SemFacet: Semantic Faceted Search over Yago*

Marcelo Arenas[†]
Dept. of Computer Science
PUC Chile

Bernardo Cuenca Grau[‡]
Dept. of Computer Science
University of Oxford

Evgeny Kharlamov[‡]
Dept. of Computer Science
University of Oxford

Šarūnas Marciuška[‡]
Dept. of Computer Science
University of Oxford

Dmitriy Zheleznyakov[‡]
Dept. of Computer Science
University of Oxford

Ernesto Jimenez-Ruiz[‡]
Dept. of Computer Science
University of Oxford

1. INTRODUCTION

Faceted search is a technique for accessing document collections that combines text search and *faceted navigation* applied to the documents' metadata. With the latter users can narrow down search results by incrementally applying multiple filters, called *facets* [18]. Though faceted search has become a mainstream commercial technology, traditional models impose severe constraints on the way faceted metadata is represented, and queries are formulated.

In particular, conventional faceted search models assume that documents are not "linked" to each other. E.g., in travel websites like *TripAdvisor.com*, one can do faceted search for hotels and restaurants, but there are no facets linking specific hotel and restaurant documents. Thus, values of facets for different kinds of documents cannot be joined in a single query. E.g., one cannot formulate a query (in *TripAdvisor*) that returns hotels with restaurants serving local food: even if we can query for hotels or restaurants independently, when we "switch view" from hotels to restaurants, the constraints imposed on hotels are lost. Data sparsity is another important issue in faceted search applications: document annotations are intrinsically incomplete, which leads to missing expected answers and facets. The meaning of queries in faceted search front-ends is also an issue: users are allowed to make a multiple choice within facets and the meaning of the corresponding queries is resolved in the back-end when queries are translated into operations over inverted indices. This is application dependent and not grounded on a formal query model that can be studied independently.

In this paper we demonstrate SemFacet, a proof-of-concept prototype implementing faceted search enhanced with Semantic Web technologies. RDF provides the required flexibility to semantically link documents in arbitrary ways, thus overcoming the limitations of conventional metadata models for faceted search. Furthermore, RDF is powerful enough to capture existing metadata models for faceted search. OWL 2 can be used to provide rich domain knowledge on top of faceted metadata. This can be exploited to capture complex dependencies between facets in a declarative and semantically unambiguous way and provide a schema-level solution to the data sparsity problem. Finally, faceted queries can be captured by SPARQL 1.1 [1], which provides a well-understood semantics and computational properties, as well as powerful application-independent infrastructure for query processing.

Our system SemFacet allows Web developers to automatically generate faceted query interfaces over any application, provided that its metadata and domain knowledge are given in RDF and OWL 2, respectively. For the initial keyword based search, SemFacet exploits Lucene [2]. Then, SemFacet implements algorithms for automatically generating facet names and their values from RDF and OWL documents, as well as for determining which facets are relevant at any point during faceted navigation. Since RDF triples can be exploited to link different types of documents, SemFacet also provides functionality to refocus the search from one type of document to another (e.g. from hotels to restaurants and vice-versa), without losing the constraints imposed thus far by means of facets. User choices in facet values are automatically translated into SPARQL 1.1 queries, which are then executed using the OWL 2 RL triple store RDFox [3].

Although our platform is generic, for this demo we have used the Yago [20] knowledge base extended with DBpedia [4] abstracts. SemFacet and further details are available online [16] and in [17].

2. SEMANTIC FACETED SEARCH

We now introduce the basics of semantic faceted search. First, we describe a metadata model that provides the required flexibility for documents to annotate other documents. It is described in an abstract way and not particularly tied to specific semantic technologies; however, it can be trivially realised using RDF. Then, we describe a query language that provides the necessary features to enable faceted navigation across different types of documents. This language is independent from existing query languages for semantic technologies, while it has a clean embedding into SPARQL 1.1.

Semantic Faceted Annotations. We next formalise an extension of conventional faceted data models where documents are annotated not only with facet values (typically strings), but also with other documents, which in turn can have their own annotations.

In the remainder of this section, we assume that V_N, V_L, and V_D are pairwise disjoint sets of (facet) names, labels, and documents, respectively, and that $V = V_N \cup V_L \cup V_D$. We will refer to elements of V_N as *facet-names* and of $V_L \cup V_D$ as *facet-values*.

DEFINITION 1. *The* semantic annotation *of a document $d \in V_D$ is a subset of $V_N \times (V_L \cup V_D)$. Moreover, d is said to be* n-annotated *with x, where $n \in V_N$ and $x \in V_L \cup V_D$, if (n, x) belongs to the semantic annotation of d.*

* The work was partially supported by the EPSRC projects Score and EXODA, B. Cuenca Grau was also supported by a Royal Society University Research Fellowship.

[†]Email: marenas@ing.puc.cl
[‡]Email: firstname.middlename.lastname@cs.ox.ac.uk

WWW'14 Companion, April 7–11, 2014, Seoul, Korea.
ACM 978-1-4503-2745-9/14/04.
http://dx.doi.org/10.1145/2567948.2577011.

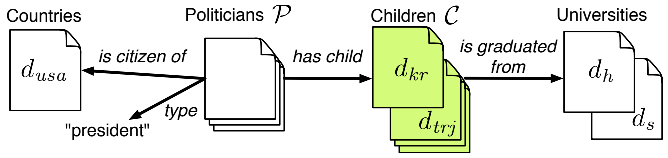


Figure 1: Example of semantic annotation

Note that in terms of this definition, the annotation of a document with conventional facets is simply a subset of $V_N \times V_L$; that is, documents cannot be used to annotate other documents.

EXAMPLE 2. *Assume that V_D includes (i) documents $\mathcal{P}$ about politicians; (ii) documents $\mathcal{C}$ about children of politicians, including a document d_{trj} about Theodore Roosevelt Jr. and d_{kr} about Kermit Roosevelt; (iii) a set of (documents about) countries, including d_{usa} about USA; and (iv) a set of (documents about) universities, including d_h for Harvard University and d_s for Stanford University. Moreover, assume that V_L includes the string* president. *As illustrated in Figure 1, the documents of $\mathcal{P}$ are* type-*annotated with values from V_L (one of which is* president*),* has_child-*annotated with document from $\mathcal{C}$, and* is_citizen_of-*annotated with documents about countries. Finally, the documents of $\mathcal{C}$ are* is_graduated_from-*annotated with university documents.*

Faceted Queries. A conventional faceted query can be seen as a filter over a set of documents: it specifies what facet names should be used to annotate a target document and what faceted values should be used for each of these names. The answer for such a query is the subset of the given documents satisfying the filter. In the semantic faceted search approach, we go further: a semantic faceted query can relate sets $S_1, \ldots, S_n$ of documents with annotations by describing relations between these sets and setting separate filters over each one of them. Analogously to conventional faceted queries, the effect of these filters is to select subsets $S'_1 \subseteq S_1, \ldots, S'_n \subseteq S_n$. In contrast to a conventional faceted query, we can now "navigate" between different types of documents and select what documents $S'_i \subseteq S_i$ are to be returned as query output.

In what follows, we formalise the notion of faceted query, starting with the notion of basic faceted query.

DEFINITION 3. *A* basic faceted query (BFQ) *over V is a pair $F = (X, \Gamma)$, where $X \in V_N$ and $\Gamma \subseteq V_L \cup V_D$.*

Intuitively, a document satisfies a BFQ (X, Γ) if it is X-annotated with *some* value in Γ. We illustrate BFQs on an example.

EXAMPLE 4. *The following are BFQs about politicians:*

$$Q^1_{\mathcal{P}} = (\mathsf{type}, \{\mathsf{president}\}), \quad Q^3_{\mathcal{P}} = (\mathsf{has\_child}, \{\}),$$
$$Q^2_{\mathcal{P}} = (\mathsf{is\_citizen\_of}, \{d_{usa}\}).$$

The query $Q^1_{\mathcal{P}}$ asks for (all the documents about) presidents, $Q^2_{\mathcal{P}}$ for politicians with American citizenship, $Q^3_{\mathcal{P}}$ for politicians with children. The empty condition $\{\}$ is used in $Q^2_{\mathcal{P}}$ since we impose no restrictions on the children. Moreover, the BFQ

$$Q_{\mathcal{C}} = (\mathsf{is\_graduated\_from}, \{d_h, d_s\}),$$

over the children of politicians in $\mathcal{C}$ asks for documents about children of politicians graduated from Harvard (d_h) or Stanford (d_s).

We now define how to combine BFQs in complex queries. Faceted search is typically initiated by a keyword search that returns the initial set of documents over which a user can set filters. To model keyword-based search, we introduce a special set `KW` disjoint from V, and assume that every element $S \in$ `KW` stands for a set of documents retrieved by a specific keyword-based search.

DEFINITION 5. *A (faceted)* query expression

$$\mathit{expr} ::= \mathit{start} \ | \ \mathit{start}/\mathit{rest}$$

over V is defined by the following grammar:

$$\begin{array}{llll} \mathit{start} ::= S \ | \ \star(S), & \mathit{rest} ::= \mathit{path} \ | \ (\mathit{path} \circ \mathit{path}), \\ \mathit{step} ::= F \ | \ \star(F), & \mathit{path} ::= \mathit{step} \ | \ \mathit{step}/\mathit{rest}, \end{array}$$

where $\circ \in \{\wedge, \vee\}$, $S \in$ `KW` and F is a BFQ over V. A (semantic) faceted query (FQ) over V *is a query expression over V where the symbol $\star$ occurs exactly once.*

The starting point (*start*) of a query expression *expr* captures the content of the initial search used in typical interfaces (prior to faceted navigation), which is given by a keyword search. In each step (*step*) of the remainder of the query expression (*rest*), a BFQ is applied to narrow down the search results. These steps are put together in paths (*path*), where each path is a sequence of BFQs that have to be applied according to the sequence order. A branching ($\mathit{path}_1 \wedge \mathit{path}_2$) can be applied in a query expression indicating that each of the search results has to satisfy the sequence of conditions specified by both path_1 and path_2. Finally, the symbol $\star$ in a query expression indicates which documents must be returned. In a faceted query this symbol is mentioned exactly once, as the result of such query is one set of documents. We illustrate the syntax and semantics of FQs in the following example.

EXAMPLE 6. *Consider the following information request about the scenario introduced in Examples 2 and 4:*

> *Find children of US presidents who graduated from Harvard University or Stanford University.*[1]

It corresponds to the following FQ:

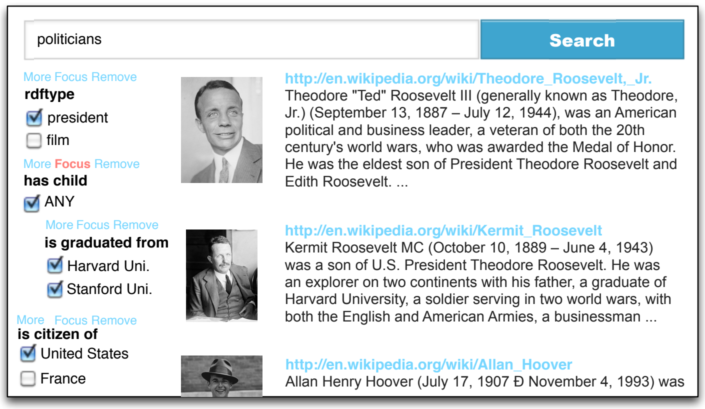
$$S_{\mathit{politician}}/(Q^1_{\mathcal{P}} \wedge Q^2_{\mathcal{P}} \wedge (\star(Q^3_{\mathcal{P}})/Q_{\mathcal{C}})), \tag{1}$$

Intuitively, the query first retrieves all documents of $S_{\mathit{politician}} \in$ `KW`, i.e., all documents returned by a keyword search with politicians. Then these results are filtered with $Q^1_{\mathcal{P}}$ and $Q^2_{\mathcal{P}}$, obtaining the set of documents about American presidents. Finally, this set is filtered by using $\star(Q^3_{\mathcal{P}})/Q_{\mathcal{C}}$, returning the set of documents about children of American presidents who graduated from Harvard or Stanford.

Semantic Faceted Search using RDF and SPARQL. Semantic annotations can be realised using different technologies. Our demo implementation exploits RDF, which we see as the most natural choice: URIs can be seen as documents of V_D, object and data properties as facet-names of V_N, literals as labels of V_L. Moreover, annotations are encoded in RDF as triples: each triple (s, p, o) corresponds to a p-annotation of a document s with a facet-value o. As we will see in Sections 3 and 4, RDF gives more than that: by combining RDF with RDFS and OWL 2, one can use reasoning to address natural limitations of current faceted search.

We now discuss how to embed our faceted query language in SPARQL 1.1, the standard language for querying RDF. One can provide semantics to our faceted query language by translation into first-order logic and show that that any FQ corresponds to a filter-free SPARQL 1.1 query Q that: *(i)* is positive (neither `OPTIONAL` nor `NOT EXISTS` nor `MINUS` construct occurs in Q); *(ii)* is tree-shaped (the dependency graph of Q's variables, where $?x$ and $?y$ are connected if there is a triple $?x\ Z\ ?y$ in Q, is tree-shaped); and

[1]A similar query was used to illustrate the Parallax system in [5].



rdftype
☑ president
☐ film

has child
☑ ANY

is graduated from
☑ Harvard Uni.
☑ Stanford Uni.

is citizen of
☑ United States
☐ France

http://en.wikipedia.org/wiki/Theodore_Roosevelt,_Jr.
Theodore "Ted" Roosevelt III (generally known as Theodore, Jr.) (September 13, 1887 – July 12, 1944), was an American political and business leader, a veteran of both the 20th century's world wars, who was awarded the Medal of Honor. He was the eldest son of President Theodore Roosevelt and Edith Roosevelt. ...

http://en.wikipedia.org/wiki/Kermit_Roosevelt
Kermit Roosevelt MC (October 10, 1889 – June 4, 1943) was a son of U.S. President Theodore Roosevelt. He was an explorer on two continents with his father, a graduate of Harvard University, a soldier serving in two world wars, with both the English and American Armies, a businessman ...

http://en.wikipedia.org/wiki/Allan_Hoover
Allan Henry Hoover (July 17, 1907 Ð November 4, 1993) was

Figure 2: Semantic Faceted Search Interface

(iii) satisfies the condition that different triples in Q cannot share more than one variable. Thus, FQs over annotated documents represented as an RDF graph can be translated in SPARQL 1.1 and evaluated using state of the art SPARQL 1.1 engines. Moreover, as we discuss in Sections 3 and 4, the restricted shape of the generated SPARQL 1.1 queries allows to do reasoning about queries w.r.t. ontologies efficiently.

EXAMPLE 7. *A SPARQL 1.1 query, where* wiki *is a name-space for Wikipedia, corresponding to Equation* (1) *is:*

```
SELECT ?z
 WHERE {?x rdf:type "president"@en ;
        ?x is_citizen_of wiki:United_States ;
        ?x has_child ?z ;
        {{?z is_graduated_from wiki:Harvard_Uni ;} UNION
         {?z is_graduated_from wiki:Stanford_Uni ;} } }
```

3. FRONT-END

We next discuss facets query formulation interfaces and how we implemented one in the SemFacet system.

3.1 Semantic Faceted Search Interface

In classical faceted search interfaces, users are presented with facet-values organised in groups according to facet-names. These groups are typically referred to as *facets*, so we will follow this terminology. Facets can be used to refine the search by selecting values; then value choices are converted into queries over the set of underlying documents. Answers to queries are typically displayed as snippets, combining images, textual descriptions, links etc.

Facets in SemFacet. Our implementation relies on conventional facets to support BFQs; complex queries are formed by *nesting* facets in a hierarchical fashion (see Figure 2 for a screenshot of SemFacet's interface).

EXAMPLE 8. *Figure 2 shows how the query in Equation* (1) *can be composed using the facets computed by* SemFacet*. In the screenshot there are 4 facets with facet-names: (i)* rdftype *where* "president" *is selected, (ii)* has_child *where* "ANY" *is selected (it is a special facet-value meaning "the value is not important" and corresponding to the empty condition* {} *in Example 4), (iii)* is_graduated_from*, where both* "Harvard_Uni" *and* "Stanford_Uni" *are selected, and (iv)* is_citizen_of*, where* "United_States" *is selected.*

Query Construction in SemFacet. Our interface supports a fragment of the query language from Definition 5, where we slightly restrict the usage of $(path \circ path)$ construct for the sake of usability of the query formulation interface.

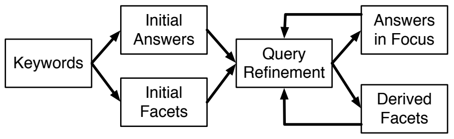


Figure 3: workflow of semantic faceted search

EXAMPLE 9. *Figure 2 captures the FQ in Equation* (1)*. Indeed, Harvard and Stanford are values within the facet* is_graduated_from*, which corresponds to the BDFQ* $Q_{\mathcal{C}}$*; in contrast, the choices of* "president" *in* rdftype *and* United_States *in* is_citizen_of *correspond to the conjunction* $Q^1_{\mathcal{P}} \wedge Q^2_{\mathcal{P}}$.

Our interface also supports the $\star()$ operator, which can be used to select a kind of output documents in faceted queries. To this end, we allow users to *focus* on facets: if the focus is set on a facet, then SemFacet outputs documents corresponding the facet-values chosen by the user in this facet.

EXAMPLE 10. *In Figure 2 the focus of the query is set on a facet with the name* has_child*; thus, the system's output consists only of documents in* $\mathcal{C}$*. More precisely, the system outputs all children of US presidents who graduated from Harvard or Stanford. On the right hand side of Figure 2, we can see the answers to the query, namely Theodore and Kermit Roosevelt and Allan Hoover. Note that the interface also has* more *and* remove *buttons, where the former one allows to expand the list of possible facet-values, and the latter one allows to hide a facet when it is not needed.*

Workflow of SemFacet. The process of constructing an FQ in SemFacet is summarised in Figure 3. The first step is to provide a set of keywords (e.g., in Figure 2 the keyword "politicians" was used), which leads to a set of initial answers and initial facets. Query refinement is then an iterative process, where users can either choose available facet-values, or refocus the query to a different facet. In response the system updates the query answers as well as the facets available to continue query refinement.

3.2 Improving the Interface using Axioms

Deriving New Annotations. Data sparsity is a quite important challenge for faceted search interfaces. Document annotations are intrinsically incomplete, which leads to both missing expected answers and facets. This problem is addressed in SemFacet by exploiting OWL 2 axioms to enrich RDF data with implicit triples.

EXAMPLE 11. *The following OWL 2 axiom and the triple*

$$\textit{SubClassOf}(\mathsf{alumHarvard}\ \textit{ObjectHasValue}(\mathsf{is\_graduated\_from}\ \mathsf{Harvard\_Uni})),$$

and (Kermit_Roosevelt rdftype alumHarvard) *imply an implicit triple* (Kermit_Roosevelt is_graduated_from Harvard_Uni).

Another benefit of axioms is that they can naturally model *hierarchical* facets (e.g., documents about hotels can be annotated with facet-values *B&B*, or *hostel*, which are both kinds of *accommodation*). In traditional faceted search applications, each document needs to be annotated with all values in a path from the root of the hierarchy to the specific value of interest (e.g., when annotating a hotel document with *B&B*, one would also need to annotate with *accommodation*). Modelling such ISA relationships between facet-values as OWL 2 axioms has the advantage that only the most specific values are required in annotations since the remaining ones can be derived using a reasoner.

Showing Relevant Information. Another important challenge for faceted search interfaces is to avoid "dead ends" (i.e., facet-value selections that lead to queries with the empty answer). In conventional faceted search applications, the detection of such dead ends is data driven, in the sense that the interface does not display facet-values for which no document exists. Axioms provide an alternative, declarative, way to detect dead ends during faceted search. E.g., in the presence of the axiom

$$\mathit{DisjointClasses}(\mathsf{USPresident}\ \mathsf{FrenchPresident})$$

once the facet-value USPresident is selected, the interface should not display the value FrenchPresident. Axioms are particularly important when annotations are inconsistent (this can happen with automatically generated annotations). SemFacet uses both axiom and data-driven techniques to avoid dead ends during search.

4. BACK-END

SemFacet is available as a Web service [16] and runs on a machine with 1vCPU, 4Gb of memory, and 20Gb of disk space. SemFacet is implemented on top of a fragment of Yago [20] and DBpedia [4] abstracts; it contains around 15 million RDF triples extended with OWL 2 axioms. A general architecture of SemFacet is in Figure 4. SemFacet's back-end relies on RDFox [3] for storing RDF triples, performing reasoning, and answering queries. RDFox is a massively parallel in-memory RDF triple store; it implements reasoning for the OWL 2 RL profile. RDFox supports only a conjunctive fragment of SPARQL 1.1; thus, to answer faceted queries, we extended its query module with a support of `UNION`.

In short, the back-end's workflow is the following. First, DBpedia abstracts are loaded to Lucene and RDFox; Yago is loaded in RDFox only. Every initial user's input via keyword based search is executed over Lucene that returns initial relevant document IDs, and a *view of IDs* is created. Based on the view of IDs, the initial facets and answer snippets compound of abstracts, thumbnails, and links to Wikipedia are generated. Then, the user performs query refinement and refocusing as described in Section 3 in the workflow of SemFacet. Faceted queries Q are translated into SPARQL 1.1 and executed by RDFox and the projections of results on the Q's first variable are intersected with the view of IDs. We rely on Lucene's ranking function to display answers under the default query focus; if the user refocuses the query, then we display answers in the order they are returned by RDFox. Note that SemFacet is implemented in such a way that both Lucene and RDFox can be substituted with any other software that provide the same functionality.

5. DEMO SCENARIO

During the demo we will show how one can find relevant information available in Wikipedia using semantic faceted search. The search will be performed over Yago using SemFacet system. The users will see several search scenarios that are hard to accomplish using conventional search engines, and we will show how they can be handle using SemFacet. Moreover, users will be invited to try to express their own information needs over SemFacet.

6. RELATED WORK

RDF has been proposed by many as a promising metadata model for faceted search [5, 6, 7, 18, 19] and several faceted search systems based on RDF have been developed [5, 8, 19]. Prominent examples are Parallax [5] and Humboldt [19]. Parallax provides faceted navigation on top of Freebase [9] and it supports a limited form of refocusing (it is possible to switch view between different sets of entities while keeping faceted constraints; however, when switching back to entities visited before, some constraints are lost); it also does not support branching in queries. Humboldt [19] provides similar functionality to Parallax. Other RDF-based systems provide more limited functionality and in particular they do not support refocusing; these include BrowseRDF [6], Tabulator [7], parallel faceted browser [8], mSpace [10], /facet [11], Piggy Bank [12], FacetedDBLP [13], a faceted DBpedia browser [14], Ontograator [15], and many others. Query model underlying all these systems is unclear; furthermore, their focus is on RDF while OWL 2 reasoning plays little or no role.

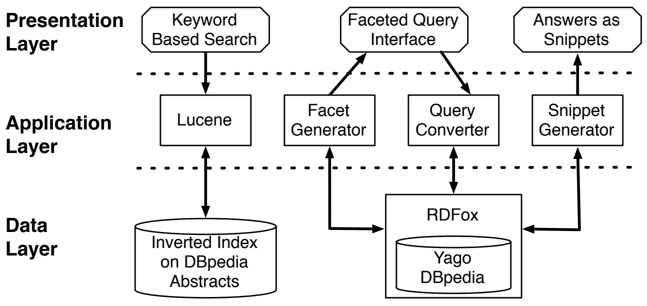


Figure 4: General architecture of SemFacet

7. CONCLUSION

We demonstrated semantic faceted search over Yago. Our approach is flexible and versatile: the same back-end implementation can be used to power faceted search over any application based on RDF and OWL 2. Our system is still an early prototype. Further work includes development of ranking functions for both facets and answers, extension of the interface to support wider fragments of our query language, experiments with larger data sets, and work on improving systems scalability and concurrency of users' access.

8. REFERENCES

[1] http://www.w3.org/TR/sparql11-query/.
[2] http://lucene.apache.org/.
[3] http://www.cs.ox.ac.uk/isg/tools/RDFox/.
[4] http://dbpedia.org/.
[5] http://sparallax.deri.ie/.
[6] https://launchpad.net/browserdf.
[7] http://www.w3.org/2005/ajar/tab.
[8] http://eventmap-ui.appspot.com/.
[9] http://www.freebase.com/.
[10] http://mspace.fm/.
[11] http://slashfacet.semanticweb.org/.
[12] http://simile.mit.edu/wiki/Piggy_Bank.
[13] http://dblp.l3s.de/.
[14] http://dbpedia.org/FacetedSearch.
[15] http://www.ontograator.org/.
[16] SemFacet: Semantic Faceted Search Project. http://www.cs.ox.ac.uk/isg/projects/SemFacet/.
[17] M. Arenas, B. Cuenca Grau, E. Kharlamov, Š. Marciuška, and D.Zheleznyakov. Towards semantic faceted search. In *WWW (Companion Volume)*, 2013.
[18] D.Tunkelang. *Faceted Search*. Morgan & Claypool Pubs.'09.
[19] G. Kobilarov and I. Dickinson. Humboldt: Exploring linked data. In *LDOW'08*.
[20] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A Core of Semantic Knowledge. In *WWW 2007*.