SPedia: A Semantics Based Repository of Scientific Publications Data

Muhammad Ahtisham Aslam(✉) and Naif Radi Aljohani

Faculty of Computing and Information Technology,
King Abdulaziz University, Jeddah, Saudi Arabia
{maaslam,nraljohani}@kau.edu.sa

Abstract. There is a noticeable increase in the number of scientific publications. These publications are being published by different publishers. *Springer* is one of those publishers which has published more than nine million scientific documents. *SpringerLink* is the portal providing the gateway to searching and accessing these published scientific documents. The structure, as well as the way, the contents are presented on the portal, provides valuable information about documents metadata such as author, ISBN, references, articles, chapters. However, this metadata is understandable by human in such a way that it facilitates the keyword-based searches through *SpringerLink* portal. At the same time this huge data about scientific documents is in silence as it is neither open nor linked to other datasets. To address these issues, we have created a semantics based repository called *SPedia* which consists of semantically enriched data about documents published by *Springer*. Currently, *SPedia* datasets consist of more than 300 million RDF triples. In this paper we describe *SPedia* and examine the quality of its extracted data by performing semantically enriched queries. The results show that *SPedia* facilitates the users to put sophisticated queries by employing semantic Web techniques instead of keyword-based searches. In addition, *SPedia* datasets can be utilized to link to other datasets available in the Linked Open Data (LOD) cloud.

1 Introduction

With the current deluge of data, we live in the fourth paradigm of scientific research which called data intensive research. The data is available in high volume about different domains such as educational data, government data, geo data etc. *Semantic Web* and *Linked Open Data (LOD)* communities have been working for more than a decade to produce semantically enriched data that is *open* as well as *linked* (i.e. *Linked Open Data (LOD)*). The purpose is to bring data on a global scale (which is currently dispersed, accessible on limited scale and available in data chunks) and to interlink different datasets so that semantically enriched queries could be made for sophisticated query answering purposes.

Many approaches have been proposed so far to extract and produce the structured and semantically enriched data from different existing data resources.

B. Cui et al. (Eds.): WAIM 2016, Part I, LNCS 9658, pp. 479–490, 2016.
DOI: 10.1007/978-3-319-39937-9_37

For example [2,3,6,14] focuses on extracting structured knowledge from Wikipedia contents, such that Semantic Web techniques can be used to ask sophisticated queries against Wikipedia. Similarly in [7,15], authors propose different methodologies to extract structured information from different unstructured data sources belonging to different domains. Another approach to extract semantically enriched data from three most popular Chinese encyclopedia sites and to link extracted data entities as Chinese LOD (*CLOD*) has been describe in [8]. In SwetoDblp project [1], authors propose a framework to create dataset in RDF form from XML document containing DBLP information.

When it comes to scientific publications such as journals, articles, books, chapters etc., there are many publishers (e.g. *Springer, IEEE, Amazon, Pearson, Thomson Reuters* etc.) that are publishing scientific documents. *Springer* is one of the leading global scientific publisher and *SpringerLink* is the worlds most comprehensive online collection of scientific documents [12]. In this paper we focus on extracting structured and semantically enriched datasets from *SpringerLink* so that semantically enriched queries can be put to huge datasets of scientific documents published by *Springer*. A very little initial work has been completed by way of extracting *LOD* only from *Springer* conferences on computer science [11]. This provides information about conferences\proceedings in the computer science domain only, leaving untouched a huge amount of remaining data. Our extracted datasets offer far more coverage of the disciplines and documents data.

We have extracted the structured data from the *SpringerLink* (as source of data) and produced a huge repository (i.e. *SPedia*) of semantically enriched data of scientific documents published by *Springer*. *SPedia* datasets consist of more than 300 million RDF triples and are available to download and experiment with at local level (in *N-Triple* format)[1]. We can use these datasets to formulate sophisticated queries about the documents published by *Springer* rather than relying on keyword-based searches through the portal. We ran SPARQL queries, visualizing document information and browsing the knowledge by using *Semantic Web* browsers (e.g. *Gruff* [5]). We also created a *SPARQL Endpoint* to pose semantically enriched queries from the extracted datasets.

The rest of the paper is organized as follows: Related work is discussed in Sect. 2. In Sect. 3 we describe extraction of structured information from *SpringerLink*. We next briefly describe the resulting datasets in Sect. 4, while Sect. 5 describes different ways to browse and query resulting datasets. Finally we conclude and discuss the future potential of our work in Sect. 6.

2 Related Work

To date, many approaches have been adopted to extract structured as well as semantically enriched data from existing sources so that users can pose semantically enriched queries. For example, in [1], the authors presented an approach

[1] http://wo.kau.edu.sa/Pages-SPedia.aspx.

to extract and produce an RDF version of the publications and author information contained in DBLP. The authors also describe their approach about crawling, parsing, and extracting structured data from source DBLP XML file. The main limitation of both source and resulting RDF datasets is that they provide only bibliographic information about documents without establishing relations between them.

The DBPedia project focused on the task of converting Wikipedia content into structured knowledge, so that Semantic Web techniques can be employed to pose sophisticated queries from Wikipedia [2,3,6]. Another project (i.e. YAGO [14]) also extracted structured information from Wikipedia. However, in [14], the authors proposed the extraction of data of 14 relationship types, ignoring extraction from Wikipedia infobox templates, in contrast to the DBPedia data extraction process. Continuing towards Wikipedia as a source, an entity based extraction approach has also been presented in [4] to extract entity based structured data from Wikipedia.

In [10], the authors proposed an approach for extraction of structured data from *The Cancer Genome Atlas (TCGA)* [15], organized as text archives in a set of directories. The approach made it easier for domain experts and bioinformatics applications to process and analyze cancer-related data by RDFizing the data. The resulting structured data is available to the public for remote querying and analysis of the structured domain knowledge.

An approach to transforming the data of the *Financial Transparency System (FTS)* of the European Commission is described in [7]. The authors give examples of queries to the resulting datasets not possible using existing HTML and XML formats of the data.

An effort to produce large-scale Chinese semantic data and to link these together as *Chinese LOD (CLOD)* is presented in [8]. In this work the authors proposed a method and strategy to extract structured data from the three largest Chinese encyclopedia sites and then link them together. They also established SPARQL Endpoint to query the extracted datasets using semantic web-based techniques.

In a similar fashion, *Springer Developer APIs* [13] provide sets of APIs to access programmatically the metadata of articles published by *Springer*. Some major limitations of these APIs are: Firstly, they provide the results in formats (e.g., mostly XML or JSON) that need to be parsed, to extract the required information, and RDFized. Secondly, they work over the traditional HTTP Get request/response method, making performance almost the same as by accessing the content online through the portal. Thirdly, these APIs provide access to 7 million articles, resulting in the straight forward absence of metadata from over 2 million published articles. Fourthly, by using these APIs structural information defining the relationships between data entities might be missed that could be detected if we access the contents through the *SpringerLink* portal. Our approach made full use of structural as well metadata content on the *SpringerLink* portal to extract the maximum number of data items and the relationships between them.

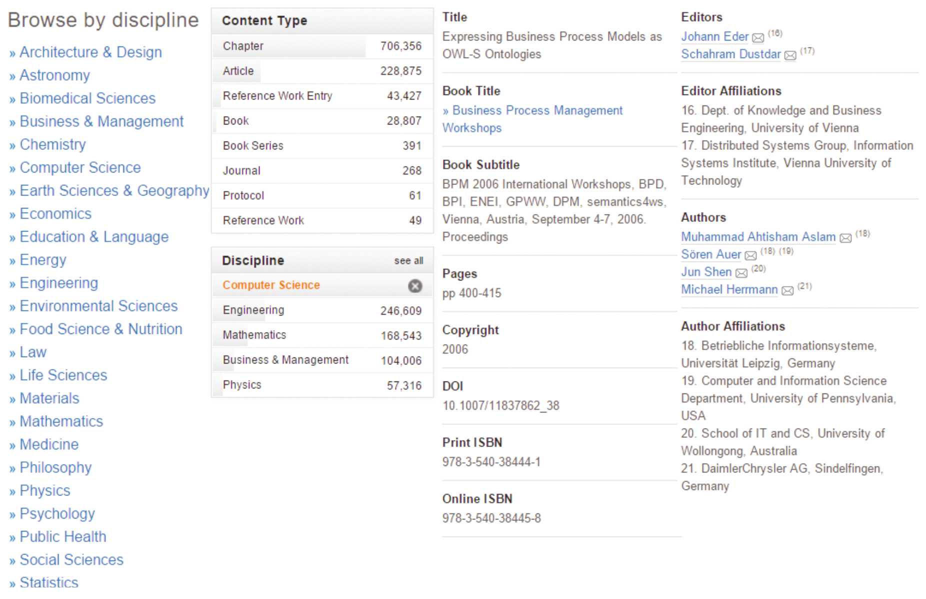
3 Extraction of Structured Information from SpringerLink

SpringerLink is the portal that we used as our source of data, so in this section we first present the content and structure of its existing data about scientific documents. We next present the *SPedia* knowledge extraction process we developed to crawl, parse, and extract the required data and to produce semantic data in RDF triples format.

3.1 SpringerLink Templates

The data source (i.e., *SpringerLink*) used in our approach contains information about the scientific documents in typical templates. These have the metadata as well as the relational information between the various documents (as shown in Fig. 1). All this information is available in HTML pages, so these were processed to extract the required structured and relational information. We extracted documents from 24 disciplines, as explained below:

Type and Discipline. As discussed above, in the data source are various types of documents (e.g., book, chapter, journal, article, etc.) and 24 disciplines (e.g., computer science, engineering, and medicine) to which a document may belong. Thus, both document type and discipline are represented by *rdf:type*.



Browse by discipline
- » Architecture & Design
- » Astronomy
- » Biomedical Sciences
- » Business & Management
- » Chemistry
- » Computer Science
- » Earth Sciences & Geography
- » Economics
- » Education & Language
- » Energy
- » Engineering
- » Environmental Sciences
- » Food Science & Nutrition
- » Law
- » Life Sciences
- » Materials
- » Mathematics
- » Medicine
- » Philosophy
- » Physics
- » Psychology
- » Public Health
- » Social Sciences
- » Statistics

Content Type	
Chapter	706,356
Article	228,875
Reference Work Entry	43,427
Book	28,807
Book Series	391
Journal	268
Protocol	61
Reference Work	49

Discipline	see all
Computer Science	
Engineering	246,609
Mathematics	168,543
Business & Management	104,006
Physics	57,316

Title
Expressing Business Process Models as OWL-S Ontologies

Book Title
» Business Process Management Workshops

Book Subtitle
BPM 2006 International Workshops, BPD, BPI, ENEI, GPWW, DPM, semantics4ws, Vienna, Austria, September 4-7, 2006. Proceedings

Pages
pp 400-415

Copyright
2006

DOI
10.1007/11837862_38

Print ISBN
978-3-540-38444-1

Online ISBN
978-3-540-38445-8

Editors
Johann Eder (16)
Schahram Dustdar (17)

Editor Affiliations
16. Dept. of Knowledge and Business Engineering, University of Vienna
17. Distributed Systems Group, Information Systems Institute, Vienna University of Technology

Authors
Muhammad Ahtisham Aslam (18)
Sören Auer (18) (19)
Jun Shen (20)
Michael Herrmann (21)

Author Affiliations
18. Betriebliche Informationsysteme, Universität Leipzig, Germany
19. Computer and Information Science Department, University of Pennsylvania, USA
20. School of IT and CS, University of Wollongong, Australia
21. DaimlerChrysler AG, Sindelfingen, Germany

Fig. 1. Different views (combined together) of *SpringerLink* template to present relational as well as metadata information of documents.

Document Metadata. Metadata properties (e.g., title, abstract, copyright, doi, print ISBN, online ISBN, print ISSN, online ISSN, publisher, volume, issue, coverage, pdf link, references, pages, publication year) about documents are presented by using properties *publication:has_Title, has_Abstract, has_Copyrigth, has_DOI, has_Print_ISBN, has_Online_ISBN, has_print_ISSN, has_Online_ISSN, has_Publisher, has_Volume, has_Issue, has_Coverage, has_PDF_Link, has_Reference, has_Pages, has_Publication_Year* respectively.

Authors, Editors and Affiliation. Every document has one or more author/s, editor/s, editor-in-chief and affiliation/s. Therefore, author/s, editor/s editor-in-chief and their affiliation/s data were extracted and presented by *publication:has_Author, has_Editor, has_Editor_In_Chief and has_Affiliation* properties.

Relations Between Documents. Documents have relationships with documents that are very important, especially when we work in domain of scientific documents. For example, a *chapter* has relation to a *book*, an *article* has relation to a *Journal* and a *reference work entry* has relation to a published *reference work* and vice versa. These relations are described by using *publication:has_Book_Chapter, is_Book_Chapter_Of, has_Article, is_Article_Of, has_Reference_Work_Entry, is_Reference_Work_Entry_Of* properties respectively.

3.2 Extraction Process

The extraction process consisted of four main steps, each of which is interlinked in a recursive manner (as shown in Fig. 2). The Fig. 2 shows that the information extraction process starts by crawling through *SpringerLink* for a particular discipline, then crawling and extracting information for a particular content type, and then for sub-content types, using a recursive approach. Here we describe each step of extraction process:

SPedia Crawler. Information about documents on *SpringerLink* is available in such a way that crawlers can be written to start crawling through documents

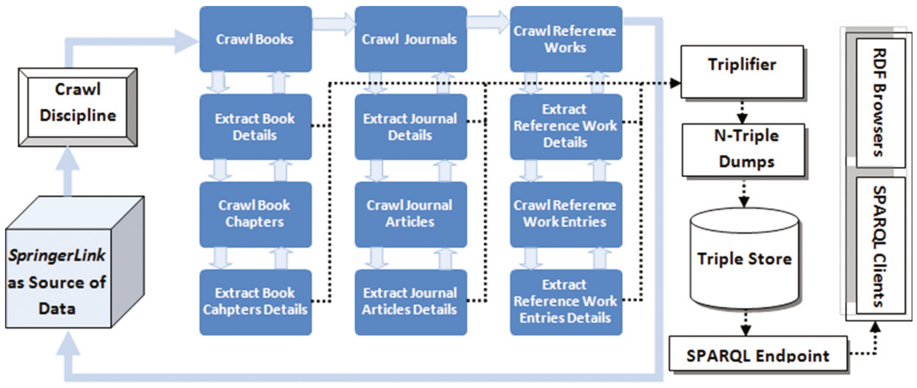


Fig. 2. Overview of the *SPedia* knowledge extraction process.

of any discipline from the first document to the last. Therefore, this module starts crawling through the *SpringerLink* portal from the first discipline and continue through every discipline by using a recursive approach (as shown in Fig. 2). In fact, crawling through a particular discipline is the starting point for further crawling through the various types of document in a particular discipline (e.g., book, journal, reference work, etc.). Also, the relations between disciplines and documents are saved during the crawling process for use in describing the relational properties (e.g., "Book" publisher:has_Book_Chapter "Book_Chapter" etc.) in later stages.

Information Parser and Extractor. The information parsing and extraction takes place at every stage of the crawling process, as well as during particular stages at which metadata of documents or relational information become available. For example, when a stage comes during the extraction process where a document (e.g., Book) exists, the information parser and extractor module gets activated. It parses the existing page and extracts the relational information (e.g., books relation to a particular discipline, or book chapters relation to a particular book, etc.) as well as metadata information (e.g., a book ISBN, pdf link, etc.). As soon as the information parsing and extraction step is complete, the process lets the crawler go on to the next item so information from every last document and its related information in the whole chain is extracted and utilized.

Triplifier. The information and metadata of documents extracted at any stage during the crawling process is triplified and saved to the data model. Every document (i.e. book, chapter, article etc.) is triplified as a resource, and every resource is triplified for their properties (e.g., title, abstract, ISBN, etc.). Relations (e.g., book *has_book_chapter*, journal *has_journal_article* etc.) are also triplified a this stage. Authors, editors, and affiliated organizations are identified as individual resources (resulting in large dataset of scientific authors and organizations). The triplifying process, together with the information parser and extraction process, also takes care of the data types of values of extracted properties.

Datasets Generator. This is the last step of the information extraction process which generates *SPedia* datasets. Data models of the information in triplicate are taken as inputs at this stage, inconsistencies in URIs are resolved and the final datasets in N-Triple format are generated. All datasets are generated at two different levels (e.g., property level, document level (further explained in Sect. 4.1)).

4 Extracted Dataset and Necessary Details

The *SPedia* datasets provide information on more than 9 million resources, including over 3.9 million journal articles, 3.1 million chapters, and 0.65 million reference work entries. Extracted datasets also provide information about more than 3100 journals, 181,000 books, and 650 reference works. Information

Table 1. *SPedia* datasets statistics.

Resource	Description	Numbers	Triples
Book	Provides information about book metadata as well as chapters of the book	181K	5.3M
Chapter	Provides information about book chapter metadata and the book in which chapter is published	3.1M	103M
Journal	Contains information about journal, its metadata as well as its sub resources i.e. articles published in a journal	3K	4M
Article	Provides information about article, the parent journal and all related metadata	3.9M	126M
Reference work	Resources of the type reference work provide information about reference works, their metadata and corresponding entries	858	0.7M
Reference work Entry	Contains information about metadata of reference work entries and their parent reference works	0.65M	11.9M
Person	Persons contains information about people as author, editor, editor-in-chief etc. with their contact information	31M	104M
Organization	Provides information about names of organization as affiliation of persons	12M	24M

about other resources, including 31 million people (e.g., authors, editors, editors-in-chief, etc.) and 12 million organizations is part of the extracted datasets. The resulting datasets consist of more than 300 million triples, including (approximately) 126 million triples on journal articles, 103 million triples on book chapters, and 11.9 million triples on reference work entries. A summary of the statistics on resources and extracted triples is provided in Table 1.

4.1 Different Levels of Extracted Datasets

SPedia datasets are available for download (in *.nt* format) at two different levels. Users can download these datasets to their requirements, starting with property-level datasets (smaller in size, but distributed in large number of property level .nt files) through to discipline level datasets (bigger in size, but with all data about documents relating to a single discipline in one .nt file). These different levels of data sets are further explained below:

I **Property Level.** Property-level datasets provide information about every property of documents in N-Triple format. For example, the *book type.nt* dataset contains information in (triple format) about the types of documents (e.g., a document type is *Book* and *Computer Science*). Similarly, the *book* *title.nt* dataset contains titles of all books published in a particular discipline. In the same way, other property level datasets (e.g., *journal hasPrint-ISBN.nt*, *article pdf link.nt* etc. provide data about the different properties of documents). Some common metadata properties of scientific documents is described in Table 2.

II **Document Type Level.** Datasets at document type level provide complete information about a particular type of document. For example, the type of document can be Book, Chapter, Article, etc. Therefore, document type-level datasets provide complete information about a particular document type belonging to a particular discipline in one dataset (e.g., the *book.nt* dataset provides all information about all books published in a particular discipline). Similarly, the *chapter.nt* dataset provides all information about all chapters published in particular discipline.

Table 2. Description of some common properties extracted for different document types.

Property	Description
type	Provides data about the document type (e.g. *type* can describe that a document is a *Chapter* in discipline *Computer Science*)
has_Title	Provides the title of the document
has_DOI	Provides the DOI of the document
has_Print_ISBN	Contains the print ISBN number of the document
has_Author	Contains the information about the authors of the document which then further contains his name, email and affiliation
has_Book_Chapter	Links every book with the every chapter publisher in that book
is_Book_Chapter_Of	This property links every chapter with the book in which chapter is published
has_Editor	Links every type of document (e.g. book, chapter, article etc.) with person (as editor)
is_Editor_Of	Links a person (as editor) with any type of document

4.2 Inconsistencies in Source Data

Since, the source of extracted datasets is the *SpringerLink* portal, many inconsistencies in the source data were noted during the extraction process. For example, sometimes information is not as per the template and heading of the topic e.g. the portal provides information about a single editor by using the heading Editor**s**. This heading leads the parsing and extraction process to extract more than one editor, although there is actually just one. Likewise, ISBNs are sometimes available in a number format and sometimes as a string (by adding the - character, which cannot be typecasted and is thus treated as an integer). Moreover, our approach uses the title of the document to create a URI for the extracted resource and sometimes this contains special characters (e.g., mathematical signs, language-based special characters, etc.) that cannot be used as part of the standard URI of a resource. In this case, such characters are treated as illegal characters and replaced by the most appropriate option, following best practice for publishing RDF vocabularies [9].

5 Accessing and Querying SPedia

SPedia datasets can be accessed from the project website and may also be queried from SPARQL Endpoint. Any third party application and semantic Web browser (e.g., Gruff [5]) can be used to query the datasets through SPARQL Endpoint, and to browse the datasets as well as to visualize the resulting data. We discuss below these scenarios of accessing and querying the resulting datasets.

5.1 Querying Through SPARQL Endpoint

The most common way to query the datasets is through SPARQL Endpoint. We have created a SPARQL Endpoint for *SPedia* datasets. This can be used to put sophisticated queries to the SPedia. Here, we give some sample queries and the results obtained from our extracted datasets.

We asked to search all articles in a journal that is published in the Mathematics discipline with the title OPSEARCH.

Example 1 (Find a Journal in Mathematics whose title is OPSEARCH and find all Articles published in that Journal.).

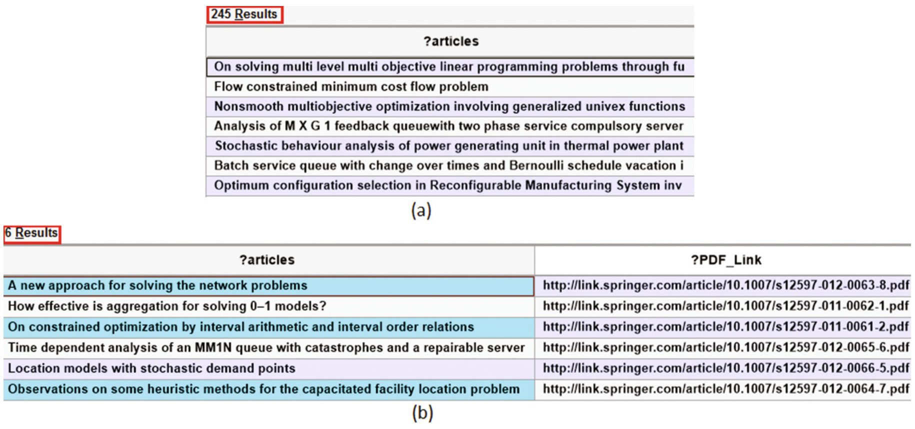
```
    PREFIX spedia:<http://www.kau.edu.sa/fcit/ontology/2015/3/v1.8# >
    select ?articles where {
        ?document rdf:type spedia:Journal.
        ?document rdf:type spedia:Mathematics.
        ?document spedia:has_Title"OPSEARCH"^^xsd:string.
        ?document spedia:has_Article ?articles.
}
```

The results that we obtained from this query are shown in the Fig. 3(a) indicating that the Journal with title *OPSEARCH* has 245 articles.

Then we further filter our results by querying for only those articles which are published in Volume 49, Issue 1 of the *OPSEARCH Journal*. We also queried for additional information (i.e. *PDF Link*) of these articles. Figure 3(b) shows six articles (with their *PDF Links*) published in Volume 49, Issue 1.

Example 2 (Find articles published in a particular volume 49 and issue 1 of the "OPSEARCH" Journal.).

```
    PREFIX spedia:<http://www.kau.edu.sa/fcit/ontology/2015/3/v1.8# >
    select ?articles ?PDF_Link where {
        ?document rdf:type spedia:Journal.
        ?document rdf:type spedia:Mathematics.
        ?document spedia:has_Title"OPSEARCH"^^xsd:string.
        ?document spedia:has_Article ?articles.
        ?articles spedia:is_In_Volume "49"^^xsd:string.
        ?articles spedia:is_In_Issue "1"^^xsd:string.
        ?articles spedia:has_PDF_Link ?PDF_Link.
}
```



245 Results

?articles
On solving multi level multi objective linear programming problems through fu
Flow constrained minimum cost flow problem
Nonsmooth multiobjective optimization involving generalized univex functions
Analysis of M X G 1 feedback queuewith two phase service compulsory server
Stochastic behaviour analysis of power generating unit in thermal power plant
Batch service queue with change over times and Bernoulli schedule vacation i
Optimum configuration selection in Reconfigurable Manufacturing System inv

(a)

6 Results

?articles	?PDF_Link
A new approach for solving the network problems	http://link.springer.com/article/10.1007/s12597-012-0063-8.pdf
How effective is aggregation for solving 0–1 models?	http://link.springer.com/article/10.1007/s12597-011-0062-1.pdf
On constrained optimization by interval arithmetic and interval order relations	http://link.springer.com/article/10.1007/s12597-011-0061-2.pdf
Time dependent analysis of an MM1N queue with catastrophes and a repairable server	http://link.springer.com/article/10.1007/s12597-012-0065-6.pdf
Location models with stochastic demand points	http://link.springer.com/article/10.1007/s12597-012-0066-5.pdf
Observations on some heuristic methods for the capacitated facility location problem	http://link.springer.com/article/10.1007/s12597-012-0064-7.pdf

(b)

Fig. 3. Results of example queries 1 and 2.

5.2 Browsing via Semantic Web Browsers

The semantic Web and LOD community has developed various tools and client applications that could be used to browse RDF data, either by loading the RDF triples directly or by connecting to SPARQL Endpoint. Following the same trend, *SPedia* datasets can be browsed by using the semantic Web browsers in tabular, as well as in visual form (as shown in Figs. 3 and 4). Figure 3 shows the results of our queries in a semantic Web browser (connected to SPARQL Endpoint) in tabular form. These can be further explored as long as linked RDF data is available or we reach a literal value. Also, as discussed before, some semantic Web browsers facilitate visualization of the datasets as well visual browsing. Figure 4 provides an example for some sample RDF records for a book and its related metadata.

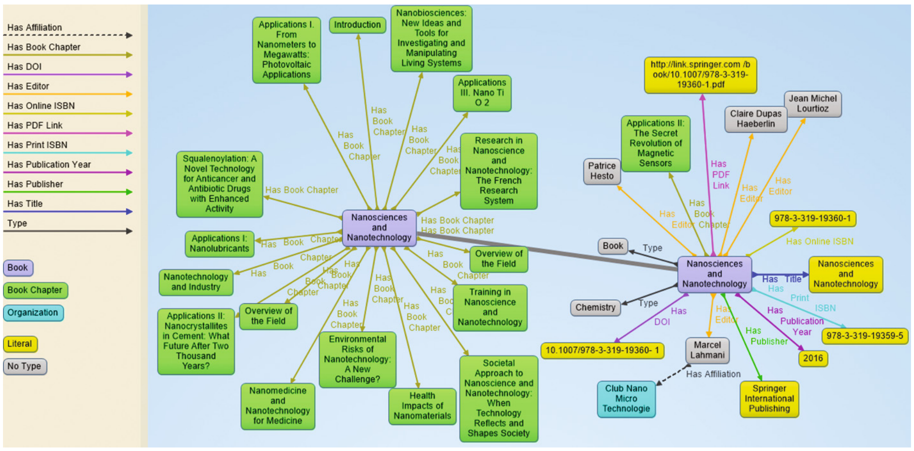


Fig. 4. Visual representation of extracted data.

6 Conclusion and Future Work

In this paper we presented *SPedia*: a semantics based repository of scientific publications data. *SPedia* aims to facilitate the semantic based search in order to allow users to ask sophisticated queries to overcome the limitation of the traditional keyword-based search. To achieve this goal, data about nine million scientific documents consisting of about 300 million RDF triples were extracted and produced as a machine-processable data. Furthermore, we created a SPARQL Endpoint to query *SPedia* datasets. The results showed that *SPedia* can play a key role in allowing users to put sophisticated queries by employing semantic Web techniques instead of keyword-based searches. In addition, *SPedia* datasets can be utilized to link to other datasets available in the Linked Open Data (LOD) cloud.

For the future work, we are aiming to extract semantically enriched structured data from publicly available resources of other publishers such as *Pearson, Elsevier, Amazon* and interlink them to explore the value of linked open scientific publications data. We are also working on increasing the coverage of the entities in *SPedia* datasets. Similarly, we are working to treat references as resources rather than strings, ultimately helping semantic Web-based agents to crawl through the interlinked data.

References

1. Aleman-Meza, B., Hakimpour, F., Budak Arpinar, I., Sheth, A.P.: Swetodblp ontology of computer science publications. Web Semant. Sci. Serv. Agents World Wide Web **5**(3), 151–155 (2007)
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., et al. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007)

3. Auer, S., Lehmann, J.: What have Innsbruck and Leipzig in common? Extracting semantics from wiki content. In: Franconi, E., Kifer, M., May, W. (eds.) ESWC 2007. LNCS, vol. 4519, pp. 503–517. Springer, Heidelberg (2007)
4. Exner, P., Nugues, P.: Entity extraction: from unstructured text to dbpedia RDF triples. In: Proceedings of the Web of Linked Entities Workshop in Conjuction with the 11th International Semantic Web Conference (ISWC 2012), pp. 58–69. CEUR-WS (2012)
5. Franz: Gruff: A grapher-based triple-store browser for allegrograph (2015)
6. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., Bizer, C.: DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. Semant. Web J. **6**(2), 167–195 (2015)
7. Martin, M., Stadler, C., Frischmuth, P., Lehmann, J.: Increasing the financial transparency of European commission project funding. Semant. Web J. **5**(2), 157–164 (2013). Special Call for Linked Dataset Descriptions
8. Niu, X., Sun, X., Wang, H., Rong, S., Qi, G., Yu, Y.: Zhishi.me - weaving Chinese linking open data. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part II. LNCS, vol. 7032, pp. 205–220. Springer, Heidelberg (2011)
9. Berrueta, D., Phipps, J.: Best practice recipes for publishing RDF vocabularies. w3c working group note (2008). http://www.w3.org/TR/2008/NOTE-swbp-vocab-pub-20080828
10. Saleem, M., Shanmukha, S., Ngonga, A.-C., Almeida, J.S., Decker, S., Deus, H.F.: Linked cancer genome atlas database. In: Proceedings of the 9th International Conference on Semantic Systems, I-SEMANTICS 2013, pp. 129–134. ACM, New York (2013)
11. Springer: Lod for conferences in computer science (2015). http://lod.springer.com/wiki/bin/view/Linked+Open+Data/About
12. Springer: Facts and figures. Springer Science+Business Media (2015). http://resource-cms.springer.com/springer-cms/rest/v1/content/20616/data/v11/Facts+and+Figures+April+2015
13. Springer: Springer—biomed central API portal (2015). https://dev.springer.com/
14. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: a core of semantic knowledge. In: Proceedings of WWW 2007, pp. 697–706 (2007)
15. Tomczak, P.C., Katarzyna, M.W.: The cancer genome atlas (TCGA): an immeasurable source of knowledge. Contemp. Oncol. **19**(1A), A68–A77 (2015)