzing their indexes of incoming external hyperlinks from other websites to the current one and thus indirectly determining its topic. Obviously, it requires a global index of the websites in the Internet, which is unfeasible for the majority of applications.

To tackle the problem of website semantic clustering, we designed and implemented an approach, which clusters webpages using inner hypertext structure of the website. The rationale of our approach is that if we group webpages with a number of hyperlinks inside group higher than hyperlinks to webpages outside the group, we can consider webpages in the group as thematically connected. With partitioning a whole website in these groups, we form a set of semantic clusters of webpages. To perform this partition, we create a link graph of the corresponding hypertext structure as a basis and apply graph-clustering algorithms based on the flow simulation principle (e.g. MCL, BorderFlow)

In particular our contributions are:

– We describe website clustering approach based on the analysis of hypertext structure of the website.
– We design and implement software system, which can construct website model and apply clustering algorithms to provide collections of thematically connected webpages.
– With help of this system, we evaluate our approach and compare it with existing text-based clustering techniques on the corpus of real-world data.
– We provide the source code of the whole framework for further reuse[1].

The paper is structured as follows: In section 2, we outline existing web data extraction systems, clustering techniques utilized in those systems as well as related work on web data extraction for ontology learning and various clustering algorithms. In section 3, we discuss the crawling approach used in our framework we present. In section 4, we present the architecture of our keyword extraction framework and describe its general workflow. We evaluate our framework in section 5 and conclude as well as outline future work in section 6.

[1] https://github.com/sainnr/website-graph-cluster

2 Related Work

The research field Web Data Extraction (WDE) is dealing with Information Extraction (IE) on the Web. An outline of existing WDE systems and their application domains is given in [9]. In particular, the authors distinguish enterprise and social web applications. The survey provides insights on the techniques for WDE as well as it gives examples for possible applications. However, it does not delve into the inner workings of WDE systems and thus lacks information on clustering approaches and their usage inside WDE systems. Another WDE work is [8], concentrated solely on an overview of WDE techniques. The authors differentiate two main types of techniques, one is based on wrapper/template induction and an another is automatic extraction.

Another related field of research is IE for Ontology Learning (OL). In [12] Suchanek describes IE from well-formed sources, that are projects like DBpedia [2], YAGO [18] and KOG [21]. All three projects aim to extract information from Wikipedia and to construct an ontology, which supports querying (e.g. SPARQL queries) and question answering. In the same paper Suchanek reports about systems, which are designed to extract information from any web page on the Web: OntoUSP [17], NELL [4] and SOFIE [19]. These systems are centered around NLP processing of web documents using different approaches, but do not perform webpage clustering.

Web clustering has an important role in WDE and Semantic Web areas. The survey [5] explains and compares various clustering techniques applied to web documents. Most of them are text-clustering algorithms, including an efficient Suffix-Tree Clustering (STC) [22]. STC has been improved and become a basis for Lingo [15], both used in Carrot^2 framework[2] [16] and its commercial successor Carrot Search[3]. While these algorithms show good results on corpus of data of average size, they have certain limitations common for all of them: required full-text analysis for every webpage, polysemy-capturing problem, limitations of bag of words model and other.

In contrast to text-based clustering, various systems employ clustering of webpages based on their HTML features. This usually includes webpage's internal document object model (DOM) tree analysis, or mixed approach, where hypertext markup combined with text content during analysis. For instance, EXALG – an information extraction approach described in [1] – clusters webpages into "equivalence classes" by sets of tokens, found in the hypertext markup. As for another example, in [6] the authors utilize clustering based on paths in the DOM trees. Authors also list other DOM structure based clustering approaches. Generally, for these approaches the similarity of webpages to be grouped in one cluster is determined by similarity of various hierarchies in internal HTML structure of each webpage.

Hyperlinks between pages are also playing important role in Web Information Retrieval. Thus, search engines utilize incoming hyperlinks data for website's

[2] http://project.carrot2.org
[3] www.carrot-search.com

pages not only to rank webpages in search results [3], but also to determine their topic and similarity with other pages [7]. But for effective analysis with incoming hyperlinks, such approach requires a large corpus of indexed websites and webpages, which is commonly available only for search engines.

3 Graph-Based Clustering Approach

In this paper, we propose the website clustering approach underlined by the following hypothesis: websites have groups of pages with a higher level of connectivity inside the group than with pages outside. We suppose, that in general, links are specified between webpages with crossing topics. A higher level of connectivity in group of webpages display that topics are intensively shared between pages in a group. Such groups of webpages are forming sections of websites referring to certain topics.

An approach is applicable to any hypertext structure in general, while a website is a particular case of such hypertext model. Formally, we represent a hypertext structure $H = (P, L)$ with directed graph $G = (V, E)$, where:

- $V = P$ is a set of vertices $v_1, v_2, \ldots, v_n \in V$, which correspond to hypertext documents.
- $E = L$ is a set of edges between vertices: $e(v, u) \in E, v, u \in V$, which are hyperlinks between documents.

To find communities in the graph, we employ existing graph clustering algorithms, based on flow simulation principle. This can be, for instance, Markov Chain Clustering (MCL) [20] or BorderFlow [14] algorithm. Results include collections of clustered vertices of the graph.

Mentioned graph clustering algorithms assume that provided as input directed graph is a weighted graph. Although weights are not required for the most of the clustering algorithms, the usage of weights can drastically improve the clustering results. This can be used as extension point of the approach. In example, weights can be pre-calculated with another clustering algorithm (e.g. text-based clustering).

We also suppose that our approach will be able to include into clusters webpages without textual content. By analysing incoming hyperlinks to such pages from others, we can discover semantics of these pages without necessity to analyse media or binary data contained in them.

Finally, we presume that performance of graph-based website clustering will be higher than common text-based clustering (which includes indexing and further index analysis). For both types of approaches, in this test we will assume that required data has been already extracted from the website, to exclude crawling and transporting costs. Required data includes connectivity structure (*"webpage A has a link to webpage B"*) and text contents from the webpages for graph-based and text-based clustering, correspondingly.

To examine our approach, in section 5 we compare the results of graph-based clustering with credible text-based algorithm Carrot[2] on the corpus of webpages

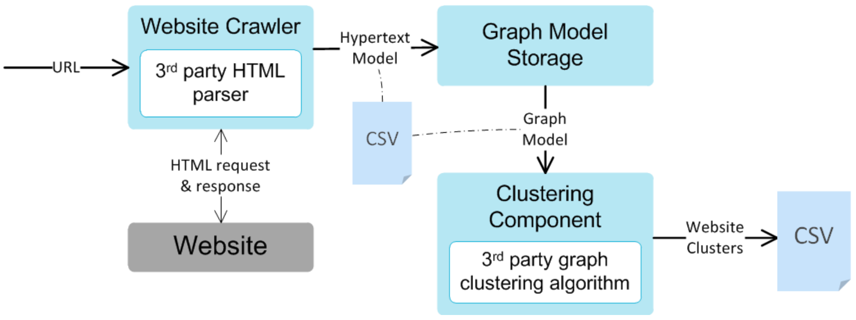

Fig. 1. The architecture of the keyword extraction process

of a real-world website. We also prove the ability to determine clusters for webpages with no text content, and estimate a performance of the approach, in comparison with existing one.

4 Architecture and Workflow

In this section, we explain the architecture and the workflow of the framework that we have designed to perform graph-based clustering of the website.

The framework consists of three major components, highlighted on the figure above (see Figure 1):

1. The *website crawler* parses the hypertext structure of the website by provided URL;
2. The *graph model storage* constructs a graph model of the hypertext from the data obtained by website crawler, and stores it for further analysis;
3. The *website clustering component* clusters the graph from graph model storage with one of supported algorithms.

As for software platform, we use Java 7 web application, running on Apache Tomcat application server. It also uses additional libraries for HTML data extraction and graph clustering. The following subsections describe Design and Workflow of each component in detail.

4.1 Parsing the Hypertext Structure of Web Sites

To extract required information from the webiste, we use straightforward HTML parser – JSOUP[4]. So far, we assume that every webpage has no asynchronous content and fully loaded at a time JSOUP connects to the host. We understand, that various modern websites intensively employ JavaScript and interactive content on the webpages, but the crawling of this sort of data is not in scope of this paper, as well as extracting data from media. This limitation can be eliminated

[4] http://jsoup.org

by using modern state of the art crawlers which are able to extract information from media and interactive contents (e.g. OXPath crawler[5]).

We have built our crawler around JSOUP and provided interface to crawl a whole website by its URL. As the result, the website crawler component parses a selected website and extracts the following types of information: (1) each webpage URL, (2) links between webpages, (3) webpage text content (not required for graph clustering).

The component outputs its crawling data into CSV file, where every row looks like the following:

[ID, URL from, URL to]

The file represents a connectivity of the website and can be used for further purposes.

4.2 Constructing the Web Site Graph

With crawling results provided from the previous component, we can easily build a graph model of the hypertext structure. As it was mentioned in section 3, we treat webpages as vertices and hyperlinks as edges. The order of pages in hyperlinks matters, because we need a directed graph to use as an input for clustering algorithm.

Regarding weights, we have no additional information extracted from the website by default, which can be treated as weight and meet our clustering approach. So we use the same weight for all edges equal to 1.

The component converts constructed graph model to the specific CSV format, expected by clustering component:

[URL from, URL to, Weight]

4.3 Graph Model Clustering

We use existing implementations of flow simulation algorithms in correspondence with the approach presented in section 3. The CUGAR Graph Clustering and Visualization Framework[6] serves well in this case. It written in Java and includes implementations of the following five algorithms: Affinity Propagation, BorderFlow, Chinese Whispers, k-Nearest Neighbors and MCL algorithm. We use BorderFlow and MCL in our approach as graph clustering algorithms. This library also allows using these algorithms through an API without heavy interfaces and visualizations.

We provide graph model data obtained from the previous component to CUGAR and execute BorderFlow and MCL for it. An API of CUGAR supports various parameters of the clustering, on which clustering results depend. We show specific values of these parameters along with clustering results in section 5.

[5] http://www.oxpath.org
[6] https://github.com/renespeck/Cugar

When clustering complete, CUGAR produces output file, in which every row contains a collection of vertices corresponding to a certain cluster. In section 5, we examine these clusters and conclude about efficiency.

5 Evaluation

To evaluate the clustering approach, described above, we have used the following techniques. At first, we showcase clustering results: a number of clusters, produced by applying chosen algorithms, average size of each cluster, standard deviation and time consumed. Then, we compare execution time and measure performance of these algorithms. After that, we estimate precision of different clustering approaches: one provided in this paper and others based on text clustering. To perform this, we compare results in semantic clustering problem, where website clustering needs to produce a number of webpage groups related to a certain topic. There are two basic requirements for clusters: 1) Defined topics should not be too broad and common, e.g. related to the whole website, 2) Each cluster should include at least three webpages. Finally, we examine how the proposed approach deals with webpages without text content, e.g. media or other specific formats.

5.1 Experimental Setup

To perform evaluation, we use a real-world website as data source. The website of Agile Knowledge and Semantic Web (AKSW) Research Group[7] provides information about AKSW research group, their projects, team members, events, etc. It has about 500 hypertext documents available on the main domain. Also, there is a set of OntoWiki webpages, which mostly contains technical and administrative information and so were not analyzed.

From the technical point of view, we use an implementation of our framework, written in Java. We use BorderFlow and MCL algorithms as graph-based clustering, provided by CUGAR library. To compare with, we use Carrot^2 workbench for text-based clustering, which provides implementations of two industry-proven clustering algorithms: STC and Lingo. Overall, we compare four different algorithms of clustering.

The processes of website crawling, clustering and keywords analyzing were performed on dualcore Intel Core i5 CPU M460 with 2.53GHz and 4Gb RAM. It runs under Windows 8.1 with Java(TM) SE Runtime Environment (build 1.8.0 45−b15), used for Java applications.

5.2 Experimental Results

As it was described in evaluation approach, we have performed website clustering using four different algorithms. Beforehand we were required to extract data from

[7] http://aksw.org

the website with help of the crawler: hypertext structure required for graph model construction and text contents for text-based clustering.

From 498 hypertext documents, found on the main domain of aksw.org, only 343 of them were unique, while others had similar URLs with duplicate content.

Table 1. Data extraction results from http://aksw.org.

	.html	*.rdf*	*.ttl*	*.csv*	*Others*
Num. of documents (%)	343 (26.6%)	479 (37.1%)	451 (34.9%)	9 (0.7%)	9 (0.7%)

In addition, there were found almost 950 non-hypertext documents of different format, linked from these pages (see Table 1).

Table 2. Clustering characteristics of chosen algorithms applied to aksw.org.

	MCL	*BorderFlow*	*Lingo*	*STC*
Number of Clusters	77	108	51	58
Avg. Docs per Cluster	12.54	9.05	31.90	97.88
Standard Deviation	77.16	21.54	19.51	47.38
Time Spent, sec.	8.9	27.8	1.9	0.2

A brief overview of cluster contents is given in Table 2.

To measure a performance of the approach using BorderFlow and MCL, we have measured 20 test clustering runs of these algorithms. Using default implementations with no additional performance improvements, we have received average 27.8 sec for clustering with BorderFlow and 8.9 sec for MCL for their better input parameters. We also used benchmark results provided by Carrot^2 workbench. Lingo and STC are tuned better and show 1.9 sec and 0.2 sec in average, correspondingly. BorderFlow and MCL computed a graph with whole 1290 documents found on aksw.org, while Lingo and STC operated only HTML documents, which are only 343. Due to technical reasons, we were unable to complete clustering with *Inflation* higher then 1.46.

There were not enough clusters with $Inflation = 1.46$ for uniform coverage of all webpages, and cluster dispersion was too high. In our expectations, the best results in matter of precision and time are provided with *Inflation* close to 2.0 (see Figure 2). We suppose that in further experiments it will produce results similar to BorderFlow clustering.

We have examined precision of the approach in our semantic clustering task. To do that, we manually checked all clusters provided by clustering algorithms and made conclusion regarding 1) the semantic similarity of pages in the cluster, 2) how cluster meets our requirements of topic broadness and minimum size.

Thus, for graph-based algorithms, we have computed a precision for every cluster and than got average from all clusters. Rather common example of precise cluster produced by BorderFlow is described below.

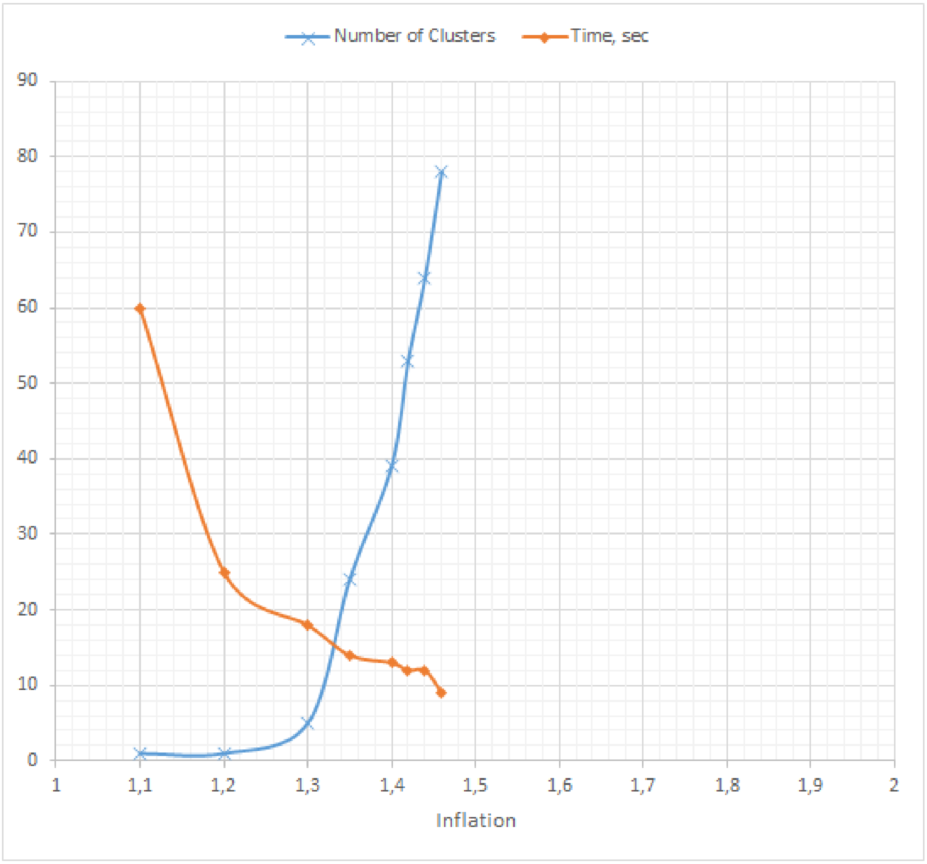

Fig. 2. Dependency of execution time and clusters number from Inflation value.

As shown on Figure 3, the sample cluster contains such URLs as http://aksw.org/DiegoEsteves, http://aksw.org/BettinaKlimek, http://aksw.org/Groups/NLP2RDF, etc. All of them relate to personal pages of the *NLP2RDF* group members. We can make sure of this after looking on the real Web page with URL (see Figure 4).

For text-based clustering, we assumed that precision equals 100% for all clusters because of the approach specifics. Nevertheless, clusters itself are often broader than needed, so STC proposed clusters under such topics as *Open*, *Systems*, *Current* and others.

So far, BorderFlow showed unpredictably good results and came with almost 95% of precision (see Table 3). MCL has failed due to technical reasons, and we are going to eliminate this issue in further experiments. Lingo solved the problem very well, producing only few clusters with broad topic with resulting precision of 96%. In comparison to Lingo, STC did not so well, returning lots of wide and uncertain topics defined by common keywords. Additionally, Lingo and STC produced a set of documents related to *Others* cluster, in other words, not related to any other

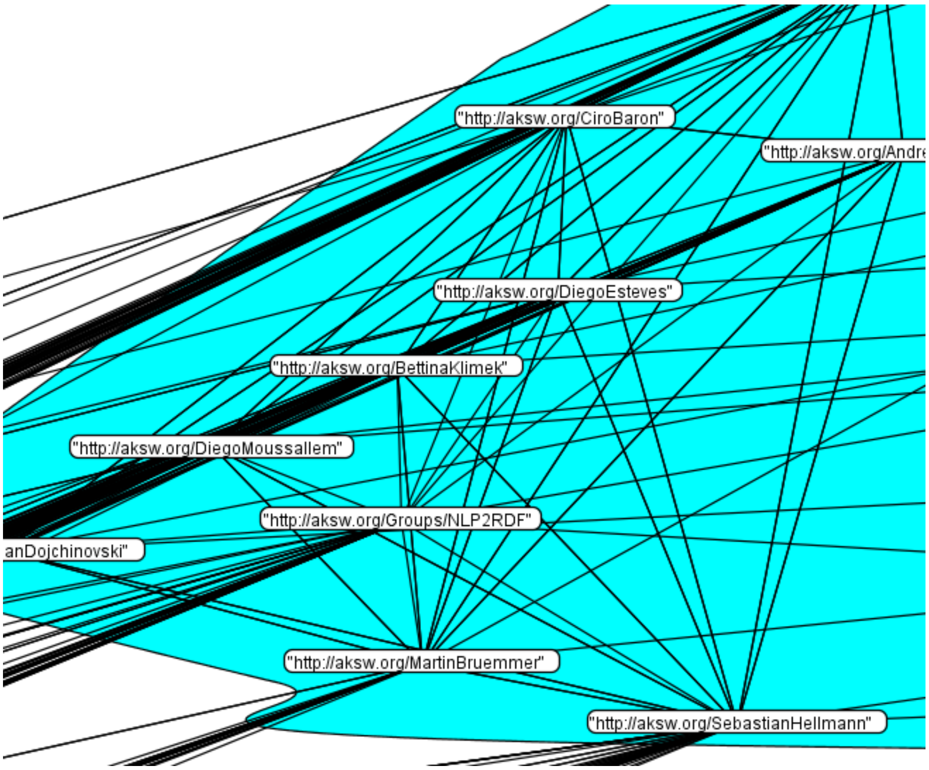
"http://aksw.org/CiroBaron"
"http://aksw.org/Andre
"http://aksw.org/DiegoEsteves"
"http://aksw.org/BettinaKlimek"
"http://aksw.org/DiegoMoussallem"
"http://aksw.org/Groups/NLP2RDF"
anDojchinovski"
"http://aksw.org/MartinBruemmer"
"http://aksw.org/SebastianHellmann"

Fig. 3. Sample cluster visualization with CUGAR framework.

cluster. While for STC such "cluster" included only 2 documents, a corresponding cluster of Lingo consists of 51 document. Graph-based algorithms have no such non-clustered documents.

Finally, we have assessed how clustering approach deals with non-text documents. As it was mentioned above, text-based clustering is inapplicable here. While graph-based algorithms correctly detected clusters for such pages. They have found exact clusters with .ttl and .rdf documents related to original Web pages with no need to analyze file contents. For other formats, like CSV or PDF, they also did well and covered such documents with clusters of well-known pages.

Table 3. Average precision of clusters & number of non-clustered documents produced by different algorithms.

	MCL	*BorderFlow*	*Lingo*	*STC*
Average Precision	15.1%	94.9%	96.1%	84.5%
Num. of Non-clustered Docs	–	–	51	2

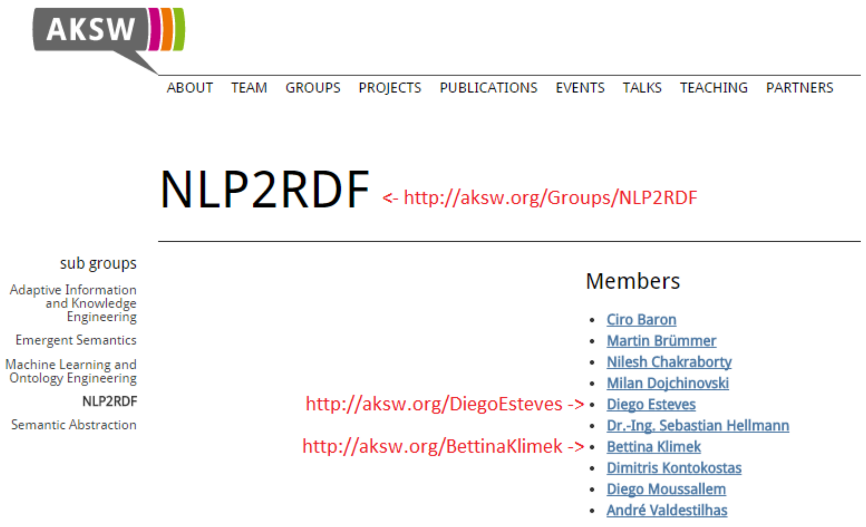
AKSW
ABOUT TEAM GROUPS PROJECTS PUBLICATIONS EVENTS TALKS TEACHING PARTNERS
NLP2RDF
<- http://aksw.org/Groups/NLP2RDF
sub groups
Adaptive Information and Knowledge Engineering
Emergent Semantics
Machine Learning and Ontology Engineering
NLP2RDF
Semantic Abstraction
Members
Ciro Baron
Martin Brümmer
Nilesh Chakraborty
Milan Dojchinovski
http://aksw.org/DiegoEsteves ->
Diego Esteves
Dr.-Ing. Sebastian Hellmann
http://aksw.org/BettinaKlimek ->
Bettina Klimek
Dimitris Kontokostas
Diego Moussallem
André Valdestilhas

Fig. 4. A screenshot taken from NLP2RDF webpage with URL highlights.

6 Conclusions and Future Work

This article describes graph-based Web site clustering framework. The framework is able to detect semantic clusters in the website with no need to analyze text contents of every page. It also can help to cover with clusters those pages, which contents are hard to analyze directly, like binary documents, media files and others. Framework shows medium performance in comparison with industry-proven and well-tuned clustering algorithms (e.g. Lingo or STC), but has wide capabilities to be improved in this direction.

Our clustering-based keyword extraction framework still has the following limitations. Most obvious is the crawling limitation i.e. the lack of interactive content support. This is an implementation-specific limit which can be bypassed by using more complex crawling tools (e.g. OXPath). As it was mentioned in section 5, we have experienced a technical issue with MCL implementation. We are going to solve it in further experiments and expect results similar to BorderFlow algorithm. Another way to improve the approach is to add weights.

Graph-based clustering algorithms support weighted graphs, so we can use an external source of weights to improve clustering precision. For instance, we can combine them with text-based clustering algorithms and use such mixed approach to reduce limitations from of approaches.

Quality measurement techniques also require further improvements. Currently used evaluation approach with manual assessment of clusters has certain limitations. This includes poor scalability to apply the approach for large websites of different knowledge domains. To verify experimental results on additional websites, new evaluation techniques will be used.

We plan extend the framework in two directions: mitigate limitations and apply the framework as a backend for existing applications (e.g. conTEXT[8]).

References

1. Arasu, A., Garcia-Molina, H.: Extracting structured data from web pages. In: Proceedings of the 2003 ACM SIGMOD international conference on Management of data, pp. 337–348. ACM (2003)
2. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.G.: DBpedia: a nucleus for a web of open data. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007)
3. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. Computer Networks and ISDN Systems **30**(1–7), 107–117 (1998). Proceedings of the Seventh International World Wide Web Conference. http://www.sciencedirect.com/science/article/pii/S016975529800110X
4. Carlson, A., Betteridge, J., Wang, R.C., Hruschka, Jr., E.R., Mitchell, T.M.: Toward an architecture for never-ending language learning. In: Proceedings of the Conference on Artificial Intelligence (AAAI) (2010)
5. Carpineto, C., Osinski, S., Romano, G., Weiss, D.: A survey of web clustering engines. ACM Computing Surveys **41**(3), July 2009. http://doi.acm.org/10.1145/1541880.1541884
6. Chakrabarti, D., Mehta, R.: The paths more taken: matching dom trees to search logs for accurate webpage clustering. In: Proceedings of the 19th International Conference on World Wide Web, pp. 211–220. ACM (2010)
7. Croft, W.B., Metzler, D., Strohman, T.: Search engines: Information retrieval in practice, chap. 4.5. Addison-Wesley Reading (2010)
8. Devika, K., Surendran, S.: An overview of web data extraction techniques. International Journal of Scientific Engineering and Technology **2**(4) (2013)
9. Ferrara, E., Meo, P.D., Fiumara, G., Baumgartner, R.: Web data extraction, applications and techniques: A survey. CoRR abs/1207.0246 (2012)
10. Hollink, V., van Someren, M., Wielinga, B.J.: Navigation behavior models for link structure optimization. User Modeling and User-Adapted Interaction **17**(4), 339–377 (2007)
11. Kosala, R., Blockeel, H.: Web mining research: A survey. ACM Sigkdd Explorations Newsletter **2**(1), 1–15 (2000)
12. Lehmann, J., Völker, J. (eds.): Studies on the Semantic Web, chap. Information Extraction for Ontology Learning. Akademische Verlagsgesellschaft - AKA GmbH, P.O. Box 41 07 05, 12117 Berlin, Germany (2014)
13. Ngomo, A.C.N., Lyko, K., Christen, V.: Coala-correlation-aware active learning of link specifications. In: The Semantic Web: Semantics and Big Data, pp. 442–456. Springer (2013)
14. Ngonga Ngomo, A.-C., Schumacher, F.: Borderflow: a local graph clustering algorithm for natural language processing. In: Gelbukh, A. (ed.) CICLing 2009. LNCS, vol. 5449, pp. 547–558. Springer, Heidelberg (2009)

[8] http://context.aksw.org/app/

15. Osinski, S., Stefanowski, J., Weiss, D.: Lingo: search results clustering algorithm based on singular value decomposition. In: Proceedings of the International Conference on Intelligent Information Systems (IIPWM 2004), Zakopane, Poland, pp. 359–368 (2004)
16. Osiński, S., Weiss, D.: Carrot^2: design of a flexible and efficient web information retrieval framework. In: Szczepaniak, P.S., Kacprzyk, J., Niewiadomski, A. (eds.) AWIC 2005. LNCS (LNAI), vol. 3528, pp. 439–444. Springer, Heidelberg (2005)
17. Poon, H., Domingos, P.: Unsupervised ontology induction from text. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 296–305. ACL 2010, Association for Computational Linguistics, Stroudsburg (2010). http://dl.acm.org/citation.cfm?id=1858681.1858712
18. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: a core of semantic knowledge. In: Proceedings of the 16th International Conference on World Wide Web, pp. 697–706. ACM (2007)
19. Suchanek, F.M., Sozio, M., Weikum, G.: Sofie: a self-organizing framework for information extraction. In: Proceedings of the 18th International Conference on World Wide Web, pp. 631–640. ACM (2009)
20. Van Dongen, S.M.: Graph clustering by flow simulation (2001)
21. Wu, F., Weld, D.S.: Automatically refining the wikipedia infobox ontology. In: Proceedings of the 17th International Conference on World Wide Web, pp. 635–644. ACM (2008)
22. Zamir, O., Etzioni, O.: Web document clustering: a feasibility demonstration. In: SIGIR 1998: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia, August 24–28 1998, pp. 46–54 (1998). http://doi.acm.org/10.1145/290941.290956