

CAF-SIAL: Concept Aggregation Framework for Structuring Informational Aspects of Linked Open Data

Atif Latif^{1,2}, Muhammad Tanvir Afzal³, Anwar Us Saeed^{1,4}, Patrick Hoefler^{5,6}, and Klaus Tochtermann^{1,5,7}

*{¹Institute for Knowledge Management , ⁵Know-Center}
Graz University of Technology, Inffeldgasse 21a, 8010 Graz, Austria,
{²atif.latif, ⁴anwar.ussaeed }@student.TUGraz.at, ⁶phoefler@know-center.at,
⁷klaus.tochtermann@TUGraz.at
³Institute for Information Systems and Computer Media (IICM)
Graz University of Technology, Inffeldgasse 16c, 8010 Graz, Austria
mafzal@iicm.edu*

Abstract

Linked Open Data (LOD) is becoming an essential part of the Semantic Web. Although LOD has amassed large quantities of structured data from diverse, openly available data sources, there is still a lack of user-friendly interfaces and mechanisms for exploring this huge resource. In this paper we highlight two critical issues related to the exploration of the semantic LOD pool by end users. We introduce a proof of concept application which helps users to search information about a concept without having to know the mechanics of the Semantic Web or Linked Data. We assume that this kind of application may lead to bridge the gap between semantic search and end users. With this application, we concentrated on two aspects:

1) A novel Concept Aggregation Framework to present the most relevant information of LOD resources in an easy to understand way.

2) A simplified keyword search mechanism which hides the complex underlying semantic search logic.

This research is intended to simplify the LOD end user interfaces so that they may be used by novice users who don't possess any prior knowledge of semantic structures.

1. Introduction

The World Wide Web can be considered as a huge repository of networked resources. Due to the exponential growth of World Wide Web, it is a challenging task for search engines to locate meaningful pieces of information from heavily

redundant and unstructured resources. The semantic paradigm of information processing suggests a solution to the above problem. Semantic resources are structured, and related semantic metadata can be used to query and search the required piece of information precisely. On the other hand, the bulk of the data currently residing on the Web is unstructured or semi-structured at best. Therefore, the W3C SWEO launched the Linking Open Data¹ movement, a community effort that motivates people to publish their information in a structured way (RDF)². LOD not only “semantifies” different kind of open data sets but also provides a framework for the interlinking of these datasets. This framework is based on the rules described by Tim Berners-Lee [1]. As of May 2009, the LOD cloud consists of over 4.7 billion RDF triples, which are interlinked by around 142 million RDF/OWL links. Although LOD has gained huge volumes of data and has attracted attention of many researchers, it still lacks broad recognition, especially in commercial domains. This is, amongst other reasons, because of complex semantic search and end user applications [2].

In the absence of official standards, DBpedia³ and Yago⁴ are considered de facto standards for classification. DBpedia is also considered a hub of linked data for interlinking and finding facts. Facts about a specific resource, extracted from the infoboxes of Wikipedia, are structured in the form of properties defined in DBpedia's ontology. This

¹<http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData>

²<http://www.w3.org/RDF/>

³<http://dbpedia.org>

⁴<http://www.mpi-inf.mpg.de/yago-naga/yago/>

vocabulary is also associated with Yago classification to identify the type (person, place organization etc.) of the resource. For instance, a query about *Arnold Schwarzenegger* returns about 260 distinct properties encapsulating nearly 900 triples in the raw RDF form. Such semantic data is not easily graspable by end users. Representing this bulk of structured information in a simple and concise way is still a challenge.

Recently, a few applications have emerged which provide user interfaces to explore LOD datasets [4, 5]. These applications use SPARQL endpoints to query LOD with Subject Predicate Object (SPO) logic. SPO logic represents a triple, which is a building block of RDF. Triples establish a relationship between two resource types. One resource is called subject and the other one is called object. The relationship between subject and object is called predicate. For example, subject (*Arnold Schwarzenegger*), predicate (*is governor of*) and object (*California*) form a triple. Now, in order to exploit LOD resources using SPARQL endpoint interfaces of recent applications, users had to understand the underlying semantic structures (triples, ontologies, properties). The same gap between semantic search and end user applications has also been identified by [3].

To overcome the aforementioned problems, we have developed a proof of concept application called CAF-SIAL. In this application we have contributed in two ways:

- 1) We introduced a Concept Aggregation Framework which selects a set of properties related to a particular informational aspect of a resource type. This approach conceptualizes the most relevant information of a resource in an easily perceivable construct.

- 2) We proposed a two step keyword search process in order to hide the underlying SPO logic. In the first step, users search for a keyword, and the system auto-suggests related entries to exactly specify the subject. Then, information related to that subject is structured using the aggregation framework. Furthermore, to avoid searching a specific property (Predicate) of the selected Subject by its name, a keyword based 'search within' facility is provided where the specified keyword is mapped to a certain property or set of properties.

The remainder of this paper is structured as follows: Section 2 discusses the state of the art and related work. Section 3 elaborates on the Concept Aggregation Framework. Section 4 describes the system architecture. Section 5 explains the overall use of the system with the help of a use case scenario. Conclusions and future works are discussed in section 6.

2. Related Work

The current state of the art with respect to the consumption of Linked Open Data for end users is RDF browsers [4, 5]. Some tools such as Tabulator [4], Disco⁵, Zitgist data viewer⁶, Marbles⁷, Object Viewer⁸ and Open link RDF Browser⁹ can explore the Semantic Web directly. All these tools have implemented a similar exploration strategy, allowing the user to visualize an RDF sub-graph in a tabular fashion. The sub-graph is obtained by dereferencing [6, 7] an URI and each tool uses a distinct approach for this purpose. These tools provide useful navigational interfaces for the end users, but due to the abundance of data about a concept and lack of filtering mechanisms, navigation becomes laborious and bothersome. In these applications, it is a tough task for a user to sort out important pieces of information without having the knowledge of underlying ontologies and basic RDF facts. Keeping in mind these issues, we suggest a keyword search mechanism to reduce the cognitive load of the users.

Regarding the problem of searching and filtering in the Web of data, a number of approaches and tools exist. One approach is to query a SPARQL endpoint that returns a set of RDF resources. There are a few tools that allow to explore a SPARQL Endpoint. NITELIGHT [8], iSparql [9], Explorator [10] are Visual Query Systems (VQS) [11] which allow visual construction of SPARQL queries, differing mainly in the visual notation employed. However, in order to use these tools, the user must have comprehensive knowledge of the underlying RDF schemata and the semantic query languages (e.g. SPARQL). In summary, current tools allow users to manipulate the raw RDF data and do not provide user-friendly interfaces.

Contrary to VQS applications, Freebase Parallax [12], the winner of Semantic web challenge 2006, is based on the idea of faceted search. Freebase Parallax is a browser for exploring and presenting the structured data in a centralized infrastructure. Similar faceted search application YARS2 [13] explores distributed datasets using SPO constructs.

The interface of Freebase seems to be closely related to the one of CAF-SIAL; however, our proposed application differs in two crucial points:

- 1) It makes use of distributed LOD.
- 2) It also provides the facility of keyword-based search for filtering particular information within the results.

To the best of authors' knowledge, the approach presented in this paper is the first one that uses arbitrary data accessible via SPARQL and

⁵ <http://www4.wiwiw.fu-berlin.de/bizer/ng4j/disco/>

⁶ <http://dataviewer.zitgist.com/>

⁷ <http://beckr.org/marbles>

⁸ <http://objectviewer.semwebcentral.org/>

⁹ <http://demo.openlinksw.com/rdfbrowser/index.html>

aggregates important facts on the basis of informational aspects.

3. Concept Aggregation Framework

The Concept Aggregation Framework aggregates relevant concepts from DBpedia and organizes the most important informational aspects related to a resource.

The scope of this application is limited to DBpedia and Yago. DBpedia covers 23 types of resources (place, person, organization etc), we selected the resource type person for our initial proof of concept application.

The Concept Aggregation Framework is shown in Figure 1. The aggregation classification layer is responsible for aggregating the most relevant information related to the person in question. This information is collected based on the list of related properties compiled at the property aggregation layer. The properties are extracted from knowledge bases shown in the aggregation knowledge bases layer.

3.1 Aggregation Knowledge Bases Layer

DBpedia, Yago and Umbel ontologies mainly contribute in the identification and classification of the resources. Two of them (DBpedia and Yago) are considered complete knowledge bases [14]. The underlying mechanism in our system is as follows:

We have generated two knowledge bases, a DBpedia Property Dump and a Yago Classification Dump. The DBpedia Property Dump is built by querying each type of a person from SNORQL query explorer¹⁰ (SPARQL endpoint of Dbpedia). Then we aggregate all the distinct property sets for each person. The formulated query for this operation is given below:

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT DISTINCT ?p
WHERE {
?s ?p ?o .
?s rdf:type <http://DBpedia.org/ontology/Artist> .
}
```

Properties for each person type are shown in Table 1.

Table 1. Person's property list

Person Type	Total	Picked Properties
Artist	2111	409
Journalist	186	55
Cleric	419	76
BritishRoyalty	252	47
Athlete	2064	496
Monarch	337	50
Scientist	421	126
Architect	132	41
PlayboyPlaymate	125	37
Politician	36	18

¹⁰<http://dbpedia.org/snorql/>

MilitaryPerson	725	158
FictionalCharacter	599	273
Criminal	287	74
CollegeCoach	282	124
OfficeHolder	1460	634
Philosopher	226	71
Astronaut	168	62
Model	211	99
Celebrity	X	X
Judge	X	X
FootballManager	X	X

The Yago Classification Dump is built by querying subclasses of Person class from SNORQL query explorer. The query looks like this:

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT DISTINCT ?s
WHERE {
?s rdfs:subClassOf
<http://DBpedia.org/class/yago/Person100007846> .
}
```

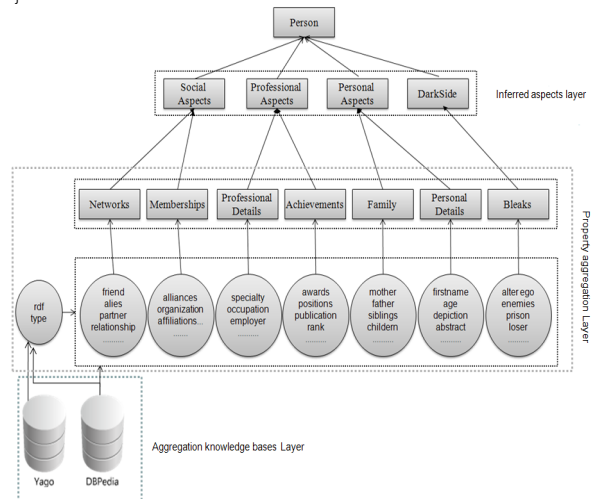


Figure 1. Concept Aggregation Framework

3.2 Property Aggregation Layer

This layer first identifies the profession type. This works in two steps. In the first step, the resource type (RDF type) is identified by using DBpedia. In case that in the retrieved set of properties, there is no property mapped within DBpedia knowledge base, the system tries to map the retrieved property to a Yago class. For example if the retrieved property is “AustrianComputerScientist” which is not listed in DBpedia knowledge base, then the system maps it to the Yago hierarchy and can infer that the person belongs to the profession of “Scientist” because “AustrianComputerScientist” is a subclass of “Scientist”.

Based on a resource type, we have extracted all the possible properties from the DBpedia Property Dump. We then have manually identified sets of properties indicating an informational concept (networks, memberships, family, achievements etc.) related to a person. These concepts are aggregated

and mapped to the related informational aspect identified in the inferred aspects layer. More than one concept may be mapped to a single informational concept defined at inferred aspects layer.

3.3 Inferred Aspects Layer

The information for a resource such as person may be organized and viewed in different informational aspects like personal, professional, social etc. The most popular search engine like Google also tries to present such informational aspects related to a topic in its top results. It has been shown in [17] that how Google rank its results to provide the most relevant contents. For example, in a response to a user query of “*Bill Clinton*”, Google top ten results are based on the Clinton personal (biography), his Professional (president, writer), videos, dark side of Clinton (negative reviews) etc. These results, however, depends on the complex link analysis of Web pages (citations to Web pages from different sources) along weight mechanisms assigned to different factors [16,15]. Google is considered as the most popular search engine having 64.2 % share in U.S search market [18]. Inspired from Google’s success in calculating and presenting the results in diverse and important informational aspects related to a query, we developed a concept aggregating framework. We have identified set of facts that should be displayed for a searched person. These are social, professional, personal and dark side.

4. Architecture

The system architecture is depicted in Figure 2. The implemented system is divided into four modules called query manager, auto-suggestion module, information retrieval module and search within property module.

The query manager is a controlling module of the application. It is responsible in translating the keyword search query into SPARQL queries.

The auto-suggestion module helps users to disambiguate entered search term.

The information retrieval module is responsible for locating the URIs and extracting related information.

The search within property module provides the facility of searching within all retrieved properties of a resource.

4.1 Auto-Suggestion Module

The query manager triggers the auto suggestion module by converting the searched keyword of a user into a SPARQL query. The local DBpedia disambiguation triple store is first used to find the set of ambiguous concepts using SPARQL query. These

results will be suggested to the user. If user select any of the suggested terms or in case of a distinct query (no auto-suggestion yielded), the searched term will be passed to the information retrieval module for further processing.

4.2 Information Retrieval Module

This module is further divided into four processes:

- 1) URI locator
- 2) LOD retrieval
- 3) Parser
- 4) Concept aggregation

The searched term is passed to the URI locator process which will query to the locally maintained datasets i.e. (DBpedia Title TS, DBpedia Person Data TS, and DBLP TS) to get a URI. If this fails, a new query is formulated for the SINDICE¹¹ Web service to locate the URI. After locating the URI of a resource, the LOD retrieval process dereferences that URI at the DBpedia server to get the respective resource RDF description. This RDF description is further passed to the Parser process. This process will parse RDF description into triples and stored them locally. Concept aggregation process is called to sort out the most important information aspect of the resource and in the end, the output is presented to the user.

4.3 Search within Property Module

This module lets the user search within all properties of a resource retrieved from the information retrieval module.

When a user enters a keyword to search some information about a resource, synset extraction process query wordnet to retrieve the synset of searched keyword. This synset will be passed to the query manager and for each word in synset it will query the local triple store through the property locator process. The property locator process matches the keyword as substring in the retrieved property set. All matched properties are then extracted and presented to the user.

5. Case Study

The system evaluation with the help of a use case scenario is explained here. We have selected a person “Arnold Schwarzenegger” affiliated with four interesting and diverse professions along with multiple awards and achievements.

¹¹ <http://sindice.com/>

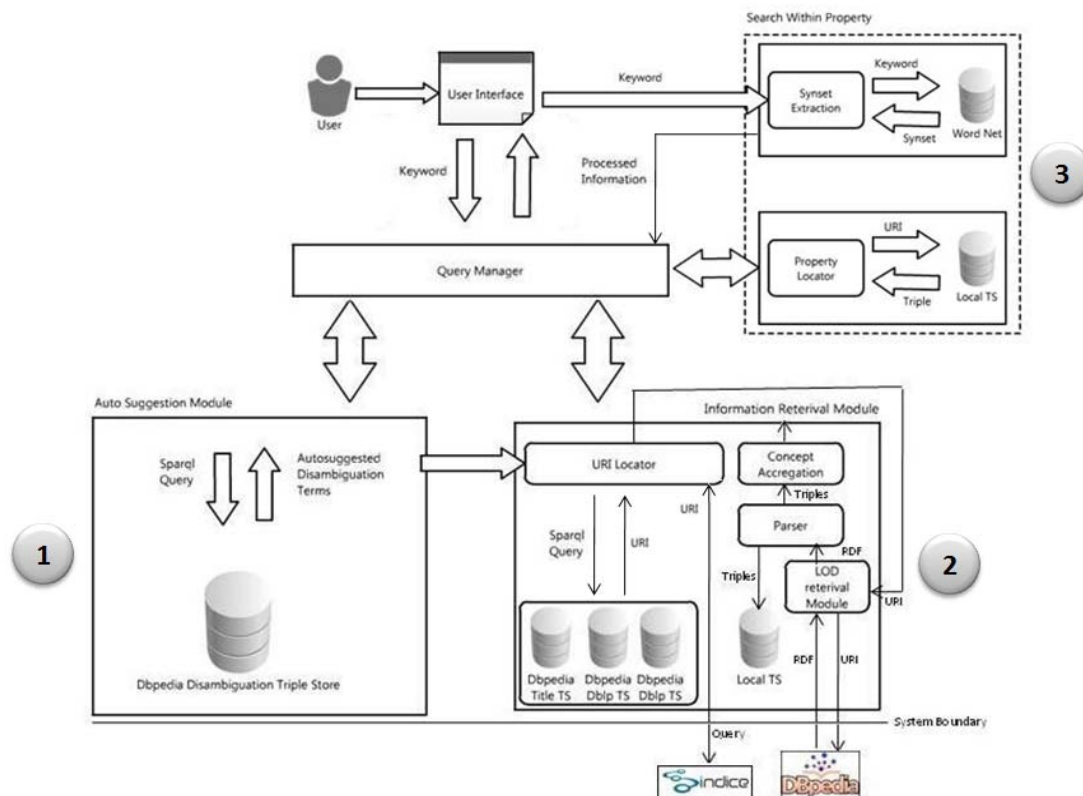
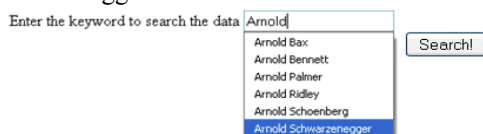


Figure 2. System Architecture

These capabilities make him a distinct person and a suitable choice for the use case. The application flow is explained as follows: User starts typing the search term “Arnold”. The persons’ names starting with the keyword “Arnold” are auto-suggested. For example “Arnold Bax”, “Arnold Bennett”, “Arnold Schwarzenegger” etc as shown below.



User selects “Arnold Schwarzenegger” to see his details as shown in Figure 3. Output is comprised of his four informational aspect named as Social, Personal, Professional and Dark side. Important properties are shown on the top for each informational aspect frame. The important property list was prepared and weighted manually.

From the screenshot it is obvious that all of his important professional details have been shown concisely and in easily graspable manner.

6. Conclusion and Future Work

This work tries to bridge the gap between semantic search and the end user. The proposed keyword-based search mechanism has simplified the

process of finding information from LOD by hiding underlying semantic logic. Users are encouraged to access the application from <http://cafsial.hoefler.st>. With the help of Concept Aggregation Framework, the information related to a resource (consisting of hundreds of properties) was structured in major and most relevant categories of informational aspects. This reduced the users’ cognitive load to find the required information. However, the set of properties were ranked manually. In our future implementation, we will develop an algorithm for the auto-clustering of properties. We also envision the extension of this application to cover resources other than person.

Acknowledgements

This contribution is partly funded by the Know-Center and the Higher Education Commission of Pakistan. The Know-Center is funded within the Austrian COMET program -- Competence Centers for Excellent Technologies -- under the auspices of the Austrian Federal Ministry of Transport, Innovation and Technology, the Austrian Federal Ministry of Economy, Family and Youth and the State of Styria. COMET is managed by the Austrian Research Promotion Agency FFG.

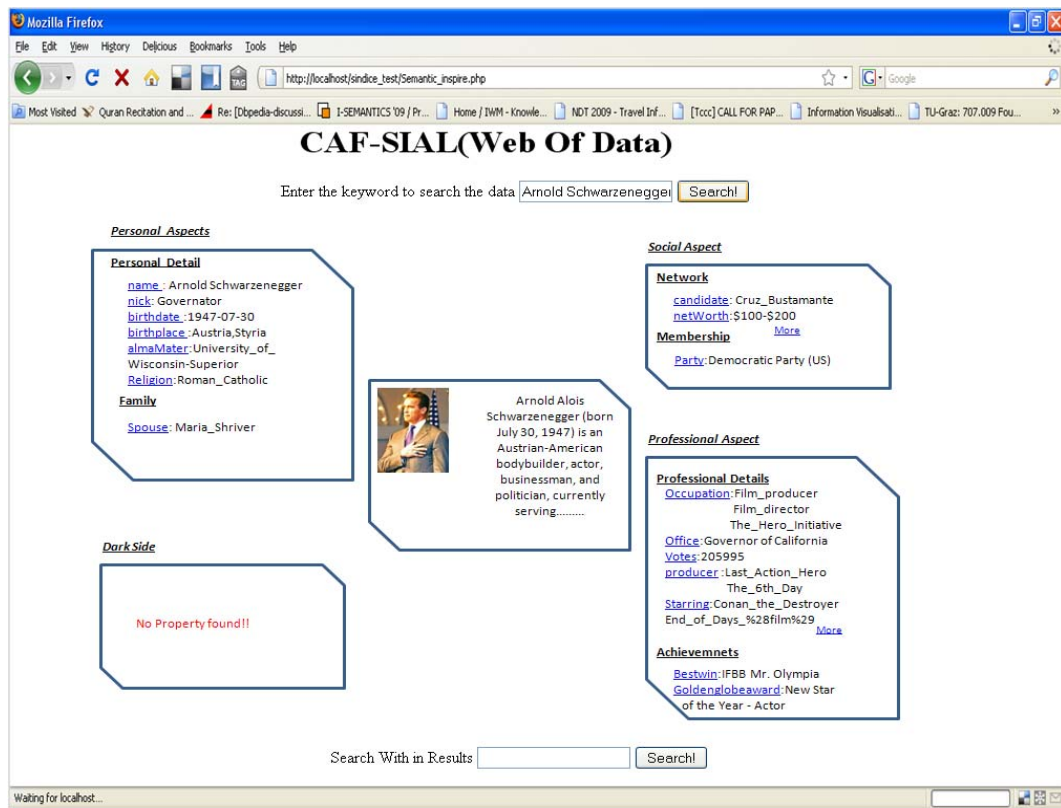


Figure 3. Screenshot

References

- [1] T. Berners-Lee, "Linked Data Design Issues"; (July 2006).
<http://www.w3.org/DesignIssues/LinkedData.html>.
- [2] A. Latif, P. Hoefler, A. Stocker, A. Saeed and C. Wagner, "The Linked Data Value Chain: A Lightweight Model for Business Engineers" Accepted for Proceedings of I-Semantic, 2009, Graz, Austria.
- [3] S. Chakrabarti, "Breaking Through the Syntax Barrier: Searching with Entities and Relations", In: Proc. PKDD'2004, Springer, Berlin Heidelberg, (2004), 9-16.
- [4] T. Berners-Lee, Y. Chen, L. Chilton, D. Connolly, R. Dhanaraj, J. Hollenbach, A. Lerer, and D. Sheets, "Tabulator: Exploring and Analyzing Linked Data on the Semantic Web." In: Proc. 3rd International Semantic Web User Interaction Workshop, 2006.
- [5] G. Kobilarov, and I. Dickinson.Humboldt, "Exploring Linked Data" In: Proc. Linked Data on the Web Workshop (LDOW2008), 2008.
- [6] "Best Practice Recipes for Publishing RDF Vocabularies", <http://www.w3.org/TR/swbp-vocab-pub/>
- [7] "Dereferencing a URI to RDF", <http://esw.w3.org/topic/DereferenceURI>
- [8] A. Russell, P. R. Smart, D. Braines and N. R. Shadbolt, "NITELIGHT: A Graphical Tool for Semantic Query Construction", In: Proc. Semantic Web User Interaction Workshop (SWUI 2008), Florence Italy, 5th April 2008.
- [9] C. Kiefer, A. Bernstein and M. Stocker, "The fundamentals of iSparql a virtual triple approach for similarity-based Semantic Web tasks ", In: Proc. ISWC. (2007), 2007.
- [10] F. C. Samur and Schwabe. Daniel, "Explorer: a tool for exploring RDF data through direct manipulation", In: Proc. Linked Data on the Web Workshop (LDOW2009), 2009.
- [11] T. Catarci, M. F. Levaldi and C. S. Batini, "Visual Query Systems for Databases: A Survey." *Journal of Visual Languages and Computing*, 8(2), 215-260, 1997.
- [12] M. Hildebrand, V. Ossenbruggen and J. Hardman. "Facet: A Browser for Heterogeneous Semantic Web Repositories." In: Proc. ISWC. (2006), 2006.
- [13] A. Harth, J. Umbrich, A. Hogan and S. Decker, "YARS2: A Federated Repository for Querying Graph Structured Data from the Web." In: Proc. ISWC. (2007), Springer, 2007.
- [14] F. M. Suchanek, G. Kasneci and G. Weikum, "Yago: A Core of Semantic Knowledge - Unifying WordNet and Wikipedia." In: Proc. 16th International World Wide Web Conference (WWW 2007), 2007.
- [15] "Google Top 10 choices for search results", <http://www.jimboykin.com/googles-top-10-choices-for-search-results/>
- [16] "Search ranking factors", <http://www.seomoz.org/article/search-ranking-factors>
- [17] S. Brin and L. Page, "The anatomy of a large-scale hypertextual Web search engine", *Computer Networks and ISDN Systems*. 30, 1998, pp. 107-117.
- [18] "ComScore Releases April 2009 U.S Search Engine Rankings", http://www.comscore.com/Press_Events/Press_Releases/2009/5/comScore_Releases_April_2009_U.S._Search_Engine_Rankings