

Robert Bembenik
Łukasz Skonieczny
Henryk Rybiński
Marek Niezgódka (Eds.)

Intelligent Tools for Building a Scientific Information Platform

Robert Bembenik, Łukasz Skonieczny, Henryk Rybiński, and Marek Niezgódka (Eds.)

Intelligent Tools for Building a Scientific Information Platform

Studies in Computational Intelligence, Volume 390

Editor-in-Chief

Prof. Janusz Kacprzyk
Systems Research Institute
Polish Academy of Sciences
ul. Newelska 6
01-447 Warsaw
Poland
E-mail: kacprzyk@ibspan.waw.pl

Further volumes of this series can be found on our homepage:
springer.com

Vol. 367. Gabriel Luque and Enrique Alba
Parallel Genetic Algorithms, 2011
ISBN 978-3-642-22083-8

Vol. 368. Roger Lee (Ed.)
Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing 2011, 2011
ISBN 978-3-642-22287-0

Vol. 369. Dominik Ryzko, Piotr Gawrysiak, Henryk Rybiński, and Marzena Kryszkiewicz (Eds.)
Emerging Intelligent Technologies in Industry, 2011
ISBN 978-3-642-22731-8

Vol. 370. Alexander Mehler, Kai-Uwe Kühnberger, Henning Lobin, Harald Lüngen, Angelika Storrer, and Andreas Witt (Eds.)
Modeling, Learning, and Processing of Text Technological Data Structures, 2011
ISBN 978-3-642-22612-0

Vol. 371. Leonid Perlovsky, Ross Deming, and Roman Ilin (Eds.)
Emotional Cognitive Neural Algorithms with Engineering Applications, 2011
ISBN 978-3-642-22829-2

Vol. 372. António E. Ruano and Annamária R. Várkonyi-Kóczy (Eds.)
New Advances in Intelligent Signal Processing, 2011
ISBN 978-3-642-11738-1

Vol. 373. Oleg Okun, Giorgio Valentini, and Matteo Re (Eds.)
Ensembles in Machine Learning Applications, 2011
ISBN 978-3-642-22909-1

Vol. 374. Dimitri Plemenos and Georgios Miaoulis (Eds.)
Intelligent Computer Graphics 2011, 2011
ISBN 978-3-642-22906-0

Vol. 375. Marenglen Biba and Fatos Xhafa (Eds.)
Learning Structure and Schemas from Documents, 2011
ISBN 978-3-642-22912-1

Vol. 376. Toyohide Watanabe and Lakhmi C. Jain (Eds.)
Innovations in Intelligent Machines – 2, 2011
ISBN 978-3-642-23189-6

Vol. 377. Roger Lee (Ed.)
Software Engineering Research, Management and Applications 2011, 2011
ISBN 978-3-642-23201-5

Vol. 378. János Fodor, Ryszard Klempous, and Carmen Paz Suárez Araujo (Eds.)
Recent Advances in Intelligent Engineering Systems, 2011
ISBN 978-3-642-23228-2

Vol. 379. Ferrante Neri, Carlos Cotta, and Pablo Moscato (Eds.)
Handbook of Memetic Algorithms, 2011
ISBN 978-3-642-23246-6

Vol. 380. Anthony Brabazon, Michael O'Neill, and Dietmar Maringer (Eds.)
Natural Computing in Computational Finance, 2011
ISBN 978-3-642-23335-7

Vol. 381. Radosław Katarzyniak, Tzu-Fu Chiu, Chao-Fu Hong, and Ngoc Thanh Nguyen (Eds.)
Semantic Methods for Knowledge Management and Communication, 2011
ISBN 978-3-642-23417-0

Vol. 382. F.M.T. Brazier, Kees Nieuwenhuis, Gregor Pavlin, Martijn Warnier, and Costin Badica (Eds.)
Intelligent Distributed Computing V, 2011
ISBN 978-3-642-24012-6

Vol. 383. Takayuki Ito, Minjie Zhang, Valentin Robu, Shaheen Fatima, and Tokuro Matsuo (Eds.)
New Trends in Agent-Based Complex Automated Negotiations, 2012
ISBN 978-3-642-24695-1

Vol. 384. Daphna Weinshall, Jörn Anemüller, and Luc van Gool (Eds.)
Detection and Identification of Rare Audiovisual Cues, 2012
ISBN 978-3-642-24033-1

Vol. 385. Alex Graves
Supervised Sequence Labelling with Recurrent Neural Networks, 2012
ISBN 978-3-642-24796-5

Vol. 386. Marek R. Ogiela and Lakhmi C. Jain (Eds.)
Computational Intelligence Paradigms in Advanced Pattern Classification, 2012
ISBN 978-3-642-24048-5

Vol. 387. David Alejandro Pelta, Natalio Krasnogor, Dan Dumitrescu, Camelia Chira, and Rodica Lung (Eds.)
Nature Inspired Cooperative Strategies for Optimization (NICSO 2011), 2011
ISBN 978-3-642-24093-5

Vol. 388. Tiansi Dong
Recognizing Variable Environments, 2012
ISBN 978-3-642-24057-7

Vol. 389. Patricia Melin
Modular Neural Networks and Type-2 Fuzzy Systems for Pattern Recognition, 2012
ISBN 978-3-642-24138-3

Vol. 390. Robert Bembeník, Łukasz Skonieczny, Henryk Rybiński, and Marek Niezgodka (Eds.)
Intelligent Tools for Building a Scientific Information Platform, 2012
ISBN 978-3-642-24808-5

Robert Bembenik, Łukasz Skonieczny,
Henryk Rybiński, and Marek Niezgódka (Eds.)

Intelligent Tools for Building a Scientific Information Platform

Editors

Dr. Robert Bembenik
Institute of Computer Science
Faculty of Electronics and
Information Technology
Warsaw University of Technology
Ul. Nowowiejska 15/19
00-665 Warsaw
Poland
E-mail: R.Bembenik@ii.pw.edu.pl

Dr. Łukasz Skonieczny
Institute of Computer Science
Faculty of Electronics and
Information Technology
Warsaw University of Technology
Ul. Nowowiejska 15/19
00-665 Warsaw
Poland
E-mail: L.Skonieczny@ii.pw.edu.pl

Prof. Henryk Rybiński
Institute of Computer Science
Faculty of Electronics and
Information Technology
Warsaw University of Technology
Ul. Nowowiejska 15/19
00-665 Warsaw
Poland
E-mail: H.Rybinski@ii.pw.edu.pl

Prof. Marek Niezgódka
Interdisciplinary Centre for Mathematical and
Computational Modelling (ICM)
University of Warsaw
Ul. Pawińskiego 5a
02-106 Warsaw
Poland
E-mail: M.Niezgodka@icm.edu.pl

ISBN 978-3-642-24808-5

e-ISBN 978-3-642-24809-2

DOI 10.1007/978-3-642-24809-2

Studies in Computational Intelligence

ISSN 1860-949X

Library of Congress Control Number: 2011939766

© 2012 Springer-Verlag Berlin Heidelberg

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilm or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

Typesetting: Scientific Publishing Services Pvt. Ltd., Chennai, India.

Printed on acid-free paper

9 8 7 6 5 4 3 2 1

springer.com

Preface

SYNAT is a program funded by the National Centre for Research and Development in Poland (NCBiR), with the main objective to set up an ICT platform for the national system of repositories that will cover content areas of science and humanities. A network of 16 academic partners have committed to implement SYNAT's concept in the form of a universal open knowledge infrastructure for the information society in Poland. Beyond the system development, a comprehensive portfolio of research problems is addressed by the network partners.

The program is scheduled for the period of three years, initiated in 2010. In view of the limited implementation time, the primary goal of the program consists in meeting the challenges of global digital information revolution viewed from the perspective of Poland. So not only the access to knowledge, at any scale, but also the forms of visibility gain for all categories of the publications and other forms documenting results of the research are foreseen.

Novel algorithms, their implementations and practical use are expected to result from SYNAT's activities. A broad range of new functionalities will get introduced and implemented upon validating their real practical features for scalability and interoperability. The result of a one year research comprises a number of works in the areas of, *inter alia*, artificial intelligence, knowledge discovery and data mining, information retrieval and natural language processing, addressing the problems of implementing intelligent tools for building a scientific information platform.

The idea of this book is based on the very successful SYNAT Project Conference (January, 2011) and the SYNAT Workshop accompanying the 19th International Symposium on Methodologies for Intelligent Systems (ISMIS 2011). During the Conference, research areas for the purpose of the platform have been outlined. The Workshop was a place of discussions on proposed solutions based on intelligent tools able to significantly improve the quality of the planned scientific information services.

The papers included in this volume cover the following topics: The SYNAT Project Concepts, Semantic Clustering, Ontology-based Systems, Multimedia Data Processing. We will now briefly summarize contents of the particular chapters.

Chapter I, “The SYNAT Project Concepts”, is dealing with issues and problems related to constructing a large IT software system capable of collecting, storing and searching information in an intelligent way.

- **Hung Son Nguyen, Dominik Ślęzak, Andrzej Skowron, and Jan G. Bazan** (“Semantic Search and Analytics over Large Repository of Scientific Articles”) present an architecture of the system aimed at searching and synthesizing information within document repositories originating from different sources, i.e., with documents provided not necessarily in the same format and the same level of detail. The system is expected to provide domain knowledge interfaces enabling the internally implemented algorithms to identify relationships between documents (as well as authors, institutions etc.), and concepts (such as, e.g., areas of science) extracted from various types of knowledge bases. The system should be scalable by means of scientific content storage, performance of analytic processes, and speed of search.
- **Linh Anh Nguyen and Hung Son Nguyen** (“On Designing the SONCA System”) present ideas and proposals for the SONCA system. The main idea is to allow combination of metadata-based search, syntactic keyword-based search and semantic search, and to use ranks of objects. Semantic search criteria may be keywords, concepts, or objects (for checking similarity). Search criteria based on metadata play the role of exact restrictions, while syntactic keywords and semantic search criteria are fuzzy restrictions. To enable metadata-based search, an appropriate document representation is used. To enable syntactic keyword-based search, each document (object) is stored together with information about the terms occurring in its text attributes. Authors provide an abstract model for the SONCA system, an instantiation of that model, some ideas for the user interface of SONCA as well as proposals for increasing efficiency of the query answering process.
- **Piotr Gawrysiak, Dominik Ryżko, Przemysław Więch, and Marek Kozłowski** (“Retrieval and Management of Scientific Information from Heterogeneous Sources”) describe the process of automated retrieval and management of scientific information from various sources, including the Internet. They describe application of semantic methods in different phases of the process. The system envisaged in the project is a scientific digital library, with automated retrieval and hosting capabilities. An overall architecture for the system is proposed.
- **Marcin Kowalski, Dominik Ślęzak, Krzysztof Stencel, Przemysław Pardel, Marek Grzegorowski, and Michał Kijowski** (“RDBMS Model for Scientific Articles Analytics”) present a relational database schema aimed at efficient storage and querying of parsed scientific articles, as well as entities referring to researchers, institutions, scientific areas, etc. An important

requirement in front of the proposed model is to operate with various types of entities, but with no increase of schema's complexity. Another aspect is to store detailed information about parsed articles in order to conduct advanced analytics in combination with the domain knowledge about scientific topics, by means of standard SQL and RDBMS management. The overall goal is to enable offline, possibly incremental computation of semantic indexes supporting end users via other modules, optimized for fast search and not necessarily for fast analytics.

Chapter II, “Semantic Clustering”, deals with semantic clustering of documents as well as problems of document representation and document representation formats facilitating such clustering.

- **Marcin Szczuka, Andrzej Janusz, and Kamil Herba** (“Semantic Clustering of Scientific Articles with Use of DBpedia Knowledge Base”) present a case study of semantic clustering of scientific articles related to Rough Sets. The proposed method groups the documents on the basis of their content and with assistance of DBpedia knowledge base. The text corpus is first treated with Natural Language Processing tools in order to produce vector representations of the content and then matched against a collection of concepts retrieved from DBpedia. As a result, a new representation is constructed that better reflects the semantics of the texts. With this new representation, the documents are hierarchically clustered in order to form partition of papers that share semantic relatedness. An assessment of clustering quality by human experts, compared to traditional approach, is presented.
- **S. Hoa Nguyen, Wojciech Świeboda, and Grzegorz Jaśkiewicz** (“Extended Document Representation for Search Result Clustering”) discuss a framework of document description extension which utilizes domain knowledge and semantic similarity. The idea is based on application of Tolerance Rough Set Model, semantic information extracted from a source text and domain ontology to approximate concepts associated with documents and to enrich the vector representation.
- **Paweł Betliński, Paweł Gora, Kamil Herba, Trung Tuan Nguyen, and Sebastian Stawicki** (“Semantic Recognition of Digital Documents”) describe document representation format together with a proof of concept of the system converting scientific articles in PDF format into this representation. Another topic presented in the article is an experiment with clustering documents by style.

Chapter III, “Ontology-based Systems”, is concerned primarily with semantics and ontologies. These notions are employed to the construction of semantically-driven knowledge bases.

- **Piotr Wasilewski** (“Towards Semantic Evaluation of Information Retrieval”) discusses fundamentals of semantic evaluation of information retrieval systems. Semantic evaluation is understood in two ways. Semantic evaluation *sensu stricto* consists of automatic global methods of information retrieval evaluation which are based on knowledge representation systems. Semantic

evaluation *sensu largo* includes also evaluation of retrieved results presented using new methods and comparing them to previously used ones. The paper focuses on semantic relevance of documents, both binary and graded, together with semantic ranking of documents. Various types of semantic value and semantic relevance are proposed and also some semantic versions of information retrieval evaluation measures are given.

- **Anna Wróblewska, Teresa Podsiadły-Marczykowska, Robert Bembenik, Grzegorz Protaziuk, and Henryk Rybiński** (“Methods and Tools for Ontology Building, Learning and Integration - Application in the SYNAT Project”) started building an experimental platform where different kinds of stored knowledge will be modeled with the use of ontologies, in particular a system ontology, accompanied by domain ontologies. The system ontology defines “system domain” (a kind of meta knowledge) for the scientific community, covering concepts and activities related to the scientific community. The paper makes a contribution to understanding semantically modeled knowledge and its incorporation into the SYNAT project. The authors present a review of ontology building, learning, and integration methods and their potential application in the project.
- **Cezary Mazurek, Krzysztof Sielski, Maciej Stroiński, Justyna Walowska, Marcin Werla, and Jan Węglarz** (“Transforming a Flat Metadata Schema to a Semantic Web Ontology: The Polish Digital Libraries Federation and CIDOC CRM Case Study”) describe the transformation of the metadata schema used by the Polish Digital Libraries Federation, to the CIDOC CRM model implemented in OWL as Erlangen CRM. In the paper the authors identify a number of problems that are common to all such transformations and propose solutions. They also present statistics concerning the mapping process and the resulting knowledge base.
- **Krzysztof Goczyla, Aleksander Waloszek, Wojciech Waloszek, and Teresa Zawadzka** (“Modularized Knowledge Bases Using Contexts, Conglomerates and a Query Language”) present a novel approach to design and development of a modularized knowledge base. The approach is oriented towards decomposition of a knowledge base into logical components, called contexts, and further, into semantic components called conglomerates. The paper shows how contexts and conglomerates concepts can work in harmony to create a maintainable knowledge base. A thorough discussion of related work is also given.

Chapter IV, “Multimedia Data Processing”, is devoted to the data mining approach for processing multimedia data like sound and images. Both hardware and software solutions are discussed.

- **Maciej Wielgosz, Ernest Jamro, Dominik Żurek, and Kazimierz Wiatr** (“FPGA Implementation of the Selected Parts of the Fast Image Segmentation”) present preliminary hardware implementation of a SVM (Support Vector Machine) algorithm (in FPGA). The work is primarily focused on the FPGA implementation aspects of the algorithm as well as on comparison of

the hardware and software performance. The approach provides a high performance of the hardware classification module.

- **Rafał Frączek and Bogusław Cyganek** (“Evaluation of Image Descriptors for Retrieval of Similar Images”) address the issue of searching for similar images in a repository. The contained images are annotated with help of the sparse descriptors. In the presented research different color and edge histogram descriptors were used. To measure distances among images the sets of their descriptors are compared. For this purpose different similarity measures were employed. The results of these experiments, as well as discussion of the advantages and limitations of different combinations of methods for retrieval of similar images are presented.
- **Andrzej Czyżewski, Adam Kupryjanow, and Bożena Kostek** (“Online Sound Restoration for Digital Library Applications”) discuss a sound restoration system conceived and engineered at the Multimedia Systems Department of the Gdansk University of Technology with regard to the principles of its design, features of operation and the achieved results. The system has been designed so that (1) no special sound restoration software is needed to perform audio restoration; (2) no skills in digital signal processing are required from the user.

Chapter V, “Intelligent Systems, Tools and Applications”, deals with the development and applications of various tools which might be used as a part of the scientific information platform.

- **Łukasz Brocki, Krzysztof Marasek, and Danijel Koržinek** (“Connectionist Language Model for Polish”) describe a connectionist language model, which may be used as an alternative to the well known n-gram models. A comparison experiment between n-gram and connectionist language models is performed on a Polish text corpus. Statistical language modeling is based on estimating a joint probability function of a sequence of words in a given language. This task is made problematic due to a phenomenon known commonly as the “curse of dimensionality”. In the presented experiments, perplexity is used as a measure of language model quality.
- **Henryk Krawczyk and Marek Downar** (“Commonly Accessible Web Service Platform - Wiki-WS”) present a SOA-enabled platform - Wiki-WS - that empowers users to deploy, modify, discover and invoke web services. The main concept of the Wiki-WS platform is searching and the invocation of web services written by different workgroups in different technologies deployed on different servers. Wiki-based web service code modification allows engineers from any place to construct and to implement components, which are mature and ready to use. There is also included presentation of sample scenarios of Wiki-WS usage and advantages derived from its deployment.

This book could not have been completed without the help of many people. We would like to thank all the authors for their contribution to the book and their effort in addressing reviewers' and editorial feedback. We would also like to thank reviewers and all program committee members of the SYNAT Workshop. Finally, we would like to thank Bożenna Skalska for her administrative work.

July 2011
Warszawa

Robert Bembenik
Łukasz Skonieczny
Henryk Rybiński
Marek Niezgódka

Contents

Chapter 1: The SYNAT Project Concepts

Semantic Search and Analytics over Large Repository of Scientific Articles	1
<i>Hung Son Nguyen, Dominik Ślęzak, Andrzej Skowron, Jan G. Bazan</i>	

On Designing the SONCA System	9
<i>Linh Anh Nguyen, Hung Son Nguyen</i>	

Retrieval and Management of Scientific Information from Heterogeneous Sources	37
<i>Piotr Gawrysiak, Dominik Ryżko, Przemysław Więch, Marek Kozłowski</i>	

RDBMS Model for Scientific Articles Analytics	49
<i>Marcin Kowalski, Dominik Ślęzak, Krzysztof Stencel, Przemysław Pardel, Marek Grzegorowski, Michał Kijowski</i>	

Chapter 2: Semantic Clustering

Semantic Clustering of Scientific Articles with Use of DBpedia Knowledge Base	61
<i>Marcin Szczuka, Andrzej Janusz, Kamil Herba</i>	

Extended Document Representation for Search Result Clustering	77
<i>S. Hoa Nguyen, Wojciech Świeboda, Grzegorz Jaśkiewicz</i>	

Semantic Recognition of Digital Documents	97
<i>Paweł Betliński, Paweł Gora, Kamil Herba, Trung Tuan Nguyen, Sebastian Stawicki</i>	

Chapter 3: Ontology-Based Systems

Towards Semantic Evaluation of Information Retrieval	107
<i>Piotr Wasilewski</i>	

Methods and Tools for Ontology Building, Learning and Integration – Application in the SYNAT Project	121
<i>Anna Wróblewska, Teresa Podsiadły-Marczykowska, Robert Bembeník, Grzegorz Protaziuk, Henryk Rybiński</i>	
Transforming a Flat Metadata Schema to a Semantic Web Ontology: The Polish Digital Libraries Federation and CIDOC CRM Case Study	153
<i>Cezary Mazurek, Krzysztof Sielski, Maciej Stroiński, Justyna Walkowska, Marcin Werla, Jan Węglarz</i>	
Modularized Knowledge Bases Using Contexts, Conglomerates and a Query Language	179
<i>Krzysztof Goczyła, Aleksander Waloszek, Wojciech Waloszek, Teresa Zawadzka</i>	
Chapter 4: Multimedia Data Processing	
FPGA Implementation of the Selected Parts of the Fast Image Segmentation	203
<i>Maciej Wielgosz, Ernest Jamro, Dominik Żurek, Kazimierz Wiatr</i>	
Evaluation of Image Descriptors for Retrieval of Similar Images	217
<i>Rafał Frączek, Bogusław Cyganek</i>	
Online Sound Restoration for Digital Library Applications	227
<i>Andrzej Czyżewski, Adam Kupryjanow, Bożena Kostek</i>	
Chapter 5: Intelligent Systems, Tools and Applications	
Connectionist Language Model for Polish	243
<i>Lukasz Brocki, Krzysztof Marasek, Danijel Koržinek</i>	
Commonly Accessible Web Service Platform — Wiki-WS	251
<i>Henryk Krawczyk, Marek Downar</i>	
Author Index	265

Semantic Search and Analytics over Large Repository of Scientific Articles

Hung Son Nguyen¹, Dominik Ślęzak^{1,2}, Andrzej Skowron¹, and Jan G. Bazan^{3,1}

¹ Institute of Mathematics, University of Warsaw
ul. Banacha 2, 02-097 Warsaw, Poland

² Infobright Inc.
ul. Krzywickiego 34 lok. 219, 02-078 Warsaw, Poland

³ Chair of Computer Science, University of Rzeszów
ul. Rejtana 16A, 35-310 Rzeszów, Poland

*Good mathematicians see analogies between theorems or theories.
The very best ones see analogies between analogies.*

– Stefan Banach [36]

Abstract. We present the architecture of the system aimed at search and synthesis of information within document repositories originating from different sources, with documents provided not necessarily in the same format and the same level of detail. The system is expected to provide domain knowledge interfaces enabling the internally implemented algorithms to identify relationships between documents (as well as authors, institutions et cetera) and concepts (such as, e.g., areas of science) extracted from various types of knowledge bases. The system should be scalable by means of scientific content storage, performance of analytic processes, and speed of search. In case of compound computational tasks (such as production of richer semantic indexes for the search improvements), it should follow the paradigms of hierarchical modeling and computing, designed as an interaction between domain experts, system experts, and appropriately implemented intelligent modules.

Keywords: semantic search, semantic information retrieval and synthesis, document analytics, document repositories, interactive and hierarchical computing, decision support, behavioral patterns, wisdom technology.

1 Introduction

This article outlines the proposed architecture, major requirements, assumptions and ideas related to the engine for semantic search and analytics developed within the “Interdisciplinary System for Interactive Scientific and Scientific-Technical Information” project financed by Polish Government in 2010-2013¹. The goal is to extend capabilities of the existing (usually Web-based, sometimes enterprise-based)

¹ http://ismis2011.ii.pw.edu.pl/post_conference_event.php

semantic search engines by deeper analysis of the semantics of user requests. Given such assumptions, we will refer to our engine as *SONCA*, which stands for *Search based on ONtologies and Compound Analytics*.

The engines such as SONCA should support a dialog of users with the text sources gathered within available repositories. It should lead not only to the search of significant documents, including their rankings [6,21], but also to intelligent systems helping users to specify and solve their problems [2,16].

An example of required functionality may be related to user's needs to understand the state of the art in a given domain of science. Surely, we can imagine various implementations addressing such requirement. The final answer to such understood user's query may summarize and synthesize various aspects, such as the most meaningful parts of representative documents related to the given area, characterization of the leading scientific groups and research initiatives, characterization of the most significant concepts and open problems, historical trends of progress in similar domains et cetera. Such framework needs to combine standard search capabilities based on keywords, dictionaries, glossaries, and ontologies [11,14,15] with hierarchical knowledge representation and reasoning [4,17,30], based on domain knowledge acquired from experts and users.

The idea of SONCA can be explained using the principles of interactive calculi on compound objects called information granules [18,29]. The user provides specification (in form of a query) in a language that is most often the natural language or its simplified fragment [40]. Then, the goal of the engine is to construct a compound information granule satisfying the formulated specification to a satisfactory degree. Establishing methods and algorithms for constructing such granules and measures of satisfiability of specifications by granules is crucial. Background for such computations may take a form of information systems [28] or relational database models [10]. Another aspect is to develop methods based on a dialog with users in order to understand their expectations (for example, a degree of advancement in a given scientific area).

Certainly, our engine should be scalable with respect to the volumes of data. It should be also at least partially scalable regarding the number of users, who, usually, would work rather with pre-computed semantic indexes, partially pre-computed results and patterns, but with some fraction of users wanting to run more ad-hoc queries against the entire data. Thus, scalability should also refer to diversity of users' needs and kinds of usage of the stored data. Scalability can be secured at the algorithmic level, e.g. by adapting various forms of approximate reasoning [8,24], as well as at the level of data representation, data processing and data structures, by using database architectures that are aimed at data analysis and compound object handling [1,34].

2 System's Assumptions and Components

SONCA is aimed at extracting and constructing information based on text repositories originating from various libraries and publishers. Documents may be in different formats, such as XML, PDF, and scans of different quality. The system

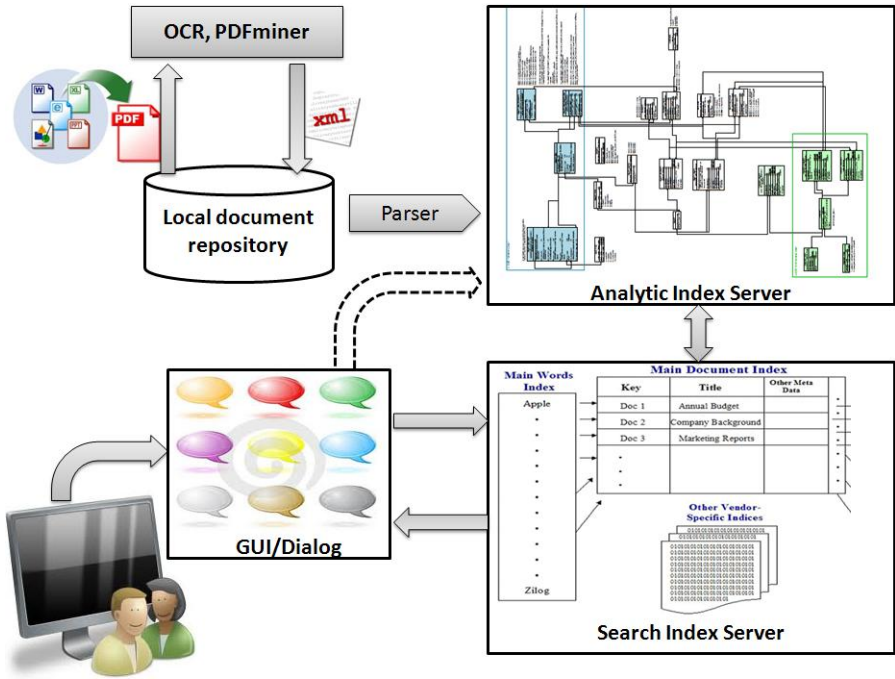


Fig. 1. The proposed architecture of SONCA

should be able to take into account various knowledge bases about the considered areas. There may be also independent sources of information about the analyzed objects, e.g., information about scientists who can be later identified as the authors of documents stored in a repository. Knowledge bases can be employed in multiple ways:

- As a means for communicating with users.
- As a means for injecting domain knowledge.
- As a specification of structures for objects and their relations.
- As a basis for discovering patterns useful for indexing objects.

We refer to [25] for further analysis how to use various sources of knowledge at various stages of our system.

Comparing to engines where documents are the main search process target, the proposed methodology has some additional features, e.g.:

- Ability to represent and search for various objects: concepts, authors, conferences, organizations, results, images, et cetera.
- Ability to use documents and knowledge bases to produce semantic indexes represented as information systems, further applicable in scalable search and information synthesis.

We refer to [26,35] for some examples of algorithms that prepare a background for semantic indexes from the above-mentioned combination of knowledge and information or use pre-computed semantic indexes for further processing.

Figure 1 outlines the system's architecture with its four major modules that can be integrated in multiple ways, also with their completely external counterparts in a more general framework:

- Local document repository of articles, including acquired information about them. They can occur in various forms (see below). They may be also given in metadata-only form.
- Analytic index server, including a framework for developing analytic operations that compute various intermediate structures leading eventually to the output semantic indexes.
- Search index server aimed at providing scalable external access to the latest available versions of semantic indexes.
- User interfaces aimed at decomposing user requests along the lines of domain-knowledge-driven hierarchical modeling.

Local document repository needs to be prepared for truly diversified forms of incomplete information. It is also important to equip it with interfaces to methods that can extract meaningful information and structures from the original files [5,31]. Below there are some observations that can influence local document repository and its interactions with other components:

- It can acquire various types of data, including:
 - i.* Scanned articles.
 - ii.* Digitalized articles (digitalized PDFs).
 - iii.* Metadata – structured (but potentially heterogeneous, given different origins of data) descriptions of articles.
- Metadata can take various forms, including:
 - i.* Input articles with no metadata assigned.
 - ii.* Input articles with complete metadata.
 - iii.* Metadata of articles that are not (yet) present in the database.
 - iv.* Information about other objects (authors, institutes, domains).
- At the stage of receiving data we assume redundancy, e.g.: some article may be already present in the database, may be loaded to the database in future or never. Even metadata with no files assigned may be relevant for analysis, so they should be treated as articles with incomplete information.

Figure 2 illustrates our motivation to choose the MongoDB software [9] to establish the local document repository. It also outlines our choice of the Infobright [33,34] and Lucene [22] software in order to represent information about articles and other entities under consideration in two synchronized forms: repository-oriented and analytics-oriented. As already discussed in Section 1, our investigations led us to applying standard RDBMS data schemas to represent available data in a form that would be useful for more compound analytic processes. We refer to [20] for our further studies in this area.

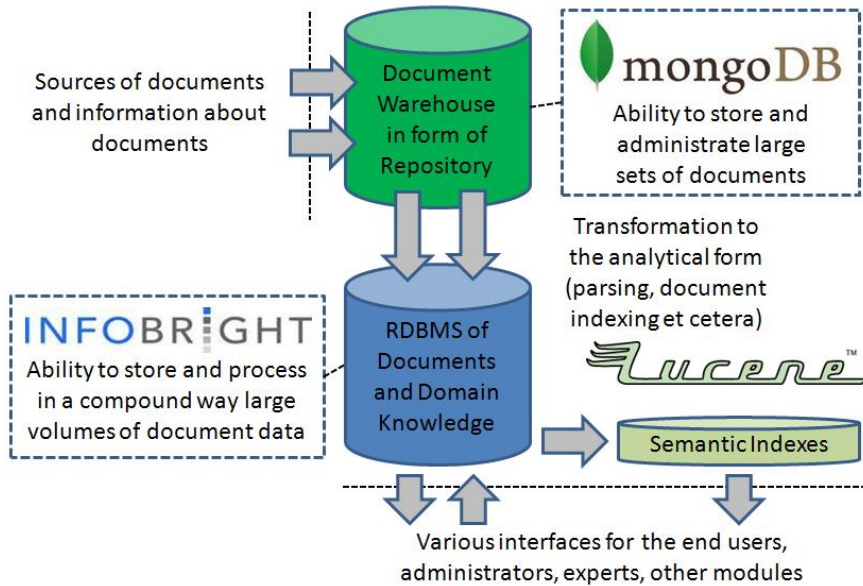


Fig. 2. The proposed software components used to implement local document repository, analytic index server, as well as data flow modules and mechanisms.

Certainly, an important aspect is also to create a framework for (possibly at least partially interactive) evaluation of each of the above components and the whole system with respect to the quality of results and efficiency of providing them to end users. While efficiency may be understood by quite standard means of speed and scalability of the search and analytic processes, evaluation of the quality may seek for analogies in information retrieval [7]. We refer to [39] for some detailed investigation in this area.

3 Further Perspectives

One of our motivations for developing the SONCA system is to extend functionality of the current enterprise and Web search engines towards the support for problem solving via enhanced search capabilities over various types of entities, information synthesis leading to answers that do not correspond to any single original entities, as well as letting advanced users omit some external interface layers and step down to the level of intermediate and core structures. While designing SONCA, we have been seeking for inspiration in many other projects and approaches, related to such domains as, e.g., social networks [23] and heterogeneous information networks [16]. Certainly there are plenty of details to be further discussed, e.g., how and in what form the results of search and analysis should be transmitted between modules and eventually reported to end users. With this respect, we can refer to, e.g., enriching original contents [2], approximate querying [33], or linguistic summaries of query results [19].

Practically all above aspects require a good means for employing domain knowledge, e.g., by developing methods similar to those for semantic Web [3,12]. In particular, one can consider learning behavioral patterns used by domain experts while solving problems [4,38]. Some hints for the research in this direction may follow from our experience with ontology-based approximation of compound concepts and identifying behavioral patterns in different applications. Interactions between Web/enterprise/repository resources and domain experts/users play an important role in learning such patterns. Thus, we plan to provide a framework for dialog between SONCA and its users, e.g., basing on interactive rough granular computations designed within the Wistech framework [18,32], where combination of personalization, interaction, and wisdom is claimed to lead to significant semantic search engine extensions.

In a broader sense the discussed challenges lead to such fundamental issues of mathematics as understanding of the concepts as proof, similarity of proofs, analogies between theorems, analogies between strategies of proofs used in different domains et cetera [36]. In real-life applications one should be ready to deal with even more compound situations. Due to uncertainty and/or necessity of overcoming infeasibility caused by computational complexity, we are forced to deal with interactive approximate reasoning schemes (plans, networks) over vague concepts instead of crisp reasoning schemes [37].

Further extensions of capabilities of engines similar to SONCA should be related to evolution of languages [13,27] applied to constructing and describing information granules related to articles and other entities that queries refer to. Therefore, we should take into account the strategies of automatic adaptive evolution of the language of patterns and granules. We should also expect evolution of domain knowledge and behavioral patterns, even if the language aimed at expressing them does not change. Thus, we should provide the means for storing new automatically learned concepts and behavioral patterns in knowledge bases that influence the processes of search and analytics.

Finally, our investigations should be continued also at a more technical level, ensuring sufficient scalability, performance, and interaction characteristics of the overall solution. For example, in [20] we outline some preliminary observations with respect to the usage of Infobright RDBMS software [34]. Although the outcomes are pretty optimistic, we will keep running scalability and performance tests over multiple platforms, paying a special attention to integration of different technologies supporting different parts of SONCA.

Acknowledgments. The authors are supported by the grant N N516 077837 from the Ministry of Science and Higher Education of the Republic of Poland and by the National Centre for Research and Development (NCBiR) under the grant SP/I/1/77065/10 by the Strategic scientific research and experimental development program: “Interdisciplinary System for Interactive Scientific and Scientific-Technical Information”.

References

1. Agrawal, R., Ailamaki, A., Bernstein, P.A., Brewer, E.A., Carey, M.J., Chaudhuri, S., Doan, A., Florescu, D., Franklin, M.J., Garcia-Molina, H., Gehrke, J., Gruenwald, L., Haas, L.M., Halevy, A.Y., Hellerstein, J.M., Ioannidis, Y.E., Korth, H.F., Kossmann, D., Madden, S., Magoulas, R., Ooi, B.C., O'Reilly, T., Ramakrishnan, R., Sarawagi, S., Stonebraker, M., Szalay, A.S., Weikum, G.: The Claremont Report on Database Research. *Commun. ACM* 52(6), 56–65 (2009)
2. Agrawal, R., Gollapudi, S., Kannan, A., Kenthapadi, K.: Enriching Education through Data Mining. In: Kuznetsov, S.O., Mandal, D.P., Kundu, M.K., Pal, S.K. (eds.) *PREMI 2011*. LNCS, vol. 6744, pp. 1–2. Springer, Heidelberg (2011)
3. Badr, Y., Chbeir, R., Abraham, A., Hassanien, A.: *Emergent Web Intelligence: Advanced Semantic Technologies*. Springer, Heidelberg (2010)
4. Bazan, J.G.: Hierarchical Classifiers for Complex Spatio-temporal Concepts. *T. Rough Sets* 9, 474–750 (2008)
5. Betliński, P., Gora, P., Herba, K., Nguyen, T.T., Stawicki, S.: Semantic Recognition of Digital Documents. In: Bembek, R., Skonieczny, Ł., Rybiński, H., Niezgodka, M. (eds.) *Intelligent Tools for Building a Scientific Information Platform*. Springer, Heidelberg (2011)
6. Breitman, K., Casanova, M., Truszkowski, W.: *Semantic Web: Concepts, Technologies and Applications*. Springer, Heidelberg (2007)
7. Butcher, S., Clarke, C., Cormack, G.: *Information Retrieval: Implementing and Evaluating Search Engines*. MIT Press (2010)
8. Cao, L.: *Data Mining and Multiagent Integration*. Springer, Heidelberg (2009)
9. Chodorow, K., Dirolf, M.: *MongoDB: The Definitive Guide: Powerful and Scalable Data Storage*. O'Reilly Media (2010)
10. Codd, E.: Derivability, Redundancy and Consistency of Relations Stored in Large Data Banks. *SIGMOD Record* 38(1), 17–36 (2009)
11. Colomb, R.: *Ontology and the Semantic Web*. IOS Press (2007)
12. Davies, J., Grobelnik, M., Mladenic, D.: *Semantic Knowledge Management: Integrating Ontology Management, Knowledge Discovery, and Human Language Technologies*. Springer, Heidelberg (2009)
13. Feldman, J.A.: *From Molecule to Metaphor: A Neural Theory of Language* (A Bradford Book). MIT Press (2006)
14. Gasevic, D., Djuric, D., Devedzic, V.: *Model Driven Engineering and Ontology Development*. Springer, Heidelberg (2009)
15. Gómez-Pérez, A., Fernández-López, M., Corcho, O.: *Ontological Engineering with Examples from the Areas of Knowledge Management, e-Commerce and the Semantic Web*. Springer, Heidelberg (2004)
16. Han, J.: Construction and Analysis of Web-Based Computer Science Information Networks. In: Kuznetsov, S.O., Ślęzak, D., Hepting, D.H., Mirkin, B.G. (eds.) *RSFDGrC 2011*. LNCS (LNAI), vol. 6743, pp. 1–2. Springer, Heidelberg (2011)
17. Helbig, H.: *Knowledge Representation and the Semantics of Natural Language*. Springer, Heidelberg (2006)
18. Jankowski, A., Skowron, A.: A Wistech Paradigm for Intelligent Systems. *T. Rough Sets* 6, 94–132 (2007)
19. Kacprzyk, J., Zadrozny, S.: Computing with words is an implementable paradigm: Fuzzy queries, linguistic data summaries, and natural-language generation. *IEEE T. Fuzzy Systems* 18(3), 461–472 (2010)

20. Kowalski, M., Ślęzak, D., Stencel, K., Pardel, P., Grzegorowski, M., Kijowski, M.: RDBMS Model for Scientific Articles Analytics. In: Bembenik, R., Skonieczny, Ł., Rybiński, H., Niezgódka, M. (eds.) *Intelligent Tools for Building a Scientific Information Platform*. Springer, Heidelberg (2011)
21. Ledford, J.L.: *Search Engine Optimization Bible*. Wiley (2009)
22. McCandless, M., Hatcher, E., Gospodnetić, O.: *Lucene in Action*, 2nd edn. Manning Publications (2010)
23. Mika, P.: Social Networks and the Semantic Web. In: *Proc. of Int. Conf. on Web Intelligence (WI)*, pp. 285–291 (2004)
24. Nguyen, H.S.: Approximate Boolean Reasoning: Foundations and Applications in Data Mining, pp. 334–506 (2006)
25. Nguyen, L.A., Nguyen, H.S.: On Designing the SONCA System. In: Bembenik, R., Skonieczny, Ł., Rybiński, H., Niezgódka, M. (eds.) *Intelligent Tools for Building a Scientific Information Platform*. Springer, Heidelberg (2011)
26. Nguyen, S.H., Świeboda, W., Jaśkiewicz, G.: Extended Document Representation for Search Result Clustering. In: Bembenik, R., Skonieczny, Ł., Rybiński, H., Niezgódka, M. (eds.) *Intelligent Tools for Building a Scientific Information Platform*. Springer, Heidelberg (2011)
27. Nolfi, S., Mirulli, M.: *Evolution of Communication and Language in Embodied Agents*. Springer, Heidelberg (2010)
28. Pawlak, Z.: Information Systems Theoretical Foundations. *Inf. Syst.* 6(3), 205–218 (1981)
29. Pedrycz, W., Skowron, A., Kreinovich, V. (eds.): *Handbook of Granular Computing*. Wiley (2008)
30. Poggio, T., Smale, S.: The Mathematics of Learning: Dealing with Data. *Notices of the AMS* 50(5), 537–544 (2003)
31. Shinyama, Y.: PDFMiner: Python PDF Parser and Analyzer (2010), <http://www.unixuser.org/~euske/python/pdfminer/>
32. Skowron, A., Stepaniuk, J., Świniarski, R.W.: Approximation Spaces in Rough-Granular Computing. *Fundam. Inform.* 100(1-4), 141–157 (2010)
33. Ślęzak, D., Kowalski, M.: Towards Approximate SQL – Infobright’s Approach. In: Szczuka, M., Kryszkiewicz, M., Ramanna, S., Jensen, R., Hu, Q. (eds.) *RSCTC 2010*. LNCS, vol. 6086, pp. 630–639. Springer, Heidelberg (2010)
34. Ślęzak, D., Wróblewski, J., Eastwood, V., Synak, P.: Bighthouse: An Analytic Data Warehouse for Ad-hoc Queries. *Proc. VLDB Endow.* 1(2), 1337–1345 (2008)
35. Szczuka, M., Janusz, A., Herba, K.: Clustering of Rough Set Related Documents with Use of Knowledge from DBpedia. In: Yao, J. (ed.) *RSKT 2011*. LNCS (LNAI), vol. 6954, pp. 394–403. Springer, Heidelberg (2011)
36. Ulam, S.: *Analogies Between Analogies: The Mathematical Reports of S. M. Ulam and His Los Alamos Collaborators*. University of California Press (1990)
37. Valiant, L.G.: Robust Logics. *Artif. Intell.* 117(2), 231–253 (2000)
38. Vapnik, V.: Learning Has Just Started (An interview with Vladimir Vapnik by Ran Gilad-Bachrach) (2008), <http://seed.ucsd.edu/joomla/index.php/articles/12-interviews/9-qlearning-has-just-startedq-an-interview-with-prof-vladimir-vapnik>
39. Wasilewski, P.: Towards Semantic Evaluation of Information Retrieval. In: Bembenik, R., Skonieczny, Ł., Rybiński, H., Niezgódka, M. (eds.) *Intelligent Tools for Building a Scientific Information Platform*. Springer, Heidelberg (2011)
40. Zadeh, L.A.: Computing with Words and Perceptions - A Paradigm Shift. In: *Proc. of Int. Conf. on Parallel and Distributed Processing Techniques and Applications (PDPTA)*, pp. 3–5 (2010)

On Designing the SONCA System

Linh Anh Nguyen¹ and Hung Son Nguyen²

¹ Institute of Informatics

² Institute of Mathematics

University of Warsaw

Banacha 2, 02-097 Warsaw, Poland

{nguyen,son}@mimuw.edu.pl

Abstract. The SYNAT project aims to develop a universal, open hosting and communication platform for network knowledge resources for science, education and open information society. The stage B13 of this project aims to develop methods and algorithms of semantic indexing, classification and retrieval using dictionaries, thesauri and ontologies as well as methods of processing and visualizing results. The methods and algorithms aim to support the dialogue with the repositories of text and multimedia resources gathered on some servers. To realize the objectives of the stages B13 and B14 of the SYNAT project we plan to develop and implement a system called SONCA (Search based on ONtologies and Compound Analytics).

We present ideas and proposals for the SONCA system. The main idea is to allow combination of metadata-based search, syntactic keyword-based search and semantic search, and to use ranks of objects. Semantic search criteria may be keywords, concepts, or objects (for checking similarity). Search criteria based on metadata play the role of exact restrictions, while syntactic keywords and semantic search criteria are fuzzy restrictions. To enable metadata-based search, an appropriate document representation is used. To enable syntactic keyword-based search, each document (object) is stored together with information about the terms occurring in its text attributes. The terms are normalized and only important ones are stored. To enable semantic search, the representation of each document (object) is extended further with the most important concepts that characterize the document. Such concepts belong to the main ontology of SONCA.

We provide an abstract model for the SONCA system, an instantiation of that model, some ideas for the user interface of SONCA as well as proposals for increasing efficiency of the query answering process.

1 Introduction

The SYNAT project [13], realized in 2010-2013 by 16 scientific institutions, aims to develop a universal, open hosting and communication platform for network knowledge resources for science, education and open information society. Our institution (MIMUW) realizes two out of 52 stages of the SYNAT project. One

¹ Faculty of Mathematics, Informatics and Mechanics, University of Warsaw.

of them is to develop methods and algorithms of semantic indexing, classification and retrieval using dictionaries, thesauri and ontologies as well as methods of processing and visualizing results. The methods and algorithms aim to support the dialogue with the repositories of text and multimedia resources gathered on some servers. To realize the objectives we plan to develop and implement a system called SONCA (Search based on ONtologies and Compound Analytics). The additional aim is to test our methods and prepare for an integration of our system with the systems developed by the other partners of the SYNAT project.

Here are some assumptions for the SONCA system:

- The system offers searching objects like publications, authors, institutions and other related information. Documents may be written in English, Polish or other languages.
- The search engine extends metadata-based search and keyword-based search with semantic search.
- The system is general-purpose, running on a large repository of gathered objects (articles, institutions, authors ...). It does not assume to rely on specific domains only.
- The system is extensible. It can work with different types of domain-specific knowledge.

In this paper, we present ideas and proposals for the searching track of the SONCA system. The main idea is to combine different search approaches including metadata-based search, syntactic keyword-based search and semantic search, and to use ranks of objects. Semantic search criteria may be keywords, concepts, or objects (for checking similarity). Search criteria based on metadata play the role of exact restrictions, while syntactic keywords and semantic search criteria are fuzzy restrictions. To enable metadata-based search, an appropriate document representation is used and the database of SONCA is designed accordingly. To enable syntactic keyword-based search, each document (object) is stored together with information about the terms occurring in its text attributes. The terms are normalized and only important ones are stored. To enable semantic search, the representation of each document (object) is extended further with the most important concepts that characterize the document. Such concepts are specified by the main ontology of SONCA, which may consist of several subontologies.

1.1 On the Web Ontology Language OWL

OWL is a standardized Web ontology language, with the first and second versions recommended by W3C in 2004 [14] and 2009 [15], respectively. These versions, OWL 1 and OWL 2, have some sublanguages (species/profiles):

- OWL 1 DL and OWL 2 DL support those users who want maximum expressiveness without losing computational completeness.

- OWL 1 Lite, OWL 2 EL, OWL 2 QL and OWL 2 RL are restricted versions with PTIME data complexity of the “DL” sublanguages.
- OWL 1 Full and OWL 2 Full support those users who want maximum expressiveness and the syntactic freedom of RDF with no computational guarantees.

OWL 1 DL is based on the description logic *SHOIN* [11], while OWL 2 DL is based on a more expressive description logic *SROIQ* [10]. Other well-known description logics (DLs) are *ALC*, *SHIQ* and *SHOIQ*. DLs are fragments of classical first-order logic and are variants of modal logics, used to describe the domain of interest by means of individuals, concepts and roles. A concept is interpreted as a set of individuals, while a role is interpreted as a binary relation between individuals. The satisfiability checking problem is EXPTIME-complete in *ALC* and *SHIQ*; NEXPTIME-complete in *SHOIN* and *SHOIQ*; and N2EXPTIME-complete in *SROIQ*.

According to the recent survey [28], the third generation ontology reasoners that support *SHIQ* or *SROIQ* are FaCT, FaCT++, RACER, Pellet, KAON2 and HermiT. The reasoners FaCT, FaCT++, RACER and Pellet are based on tableaux, KAON2 is based on a translation into disjunctive Datalog, and HermiT is based on hypertableaux. That is, all of the listed reasoners except KAON2 are tableau-based. According to [30], KAON2 provides good performance for ontologies with rather simple TBoxes, but large ABoxes²; however, for ontologies with large and complex TBoxes, existing tableau-based reasoners still provide superior performance. The current version of KAON2 does not support nominals and cannot handle large numbers in cardinality statements. The reasoners FaCT, FaCT++, RACER and Pellet use traditional tableau decision procedures like the ones for *SHIQ* [12], *SHOIQ* [11], *SROIQ* [10]. These procedures use backtracking to deal with disjunction (e.g., the union constructor). Their search space is an “or”-tree of “and”-trees. Despite advanced blocking techniques (e.g., anywhere blocking), their complexities are non-optimal (e.g., NEXPTIME instead of EXPTIME for *SHIQ*; N2EXPTIME instead of NEXPTIME for *SHOIQ*; and N3EXPTIME instead of N2EXPTIME for *SROIQ*). Similarly, the decision procedure of HermiT also has a non-optimal complexity [6]. To obtain optimal and efficient tableau decision procedures for DLs one may consider the optimization techniques developed in [7,8,38,32,40,39,36,37,35].

The profiles OWL 2 EL, OWL 2 QL and OWL 2 RL are restricted sublanguages of OWL 2 FULL with PTIME data complexity. They are based on the families of DLs \mathcal{EL} [12], DL-Lite [3] and DLP (Description Logic Programs) [9], respectively. As fragments of DLs with PTIME data complexity there are also Horn-*SHIQ* [25], Horn-*SROIQ* [42], and the deterministic Horn fragments of *ALC* and regular description logics [31,33]. Formalisms that combine a DL with a rule language were studied, among others, in [27,4,26,29,5].

² A TBox is a set of terminology axioms, including role axioms. In the literature of DLs, sometimes role axioms are grouped into an RBox. An ABox is a set of individual assertions.

1.2 Using Ontologies for the SONCA System

Ontologies can be used for:

- increasing flexibility in using different names for the same attribute/relation
- understanding the meanings of terms occurring in documents.

The first purpose is useful for the user interface (e.g., for transforming a user’s query to a formal query) and for integrating modules implemented by different partners of the SYNAT project. Ontologies in OWL or similar languages can be used for this purpose.

The second purpose is useful for:

- matching or measuring similarity between:
 - term – concept – document
 - concept – document
 - document – concept – document
- answering queries
- the user interface (for navigation and improving queries).

The second purpose seems more important (or at least harder to be fulfilled) than the first one. In our opinion, ontologies in OWL are not suitable for the second purpose due to the following reasons:

- Most documents of the SONCA system (like publications and webpages) are unstructured or semi-structured. They are PDF files, (OCR’ed) texts, HTML or XML documents, but not OWL-based documents.
- Reasoning in OWL DL has a very high complexity.
- The knowledge base of a very large system like SONCA may be inconsistent. Despite that paraconsistent reasoning has been studied for DLs (e.g., in [41,34]), the techniques may be not advanced enough.

For these reasons, we propose to use a thesaurus extended with fuzzy relationships between the concepts to “understand” the meanings of terms occurring in documents. We call that thesaurus the *main ontology* of SONCA. It may consist of several subontologies. We can construct it from sources like DBpedia [16], WordNet [17] and plWordNet [18]. In this paper we address the construction of this main ontology and its use for query answering. We omit ontologies in OWL, which does not mean they are not useful for SONCA and SYNAT.

1.3 The Contributions and the Structure of This Work

In this paper we provide an abstract model for the SONCA system, an instantiation of that model, some ideas for the user interface of SONCA as well as proposals for increasing efficiency of the query answering process.

The abstract model includes notions and requirements of:

- the collection of terms
- the main ontology

- three abstract document-representations
- abstract queries (together with their meanings).

The second document representation extends the first one with information about terms occurring in the document, and the third document representation extends the second one with *ObjectRank* and *RelatedConcepts* of the document.

Our instantiation of that abstract model provides definitions and detailed descriptions for the model, including:

- a function computing concepts similar to a given concept
- a method of computing *ObjectRank* of an object
- a function computing *RelatedConcepts* of an object
- several functions measuring similarity between terms, concepts and objects.

The rest of this paper is structured as follows. In Section 2 we present our abstract model of SONCA. In Section 3 we present our instantiation of that model. Section 4 is devoted to the user interface of SONCA. Section 5 contains our proposals for increasing efficiency of the query answering process. Section 6 contains some concluding remarks. Appendix A is devoted to DBpedia and its use for the SONCA system.

2 An Abstract Model for SONCA

2.1 Terms, Concepts and Ontologies

A *term* is a string representing a word or a phrase of a few words, which has been normalized using some algorithm (e.g., a stemming algorithm). The maximal number of words in a term is parameterized by *MaxTermLength* and chosen appropriately. The importance of a term t_i in a text d_j (which can be an element of a document) is usually measured by *tf-idf* _{t_i, j} (term frequency - inverse document frequency), which is defined as $tf_{i,j} \times idf_i$, where $tf_{i,j}$ stands for the frequency of t_i in d_j , and idf_i stands for the general importance of term t_i in the set of all documents of the system. There are different possible definitions of $tf_{i,j}$ and idf_i [19].

The system contains information about important terms, which is a collection TERMS of pairs (t_i, idf_i) . Only terms t_i with high enough idf_i will be stored in this collection. The threshold for idf is parameterized by *MinIdf* and chosen appropriately.

Consider, for example, the text “information and decision systems”. As terms in this text one can consider “information”, “and”, “decision”, “system”, “decision system”, “information and decision”, “information and decision system”. Notice that we exclude the phrases “information and”, “and decision”, “and decision systems”. Among the listed terms, “and” probably has very low idf value and is not stored.

The system implements a function $Terms(d_j, h)$ that, for a given text d_j and a *tf-idf* threshold h , returns information about the most important terms t_i occurring in d_j in the form of a collection of pairs $(t_i, tf_{i,j})$ with $tf-idf_{i,j} \geq h$.

The system also implements a function $Sim_{st}(s, t)$, which measures the similarity between a string and a term.

A *concept* is understood here as a meaning, which is specified, amongst others, by an ID and a list of pairs (*label*, *weight*), where *label* is a term and *weight* is a real number in $(0, 1]$ representing the accuracy that the term has that meaning.

The system implements a function $Sim_{tc}(t, c)$, which measures the similarity between a term t and a concept (with ID) c , i.e., the accuracy that the term t has the meaning c .

An *ontology* is a compact data structure containing information about concepts and the relationships between them. For a given ontology, the system implements a function $SimilarConcepts(c, n, s)$, where c is a concept (ID), n is a natural number and s is a real number in $(0, 1]$. This function returns a list of no more than n pairs (c_i, s_i) such that c_1, c_2, \dots are the concepts of the ontology most similar to c and $s_i \geq s$ is the similarity between c and c_i . (This similarity measure between concepts need not be a commutative operator.) The system also implements a function $PossibleConcepts(t)$ which returns the set of concepts which may correspond to the term t .

2.2 Abstract Document-Representations

The database of SONCA contains a set of objects, which may be publications, authors, institutions, etc. Each object is represented as a tree-like document (like an XML document). The original form of a publication may be an image. However, in the current paper we assume that such a document has been OCR'ed and we have its text-based representation, possibly with references to images representing the whole paper and its figures. Images are not treated as documents; they are stored in a specific way and only used for visualization in user interfaces, but not for the search process. Using the terminology of XML, an object is an element; each complex element has a name, possibly some attributes, and contents, which are simple values (e.g., texts, numbers, references) or complex elements; each attribute has a name and a simple value. Elements and attributes of XML are similar, but with the differences that, values of attributes are simple (i.e. without structure), attributes are functional, and the order of attributes are inessential, while the order of elements is essential.

In the current paper, we use the term “attribute” to refer to both “attribute” and “element” as the ones of XML. When necessary we use adjectives like simple, complex, functional to describe attributes. In our settings, the order of attributes with the same name in a given scope is essential. Documents and attributes have types, which are simple or complex. A value of a simple type is either a literal (e.g., a number, a date, a reference, or a short text) or a long text.

We give below an exemplary document, which consists of pairs of the form (an attribute name: a value or a list of values), where values may be complex structures:

ID: 1024

Type: 1

```

Title: "Rudiments of rough sets"
AuthorID: 1121, 1432
JournalID: 2124
Volume: 177
Number: 1
Year: 2007
Pages: "3-27"
Abstract: ...
Keyword: "vague concepts", "information and decision systems", ...
Section:
  (Section 1):
    Title: "Introduction"
    Text: ...
    Citation:
      (Citation 1):
        Reference: 3243, 3324, 4532
        Text: ...
    ...
  ...
  ...

```

We distinguish *external objects* as the ones with ID (like publications and authors) from *internal objects* that do not have ID (like sections in a paper). By a *document*, in a narrower sense, we mean a publication, but in a broader sense, we mean any external object.

By ADR1 we call the above discussed abstract representation of documents.

By ADR2 we call the representation that extends ADR1 with information about the most important terms occurring in a document. In particular, in ADR2 each simple attribute of a document whose value is a text d_j will be stored together with $Terms(d_j, h)$, where the *tf-idf* threshold h is parameterized ($h = MinTfIdf$ [full path of the attribute]) and chosen for each attribute in an appropriate way. Furthermore, we assume that, for each document x in ADR2, the set of all normalized words of text attributes of x , denoted by $words(x)$, and the set of all terms of text attributes of x , denoted by $terms(x)$, are stored together with x .

By ADR3 we call the representation that extends ADR2 for an external object with information about *ObjectRank* and the most important concepts related with that object. This representation is discussed below.

Some attributes link different external objects (e.g., authorship and references). In this way, objects are connected and create a graph like a web. *ObjectRank* serves the same role as *PageRank* (see, e.g., [20]).

In ADR3, each publication is stored with a list of pairs $(c_1, s_1), \dots, (c_n, s_n)$, where c_1, \dots, c_n is the list of the concepts most related to the topic of the publication, and each s_i represents the similarity between c_i and the topic, in the scale $(0, 1]$. Only pairs (c_i, s_i) with high enough s_i are stored. That is, we take the pairs (c_i, s_i) with s_i greater than or equal to a parameter *MinConSimForDoc*. The number of such pairs is bounded by a limit parameterized by *MaxConceptsForDoc*, which is chosen appropriately.

Similarly, other external objects like authors, journals or conferences are stored with a list of the most important related concepts (representing the fields the author works in or the fields the journal or conference involves).

We use *RelatedConcepts* as the attribute representing the list of the most important related concepts of a given document in ADR3.

Documents are added to the system at some stage in ADR1. They are then extended to ADR2, and after that to ADR3, possibly in a few stages (for improving the quality of the additional information).

2.3 Abstract Queries

In this subsection we define a general form for queries in SONCA. Such a form is not provided by the user, but by the user interface module. It is a form which requires preprocessing before the query is passed to the search module.

We assume that the system implements the following functions:

- $Sim_{so}^{syn}(x, y)$ is a function that maps a pair consisting of a phrase (string) x and an external object (e.g., a publication) y to a real number in $[0, 1]$ representing the syntactic similarity between x and y .
- $Sim_{so}^{sem}(x, y)$ (resp. $Sim_{co}(x, y)$ or $Sim_{oo}(x, y)$) is a function that maps a pair consisting of a phrase (resp. a concept or an external object³) x and an external object (e.g., a publication) y to a real number in $[0, 1]$ representing the semantic similarity between x and y .

Simple attributes whose values are expected to be long texts (e.g., the text of a section) are called *long-text attributes*. The part of the database of SONCA without values of long-text attributes is called the *metadata* of the system.

A *general query* is a tuple

$$\Phi = (\Phi_{Metadata}, \Phi_{SynPhrases}, \Phi_{SemPhrases}, \Phi_{Concepts}, \Phi_{SimObjects}, \Phi_{weighting})$$

where

- $\Phi_{Metadata}$ is a logical formula over the metadata of the system, with only one free variable standing for the queried object; the formula can be an SQL-like query, a first-order logic formula, a fixpoint formula, a rule-based query (e.g., a Datalog or stratified Datalog⁻ query), a description-logic-like query, etc
- $\Phi_{SynPhrases}$ is
 - either a set of weighted phrases, i.e., a set of pairs (s_i, w_i) , where each s_i is a string and the corresponding w_i is a weight for s_i (the phrases from $\Phi_{SynPhrases}$ are syntactically matched with queried objects)
 - or a Boolean combination using \wedge and \vee of fuzzy expressions of the form $Sim_{so}^{syn}(s_i, x)$ with the same variable x , where each s_i is a phrase and x stands for the queried object; in this case, $\Phi_{SynPhrases}$ is a fuzzy function with argument x

³ e.g., a publication or an author.

- $\Phi_{SemPhrases}$ can be described similarly as for $\Phi_{SynPhrases}$ except that the word “syntactically” is replaced by “semantically” and $Sim_{so}^{syn}(s_i, x)$ is replaced by $Sim_{so}^{sem}(s_i, x)$
- $\Phi_{Concepts}$ is
 - either a set of weighted concepts (used to matching with queried objects)
 - or a Boolean combination using \wedge and \vee of fuzzy expressions of the form $Sim_{co}(c_i, x)$ with the same variable x , where each c_i is a concept and x stands for the queried object; in this case, $\Phi_{Concepts}$ is a fuzzy function with argument x
- $\Phi_{SimObjects}$ is
 - either a set of weighted objects (used to “semantically” matching with queried objects w.r.t. similarity of topics/fields)
 - or a Boolean combination using \wedge and \vee of fuzzy expressions of the form $Sim_{oo}(o_i, x)$ with the same variable x , where each o_i is an external object and x stands for the queried object; in this case, $\Phi_{SimObjects}$ is a fuzzy function with argument x
- $\Phi_{weighting}$ is a (monotonic) function from $[0, 1]^4 \times \mathbb{R}$ to \mathbb{R} .

Notice that the given form of queries enables combination of searching based on metadata, searching based on syntactic keywords and “semantic” searching.

The user interface module may interact with the user to improve the query by eliminating $\Phi_{SemPhrases}$, $\Phi_{SimObjects}$ and modifying $\Phi_{Concepts}$. It is more likely that the user will give a list of phrases (resp. concepts, objects) instead of a set of weighted phrases (resp. concepts, objects). In that case, the system uses an appropriate conversion method. (Note that the first element in a list usually has a greater weight than the others.)

The result of a query Φ is specified as follows:

- Assume an appropriate definition for the fuzzy Boolean operators \wedge and \vee . For example, $(x \wedge y) = (x.y)$ and $(x \vee y) = (x + y - x.y)$. An alternative definition is $(x \wedge y) = \min(x, y)$ and $(x \vee y) = \max(x, y)$.
- Let X be the set of all external objects x (with an appropriate type) of the system such that $\Phi_{Metadata}(x)$ holds.
- For $\Phi_{SynPhrases} = \{(s_1, w_1), \dots, (s_n, w_n)\}$ and $x \in X$, define the matching score of $\Phi_{SynPhrases}$ for x as follows:

$$ms_s^{syn}(\Phi_{SynPhrases}, x) = \frac{\sum_{i=1}^n (w_i \times Sim_{so}^{syn}(s_i, x))}{\sum_{i=1}^n w_i}.$$

- If $\Phi_{SynPhrases}$ is a fuzzy function then define

$$ms_s^{syn}(\Phi_{SynPhrases}, x) = \Phi_{SynPhrases}(x)$$

- For $\Phi_{SemPhrases} = \{(s_1, w_1), \dots, (s_n, w_n)\}$ and $x \in X$, define the matching score of $\Phi_{SemPhrases}$ for x as follows:

$$ms_s^{sem}(\Phi_{SemPhrases}, x) = \frac{\sum_{i=1}^n (w_i \times Sim_{so}^{sem}(s_i, x))}{\sum_{i=1}^n w_i}$$

- If $\Phi_{SemPhrases}$ is a fuzzy function then define

$$ms_s^{sem}(\Phi_{SemPhrases}, x) = \Phi_{SemPhrases}(x)$$

- For $\Phi_{Concepts} = \{(c_1, w_1), \dots, (c_n, w_n)\}$ and $x \in X$, define the matching score of $\Phi_{Concepts}$ for x as follows:

$$ms_c(\Phi_{Concepts}, x) = \frac{\sum_{i=1}^n (w_i \times Sim_{co}(c_i, x))}{\sum_{i=1}^n w_i}$$

- If $\Phi_{Concepts}$ is a fuzzy function then define

$$ms_c(\Phi_{Concepts}, x) = \Phi_{Concepts}(x)$$

- For $\Phi_{SimObjects} = \{(o_1, w_1), \dots, (o_n, w_n)\}$ and $x \in X$, define the matching score of $\Phi_{SimObjects}$ for x as follows:

$$ms_o(\Phi_{SimObjects}, x) = \frac{\sum_{i=1}^n (w_i \times Sim_{oo}(o_i, x))}{\sum_{i=1}^n w_i}$$

- If $\Phi_{SimObjects}$ is a fuzzy function then define

$$ms_o(\Phi_{SimObjects}, x) = \Phi_{SimObjects}(x)$$

- For $x \in X$, define the matching score of Φ for x as follows:

$$ms(\Phi, x) = \Phi_{weighting}(y_1, y_2, y_3, y_4, y_5)$$

where

$$\begin{aligned} y_1 &= ms_s^{syn}(\Phi_{SynPhrases}, x) \\ y_2 &= ms_s^{sem}(\Phi_{SemPhrases}, x) \\ y_3 &= ms_c(\Phi_{Concepts}, x) \\ y_4 &= ms_o(\Phi_{SimObjects}, x) \\ y_5 &= ObjectRank(x) \end{aligned}$$

- The result of the query Φ is the set of objects $x \in X$ that have the highest scores $ms(\Phi, x)$. The maximal number of objects to return is parameterized by *MaxResults*.

2.4 Summary

The proposed abstract model for SONCA depends on:

- computing a collection of terms (TERMS) and defining a function $Terms(d, h)$, where d is a text and h is a *tf-idf* threshold
- using a general-purpose ontology with the following functions:

- $PossibleConcepts(t)$, where t is a term
 - $Sim_{tc}(x, y)$, where x is a term and y is a concept
 - $SimilarConcepts(c, n, s)$, where c is a concept, n is the maximal number of returned concepts which are similar to c , and s is a threshold for similarity
- using the abstract document-representation ADR3 (in particular, computing and using $ObjectRank$ and $RelatedConcepts$ of objects)
 - defining and implementing the following similarity functions:
 - $Sim_{st}(x, y)$, where x is a string and y is a term
 - $Sim_{so}^{syn}(x, y)$, where x is a string and y is an object (a document)
 - $Sim_{so}^{sem}(x, y)$, where x is a string and y is an object (a document)
 - $Sim_{co}(x, y)$, where x is a concept and y is an object (a document)
 - $Sim_{oo}(x, y)$, where x and y are objects (documents)
 - using appropriate values for the parameters $MaxTermLength$, $MinIdf$, $MinTfIdf[-]$, $MaxConceptsForDoc$, $MinConSimForDoc$, $MaxResults$
 - using an appropriate definition for the fuzzy Boolean operators \wedge and \vee .

Function Sim_{so}^{sem} may be defined by using $Terms$, $PossibleConcepts$, Sim_{st} , Sim_{tc} , $SimilarConcepts$ and the attributes $RelatedConcepts$ of objects. Functions Sim_{co} and Sim_{oo} may be defined by using function $SimilarConcepts$ and the attributes $RelatedConcepts$ of objects.

3 An Instantiation of the Abstract Model

In this section, we describe how the abstract model presented in the previous section can be implemented. There are other possible realizations of that abstract model, and what proposed in this section also allows different ways of implementation.

3.1 Collection TERMS and Involved Functions

Collection TERMS can be constructed either by extracting terms from the database of SONCA or by importing terms from existing datasets. Consider the first case. Assume that most documents have been stored in the SONCA system using ADR1 representation. For each text d occurring in the system, every phrase of d consisting of no more than $MaxTermLength$ words is checked and, if it is acceptable as a term, will be normalized and added to the collection TERMS. For example, a phrase like “and decision” is not accepted as a term, and a phrase “decision systems” can be normalized to “decision system”. At the end, the measure idf_i of each element t_i of TERMS will be computed; if $idf_i > MinIdf$ then idf_i will be stored together with t_i in TERMS, else the term t_i is deleted from TERMS. After that we assume that idf values of elements of TERMS are rescaled to fit into the interval $(0, 1]$. For the case TERMS is constructed by importing terms from existing datasets, the parameters $MaxTermLength$ and $MinIdf$ are also used in an appropriate way, and rescaling to the interval $(0, 1]$ is also performed.

Function $Terms(d, h)$, where d is a text and h is a *tf-idf* threshold, may be defined as follows:

- initialize rs to an empty list
- for every phrase t of d consisting of no more than $MaxTermLength$ words
 - let t' be the normalized form of t
 - if t' occurs in **TERMS** then
 - * let x be the *tf* measure of t in d
 - * let y be the *idf* measure of t in **TERMS**
 - * if $x \cdot y \geq h$ then add the pair (t, x) to rs
- sort rs decreasingly w.r.t. the second coordinates of the pairs
- return rs .

Having collection **TERMS** and function $Terms(d, h)$ defined, documents represented in ADR1 can be enriched to ADR2 in the usual way.

Function $Sim_{st}(x, y)$ for a string x and a term y from **TERMS** may be defined as follows, where $SimST1$ and $SimST2$ are parameters of SONCA with values in $(0, 1]$:

- if the normalized form of x is equal to y then return 1
- let s_x be the set of normalized forms of words occurring in x
- let s_y be the set of normalized forms of words occurring in y
- if $s_x = s_y$ then return $SimST1$ else $s := s_x \cap s_y$
- if $s = s_x$ then $a := 1$ else $a := SimST2$
- if $s = s_y$ then $b := 1$ else
 - let b_0 be the maximal *idf* measure of a word from s w.r.t. **TERMS**
 - let b_1 be the *idf* measure of y w.r.t. **TERMS**
 - $b := (b_0/b_1) \times (|s|/|s_y|)$ (one can modify this formula somehow, e.g., by applying *sqrt* to the whole formula or a part of it)
- if the words of s occur in x in the same order as in y (ignoring the differences caused by normalization) then $c := 1$ else $c := SimST1$
- return $a \times b \times c$.

3.2 The Main Ontology and Involved Functions

The SONCA system may use a number of ontologies, but they are combined into one, which is called the *main ontology* of SONCA. This ontology contains information about concepts and relationships between them.

A concept is specified by the concept ID, the ID of the component ontology (if necessary), and a list of pairs (*label, weight*) called the *list of labels* of the concept, where *label* is a term and *weight* is a real number in $(0, 1]$ representing the accuracy that the term has the concept’s meaning. In the case information about the *weight* is not available, we assume that it is equal to 1. Information about concepts may be stored in a few tables (e.g., component ontologies, concepts, labels of concepts). IDs of concepts are unique. For convenience, sometimes we just write “concept” to mean “concept-ID”.

We index essential words occurring in labels of concepts by using a table WORD-CONC consisting of pairs (*word*, *concept-ID*) together with an index on the attribute “word”. For each word w occurring in label l of a concept c : let w' be the normalized form of w ; if w' occurs in TERMS then add the pair (w, c) to that table.

We use a table CONC-CONC to store relationships between concepts. It consists of tuples of the form $(ID_1, ID_2, type, similarity)$, where ID_1 and ID_2 are IDs of two concepts, *type* is the type of the relationship, and *similarity* is a real number in $(0, 1)$ representing the degree that the first concept is similar to the second concept. In the case information about the *similarity* is not available, it will be computed from the *type*, according to some strategy based on parameters. For example, the type “is a subconcept of” (e.g., *Man* is a subconcept of *Human*) may have *similarity* = 0.8, while the type “is a superconcept of” may have *similarity* = 0.4 (notice the asymmetry).

The main ontology of SONCA can be constructed from DBpedia, in particular, from the datasets “Articles Categories”, “Categories (Skos)”, “Links to YAGO2” and “Word Net Classes”. See the appendix for more details.

Function *PossibleConcepts*(t), which returns the set of concepts which may correspond to the term t , can be defined as follows:

- if the dataset “Disambiguation Links” of DBpedia can be used to determine the concepts corresponding to t then use it and return the result
- $rs := \emptyset$
- add to rs all concepts that have t as a label
- for each word w occurring in t :
 - let w' be the normalized form of w
 - for each pair (w', c) occurring in WORD-CONC, add c to rs
- return rs .

Function *Sim_{tc}*(x, y), where x is a term and y is a concept, may be defined as follows:

- let the list of labels of y be $(l_1, w_1) \dots (l_k, w_k)$
- let $t\text{-conorm}_{tc}$ be the $t\text{-conorm}$ used for *Sim_{tc}* (e.g., \perp_{max} or \perp_{sum})⁴
- return $t\text{-conorm}_{tc}\{Sim_{st}(x, l_i) \times w_i \mid 1 \leq i \leq k\}$.

Function *SimilarConcepts*(c, n, s), where c is a concept, n is the maximal number of expected concepts similar to c , and s is a threshold for similarity, may be defined as shown on page 22.

3.3 Computing *ObjectRank* and *RelatedConcepts* of Objects

Recall that *ObjectRank* and *RelatedConcepts* of objects are information to be added to ADR2 to obtain ADR3. To compute *ObjectRank* values of publications, authors and institutions we create a graph consisting of those objects as vertices, with edges specified as follows:

⁴ $\perp_{max}(a, b) = \max(a, b)$ and $\perp_{sum}(a, b) = a + b - a \cdot b$, for $a, b \in [0, 1]$.

Function. SimilarConcepts(c, n, s)

```

let  $t$ -conorm-SimCon be the  $t$ -conorm used for SimilarConcepts;
 $rs := \{(c, 1)\}$ ,  $min := 1$ ,  $count := 1$ ,  $limit := s$ ,  $used := \emptyset$ ;
while ( $rs \setminus used$ )  $\neq \emptyset$  do
  let  $(c_1, w_1)$  be a pair from  $rs \setminus used$  with the largest  $w_1$ ;
  add  $(c_1, w_1)$  to  $used$ ;
  foreach tuple  $(c_1, c_2, -, w_2)$  in table CONC-CONC do
     $w'_2 := w_1 \times w_2$ ;
    if  $w'_2 \geq limit$  and  $(count < n$  or  $w'_2 > min)$  then
      if there exists  $(c_2, w''_2) \in rs$  then
        | replace  $(c_2, w''_2)$  in  $rs$  by  $(c_2, t$ -conorm-SimCon( $w'_2, w''_2$ ))
      else if  $count < n$  then
        | add  $(c_2, w'_2)$  to  $rs$  and increase  $count$  by 1
      else
        | find a pair  $(c_3, min) \in rs$ ;
        | replace  $(c_3, min)$  in  $rs$  by  $(c_2, w'_2)$ 
        update the variable  $min$  to  $min\{w \mid (-, w) \in rs\}$ ;
      if  $count = n$  then  $limit := min$ 
  return  $rs$ 

```

- for each publication x :
 - for each publication x' cited by x , add the edge (x, x') to the graph
 - for each author y of x , add the edges (x, y) and (y, x) to the graph
 - for each institution z declared in x as an affiliation of an author of x , add the edge (x, z) to the graph
- for each author y and each current institution z of y , add the edges (y, z) and (z, y) to the graph.

Having that graph constructed, *ObjectRank* values of the objects can be computed as *PageRank* values of Web pages, treating edges of the graph as links between Web pages. Some modifications can be investigated, e.g., assigning weights to the edges, starting with different *ObjectRank* values for different objects, or adding other kinds of edge/vertex to the graph.

Values *RelatedConcepts* of objects can be computed in the following order:

1. values *RelatedConcepts* of journals and conferences
2. preliminary values *RelatedConcepts* of publications, computed without using citations and information about authors
3. preliminary values *RelatedConcepts* of authors
4. refined values *RelatedConcepts* of publications and authors.

Consider computing *RelatedConcepts* for a journal/conference x which is described by a list of topics (t_1, \dots, t_k) , where each t_i is a term (already normalized). When such a description is not available, use the name of x as a “topic”. It can be assumed that each t_i corresponds to a concept, or at most two concepts. Here, we can ignore the limits *MinConSimForDoc* and *MaxConceptsForDoc*. *RelatedConcepts* of x may be computed as follows, where

ParamRC1, *ParamRC2*, *ParamRC3* are parameters (e.g., with values 100, 0.4, 0.8, respectively):

- $rs := \emptyset$
- for each $1 \leq i \leq k$:
 - choose $y_i \in PossibleConcepts(t_i)$ with a maximal value $Sim_{tc}(t_i, y_i)$
 - add the pair $(y_i, Sim_{tc}(t_i, y_i))$ to rs
- for each $1 \leq i \leq k$:
 - let $C := \{c \mid (c, _) \in SimilarConcepts(y_i, ParamRC1, ParamRC2)\}$
 - choose $z_i \in PossibleConcepts(t_i) \setminus C$ with a maximal value $Sim_{tc}(t_i, z_i)$
 - if $Sim_{tc}(t_i, z_i) \geq Sim_{tc}(t_i, y_i) \times ParamRC3$ or
 $(Sim_{tc}(t_i, z_i) \geq Sim_{tc}(t_i, y_i) \times ParamRC2$ and z_i is “supported” by rs)
 then add the pair $(z_i, Sim_{tc}(t_i, z_i))$ to rs
- return rs .

Recall that a publication in ADR1 form has a tree-like structure, where each leaf can be specified as a pair (a, d) with a being the sequence of component attributes forming the path from the root to the leaf and d being the value of the leaf (i.e., the value of the last attribute in the sequence). One can treat a as a complex attribute. Let $WeightForRC[a]$ be a parameter standing for the weight of a for computing values $RelatedConcepts$ of publications. Different complex concepts may have different weights. For example, an explicit keyword or a term in the title should have a higher weight than a term occurring in the text of a section.

Let x be a publication in ADR2 form. A preliminary value $RelatedConcepts$ for x can be computed by function `RelatedConceptsOfPub(x)` given on page 24, where `UsingSupportsForRC` is a Boolean parameter, `ParamRC4` is a parameter of type natural number, `ParamRC5` and `ParamRC6` are parameters with values in $(0, 1]$. Exemplary values for these parameters are `ParamRC4` = 100, `ParamRC5` = 0.1 and `ParamRC6` = 0.6. A term t in a text d of a complex attribute a may corresponds to a concept c . If w_1 is the *tf-idf* measure of t in d w.r.t. `TERMS`, $w_2 = Sim_{tc}(t, c)$ and w_3 is the weight of a , then the weight of c for x is increased by $w_1 \times w_2 \times w_3$. The function `RelatedConceptsOfPub` accumulates weights for each possible related concept. In the case the flag `UsingSupportsForRC` is set on, “support” for each related concept c is computed, based on its similarity to the other related concepts, and is used to modify the weight of c . After that, concepts with too small weights are eliminated, and weights of the remaining concepts are rescaled to fit into the interval $(0, 1]$.

In the case values of $WeightForRC[_]$ are normalized to the interval $(0, 1]$, one can use a *t-conorm* denoted by `t-conorm-RCP` to modify function `RelatedConceptsOfPub` as follows (which may result in a worse function):

- replace $w' + WeightForRC[a_i] \times w$ in line 2 by

$$t\text{-conorm-}RC_P(w', WeightForRC[a_i] \times w)$$

- replace $w' + w''$ in line 2 by `t-conorm-RCP(w', w'')`
- delete line 2 (i.e., rescaling is not needed anymore).

Function. RelatedConceptsOfPub(x)

Input: a publication x in ADR2 form (specified as a document ID).

Output: a set of weighted concepts which are the most related to the topic of x .

```

1 let all the text leaves of the tree-representation of  $x$  be  $(a_1, d_1), \dots, (a_n, d_n)$ , not
  counting the terms used for extending ADR1 to ADR2;
2  $rs := \emptyset$ ;
3 foreach  $1 \leq i \leq n$  do
4    $rs_i := \emptyset$ ;
5   let  $(t_1, w_1), \dots, (t_k, w_k)$  be the list of pairs (term, frequency) associated with
      $d_i$  in the ADR2 of  $x$ ;
6   foreach  $1 \leq j \leq k$  do
7     foreach  $c \in PossibleConcepts(t_j)$  do
8        $\lfloor$  add the pair  $(c, w_j \times idf(t_j) \times Sim_{tc}(t_j, c))$  to  $rs_i$ 
9   delete from  $rs_i$  pairs  $(c, w)$  with “too low”  $w$ ;
10  foreach  $(c, w) \in rs_i$  do
11    if there exists  $(c, w') \in rs$  then
12       $\lfloor$  replace  $(c, w')$  in  $rs$  by  $(c, w' + WeightForRC[a_i] \times w)$ 
13    else add the pair  $(c, WeightForRC[a_i] \times w)$  to  $rs$ 
14 delete from  $rs$  pairs  $(c, w)$  with “too low”  $w$ ;
15 if UsingSupportsForRC then
16    $supports := rs$ ;
17   foreach  $(c, w) \in rs$  do
18     foreach  $(c', w') \in SimilarConcepts(c, ParamRC4, ParamRC5)$  do
19       if there exists  $(c', w'') \in supports$  then
20          $\lfloor$  replace  $(c', w'')$  in  $supports$  by  $(c', w' + w'')$ 
21   foreach  $(c, w') \in supports$  do
22      $\lfloor$  let  $w$  be the value such that  $(c, w) \in rs$ ;
23      $\lfloor$  replace  $(c, w)$  in  $rs$  by  $(c, w \times ParamRC6 + w' \times (1 - ParamRC6))$ 
24 keep in  $rs$  only MaxConceptsForDoc pairs  $(c, w)$  with the highest  $w$ ;
25 rescale values of the second components of pairs from  $rs$  to the interval  $(0, 1]$  in
   an appropriate way (to reflect similarities of the corresponding concepts to the
   topic of the publication  $x$ );
26 delete from  $rs$  pairs  $(c, w)$  with  $w < MinConSimForDoc$ ;
27 return  $rs$ 

```

Having preliminary values *RelatedConcepts* of publications computed, preliminary values *RelatedConcepts* of authors representing their research areas can be computed next. We can assume that the topics of publications of an author x are more concrete than the main research areas of x . A preliminary value *RelatedConcepts* for an author x can be computed by function *ResearchAreasOfAuthor*(x) given on page 25, where *ParamRC7* is a parameter of type natural number and *ParamRC8* is a parameter with value in $(0, 1]$. Exemplary values for these parameters are *ParamRC7* = 100 and *ParamRC8* = 0.4.

Function. $\text{ResearchAreasOfAuthor}(x)$

```

1 let  $y_1, \dots, y_n$  be the publications of  $x$ , already in ADR3 form;
2  $rs := \emptyset$ ;
3 foreach  $1 \leq i \leq n$  do
4   foreach  $(c, w) \in \text{RelatedConcepts}(y_i)$  do
5     foreach  $(c', w') \in \text{SimilarConcepts}(c, \text{ParamRC7}, \text{ParamRC8})$  do
6       if there exists  $(c'', w'') \in rs$  then
7         | replace  $(c'', w'')$  in  $rs$  by  $(c', w'' + w \times w')$ 
8       else add  $(c', w \times w')$  to  $rs$ 
9 keep in  $rs$  only  $\text{MaxConceptsForDoc}$  pairs  $(c, w)$  with the highest  $w$ ;
10 rescale values of the second components of pairs from  $rs$  to the interval  $(0, 1]$  in
    an appropriate way (to reflect similarities of the corresponding concepts to the
    research areas of the author  $x$ );
11 delete from  $rs$  pairs  $(c, w)$  with  $w < \text{MinConSimForDoc}$ ;
12 return  $rs$ 

```

Function. $\text{Sim}_{so}^{syn}(x, y)$

Input: a string x and an object y .

Output: a value in $[0, 1]$ representing similarity between x and y .

```

1 let the set of normalized words of  $x$  be  $\{u_1, \dots, u_m\}$ ;
2  $sum := 0, sum_2 := 0$ ;
3 foreach  $1 \leq i \leq m$  do
4    $sum := sum + \text{idf}(u_i)$ ;
5   if  $u_i \in \text{words}(y)$  then  $sum_2 := sum_2 + \text{idf}(u_i)$ 
6 let the set of terms of  $x$  be  $\{t_1, \dots, t_n\}$ ;
7  $sum' := 0, sum'_2 := 0$ ;
8 foreach  $1 \leq i \leq n$  do
9    $sum' := sum' + \text{idf}(t_i)$ ;
10  if  $t_i \in \text{terms}(y)$  then  $sum'_2 := sum'_2 + \text{idf}(t_i)$ 
11 return  $(sum_2/sum) \times \text{ParamSim1} + (sum'_2/sum') \times (1 - \text{ParamSim1})$ 

```

One can also use a t -conorm denoted by $t\text{-conorm-}RC_A$ to modify function $\text{ResearchAreasOfAuthor}$ as follows (which may result in a worse function):

- replace $w'' + w \times w'$ in line 3 by $t\text{-conorm-}RC_A(w'', w \times w')$
- delete line 3 (i.e., rescaling is not needed anymore).

In the next stage, values RelatedConcepts of publications can be refined by taking into account also preliminary values RelatedConcepts of the cited papers and the authors. After that, values RelatedConcepts of authors can also be refined by taking into account also refined values RelatedConcepts of their publications. We do not go into details, but want to emphasize that weights of topics of cited papers should be small enough unless it is known which cited papers are most related to the considered publication.

3.4 Computing Other Similarity Functions

Function $Sim_{so}^{syn}(x, y)$, where x is a string and y is an object, may be defined as shown on page 25, where parameter $ParamSim1$ has a value in $(0, 1]$, e.g., 0.7.

Function. $Sim_{so}^{sem}(x, y)$

Input: a string x and an object y .

Output: a value in $[0, 1]$ representing similarity between x and y .

let t -conorm- Sim_{so}^{sem} be the t -conorm used for Sim_{so}^{sem} ;

$rs := 0$;

let $Terms(x, ParamSim2) = ((t_1, w_1), \dots, (t_n, w_n))$;

foreach $1 \leq i \leq n$ **do**

foreach $c \in PossibleConcepts(t_i)$ **do**

foreach $(c', w) \in SimilarConcepts(c, ParamSim3, ParamSim4)$ **do**

if there exists $(c', w') \in RelatedConcepts(y)$ **then**

$rs := t$ -conorm- $Sim_{so}^{sem}(rs, Sim_{st}(x, t_i) \times Sim_{tc}(t_i, c) \times w \times w')$

return rs

Function. $Sim_{co}(x, y)$

Input: a concept x and an object y .

Output: a value in $[0, 1]$ representing similarity between x and y .

let t -conorm- Sim_{co} be the t -conorm used for Sim_{co} ;

$rs := 0$;

foreach $(c, w) \in SimilarConcepts(x, ParamSim3, ParamSim4)$ **do**

if there exists $(c, w') \in RelatedConcepts(y)$ **then**

$rs := t$ -conorm- $Sim_{co}(rs, w \times w')$

return rs

Function. $Sim_{oo}(x, y)$

Input: objects x and y .

Output: a value in $[0, 1]$ representing similarity of y to x .

let t -conorm- Sim_{oo} be the t -conorm used for Sim_{oo} ;

$concepts := \emptyset$;

foreach $(c, w) \in RelatedConcepts(x)$ **do**

foreach $(c', w') \in SimilarConcepts(c, ParamSim3, ParamSim4)$ **do**

if there exists $(c', w'') \in concepts$ **then**

 replace (c', w'') in $concepts$ by $(c', t$ -conorm- $Sim_{oo}(w'', w \times w'))$

else add $(c', w \times w')$ to $concepts$

$sum := 0, sum_2 := 0$;

foreach $(c, w) \in concepts$ **do**

$sum := sum + w$;

if there exists $(c, w') \in RelatedConcepts(y)$ **then**

$sum_2 := sum_2 + \min(w, w')$

return sum_2 / sum

Function $Sim_{so}^{sem}(x, y)$, where x is a string and y is an object, may be defined as shown on page 26, where parameter $ParamSim2$ is a *tf-idf* threshold, parameter $ParamSim3$ is of type natural number, and parameter $ParamSim4$ has a value in $(0, 1]$. Exemplary values for $ParamSim3$ and $ParamSim4$ are 100 and 0.4, respectively.

Function $Sim_{co}(x, y)$, where x is a concept and y is an object, may be defined as shown on page 26.

Function $Sim_{oo}(x, y)$, which measures similarity of an object y to an object x , may be defined as shown on page 26. This function first extends the list of weighted concepts related with x , and then checks similarity of y to that list of concepts.

4 On the User Interface for SONCA

As mentioned before, the SONCA system enables combination of metadata-based search, syntactic keyword-based search and semantic search. A simple search may depend only on basic metadata. Most existing websites for searching publications use such kind of search, including:

- SCOPUS Search 21
- Advanced Search - Scirus
- Google Advanced Scholar Search
- CiteSeerX Advanced Search
- Advanced Product Search - Elsevier
- arXiv.org Search
- Biomedical Search 22.

Among the above websites the interface of “SCOPUS Search” seems most flexible. It allows also “author search” and “affiliation search”. Note that syntactic keyword-based search is used for the attribute “abstract” in some of the mentioned websites. The website “Google Advanced Scholar Search” puts more emphasis on syntactic keyword-based search, but still uses some metadata.

A simple user interface for SONCA may use fields for typing in:

- conditions on basic metadata
- a list or a Boolean expression of syntactic keywords
- a list or a Boolean expression of semantic keywords.

Basic metadata may include, e.g., the attributes used for “SCOPUS Search”. We may allow also attributes like *ObjectRank*, the number of citations of a publication, or the H-index of an author. Using the simple interface, the system decides itself what weights to associate with the keywords, how to combine the conditions on fuzzy matching, and how to weight the results.

An advanced user interface for SONCA should allow sophisticated ways for:

- specifying conditions on metadata
- providing a list of weighted syntactic keywords, possibly for some concrete long-text attributes or for the whole queried document

- specifying conditions on fuzzy matching using semantic keywords, concepts and objects
- specifying a method for combining the conditions and weighting the results.

Consider the first item of the above list. Here are some proposals:

- We can define a user logical data model, based on the relational database model of SONCA, by using appropriate views and hiding or blocking access to certain information. We can then allow the user to write SQL-like queries using that data model.
- Alternatively, we can allow the user to construct description-logic-like queries by using concept names, role names and constructors from a certain list.
- We can allow the user to query an external system (e.g., DBpedia by using SPARQL) and use the returned results in specifying a query to the SONCA system. We can also allow using search results returned by SONCA to specify another query to SONCA.
- We can allow the user to specify and use additional predicates by some recursive rule language, e.g., semipositive Datalog⁺ with the standard semantics.

The idea of querying an external system like DBpedia can also be adopted for specifying keywords or concepts for a query to SONCA.

The part involving semantic search in a query may consist of three components $\Phi_{SemPhrases}$, $\Phi_{Concepts}$ and $\Phi_{SimObjects}$. By preprocessing the query and interacting with the user, the interface module may convert that part to a list or an expression of only weighted concepts.

The interface module may help the user in constructing or refining a query by using the main ontology. The approach of the DOPE Browser [23] for navigating and refining queries is interesting and can be adopted for the SONCA system.

5 On Efficient Computation

There are two kinds of computation: pre-computation, e.g., for computing *RelatedConcepts* and *ObjectRank* of objects; and online computation, e.g., for answering queries. Accuracy seems more important for pre-computation, while efficiency seems more important for online computation. In this section, we present some proposals for increasing efficiency of the query answering process.

Consider the case when the part involving semantic search of a query consists of a set of weighted phrases, a set of weighted concepts, and a set of weighted objects (used for checking similarity). That part can be converted to a set of weighted concepts, e.g., by function *ConvertSemSubquery* given on page [29]. The conversion result can be presented to the user and improved by him/her.

Having a set of weighted concepts for matching with documents, we first extend it by function *ExtendConcepts* given on page [29] and then match concepts of the resulting set directly with the documents by function *DirectSemMatching* given on page [30].

Our first proposal is to preprocess the part involving semantic search of a query by using functions *ConvertSemSubquery* and *ExtendConcepts* (or similar

Function. ConvertSemSubquery($\Phi_{SemPhrases}, \Phi_{Concepts}, \Phi_{SimObjects}$)

Input: $\Phi_{SemPhrases} = \{(s_1, u_1), \dots, (s_h, u_h)\}$, $\Phi_{Concepts} = \{(c_1, v_1), \dots, (c_k, v_k)\}$, $\Phi_{SimObjects} = \{(o_1, w_1), \dots, (o_l, w_l)\}$, where each s_i is a string, each c_i is a concept, each o_i is an object, and each u_i (resp. v_i, w_i) is the weight of s_i (resp. c_i, o_i).

Output: a set of weighted concepts.

```

1 let t-conorm-CSS be the t-conorm used for this function;
2 concepts :=  $\{(c_1, v_1), \dots, (c_k, v_k)\}$ ;
3 foreach  $1 \leq i \leq h$  do
4   let Terms( $s_i, ParamSim2$ ) =  $((t_1, x_1), \dots, (t_n, x_n))$  and  $sum = \sum_{i=1}^n x_i$ ;
5   foreach  $1 \leq j \leq n$  do
6     foreach  $c \in PossibleConcepts(t_j)$  do
7       if there exists  $(c, v) \in concepts$  then
8         replace  $(c, v)$  in concepts by
9          $(c, t\text{-conorm-CSS}(v, u_i \times (x_j / sum) \times Sim_{st}(s_i, t_j) \times Sim_{tc}(t_j, c)))$ 
10        else add  $(c, u_i \times (x_j / sum) \times Sim_{st}(s_i, t_j) \times Sim_{tc}(t_j, c))$  to concepts
11 foreach  $1 \leq i \leq l$  do
12   let  $sum = \Sigma \{v \mid \text{there exists } (c, v) \in RelatedConcepts(o_i)\}$ ;
13   foreach  $(c, v) \in RelatedConcepts(o_i)$  do
14     if there exists  $(c, v') \in concepts$  then
15       replace  $(c, v')$  in concepts by  $(c, t\text{-conorm-CSS}(v', w_i \times v / sum))$ 
16       else add  $(c, w_i \times v / sum)$  to concepts
16 return concepts

```

Function. ExtendConcepts(*concepts*)

Input: a set $concepts = \{(c_1, w_1), \dots, (c_k, w_k)\}$, where each c_i is a concept and w_i is the weight of c_i .

Output: another set of weighted concepts.

```

1 let t-conorm-EC be the t-conorm used for this function;
2 rs :=  $\emptyset$ ;
3 foreach  $1 \leq i \leq k$  do
4   foreach  $(c, w) \in SimilarConcepts(c_i, ParamSim3, ParamSim4)$  do
5     if there exists  $(c, w') \in rs$  then
6       replace  $(c, w')$  in rs by  $(c, t\text{-conorm-EC}(w', w_i \times w))$ 
7       else add  $(c, w_i \times w)$  to rs
8 return rs

```

ones) and then apply function *DirectSemMatching* (or a similar one) to score documents from a given set X w.r.t. that part of the query. This replaces the functions ms_s^{sem} , ms_c , ms_o and requires a change for the function ms proposed in Section 2.3. The point is that preprocessing the query does not depend on documents and can be done efficiently, while direct matching can also be done efficiently using set-oriented operations.

Function. DirectSemMatching($concepts, X$)

Input: a set $concepts = \{(c_1, w_1), \dots, (c_k, w_k)\}$, where each c_i is a concept and w_i is the weight of c_i ; and a set X of objects

Output: a set of pairs (x, ms) , where $x \in X$ and ms is a matching score for x w.r.t. $concepts$

```

1   $rs := \emptyset$ ;
2   $sum := w_1 + \dots + w_k$ ;
3  foreach  $x \in X$  do
4  |    $sum_2 := 0$ ;
5  |   foreach  $1 \leq i \leq k$  do
6  |   |   if there exists  $(c_i, w'_i) \in RelatedConcepts(x)$  then
7  |   |   |    $sum_2 := sum_2 + \min(w_i, w'_i)$ 
8  |   |   add  $(x, sum_2/sum)$  to  $rs$ 
9  return  $rs$ 

```

Given a query $\Phi = (\Phi_{Metadata}, \Phi_{SynPhrases}, \Phi_{SemPhrases}, \Phi_{Concepts}, \Phi_{SimObjects}, \Phi_{weighting})$, we can apply $\Phi_{Metadata}$ first to restrict the set of possible results to a set X of objects. If the component $\Phi_{Metadata}$ is empty or the returned set X is still too big, it is desirable to restrict the set further before scoring the objects using the other components of Φ . For this aim one can adopt the restriction that each result of the search must be an object containing a term specified by $\Phi_{SynPhrases}$. If the set is still too big, a further restriction may be the assumption that each result of the search must be an object “related” to a concept specified by $\Phi_{SemPhrases}$, $\Phi_{Concepts}$ or $\Phi_{SimObjects}$ (e.g., a concept specified by $ExtendConcepts(ConvertSemSubquery(\Phi_{SemPhrases}, \Phi_{Concepts}, \Phi_{SimObjects}))$).

6 Conclusions

We have designed the SONCA system using the query-oriented approach. Our design allows combination of metadata-based search, syntactic keyword-based search and semantic search, and the use of *ObjectRank*. In our approach, semantic search is based on measuring similarity between terms, concepts and documents. Our proposals have carefully been chosen to guarantee practicability of the system. However, implementation and evaluation were not undertaken, and it is highly probable that the design, in particular, the detailed definitions of functions, can be significantly improved.

Acknowledgements. This work was supported by the National Centre for Research and Development (NCBiR) under Grant No. SP/I/1/77065/10 by the strategic scientific research and experimental development program: “Interdisciplinary System for Interactive Scientific and Scientific-Technical Information”.

References

1. Baader, F., Brandt, S., Lutz, C.: Pushing the EL envelope. In: Proc. IJCAI 2005, pp. 364–369. Morgan-Kaufmann Publishers (2005)
2. Baader, F., Brandt, S., Lutz, C.: Pushing the EL envelope further. In: Proc. of the OWLED 2008 DC Workshop on OWL: Experiences and Directions (2008)
3. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Tractable reasoning and efficient query answering in description logics: The L-Lite family. *J. Autom. Reasoning* 39(3), 385–429 (2007)
4. Drabent, W., Małuszyński, J.: Well-founded semantics for hybrid rules. In: Marchiori, M., Pan, J.Z., de Sainte Marie, C. (eds.) RR 2007. LNCS, vol. 4524, pp. 1–15. Springer, Heidelberg (2007)
5. Eiter, T., Ianni, G., Lukasiewicz, T., Schindlauer, R.: Well-founded semantics for description logic programs in the Semantic Web. *ACM Trans. Comput. Log.* 12(2), 11 (2011)
6. Glimm, B., Horrocks, I., Motik, B.: Optimized description logic reasoning via core blocking. In: Giesl, J., Hähnle, R. (eds.) IJCAR 2010. LNCS, vol. 6173, pp. 457–471. Springer, Heidelberg (2010)
7. Goré, R., Nguyen, L.A.: EXPTIME tableaux with global caching for description logics with transitive roles, inverse roles and role hierarchies. In: Olivetti, N. (ed.) TABLEAUX 2007. LNCS (LNAI), vol. 4548, pp. 133–148. Springer, Heidelberg (2007)
8. Goré, R., Widmann, F.: Sound global state caching for *ALC* with inverse roles. In: Giese, M., Waaler, A. (eds.) TABLEAUX 2009. LNCS (LNAI), vol. 5607, pp. 205–219. Springer, Heidelberg (2009)
9. Grosz, B.N., Horrocks, I., Volz, R., Decker, S.: Description logic programs: combining logic programs with description logic. In: Proc. WWW 2003, pp. 48–57 (2003)
10. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible *SROIQ*. In: Proc. KR 2006, pp. 57–67. AAAI Press (2006)
11. Horrocks, I., Sattler, U.: A tableau decision procedure for *SHOIQ*. *J. Autom. Reasoning* 39(3), 249–276 (2007)
12. Horrocks, I., Sattler, U., Tobies, S.: Practical reasoning for very expressive description logics. *Logic Journal of the IGPL* 8(3) (2000)
13. <http://www.synat.pl>
14. <http://www.w3.org/TR/owl-guide/>
15. <http://www.w3.org/TR/owl2-overview/>
16. <http://wiki.dbpedia.org/About>
17. <http://wordnet.princeton.edu/>
18. <http://plwordnet.pwr.wroc.pl/wordnet/>
19. <http://en.wikipedia.org/wiki/Tf-idf>
20. <http://pl.wikipedia.org/wiki/PageRank>
21. <http://www.scopus.com/search/form.url>
22. <http://www.biomedsearch.com/search.html>
23. <http://www.w3.org/2001/sw/swec/public/UseCases/Elsevier/>
24. <http://www.w3.org/TR/skos-reference/>
25. Hustadt, U., Motik, B., Sattler, U.: Reasoning in description logics by a reduction to disjunctive Datalog. *J. Autom. Reasoning* 39(3), 351–384 (2007)
26. Knorr, M., Alferes, J.J., Hitzler, P.: A coherent well-founded model for hybrid MKNF knowledge bases. In: Proc. ECAI 2008, Frontiers in Artificial Intelligence and Applications, vol. 178, pp. 99–103. IOS Press (2008)

27. Levy, A.Y., Rousset, M.-C.: Combining Horn rules and description logics in CARIN. *Artif. Intell.* 104(1-2), 165–209 (1998)
28. Mishra, R.B., Kumar, S.: Semantic web reasoners and languages. *Artif. Intell. Rev.* 35(4), 339–368 (2011)
29. Motik, B., Rosati, R.: Reconciling description logics and rules. *J. ACM* 57(5) (2010)
30. Motik, B., Sattler, U.: A comparison of reasoning techniques for querying large description logic ABoxes. In: Hermann, M., Voronkov, A. (eds.) *LPAR 2006. LNCS (LNAI)*, vol. 4246, pp. 227–241. Springer, Heidelberg (2006)
31. Nguyen, L.A.: A bottom-up method for the deterministic horn fragment of the description logic \mathcal{ALC} . In: Fisher, M., van der Hoek, W., Konev, B., Lisitsa, A. (eds.) *JELIA 2006. LNCS (LNAI)*, vol. 4160, pp. 346–358. Springer, Heidelberg (2006)
32. Nguyen, L.A.: An efficient tableau prover using global caching for the description logic \mathcal{ALC} . *Fundam. Inform.* 93(1-3), 273–288 (2009)
33. Nguyen, L.A.: Horn knowledge bases in regular description logics with PTime data complexity. *Fundam. Inform.* 104(4), 349–384 (2010)
34. Nguyen, L.A.: Paraconsistent and approximate semantics for the OWL 2 web ontology language. In: Szczuka, M., Kryszkiewicz, M., Ramanna, S., Jensen, R., Hu, Q. (eds.) *RSCTC 2010. LNCS*, vol. 6086, pp. 710–720. Springer, Heidelberg (2010)
35. Nguyen, L.A.: A cut-free exptime tableau decision procedure for the logic extending converse-PDL with regular inclusion axioms. *CoRR*, abs/1104.0405 (2011)
36. Nguyen, L.A.: Cut-free EXPTIME tableaux for checking satisfiability of a knowledge base in the description logic \mathcal{ALCZ} . In: Kryszkiewicz, M., Rybinski, H., Skowron, A., Raś, Z.W. (eds.) *ISMIS 2011. LNCS*, vol. 6804, pp. 465–475. Springer, Heidelberg (2011)
37. Nguyen, L.A.: A cut-free expTime tableau decision procedure for the description logic SHI. In: Jędrzejowicz, P., Nguyen, N.T., Hoang, K. (eds.) *ICCCI 2011, Part I. LNCS(LNAI)*, vol. 6922, pp. 572–581. Springer, Heidelberg (2011), <http://arxiv.org/abs/1106.2305>
38. Nguyen, L.A., Szałas, A.: EXPTIME tableaux for checking satisfiability of a knowledge base in the description logic \mathcal{ALC} . In: Nguyen, N.T., Kowalczyk, R., Chen, S.-M. (eds.) *ICCCI 2009. LNCS (LNAI)*, vol. 5796, pp. 437–448. Springer, Heidelberg (2009)
39. Nguyen, L.A., Szałas, A.: Checking consistency of an ABox w.r.t. global assumptions in PDL. *Fundam. Inform.* 102(1), 97–113 (2010)
40. Nguyen, L.A., Szałas, A.: Tableaux with global caching for checking satisfiability of a knowledge base in the description logic SH. *T. Computational Collective Intelligence* 1, 21–38 (2010)
41. Nguyen, L.A., Szałas, A.: Three-valued paraconsistent reasoning for semantic web agents. In: Jędrzejowicz, P., Nguyen, N.T., Howlet, R.J., Jain, L.C. (eds.) *KES-AMSTA 2010. LNCS*, vol. 6070, pp. 152–162. Springer, Heidelberg (2010)
42. Ortiz, M., Rudolph, S., Simkus, M.: Worst-case optimal reasoning for the Horn-DL fragments of OWL 1 and 2. In: *Proc. KR 2010. AAAI Press* (2010)

A Appendix: On DBpedia

DBpedia is a very useful source of structured information. It allows us to ask sophisticated queries against Wikipedia, and to link other data sets on the Web to Wikipedia data. DBpedia can be used as a main ontological source for SONCA, as it contains a large multi-domain ontology, links to Wikipedia pages and connections to other information sources like WordNet.

The DBpedia data set describes more than 3.5 million “things” with over half a billion “facts”. The current version DBpedia 3.6 is available for download at <http://wiki.dbpedia.org/Downloads36>. The most important datasets for SONCA are:

Articles Categories: links from resources to categories using the SKOS vocabulary [24].

Categories (Skos): information about which concept is a category and how categories are related using the SKOS Vocabulary.

Disambiguation Links

Links to YAGO2: YAGO type information for DBpedia resources and the YAGO class hierarchy (the YAGO Classification is derived from the Wikipedia category system using Word Net).

Word Net Classes: classification links to RDF representations of WordNet classes.

The dataset Ontology Infobox Properties is one of the four high-quality “mapping-based” datasets in the /ontology/ namespace of DBpedia, but we can use it as an “external source”. Other useful datasets for SONCA are, for example, Homepages, Persondata, External Links, Redirects, Links to DBLP, Links to Cyc.

Information from datasets like Categories (Labels), Titles, Links to Wikipedia Article can (probably) be computed from identifications of “things”.

The dataset DBpedia Ontology is too small and seems useless for SONCA. For example, it does not contain words/concepts “Computer Science” (in any combination of cases). Consequently, the dataset Ontology Infobox Types, which contains triples of the form \$object rdf:type \$class (concept) is not useful for SONCA either.

Here is some exemplary information from the preview of datasets:

– Articles Categories:

- <http://dbpedia.org/resource/Aristotle>
<http://purl.org/dc/terms/subject>
<http://dbpedia.org/resource/Category:Metaphysicians> .
- <http://dbpedia.org/resource/Aristotle>
<http://purl.org/dc/terms/subject>
http://dbpedia.org/resource/Category:Humor_researchers .
- <http://dbpedia.org/resource/Aristotle>
<http://purl.org/dc/terms/subject>
http://dbpedia.org/resource/Category:History_of_science .

- ... (on “Aristotle”)
 - We have a problem here. Namely, in DBpedia, “concept” means a concept defined in the dataset DBpedia Ontology. As mentioned before, this dataset is too small and rather useless for SONCA. Hence, we have to take “categories” as concepts in a broader sense. But, “Aristotle” is not an instance of “History_of_science”. An alternation is to use the YAGO Classification.
- Categories (Skos):
- <http://dbpedia.org/resource/Category:Futurama> <http://www.w3.org/2004/02/skos/core#broader> http://dbpedia.org/resource/Category:Comic_science_fiction .
 - <http://dbpedia.org/resource/Category:Futurama> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/2004/02/skos/core#Concept> .
 - http://dbpedia.org/resource/Category:World_War_II <http://www.w3.org/2004/02/skos/core#broader> http://dbpedia.org/resource/Category:Wars_involving_Poland .
- Disambiguation Links:
- <http://dbpedia.org/resource/Alien> <http://dbpedia.org/ontology/wikiPageDisambiguates> http://dbpedia.org/resource/Alien_%28law%29 .
 - <http://dbpedia.org/resource/Alien> <http://dbpedia.org/ontology/wikiPageDisambiguates> http://dbpedia.org/resource/Alien_%28franchise%29 .
 - <http://dbpedia.org/resource/Alien> <http://dbpedia.org/ontology/wikiPageDisambiguates> http://dbpedia.org/resource/Alien_%28film%29 .
 - Information of this kind is useful for tagging documents with concepts.
- Links to YAGO2:
- <http://dbpedia.org/resource/Aristotle> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://dbpedia.org/class/yago/AncientGreekPhilosophers> .
 - <http://dbpedia.org/resource/Aristotle> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://dbpedia.org/class/yago/4th-centuryBCPhilosophers> .
 - <http://dbpedia.org/resource/Aristotle> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://dbpedia.org/class/yago/HumorResearchers> .
 - ... (on “Aristotle”)
- Word Net Classes:
- http://dbpedia.org/resource/%24_%28film%29 http://dbpedia.org/property/wordnet_type <http://www.w3.org/2006/03/wn/wn20/instances/synset-movie-noun-1> .

- http://dbpedia.org/resource/%26_%28song%29
http://dbpedia.org/property/wordnet_type
http://www.w3.org/2006/03/wn/wn20/instances/synset-phonograph_record-noun-1>.
- http://dbpedia.org/resource/%2703_Bonnie_%26_Clyde
http://dbpedia.org/property/wordnet_type
http://www.w3.org/2006/03/wn/wn20/instances/synset-phonograph_record-noun-1>.
- The reason for using this dataset is that a keyword may belongs to different concepts and WordNet contains information about “frequency counts” that is useful for tagging documents with concepts.

Note that YAGO2 Class and DBpedia Category are different notions. The dataset Links to YAGO2 provides classes together with a good relation “is-instance-of” from objects (resources) to classes. As mentioned before, there is no relation like “is-instance-of” from resources to categories. A similar relation from resources to categories is called “subject”. For example, “Aristotle” has a subject belonging to the category “History_of_science”.

There are the following semantic relations between categories (skos:semanticRelation), given in a hierarchy:

- skos:related
 - skos:relatedMatch
- skos:broaderTransitive
 - skos:broader
 - * skos:broadMatch
- skos:narrowerTransitive
 - skos:narrower
 - * skos:narrowMatch
- skos:mappingRelation
 - skos:closeMatch
 - * skos:exactMatch
 - skos:relatedMatch
 - skos:broadMatch
 - skos:narrowMatch

We can restrict to categories of DBpedia, without using Yago classes. With this assumption, the most important notions and information from DBpedia for SONCA are:

- resource (object) and the relationship resource-category (subject) from the dataset Articles Categories
- the relationships category-category (skos:semanticRelation) from the dataset Categories (Skos)
- information from the datasets Disambiguation Links and Word Net Classes (used for choosing resources/categories and tagging documents with them)
- properties from the dataset Ontology Infobox Properties, which will be used as an additional feature for query languages and user interfaces for SONCA and are not required being stored in the database of SONCA.

Of course, other information from DBpedia (e.g., abstracts) are also useful for SONCA to a certain extent.

Retrieval and Management of Scientific Information from Heterogeneous Sources*

Piotr Gawrysiak, Dominik Ryzko, Przemysław Więch, and Marek Kozłowski

Institute of Computer Science, Warsaw University of Technology,
Nowowiejska 15/19, 00-655 Warsaw, Poland
{p.gawrysiak,d.ryzko,pwiech,m.kozlowski}@ii.pw.edu.pl

Abstract. The paper describes the process of automated retrieval and management of scientific information from various sources including the Internet. Application of semantic methods in different phases of the process is described. The system envisaged in the project is a scientific digital library, with automated retrieval and hosting capabilities. An overall architecture for the system is proposed.

1 Introduction

Rapid advancements in computing and networking technology, that took place during the last two decades, transformed deeply the nature of scientific research worldwide. Nowadays, it is difficult to even imagine conducting a successful research project – both in humanities and in engineering – without exploiting vast knowledge resources provided by the global Internet, and without using the same network to disseminate research results.

The nature of contemporary Internet, used as a research tool, is however drastically different from what was envisioned in the 90-ties. The Internet is just a haphazard collection of non-coordinated knowledge sources. Most valuable repositories are not even centrally controlled. It is sometimes very difficult to evaluate the quality of data contained in non-professional sources, such as some Open Access journals [11]. The situation described above basically means that the concept of Semantic Web [9], promising the coordinated global network of information, failed to materialize. One of the primary reasons for this failure is the difficulty of creating and maintaining useful ontologies, describing all aspects of the world, that would drive exchange of information in the Semantic Web [6], and only such approach would fulfill the needs of a general purpose semantic search engine. The main reason for this is a state of ontology engineering, which is still mostly a manual process, very time-consuming, expensive and error prone. While some automated, or at least semi-automated, ontology building methods that are able to leverage the amount of information present in ever growing repositories of text

* This work is supported by the National Centre for Research and Development (NCBiR) under Grant No. SP/I/1/77065/10 by the strategic scientific research and experimental development program: „Interdisciplinary System for Interactive Scientific and Scientific-Technical Information”.

data (e.g. obtainable via the Internet) have been created [4], their quality is still vastly inadequate.

In this position paper we argue that building some dedicated ontologies, especially as applied to scientific data and scientific communities, may improve the process of data acquisition. We believe that using contemporary knowledge discovery and natural language processing algorithms and methods, we can achieve much of the goals (as seen from an end user perspective) of the Semantic Web vision, which does not seem to be feasible as applied to the whole Internet. In particular, while it is not feasible to create ontologies to manage whole scientific knowledge, we can create domain ontologies for specific branches of science, which will significantly help in managing knowledge and in gathering new information.

The paper extends [5] and describes the design principles of the PASSIM project. It is a part of a broader strategic initiative of the Polish Ministry of Education and Scientific Research called SYNAT, aimed at creation of universal, open communication and hosting platform for scientific resources. Within PASSIM, led by Warsaw University of Technology, an integrated knowledge base will be created, which will enable acquisition of data from heterogeneous and distributed sources, its safe storage and analysis. On top of hosting capabilities, new services and applications will be built. The project has started in 2010 and will last till the end of 2012. The initial phase of the project has finished with a analysis of application of Artificial Intelligence methodologies in the process of data acquisition. The current phase involves development of algorithms supporting this process. Later on selected algorithms will be implemented and tested on real data.

This paper is structured as follows. Section 2 describes challenges targeted by the project and existing research results which can be used to solve them. In Section 3 requirements for the PASSIM project are listed. In Section 4 solutions for system implementation are proposed. Finally, Section 5 summarizes the results.

2 Challenges and Existing Solutions

As mentioned in the introduction, one of the main challenges in PASSIM is automated acquisition of knowledge from various structured and unstructured sources. Among these sources the Internet will play a major role. Despite the overwhelming amount of irrelevant and low quality data, there are several useful resources. This includes researchers' homepages and blogs, homepages of research and open source projects, emerging open access journals, university tutorials, software and hardware documentation, conference and workshop information etc. Finding, evaluating and harvesting such information is a complex task but nevertheless it has to be taken up in order to provide PASSIM users with a wide range of up to date resources regarding science as well as past and ongoing research activities.

Several approaches to harvesting information from the Internet have been proposed in the past. The most popular approach nowadays is the use of general purpose search engines. The improvement in the search quality caused that a vast majority of users say the Internet is a good place to go for getting everyday information [8]. Sites like Google.com, Yahoo.com, Ask.com provide tools for ad-hoc queries based on the keywords and page rankings. This approach, while very helpful on the day to day basis, is

not sufficient to search for large amounts of specialized information. General purpose search engines harvest any type of information regardless of their relevance, which reduces efficiency and quality of the process. Another, even more important drawback for scientists is that they constitute only a tiny fraction of the population generating web traffic and really valuable pages constitute only a fraction of the entire web. Page ranks built by general purpose solutions, suited for general public will not satisfy quality demands of a scientist. One can use Google Scholar, Citeseer or other sites to get more science-oriented search solutions. Although this may work for scientific papers and some other types of resources, still countless potentially valuable resources remain difficult to discover.

Another approach to the problem is web harvesting, based on creating crawlers, which search the Internet for pages related to a predefined subject. This part of information retrieval is done for us if we use search engines. However, if we want to have some influence on the process and impose some constraints on the document selection or the depth of the search, we have to perform the process by ourselves. A special case of web harvesting is focused crawling. This method introduced by Chakrabarti et al. [3] uses some labeled examples of relevant documents, which serve as a starting point in the search for new resources.

The task of retrieving scientific information from the web has already been approached. In [10] it is proposed to use meta-search enhanced focused crawling, which allows to overcome some of the problems of the local search algorithms, which can be trapped in some sub-graph of the Internet.

The main motivation for the work envisaged in the PASSIM project is to create a comprehensive solution for retrieval of scientific information from the heterogeneous resources including the web. This complex task will involve incorporating several techniques and approaches. Search engines can be used to find most popular resources with high ranks, while focused crawling can be responsible for harvesting additional knowledge in the relevant subjects. Additional techniques will have to be used to classify and process discovered resources.

Since many users will use the system simultaneously, a distributed architecture will be required. While this has several benefits regarding system performance, additional measures have to be taken in order to avoid overlap in the search process [2]. Various parallel techniques for searching the web have already been proposed [1]. In the PASSIM project multi-agent paradigms will be used, which propose intelligent, autonomous and proactive agents to solve tasks in a distributed environment.

3 Project Requirements

The system that is the subject of this paper would be a direct result of a distributed research project, funded by the Polish Ministry of Higher Education. The project is part of a larger effort aiming to improve the state of scientific research in Poland, and also in Central and Eastern Europe. One of the main obstacles hindering the growth of scientific research and perhaps even more importantly making the distribution of research results more difficult than necessary is lack of information exchange systems pertaining to these results. In short, the lack of standards and systems supporting storage

of scientific oriented information caused large distribution of repositories, with most of them created in unstructured formats. Even relatively easy to structure information such as bibliographical data is not stored in an easy to process way. These problems are common both for large universities and research institutes. As a result, the visibility of Polish science in the Internet is very poor and particular centers of research are often not aware of the work conducted by others. Obviously as a result scientific collaboration between Polish institutions is relatively rarely implemented in practice. Much more important effect is however a distorted view of Polish science, because the apparent activity of local scientists is much lower than their real effort.

The system to be developed in the PASSIM project is thought to be a heterogeneous repository of data from various structured and unstructured sources. Hosting capabilities will be provided to address issues described above. This will be helpful especially with respect to low funded centers of research, which do not have capabilities to set up extensive repositories. It is also required to acquire data from external sources. This includes large repositories of scientific papers, information about researchers and projects. In short, the project aims to create a backbone for scientific information storage in Poland.

The envisaged system should also be able to retrieve potentially useful unstructured information from the Internet. Blogs, forums, homepages, projects, conference calls, science funding schemes etc. constitute a large fraction of available knowledge scattered across the Web. Using such sources means that acquired data can contain missing information, errors or overlaps. The system should be able to: identify duplicates, merge partly overlapping objects, identify object versions, verify completeness of data objects (e.g. bibliography items), identify key words and proper names etc. This process can be greatly improved by relying on existing repositories (e.g. Geonames, LCSH, VIAF). Also identification of synonyms and homonyms should help with the disambiguation. The same techniques should be applied to interpret user queries, which is described later in the paper.

It is required that the system will be able to perform search for new resources, especially in the areas heavily searched by the end users. The user should also be able to query the system repository but also start an off-line search process in order to discover resources according to specific requirements. Any new findings relevant to a particular user profile should be reported. Once discovered, sources of data have to be monitored in order to track any changes to their contents.

The data harvesting process will involve a feedback loop. A user will be able to rate the relevance of the resources found. This information will be used to improve the search process as well as classification of documents. Based on the feedback, a user profile should be built, to personalize retrieval and presentation of information for a particular scientist.

Information storage capabilities drafted above constitute, however, only a part of the system's capabilities. Apart from just being able to store results of research in the form of publications and experimental data, the system should also support the research process itself, by providing tools for rapid dissemination of partial research efforts, for discovery of institutions or groups with similar research interests or even supporting some computationally intensive applications on the system infrastructure itself. The

system is not being designed as a computational grid, however the distributed nature of storage subsystem means, that it can be also, for some specific use cases, utilized to perform some calculations (such as creating visualizations or statistical and/or knowledge discovery computations).

In a sense, the system's functionality in this context (i.e. not directly related to storage and distribution of bibliographical data) will be similar to the functionality of a social network system. Obviously calling such a system the scientific social network (or even "Facebook for scientists") might be an overstatement. However, the similarities between Facebook and the system are also visible in a way in which its services are exposed to external parties. The system is structured as a set of modules with well-defined functionality and data types that can be interconnected by external institutions by using public system APIs. In this way it is possible to extend the functionality of the system – or rather build other, specialized research support tools basing on the system infrastructure. Such tools could be both of commercial and open nature and possibly in the long run a larger ecosystem of services, supporting the scientific community, could be created.

4 Envisaged Solution

This section describes the approach taken in order to reach the goals of the PASSIM project. This includes overall system architecture, information retrieval techniques, knowledge management and presentation of data to the end users.

4.1 Architecture

The requirements described in the previous chapter indicate an explicit distributed nature of the problems to be addressed. On the data acquisition side, the Internet is a network of loosely connected sources, which can be processed more or less in parallel. On the end user side, each one of them can generate concurrent requests for information. At the same time these parallelisms do not forbid overlap or contradiction. All of the above calls for a highly distributed architecture, with autonomy of its components, yet efficient communication and synchronization of actions between them.

The envisaged approach is based on multi-agent paradigms, which introduce a concept of an intelligent, autonomous and proactive agent. Various agent roles have been identified while analyzing processes to be implemented in the system. An example process of user query answering has been shown in Figure 1.

Personal agents will be responsible for interaction with end users. They will receive queries, preprocess them, pass to the knowledge layer and present results returned from the system. User feedback will also be collected here. Personal agents will store history of user queries and maintain a profile of interests to improve results and proactively inform the user about new relevant resources.

The main data acquisition process will be performed by specialized harvesting agents. Their task will be twofold. Firstly, they will perform a continuous search for new relevant resources. Secondly, they will perform special searches for specific queries or groups of queries. The main task of harvesting agents will be to manage a group of web crawlers to perform the physical acquisition of data.

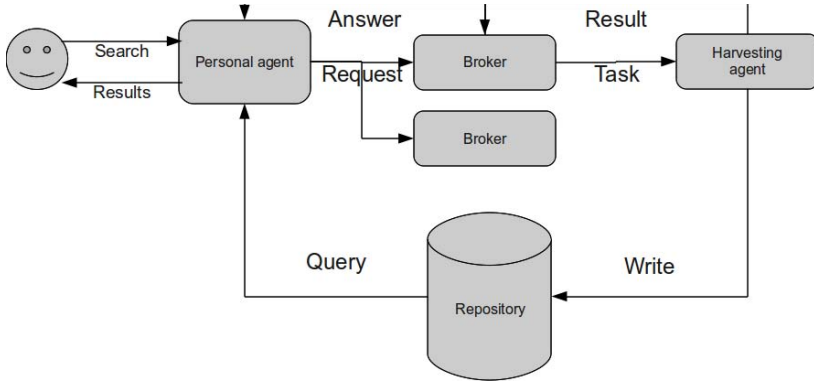


Fig. 1. Query answering process

Different users can generate queries, which return partially overlapping results. This means some search tasks should be merged. On the other hand the same results can be delivered from different sources and only one of them should be used. All this means that matchmaking and coordination between demand and supply of data generates some sophisticated problems. This can be mitigated by introduction of brokering agents (also called middle agents) [7], whose purpose is to coordinate efficient matching between personal agents and harvesting agents.

Special agents should be dedicated to the process of managing data already incorporated into the system. They will be responsible for finding missing data, inconsistencies, duplicates etc. Finding such situations will result in appropriate action e.g. starting a new discovery process to find new information, deletion of some data, marking for review by administrator etc.

The bottom layer of the system will consist of a group of web crawlers. They will search the Internet for relevant resources and pass the data to appropriate agents responsible for its further processing. The crawlers will use various methods (heuristics, machine learning etc.) to perform focused crawling for new documents based on classified examples. The general architecture of the system has been shown in Figure 2.

While the system will be designed as a set of autonomous agents, we can also look at the system architecture from a data flow perspective. Figure 3 shows how information is passed between functional components of the system. The bottom component in this figure is the data *repository*, which holds all gathered information and provides access for reading, updating and searching its contents. The repository is backed up by an ontology defining the semantics of the contained data. As discussed earlier, we assume several domain specific ontologies maintained by communities interested in particular subjects as well as a single ontology describing general concepts (e.g. conference, paper, author etc.). The remainder of the components can either be contained in one or more agents or can be viewed as tasks, which operate inside the system.

One of the processing paths deals with crawling Web resources and inserting pre-processed structured data into the repository. The starting points for the crawler can be provided by the results of a search engine, which is queried with selected keywords. This processing path will be described in detail further in the paper.

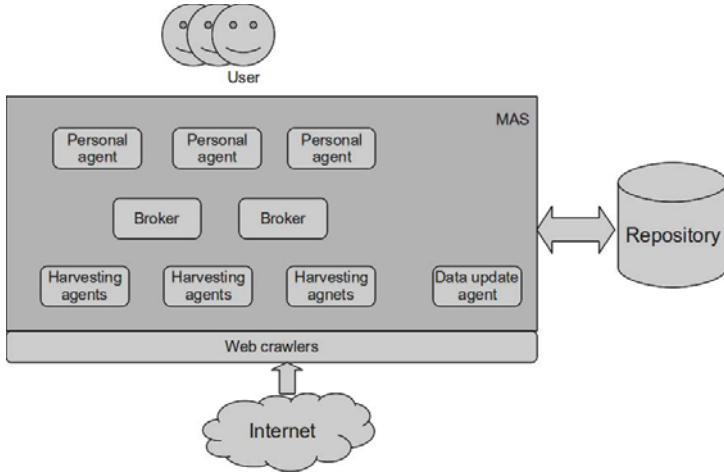


Fig. 2. System architecture

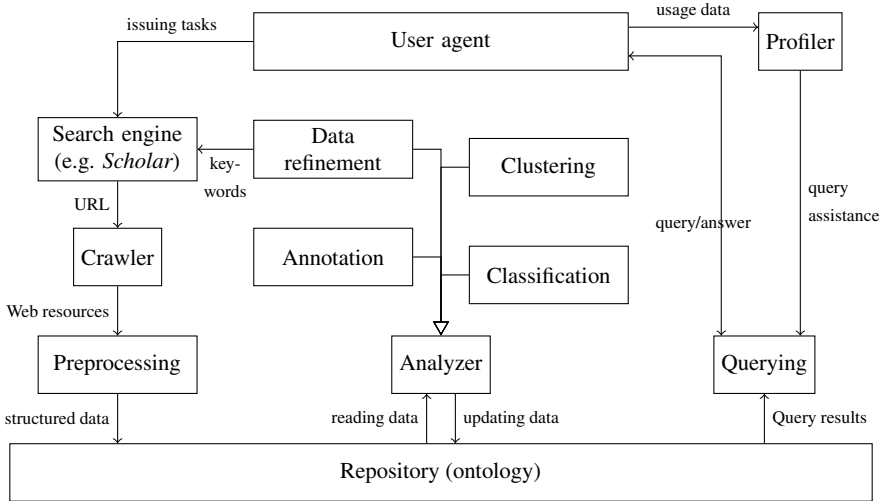


Fig. 3. System data flow

The central part of the diagram presents an architecture for building components, which further process the information that is stored in the data repository. An *analyzer* can execute a processing algorithm such as classification or clustering and based on the results it can add annotations or refine the data, which was input to the data storage. The *data refinement* component is intended to utilize the previously mentioned crawling subsystem to search for additional information about the objects occurring in the data.

The *user agent*, which directly interacts with the user of the system, receives queries and tasks from the user. This agent then queries the data repository for answers or issues tasks to the crawling modules to asynchronously find information relevant for the user. Additionally, the user agent utilizes the *profiler* to collect usage information and personalize the results for the particular user.

4.2 Web Crawling

A focused crawler is a program that traverses the Internet by choosing relevant pages to a predefined topic and neglecting those out of concern. The main purpose of such a program is to harvest more information on the topic that matches the expectation of the user while reducing the number of web pages indexed. A focused crawler has three main components: URL queue (container of unvisited pages), downloader (downloads resources from WWW), classifier (compound model which categorizes the type of information resource, and its domain).

The crawler's classifier includes two modules: extraction module and relevance analysis. The extraction module parses the web page, and identifies the main parts of it. Humans can easily distinguish the main content from navigational text, advertisements, related articles and other text portions. A number of approaches have been introduced to automate this distinction using a combination of heuristic segmentation and features. In PASSIM we would deploy the solution proposed in [13]. In this work there were used combination of two features - number of words and link density. This approach leads to simply classification model that achieves high accuracy. Web pages are segmented into atomic text blocks using html tags. The found blocks are then annotated with features and on this basis classified into content or boilerplate. The features are called shallow text ones. They are higher, domain and language independent level (i.e. average word length, average sentence length, absolute number of words). Atomic text blocks are sequence of characters which are separated by one or more html tags, except for A tags. The presence of headline tag, paragraph, division text tag are used to split content of web page into set of structural elements. To train and test classifier for various feature combinations we would use well known scientific conference, journal page, home pages of scientists, research institutes. The labeled set is then split into training and a test set (using i.e. 10-fold cross validation) and fed into a classifier model (Support Vector Machine).

Relevance analysis uses the significant parts of web resource, which were detected by above described extraction module. It would use intelligent classifier to categorize resource as scientific or not. It is also possible to do first domain classification, and label document to the field of science (i.e. biology, history, computers). The analysis of topic similarity is the most important stage for a topic-specific web crawling. The relevancy can be determined by various techniques like the cosine similarity between vectors, probabilistic classifiers, or BP neural network.

In the article [12] authors used Bayes Classifiers and improved TF-IDF algorithm to extract the characteristics of page content and compute relevance to topic. In the paper [14] was described the design and implementation of a university focused crawler that runs on BP network classifier for prediction of the links leading to relevant pages. BP is a three or multi-level topology structure. In [14] a three layer error back-propagation

neural network is used as a network model for predicting the relevance of the crawled page. Before training there is manually built database of keywords, which are closely connected with topics. The fetched pages are sent to extraction module in order to discover significant text parts, which are further matched with database of keywords, and the outcome is a boolean model. The number of inputs of neural network corresponds to the number of keywords in the database. The number of outputs corresponds to the number of topics. Trained BP network classifier achieves precision about 75%, which is the percentage of the Web pages crawled that are relevant to topics.

During relevance analysis it may be useful to identify type of resource which was positive categorized. Machine learning classifier is trained with features, which includes i.e. hosting domain, non HTML markup words, URL of page, outgoing links. Such model could be enough relevant to classify resource as home-page, institute page, conference, or blog. Type of resource may be used as a filter during invoking searching process.

We have two types of focused crawlers. First is used in harvesting mode to detect all probably scientific resource, which are further processed by middle layer (specific agents). The second type of crawler is used to find resources relevant to natural language query (NL query). Natural query would be provided by user in similar way as we type queries in google searcher. Next, this query would be analyzed in the context of ontology which describe meta-data of conferences, journals, articles, institutes, researchers. Ontology give us ability to make user's query much more semantic and structured. This approach achieves advantage over google like searcher engines. General purpose searchers could not be ontology-driven, because there is impossible to build ontology of whole world.

Web crawlers have URL queue which contains a list of unvisited web resources. In harvesting mode this queue would be initialized with seed URLs. Seed URLs may be built on data taken from DBLP, Citeseer. Those two sources have links to relevant sources, but also meta-data concerning author names, title, publication date. Found urls within DBLP, Citeseer would build initial URL queue. The mentioned meta-data would be entered to google scholar and returned urls could enrich seed entries. The classifier analyzes whether the content of parsed pages is related to topic or not. If the page is relevant, the URLs extracted from it will be added to queue, otherwise will be discarded.

The process of NL query-driven searching is as following:

- We begin with an existing manually-created ontology, describing publications, scientist, conferences etc. From hierarchically-structured ontology, using given natural language query we generate set of alternative queries taking under account concepts and relations in the ontology.
- We use a topic-specific spider(focused crawler) to submit these queries to a variety of Web Search engines and digital libraries. The spider downloads the potentially relevant documents listed on the first page (top-ranked). We also provide options to customize the number of returned results, the formats of returned resources.
- Next crawler applies classifier to filter out documents that match the query but do not belong to scientific world. If in the query is included clear information about domain of interest, focused crawler initially categorizes document to specific domain (i.e. biology, history).

- If the user choose the type of searched resources(only blog, or only home-pages) there is used described above machine learning classifier to filter out irrelevant types.
- Collection of significant resources is processed by information extraction module(IEM). It extracts structured and useful information from the actual text of filtered resources. Simple extraction is used by crawlers to identify outgoing links and add them to crawler's URL query. Much more complex extraction operations (as summarizing) are done in the knowledge system.
- Each relevant resources are automatically described by agents with attributes from ontology. If some information are not presented in the selected document, there is sent query to focused crawlers. In this way we step-by-step fulfill meta-information.

4.3 Information Retrieval

Before becoming available to end users, knowledge has to be discovered and retrieved. As discussed in the Introduction, using general purpose search engines is not an option, while retrieveing specialized information. Therefore, more sophisticated method like focused crawling have to be adopted in order to discover and retrieve valuable data from the Web.

In the search process several classes of resources have to be discovered for various fields of science. Therefore, the data has to be properly classified according to the type of information it represents (e.g. scientific paper, blog, conference homepage etc.). Once we have such classification we can use an ontology to decompose the document into appropriate components. For example, if we deal with a scientific paper, we will expect to find title, authors, affiliation, abstract etc. In the case of a conference website, the ontology will tell us what are the roles related to a scientific conference (general chair, organizing chair, program committee member etc.), what is a special session, a paper and so on. Another dimension for classificaion of resources is the field of science which they belong to.

Classified and decomposed documents will be stored in the system repository. From there they can be accessed by the system users. They will also undergo further processing in order to improve knowldge quality. Duplicates will be eliminated, missing information filled, inconsistencies resolved etc.

4.4 Semantic Analysis

When harvesting a new piece of knowledge from the web the system must know its semantics. Unless it is stated explicitly what is a scientific conference, how is it related to papers, sessions, chairman etc., it is not possible to extract automatically any useful information. To allow this task special ontology will be built called knowledge base ontology. It will define most important concepts and their respective relation. This step will be performed manually or semi-automatically. The knowledge base ontology will be used not only to provide semantics to the documents retrieved from the Internet, but also it will allow better interpretation of user queries.

Another group of ontologies will be constituted of domain ontologies. They will contain knowledge about respective fields of science, which will be covered by the project. These ontologies will be build automatically or semi-automatically out of the knowledge gathered in the system.

5 Conclusions

In the paper the concept of scientific knowledge acquisition from the internet in the PASSIM project has been presented. It has been shown why special approach is needed here and how semantic technologies play a crucial role in the process. The requirements for the system have been listed and problems to be faced have been outlined. The paper describes also envisaged solution and a general architecture of the system to be developed. The most important technologies selected to achieve the task are multi-agent systems and ontologies.

In the next stages of the project specific algorithms will be selected and implemented. It is important to verify system performance and usability across various branches of science and with large amounts of data. It seems obvious results will vary here. Some branches of science like Computer Science generate large amount of structured data and valuable knowledge bases, which can be used as starting points (e.g. DBLP). Other especially social sciences will require much more effort to retrieve significant knowledge. From the point of view of semantic technologies, the important question to be answered is how complex ontologies will be sufficient to allow retrieval of interesting information.

References

1. Bra, P., Post, R.: Searching for arbitrary information in the www: The fish-search for mosaic. In: Second World Wide Web Conference, WWW2 (1999)
2. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. Addison-Wesley Longman Publishing Co., Inc., Boston (1999)
3. Chakrabarti, S., van den Berg, M., Dom, B.: Focused crawling: A new approach to topic-specific web resource discovery. *Computer Networks* 31(11-16), 1623–1640 (1999)
4. Gawrysiak, P., Rybinski, H., Protaziuk, G.: Text-Onto-Miner - a semi automated ontology building system. In: Proceedings of the 17th International Symposium on Intelligent Systems (2008)
5. Gawrysiak, P., Ryżko, D.: Acquisition of scientific information form the Internet - The PASSIM Project Concept. In: Proceedings of the International Workshop on Semantic Interoperability - IWSI (accepted for publishing, 2011)
6. Gomez-Pérez, A., Corcho, O.: Ontology Specification Languages for the Semantic Web. *IEEE Intelligent Systems* 17(1), 54–60 (2002)
7. Klusch, M., Sycara, K.P.: Brokering and matchmaking for coordination of agent societies: A survey. In: *Coordination of Internet Agents: Models, Technologies, and Applications*, pp. 197–224. Springer, Heidelberg (2001)
8. Manning, C.D., Raghavan, P., Schuetze, H.: *An Introduction to Information Retrieval*. Cambridge University Press (2008)
9. McIlraith, S.A., Son, C.T., Zeng, H.: Semantic Web Services. *IEEE Intelligent Systems* 16(2), 46–53 (2001)

10. Qin, J., Zhou, Y., Chau, M.: Building domain-specific web collections for scientific digital libraries: a meta-search enhanced focused crawling method. In: Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries (2004)
11. Suber, P.: Open access overview (2004), <http://www.earlham.edu/peters/fos/overview.htm>
12. Wenxian, W., Xingshu, C., Yongbin, Z., Haizhou, W., Zongkun, D.: A focused crawler based on Naive Bayes Classifier. In: Third International Symposium on Intelligent Information Technology and Security Informatics (2010)
13. Kohlschutter, C., Fankhauser, P., Nejd, W.: Boilerplate detection using shallow text features. In: Proceedings of the Third ACM International Conference on Web Search and Data Mining (2010)
14. Hua, J., Bing, H., Ying, L., Dan, Z., YongXing, G.: Design and Implementation of University Focused Crawler Based on BP Network Classifier In: Second International Workshop on Knowledge Discovery and Data Mining (2009)

RDBMS Model for Scientific Articles Analytics

Marcin Kowalski¹, Dominik Ślęzak^{1,2}, Krzysztof Stencel¹,
Przemysław Pardel^{3,1}, Marek Grzegorowski¹, and Michał Kijowski¹

¹ Faculty of Mathematics, Informatics and Mechanics, University of Warsaw
ul. Banacha 2, 02-097 Warsaw, Poland

² Infobright Inc.

ul. Krzywickiego 34 lok. 219, 02-078 Warsaw, Poland

³ Chair of Computer Science, University of Rzeszów
ul. Rejtana 16A, 35-310 Rzeszów, Poland

Abstract. We present the relational database schema aimed at efficient storage and querying parsed scientific articles, as well as entities corresponding to researchers, institutions, scientific areas, et cetera. An important requirement in front of the proposed model is to operate with various types of entities, but with no increase of schema's complexity. Another aspect is to store detailed information about parsed articles in order to conduct advanced analytics in combination with the domain knowledge about scientific topics, by means of standard SQL and RDBMS management. The overall goal is to enable offline, possibly incremental computation of semantic indexes supporting end users via other modules, optimized for fast search and not necessarily for fast analytics, as well as direct ad-hoc SQL access by the most advanced users.

Keywords: RDBMS systems, database schema optimization, SQL-based analytics, document repositories, information retrieval and synthesis.

1 Introduction

One of the major functionalities of the “Interdisciplinary System for Interactive Scientific and Scientific-Technical Information” – a scientific project financed by Polish Government in 2010-2013^[1] – is to analyze and intelligently index large collections of scientific articles ^[12]. From technical point of view, it requires fast access to heterogeneous sources of articles, as well as convenient means for storing information retrieved from those articles. Such tasks may be partially compared to building and using various types of indexes in the Web search and enterprise search solutions. However, one should note that building truly meaningful indexes and other, more compound types of information may require processing and comparing huge sets of articles. Also, the level of detail while accessing and comparing those articles may be crucial for the quality of results.

During initial investigation, as in many other real-life projects, we quickly realized that there might be no single database/repository methodology perfectly

¹ http://ismis2011.ii.pw.edu.pl/post_conference_event.php

addressing all of the above expectations [1]. For example, dedicated document stores seem to be the best ones for gathering and managing original files and metadata of articles acquired from different sources, with an additional option of applying structural retrieval / OCR algorithms to their particular *instances*. On the other hand, standard RDBMS engines or key-value stores are appropriate for providing results of the indexing and analytic processes to end users and external tools aiming at search and visualization of the scientific contents. Going further, more analytic-oriented RDBMS solutions are a better choice for managing information about articles and other related entities in a form of data tables that can be efficiently queried using aggregate, nested, quite often ad-hoc SQL statements generated by advanced users or generated automatically by the data mining algorithms aimed at ranking, grouping, et cetera.

In this paper, we concentrate on the last of above aspects, particularly, a relational model for information about articles and related entities that may be appropriate for compound SQL-based analytics. At the beginning, our intension was to store in such model only relatively high-level information and metadata available for articles, in combination with the history of system's usage and the domain knowledge about scientific areas – everything expressed within a unified relational database schema. Actually, we could see similar attempts also in other projects [7], which provided us with additional inspiration and technical hints. However, encouraged by some recent examples of data warehouses storing far more detailed information about compound entities, such as natural language [8] and multimedia [15], we decided to enrich our originally planned model by fully parsed texts of articles (and descriptions of other entities).

The paper is organized as follows: Section 2 outlines fundamental requirements for the relational database schema and its integration with other modules in the project. Section 3 contains preliminary analysis of the most significant challenges and their possible solutions. Section 4 describes three major areas of our schema: generic, analytic, and ontological, populated with *instances*, *objects*, and *concepts*, respectively. Section 5 concludes the paper.

2 Requirements

In order to establish fully functional solution, we keep information about articles and their related entities in two forms – document repository optimized for navigation and RDBMS optimized for analytics. This way, we achieve duality of scientific content storage which has important implications for efficiency and scalability of algorithms designed within the project [12]. It is thus important to consider two levels of requirements: the specifics of relational database schema and the way of interaction of RDBMS part of solution with other modules. Let us outline the main requirements at the first level:

- there are three kinds of information to be stored: gained directly from parsed documents, related to domain knowledge delivered by experts or gained from well-known sources, and retrieved by combining two above kinds

- it has to be prepared for noisy and incomplete information (e.g.: missing metadata, different input file formats, partially known article structure)
- large volumes of data to be stored in a form enabling advanced analytics
- multiple *instances* of the same *objects* (articles, scientists, institutes) may occur because algorithms aimed at avoiding such cases prior to loading data into RDBMS may be insufficient
- it should be open for adding new types of information and entities but in a flexible way, i.e., additions should not enforce schema modifications
- it should enable indexing, querying, and generally reasoning about each part (e.g.: abstract, section) of an article separately and, if necessary, combining results in an article’s structure-aware way.

As for the second level of requirements, RDBMS part is going to rebuild its content in an offline fashion. The latest, not yet processed articles (we use timestamp to identify such articles) can be parsed cyclically (e.g.: every week) and then loaded to RDBMS in a bulk load style. The results of SQL-based algorithms working on the relational database can be exported cyclically to external layers, to support search capabilities and user interfaces. On the other hand, the whole data stored in RDBMS should be permanently available to advanced users who (when permitted) can generate direct SQL statements and project members working on algorithmic enhancements (often requiring dynamic creation of new intermediate tables or unmaterialized views).

Given the above scenarios, we tend to look at RDBMS solutions with good data compression and/or abilities to distribute data and query workloads. However, we need to remember that standard MPP database solutions [6] or, e.g., extensions towards MapReduce computations [13] may be risky given a need to process truly compound and diversified analytics. More information about applied software can be found in [12]. In this paper, we focus mainly on the relational database schema specification, although we also comment on some performance results obtained by using one of RDBMS engines [16].

3 Preliminaries

3.1 Multiple *Instances* of the Same *Object*

One of the problems that we encountered during the schema design was the existence of potentially many *instances* of the same article in various suppliers’ resources (given some analogies to popular search engines, we may call it duplication). Such cases may actually occur also for other types of *objects*, e.g., scientists or institutes. While parsing and loading information about new articles, we retrieve lots of meaningful information about their authors, references, et cetera. However, in order to keep the loading processes as simple and realistic as possible, we do not conduct full checking whether (and to what degree) the retrieved entities are *instances* of *objects* already stored in RDBMS part.

From the system effectiveness and results representation points of view, it is of course more useful to get information about, e.g., a book in general rather than

about its specific PDF or scanned versions. It brings us to the task of recognizing such situations within RDBMS and storing the relevant information at the level of equivalence classes of *instances* corresponding to the same *objects*. As already mentioned, it might be partially implemented online, during data load, however too sophisticated techniques would significantly decrease the load speed. It may also yield biased results as there is an asymmetry of information between the beginning and the end of load. Thus, we should rather implement and trigger the matching (deduplication) mechanisms after each bigger data load. There are surely plenty of possible matching techniques basing on *instances*' metadata or their internal structures (see e.g. [17]). Most of them can be successfully translated into SQL-based analytic processes, additionally supported (if necessary) by accessing the document store counterpart discussed in Section 2.

As a result, we introduce double identification for entities. First, during new publications (and other types of entities or their parts) are added, the new *instances* are created, assigned with the first level identifiers (column `id_instance`). Then, the matching algorithms merge *instances* (or their parts) into *objects*, which are basically the classes of (parts of) *instances*, assigned with the second level identifiers (column `id_object`). All answers for queries submitted to the system by users are assumed to concern the `id_object` level.

3.2 Sources of Information

There may be multiple algorithms for retrieving information from articles and other entities. In order to identify entities obtained using different algorithms, we introduce the *source of information*. Each (version of) algorithm has its corresponding `id_source`. Algorithms's description is available in additional table `SOURCE` comprising of columns `id_source` (INT) and `source_name` (VARCHAR, e.g.: 'wikipedia_eng_20101231' or 'heuristic_X').

3.3 DICTIONARY Data Type

We introduce the `DICTIONARY` column type, which reflects a kind of internal normalization of relational database schema. Each value of a column of `DICTIONARY` type is assumed to be internally encoded as an integer, stored effectively inside a database engine. Some RDBMS technologies actually support this kind of functionality [16]. Such mechanism is then totally transparent to end users, because columns remain visible with their original data types.

3.4 Adaptation of Existing Solutions

We use experiences of some finished or lasting projects and approaches to design a model meeting requirements formulated in Section 2, according to relatively standard modeling processes [18]. We adapted some solutions related to types of relations and multilingual sources from CERIF project [7]. We were also inspired by some entity-attribute-value (eav) variants [5] and some techniques of storing xml files in relational databases [4].

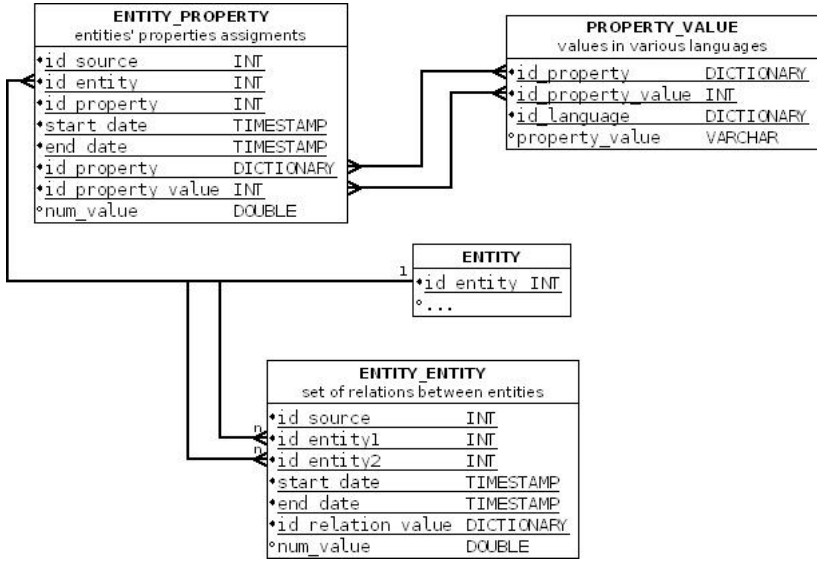


Fig. 1. Universal schema of assigning properties to entities and relations to entity pairs. In Section 4, we consider three sets of tables for three various entity semantics.

We store all properties of entities and all relations between entities in two tables. Thus, in order to add a new type of property or relation to the system, it is enough to insert a new row into one of tables displayed in Figure 1. This way, we avoid creating too many tables. We take the advantage of this universal solution in several areas of our model. The values of relations (e.g. 'IsAuthorOf', 'IsAffiliatedTo') are coded in DICTIONARY manner and stored in table ENTITY_ENTITY. The values of properties are coded as integers and stored in table ENTITY_PROPERTY. Their values can be decoded using table PROPERTY_VALUE with specific lingual context. For example, we can store the same person alias as 'Kolmogorov' (for English) and 'Kolmogorow' (Polish).

Where justified, there is temporality of described assignment for properties and relations taken into account. It is done by introducing validation time windows. There are two timestamps – `start_date` and `end_date`. If a given property or relation is not temporal, then we set up its corresponding `start_date` and `end_date` to minimum and maximum possible values, respectively.

There is always some tradeoff between effectiveness of storing data and easiness of their usage. One might consider our approach ineffective because of the size of tables ENTITY_PROPERTY and ENTITY_ENTITY but there are RDBMS solutions that can easily cope with this problem [16]. In future, in order to further increase effectiveness of a database engine, we can also consider some techniques of organizing data (such as partitioning, sorting, clustering), as well as some analytic algorithms that can work with faster approximate SQL queries.

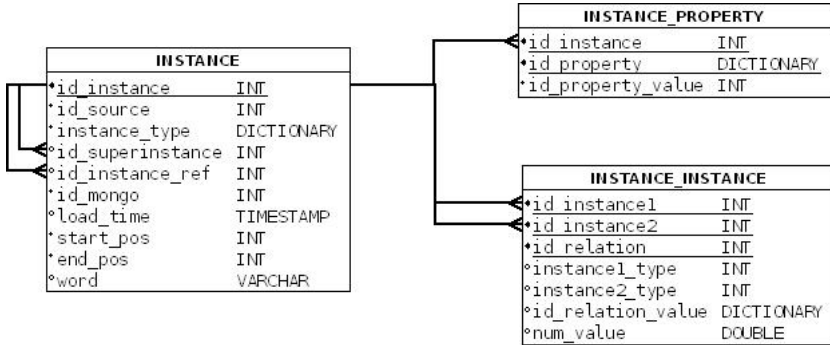


Fig. 2. Generic part of the proposed schema

4 Three Major Areas

4.1 Generic Area: *Instances*

This area contains only tables in which the primary keys include `id_instance`. Table `INSTANCE` corresponds to detailed information about *instances* – entities that we are able to distinguish while parsing the raw data (xml files, which may result from various techniques of structural document segmentation and generally understood OCR [2,12]). At the current stage of our project these are: persons, documents, and organizations. This table contains information about structures of documents. In particular, we store information about documents decomposed onto parts such as abstract, section, bibliography et cetera. Each decomposed part is treated as a separate *instance* (together with information about its structural membership to a higher-level *instance*), which enables us to operate with the parts of documents when creating semantic indexes and conducting ad-hoc analytics. For example, investigation of trends in some domains of science may be enriched by the analysis of occurrences of *concepts* related to that domain in the concluding sections (summaries, directions for further research) of earlier articles followed by their analogous occurrences in technical sections of later articles. We go back to this topic in Section 4.4.

Some discussion how to efficiently store various xml-like structures in a relational database can be found, e.g., in [10]. The proposed schema refers to the state of the art in this area, although it contains also some specific solutions reflecting the nature and quality of the input data. Generally, we may expect two categories of *instances* – those corresponding to the (parts of) input articles (or input data related to other types of entities) and those extracted from the input articles but not corresponding directly to any of them. (Although they may later turn out to be *instances* of some *objects* already stored in the database; see Section 3.1.) For the sake of consistency and simplicity, we represent them in the same way, with some parts of metadata and structural information missing depending on category and quality of parsed information.

Table 1. Columns in table INSTANCE

column name	column type	note
<code>id_instance</code>	INT	id of (a part of) publication, person, institution
<code>id_source</code>	INT	it indicates which parsing method was applied
<code>instance_type</code>	DICTIONARY	type of <i>instance</i> , e.g.: document, person, abstract
<code>id_superinstance</code>	INT	id of the direct higher-level <i>instance</i> in hierarchy
<code>id_instance_ref</code>	INT	id of an <i>instance</i> that <i>instance</i> was parsed from
<code>id_mongo</code>	VARCHAR	id of the original document stored in MongoDB
<code>load_time</code>	TIMESTAMP	time of loading the package containing <i>instance</i>
<code>start_pos</code>	INT	the beginning of <i>instance</i> 's range (see Figure 3)
<code>end_pos</code>	INT	the end of <i>instance</i> 's range (see Figure 3)
<code>word</code>	VARCHAR	it is NULL for <i>instance_type</i> other than 'word'

Whenever some portion of text or information from other sources (e.g.: various forms of metadata) is classified as an *instance*, new records in table INSTANCE are generated, with new `id_instance` associated. As mentioned earlier, at this stage, we do not take into account that it may be a new *instance* of an *object* that already exists in the system. The attributes of table INSTANCE are displayed in Figure 2. In particular, we decided to use MongoDB 3 as the store of collected articles. Thus, column `id_mongo` refers to identifiers of the corresponding articles in the store. (It applies only to the first above category of *instances* – those corresponding to input articles.) However, we may think about this column in a more abstract way, as an identifier of original, not parsed content.

As already pointed out, table INSTANCE reflects the structure of publications and their decomposition onto parts (with words as the lowest hierarchy level). Hierarchy levels are encoded by column `instance_type`. Decomposition is performed by the parsing algorithms. It is reversible, i.e., no information is lost. Figure 3 illustrates relationship between an nxml file and its INSTANCE content.

When a new *instance* is detected during the parsing process, the piece of data which enables us to distinguish it is referred using column `id_instance_ref`. The details are available in Table 1. Thus, column `id_instance_ref` creates a link to *instances* of other *objects* detected in a given *instance* (e.g.: items in the bibliography). In particular, one may think about `id_instance_ref` as responsible for linkage between two above-mentioned categories of *instances* that are represented in a uniform way in our schema.

All metadata enclosed in input files or gained from other sources are stored in two tables: INSTANCE_PROPERTY and INSTANCE_INSTANCE. The latter one includes connections between persons and organizations, which should be revealed during analyzing publication (e.g.: affiliations), between organizations and publications (e.g.: editors of documents from references) and between persons and publications (e.g.: relation of being an author of publication).

In this part of the model, we also use the structure illustrated by Figure 1, for entity = *instance*.

	id_instance	id_superinstance	start_pos	end_pos	instance_type	word
<Document>	1	0	13		Document	
<Abstract>	2	1	0	6	Abstract	
This is an exemplary abstract.	3	2	0	1	word	This
<\Abstract>	4	2	1	2	word	is
<Section>	5	2	2	3	word	an
In this section we present nothing.	6	2	3	4	word	exemplary
<\Section>	7	2	4	5	word	abstract
<\Document>	8	2	5	6	word	.
	9	1	6	12	Section	
	10	9	6	7	word	In
	11	9	7	8	word	this
	12	9	8	9	word	section
	13	9	9	10	word	we
	14	9	10	11	word	present
	15	9	11	12	word	nothing
	16	9	12	13	word	.

Fig. 3. An example of nxml file and its corresponding content in table `INSTANCE`

As already noted in Section 3.4, one might claim that it is unrealistic to put such a huge volume of data into a relational database schema. In order to verify it, we conducted a simple experiment with the RDBMS software introduced in [16], optimized with respect to data compression [9] and SQL-based analytics [14]. We parsed and loaded 5,000 scientific articles according to the proposed schema. The size of nxml representations of those articles was about 350 MB. We measured compression ratios for data stored in the applied RDBMS solution. Physical size of table `INSTANCE` turned out to be almost 40 times smaller than the corresponding tabular data obtained from the parsing algorithm. The sizes of tables `INSTANCE_INSTANCE`, `INSTANCE_PROPERTY`, and `PROPERTY_VALUE` were, respectively, 30, 20, and 5.5 times smaller than their corresponding inputs. We also measured performance of some examples of SQL statements reflecting, e.g., matching of *instances* of the same *objects*, or labeling *objects* with their most strongly represented *concepts*. Although 5,000 documents is significantly less than we should expect eventually in practice, we could clearly see that the observed SQL speed would be fully satisfactory even for far larger data.

4.2 Analytic Area: *Objects*

In this area we store tables, which are results of analytic algorithms performed on raw data stored in generic area. Again, we refer to the structure presented in Figure 1, this time for entity = *object*.

Let us start with table `OBJECT_MATCH`, which contains results of the matching algorithms, which group parts of *instances* (the whole *instance* is also treated as a 'part') in classes named *objects* and assign them with integer identifiers `id_object`. Thus, `OBJECT_MATCH` has two attributes: `id_instance`, which is `INSTANCE`'s primary key and `id_object`, which yields that every *instance*'s part

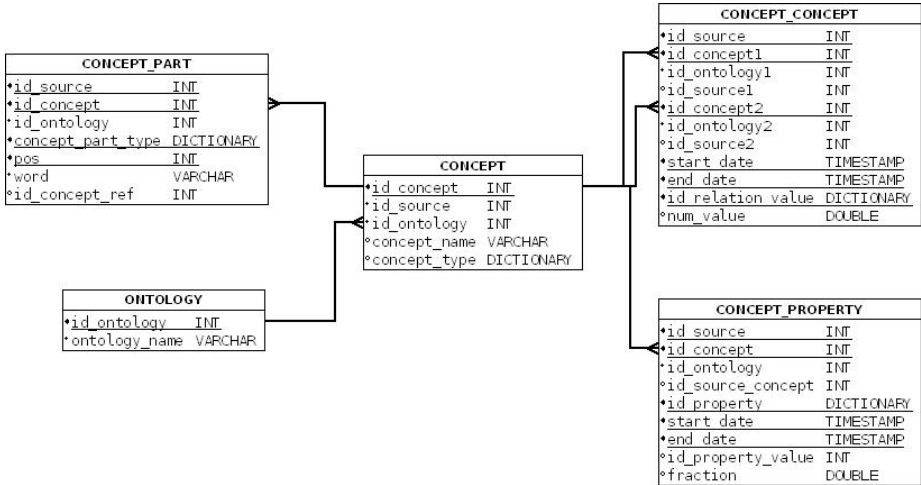


Fig. 4. Ontological part of the proposed schema

belongs to a group corresponding to an *object*. In the experimental phase of our project, we test multiple algorithms matching *instances*. We assume that `id_object` encodes a type of algorithm used to create the corresponding *object*. Eventually, once we verify which strategy is optimal, we will use a unique matching algorithm for the whole framework.

After all, analytic part of our relational database model is quite analogous to generic part displayed in Figure 2. The difference is that now we work at the level of `id_object` instead of `id_instance`. The corresponding attributes in tables describing *objects* are filled in by algorithms processing data in tables `OBJECT_MATCH` and `INSTANCE`. For example, table `OBJECT_OBJECT` stores information about relations between *objects* computed based on relations between *instances* in table `INSTANCE_INSTANCE`. This level is also a place for more intelligent algorithms that find a wider range of types and degrees of relationships between *objects* than it is present at the level of *instances*.

4.3 Ontological Area: *Concepts*

From syntactic point of view, one can treat ontologies as models of knowledge in which there can be distinguished entities, their properties and relations between them. This fits to our approach, therefore, we are going to store information retrieved from ontologies in tables similar to these described in Section 3.4. In case of ontologies, we will talk about *concepts* instead of *instances* (see Figure 4). Our major challenge in this area was to provide fairly universal framework for storing information about *concepts* acquired from different sources, such as, e.g., Wikipedia (full articles) or Wordnet (only lexems). Generally speaking, as a *concept* we treat every entity that can be retrieved from a source provided

CONCEPT_OBJECT	
relations between concepts and objects	
*id_source	INT
*id_object	INT
*id_concept	INT
*id_source_concept	INT
*id_ontology	INT
*start_date	TIMESTAMP
*end_date	TIMESTAMP
*id_relation_value	DICTIONARY
*num_value	DOUBLE

Fig. 5. Linking *concepts* and *objects* in table CONCEPT_OBJECT

by an expert (and using an algorithms accepted by an expert), which can have properties and some describable relations to other *concepts*.

Properties of *concepts* (e.g.: name of a *concept*) and relations between *concepts* are assumed to be temporal.

Apart from tables described in Section 3.4, we also need representation of *concepts* as texts. Such information is contained in table CONCEPT built similarly to INSTANCE. As one can represent a *concept* by different text descriptions, we introduce column `concept_part_type`. There is, however, a slight difference between this column and `instance_type` in table INSTANCE – as for now, we did not find a reason for introducing hierarchy for *concept* descriptions like in case of *instances*. Hence, `concept_part_type` is not the key in table CONCEPT. Also, we do not need to consider column `concept_superpart`.

We decided to separate the above subset of tables from generic part as we assume that ontologies will be loaded in more supervised way, so there is no place for noise or incompleteness. There is also a principal difference in usage of both parts. *Instances* reflect lower level information – we use them mainly for producing *objects*, not for retrieving new knowledge like in case of *concepts* and *objects* in the next subsection. Differences in data supply process are also the reason for distinguishing between *object* and *concept* areas.

4.4 Linking Concepts and Objects

Table CONCEPT_OBJECT (Figure 5) consists of relations between ontological *concepts* and analytic *objects*. One of the most important relations which we can derive, store and use is labeling documents or scientists with topics from ontologies (e.g. from Wikipedia). Going further, we can do it not only for the whole articles but also for their particular parts, which can lead to some interesting article structure-aware analytics. Querying the parts of documents is quite well-established area of research (see e.g. [11]). We have already mentioned about it in Section 4.1, although it may now make more sense for *objects* instead of their *instances*. One more example may be to reason about the most promising areas of science, e.g., by means a query formulated by a student who searches for potential topics of future thesis, where SQL-based heuristics may extract topics

occurring frequently in the concluding parts of articles but with no significant representation in bibliographies and major parts of articles.

Table `CONCEPT_OBJECT` is built analogously to `ENTITY_ENTITY` in Figure 1, where the first entity becomes analytic *object* and the second entity becomes ontological *concept*. The parts of *objects* (having the status of *objects* as well) can be also, e.g., equations, tables or figures in a publication. Thus, table `CONCEPT_OBJECT` may include a number of interesting relations, e.g., labeling specific figures from specified books with *concepts* such as 'Pythagorean theorem', or reflecting our previously discussed examples related to analyzing scientific trends and search for interesting scientific topics.

5 Conclusion

The presented model of storing and processing information related to scientific content has resulted from a number of design iterations that took into account various requirements, such as diversity and incompleteness of data sources, occurrence of multiple *instances* of the same *objects*, a need of dealing with potentially growing amount of types of *objects* without making the database schema overcomplicated, as well as analyzing *objects* with respect to various types relationships with *concepts* representing domain knowledge provided by experts or extracted semi-automatically from available sources. The obtained relational database schema contains core tables for three main layers – generic (*instances*), analytic (*objects*), and ontological (*concepts*) – in analogous formats, with ability to create additional intermediate tables and views whenever necessary.

Our major motivation to apply RDBMS framework was to provide a dual form of storing scientific content. We noticed that different tasks in our project required different characteristics of data access and processing, including massive comparisons of large collections of parts of documents. Ability to use SQL over clearly defined schema may open a variety of analytic possibilities. Appropriately chosen database technologies may enable to store huge amounts of parsed data and run SQL-based analytic tasks satisfactorily fast.

Acknowledgments. The authors are supported by the grant N N516 077837 from the Ministry of Science and Higher Education of the Republic of Poland and by the National Centre for Research and Development (NCBiR) under the grant SP/I/1/77065/10 by the Strategic scientific research and experimental development program: “Interdisciplinary System for Interactive Scientific and Scientific-Technical Information”.

References

1. Agrawal, R., Ailamaki, A., Bernstein, P.A., Brewer, E.A., Carey, M.J., Chaudhuri, S., Doan, A., Florescu, D., Franklin, M.J., Garcia-Molina, H., Gehrke, J., Gruenwald, L., Haas, L.M., Halevy, A.Y., Hellerstein, J.M., Ioannidis, Y.E., Korth, H.F., Kossmann, D., Madden, S., Magoulas, R., Ooi, B.C., O'Reilly, T., Ramakrishnan, R., Sarawagi, S., Stonebraker, M., Szalay, A.S., Weikum, G.: The Claremont Report on Database Research. *Commun. ACM* 52(6), 56–65 (2009)

2. Betliński, P., Gora, P., Herba, K., Nguyen, T.T., Stawicki, S.: Semantic Recognition of Digital Documents. In: Bembeník, R., Skonieczny, Ł., Rybiński, H., Niezgódka, M. (eds.) *Intelligent Tools for Building a Scientific Information Platform*. Springer, Heidelberg (2011)
3. Chodorow, K., Dirolf, M.: *MongoDB: The Definitive Guide: Powerful and Scalable Data Storage*. O'Reilly Media (2010)
4. Grust, T.: Accelerating XPath Location Steps. In: *Proc. of Int. Conf. on Management of Data (SIGMOD)*, pp. 109–120 (2002)
5. Hammond, W., Stead, W., Straube, M.: A Chartless Record-Is It Adequate? In: *Proceedings of the Annual Symposium on Computer Application in Medical Care*, vol. 7, pp. 89–94 (1982)
6. Hellerstein, J.M., Stonebraker, M., Hamilton, J.R.: Architecture of a Database System. *Foundations and Trends in Databases* 1(2), 141–259 (2007)
7. Jörg, B., Jeffery, K., van Grootel, G., Asserson, A., Dvorak, J., Rasmussen, H.: CERIF, - 1.2 Full Data Model (FDM) Introduction and Specification (2008), http://www.eurocris.org/Uploads/Web/%20pages/CERIF2008/Release_1.2/CERIF2008_1.2_FDM.pdf
8. Kobdani, H., Schütze, H., Burkovski, A., Kessler, W., Heidemann, G.: Relational Feature Engineering of Natural Language Processing. In: *Proc. of Int. Conf. on Information and Knowledge Management (CIKM)*, pp. 1705–1708 (2010)
9. Kowalski, M., Ślęzak, D., Toppin, G., Wojna, A.: Injecting Domain Knowledge into RDBMS – Compression of Alphanumeric Data Attributes. In: Kryszkiewicz, M., Rybinski, H., Skowron, A., Raś, Z.W. (eds.) *ISMIS 2011*. LNCS, vol. 6804, pp. 386–395. Springer, Heidelberg (2011)
10. Mihajlović, V., Blok, H.E., Hiemstra, D., Apers, P.M.G.: Score Region Algebra: Building a Transparent XML-R Database. In: *Proc. of Int. Conf. on Information and Knowledge Management (CIKM)*, pp. 12–19 (2005)
11. Navarro, G., Baeza-Yates, R.A.: Proximal Nodes: A Model to Query Document Databases by Content and Structure. *ACM Trans. Inf. Syst.* 15(4), 400–435 (1997)
12. Nguyen, H.S., Ślęzak, D., Skowron, A., Bazan, J.G.: Semantic Search and Analytics over Large Repository of Scientific Articles. In: Bembeník, R., Skonieczny, Ł., Rybiński, H., Niezgódka, M. (eds.) *Intelligent Tools for Building a Scientific Information Platform*. Springer, Heidelberg (2011)
13. Pavlo, A., Paulson, E., Rasin, A., Abadi, D.J., DeWitt, D.J., Madden, S., Stonebraker, M.: A Comparison of Approaches to Large-scale Data Analysis. In: *Proc. of Int. Conf. on Management of Data (SIGMOD)*, pp. 165–178 (2009)
14. Ślęzak, D., Eastwood, V.: Data Warehouse Technology by Infobright. In: *Proc. of Int. Conf. on Management of Data (SIGMOD)*, pp. 841–846 (2009)
15. Ślęzak, D., Sosnowski, Ł.: SQL-Based Compound Object Comparators: A Case Study of Images Stored in ICE. In: Kim, T.-h., Kim, H.-K., Khan, M.K., Kiumi, A., Fang, W.-c., Ślęzak, D. (eds.) *ASEA 2010*. Communications in Computer and Information Science, vol. 117, pp. 303–316. Springer, Heidelberg (2010)
16. Ślęzak, D., Wróblewski, J., Eastwood, V., Synak, P.: BrightHouse: An Analytic Data Warehouse for Ad-hoc Queries. *Proc. VLDB Endow.* 1(2), 1337–1345 (2008)
17. Tekli, J., Chbeir, R., Yétongnon, K.: An Overview on XML Similarity: Background, Current Trends and Future Directions. *Computer Science Review* 3(3), 151–173 (2009)
18. Teorey, T., Lightstone, S., Nadeau, T.: *Database Modeling & Design: Logical Design*, 4th edn. Morgan Kaufmann (2005)

Semantic Clustering of Scientific Articles with Use of DBpedia Knowledge Base*

Marcin Szczuka, Andrzej Janusz, and Kamil Herba

Faculty of Mathematics, Informatics, and Mechanics, The University of Warsaw
Banacha 2, 02-097 Warsaw, Poland

szczuka@mimuw.edu.pl, janusza@mimuw.edu.pl, k.herba@students.mimuw.edu.pl

Abstract. A case study of semantic clustering of scientific articles related to Rough Sets is presented. The proposed method groups the documents on the basis of their content and with assistance of DBpedia knowledge base. The text corpus is first treated with Natural Language Processing tools in order to produce vector representations of the content and then matched against a collection of concepts retrieved from DBpedia. As a result, a new representation is constructed that better reflects the semantics of the texts. With this new representation, the documents are hierarchically clustered in order to form partition of papers that share semantic relatedness. The steps in textual data preparation, utilization of DBpedia and clustering are explained and illustrated with experimental results. Assessment of clustering quality by human experts and by comparison to traditional approach is presented.

Keywords: Text mining, semantic clustering, DBpedia, document grouping, rough sets.

1 Introduction

This article presents a case study of semantic clustering of scientific articles related to the area of Rough Sets. We have undertaken this study in order to answer the need for developing semantic methods for document processing expressed in the major project (SYNAT) we are involved in.

The SYNAT project (abbreviation of Polish “**SY**stem **NA**uki i **TE**chniki”) is a large, national R&D program of Polish government aimed at establishment of a unified network platform for storing and serving digital information in widely understood areas of science and technology. Within the frame of this project we are concerned with devising and implementing methods that would allow for indexing, searching, and retrieving the documents from the possibly vast collection using their semantic content (the knowledge they contain).

* The authors are supported by the grant N N516 077837 from the Ministry of Science and Higher Education of the Republic of Poland and by the National Centre for Research and Development (NCBiR) under Grant No. SP/I/1/77065/10 by the strategic scientific research and experimental development program: “Interdisciplinary System for Interactive Scientific and Scientific-Technical Information”.

The idea that we pursue in this study is to perform semantic grouping (clustering) of documents based on their associations with concepts drawn from the DBpedia knowledge base. If done right, such clustering would make a good start for, e.g., a system with extended search features, capable of returning results that are topically close to the search terms, not just those that actually contain the terms from the query (semantic vs. syntactic). This approach is in line with the general trend of finding semantic similarities between documents with assistance of additional knowledge sources (ontologies, thesauri, taxonomies, Wikipedia) in order to obtain more meaningful and useful results, as described in [4,9,14,8].

In our case study we use a text corpus consisting of scientific papers related to Rough Sets. We hope that in this way we will gain some additional insight into our own field of research, verify (positively or negatively) some hypotheses and common beliefs, and possibly find some new. At the same time, since we know the document corpus well, we can use our own expertise to judge the quality of clustering solution.

The article is organized as follows. Section 2 describes the methodology and motivation behind our approach. Then we describe our data set (Section 3) and DBpedia knowledge base (Section 4), providing some details about their characteristics and the way they were collected and prepared for experiments. Section 5 contains description of the actual experiment and explanation of its results. We finish with conclusions and directions for further work in Section 6.

2 The Purpose and Methodology of the Study

The purpose of this experimental study is two-fold.

1. We want to test and verify methodology for document grouping (clustering) based on their semantic content and using a knowledge base. In particular, we want to identify the best configuration for various steps in the process, one that is both computationally feasible and produces meaningful clusters of documents. The goal is to establish a procedure that we will be able to apply semi-automatically to various future text corpora. Since the area of Rough Sets is close to us, we are able to better evaluate the results of experiments on the corpus of texts collected in this field of research. As a consequence, we can identify strengths and weaknesses of the method under scope.
2. We want to learn as much as the methodology permits about our corpus of documents (research papers) related to the area of Rough Sets. Since the individual documents used for this case study (Section 3) are familiar to us, we want to discover the semantic structure of the corpus as a whole and draw some conclusions regarding the features of publications in this scientific area. In particular, we are interested in identifying the most prevalent concepts that characterize this corpus.

Figure 1 shows the general layout of the method that we employ in our case study. The methodology of our was inspired by the Explicit Semantic Analysis (ESA)

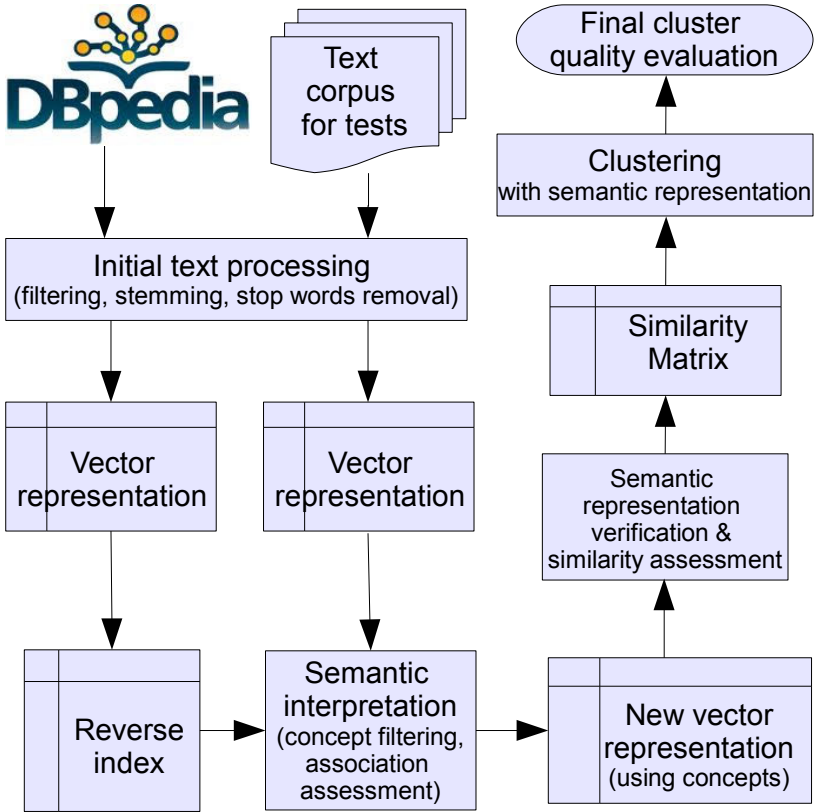


Fig. 1. The general scheme of the experiment

approach presented in IJCAI paper [4]. Since this article is quite involved, the method which it discusses requires more detailed explanation. The data sources, i.e., collection of documents and DBpedia knowledge base, together with the NLP¹ methods for their pre-processing ("Initial text processing" box in Fig. 1) are described in Sections 3 and 4, respectively. In our approach, after initial processing, both collections of texts (the corpus and the DBpedia abstracts) are converted to the *bag-of-words* (word-vector) representation. The bag-of-words representation of a text (document) is a vector based on vocabulary, i.e., the collection of unique words (stems) in the corpus.

Assume, that after initial processing of a text corpus $D = \{T_1, \dots, T_M\}$ we have collected a vocabulary consisting of n unique terms (stems) w_1, \dots, w_n . Then, any text (document) T_j in the corpus is represented by a vector of the form $\langle v_1, \dots, v_n \rangle \in \mathbb{R}_+^n$, where each coordinate v_i is a value of importance measure for i -th term (word, stem) in vocabulary (w_i), relative to this document. The

¹ Natural Language Processing (NLP) tools as in [3].

most common measure used to calculate v_i is the *tf-idf* (term frequency-inverse document frequency) index (see [3]) defined, relative to document T_j and corpus D , as:

$$v_i = tf_{i,j} \times idf_i = \frac{n_{i,j}}{\sum_{k=1}^N n_{k,j}} \times \log \left(\frac{M}{|\{j : n_{i,j} \neq 0\}|} \right) \quad (1)$$

where $n_{i,j}$ is the number of occurrences of the considered word w_i in the document T_j .

To speed up semantic interpretation, we build an inverted index, which maps each word into a list of DBpedia concepts c_1, \dots, c_N in which it appears. With the inverted index we run a semantic interpreter (Semantic Interpretation in Fig. II). Given a text from the corpus, the semantic interpreter iterates over words it contains, retrieves corresponding entries from the inverted index and merges them into a weighted vector of concepts that represents the given text.

Let $T = \langle w_i \rangle_{i=1}^n$ be input text, and let $\langle v_1, \dots, v_n \rangle$ be its tf-idf vector, where v_i is the weight of word w_i . Let k_{ij} be an inverted index entry for word w_i , where k_{ij} quantifies the strength of association of word w_i with knowledge base concept c_j , $j \in \{1, \dots, N\}$. Then, the new vector representation of T is calculated as:

$$\langle \sum_{i:w_i \in T} v_i k_{ij} \rangle_{j=1}^N. \quad (2)$$

We will refer to this new vector representation using the notion of *bag-of-concepts*. Note, that in the sum in formula (2) we only consider the words that actually appear in the text T .

The new vector (bag-of-concepts) representation makes it possible to examine relations between concepts and documents, identify and filter key concepts for the given document corpus, and – most importantly – calculate semantic similarity between texts by comparing their bag-of-concepts representations. For technical reasons we choose to store all semantic similarity values for pairs of texts in a structure called *Similarity Matrix*. Entries in this matrix are used to numerically represent the proximity between documents (their bag-of-concept representations), which in our case is calculated using cosine distance.

The fact that we can calculate semantic similarity (distance) between documents gives us the means to perform clustering. Considering that we want to obtain a meaningful grouping of documents we decided to use an agglomerative hierarchical clustering. In order to decide for how many clusters we should divide our data we use a cluster quality measure, in particular the silhouette coefficient. For detailed description of agglomerative clustering, silhouette coefficient, and cluster distance refer to [12].

The quality of resulting clusters is evaluated manually with help of experts in the area of Rough Sets as well as compared with results of a different clustering method. In order to have a reference point we perform a “classical” agglomerative clustering on the bag-of-words representation of documents, without any use of knowledge base. The resulting partition of documents is then compared with

our approach using various measures for cluster consistency as well as manual evaluation of cluster meaningfulness.

3 Data Acquisition and Preparation

For our case study we have used 349 documents in PDF format. These documents are selected from the collection of papers published by the members and associates of the Group of Logic at the University of Warsaw. The subset used for our experiment is significantly smaller than the entire collection, which consists of over 600 publications. While choosing documents for this subset we have used the following criteria:

- We restricted publications to those published in last 15 years (between 1996 and 2011) and written in English.
- We have only chosen “regular” articles, i.e., standard journal, book, and conference papers. They roughly correspond to Bib \TeX categories: **article**, **inproceedings**, and **incollection**.
- Papers that are very short (extended abstracts) or unusually long (mini-monographs) have been left out.
- Some articles have been removed from the study due to technical difficulties they posed. This was mostly due to problems with incorrect PDF format and usually concerned older (pre-2003) publications.

There were several reasons for using the above criteria in the process of constructing initial data sample for our study. The most important are as follows:

- We wanted the corpus of documents to be relatively *regular*. Since our ultimate goal is the grouping (clustering), we tried to eliminate outliers early on. The idea is to have well-comparable documents and then do the clustering on the basis of their semantic content rather than attributes of their syntactic composition, such as size, level of complication or number of words.
- We have chosen the collection of documents that were created over the years in our group in order to have good understanding of the corpus from the very beginning. Since we know the field and in many cases have direct contact with authors, we can evaluate the outcome with greater ease and confidence. This is a big advantage, especially for an initial, explorative study such as the one that we conduct. It gives us the ability to clearly identify strong and weak points in our methodology.
- We have decided to use this particular number of documents (349) because we wanted to construct a corpus which would be as representative for the area of Rough Sets as it is possible in the given circumstances. The 349 documents in our collection correspond to roughly 10% of all documents of this kind listed in the Rough Set Database system (RSDS [5]). At the moment of writing, the RSDS contains 3641 bibliographical notes that belong to categories that we are interested in.

- Last, but not the least, we selected this particular document corpus because we have both access to their PDF versions and the limited copyrights that allow us to re-use (but not re-distribute) them.

The original PDF documents were first converted to a pure text format with the use of Python script based on PDFMiner library [11]. All documents were divided into blocks of plain text. Based on certain text statistics the script extracted only the text contained in paragraphs, sections and their titles. It has to be underlined that author names article and page headers, footers, tables, equations and other parts of text which were irrelevant and could bias further analysis were discarded. The purpose of this step was to remove various artifacts and clarify text files before attempting to calculate word frequencies and clustering.

This step, although it may appear simple, proved to be troublesome at times. Typical problems at this stage are associated with conversion of hyphenated (broken between lines) words and ligatures (e.g., **fi** in “classification”) back to their original (textual) form. These problems were partly resolved with use of an English dictionary which made it possible to guess the right encoding of some characters by determining whether words created after substitution of missing characters were proper English terms. Articles contained also a great amount of mathematical symbols which were encoded in PDF files in various, sometimes very unexpected way. These unusual characters were filtered out as well. Additionally, the bibliography section (references) was removed from each of selected text files. It was done in order to assure that we perform analysis on actual semantic content of the document and to reduce the influence of certain words contained in references, like: publisher, journal name, etc.

The corpus of 349 plain text files was then processed in order to calculate word-vector (bag of words) representations in the next step. First, stop words were removed and then we have performed stemming on the set of words contained in these documents. For stemming of both documents and DBpedia abstracts (as described in Section 4) we use a version of popular Porter’s algorithm (cf. [7]). Initially, the corpus contained 35507 unique words (excluding stop words). After stemming we have obtained 26800 unique words (stems) to work with. On average a single document in the collection contains 3524 stems, with minimum of 362 and maximum of 13640.

4 The DBpedia Knowledge Base

According to its creators, the DBpedia (cf. [13,11]) is a community effort to extract structured information from Wikipedia (cf. [15]) and to make this information available on the Web. DBpedia allows to ask queries against Wikipedia data and structure, and to link other data sets to Wikipedia data. In layman terms, DBpedia is a snapshot of the original Wikipedia with mostly preserved structure, but reduced content.

For the purpose of our study we needed to use DBpedia as an enriched dictionary. The version of DBpedia that we use (version 3.5.1 for English Wikipedia) contains 3,257,133 notions (so called *things*). Each DBpedia *thing* represents

a single Wikipedia concept (a single Wikipedia page including disambiguation pages and lists). Due to the distributed and asynchronous nature of the process in which the Wikipedia is created by members of its community, there are some consistency and regularity issues with it. Much of these issues are inherited by DBpedia, which results in some problems related to conflicting or expired names for concepts and categories.

In DBpedia, pages from the original Wikipedia are represented only by their abstracts. For most of the DBpedia concepts there is also additional information derived from Wikipedia, such as classification to Wikipedia categories. There are 3,144,262 abstracts available in DBpedia 3.5.1, but they are very diverse in their length and quality. The length of abstracts vary from empty (0 words) to quite long ones (the longest has 16850 words), with an average of 101 words per abstract. Most of those texts are well formatted and structured but there are exceptions, e.g., some contain only L^AT_EX-styled source code of tables or figures which were, probably unintentionally, placed in the abstract section of the corresponding Wikipedia article. There are also cases when a whole text of Wikipedia page is placed in the abstract which results in considerably longer DBpedia representations.

Taken altogether, DBpedia 3.5.1 entries constitute a text corpus consisting of 316,631,010 words (after filtration). The number of unique words, before stemming and filtering, is 2,818,483. There are 560,049 *categorical notions* (Wikipedia categories) of which 449,140 are *direct*, i.e., contain some concepts and the rest are *indirect*, i.e., they contain only other categories.

5 Experimental Evaluation of the Approach

Our experiment was conducted in three main steps which we implemented in R System ([10]). First, DBpedia and the selected text collection were preprocessed. Each DBpedia entry and a document was cleaned, in particular: its encoding was changed to UTF-8, words that contained special characters (!@#%&*+)=) or numbers were removed, the most common shortcuts were expanded, and the most common words from a special *stop word list*² were removed. The Porter’s algorithm [7], implemented in the *Rstem* library, was used for finding stems of words. The stems that occurred less than three times in DBpedia were also eliminated from the texts. Finally, the concepts that were represented by less than 10 unique stems were removed from the knowledge base. As a result, the size of the knowledge base was reduced to around 2.5 million concepts described by approximately 850 thousands of unique stems.

In the second step, the bag-of-concepts representations of texts from the rough set corpus were created using the method described in Section 2. A modified tf-idf index was used to assess the relevance of words (stems) to documents and to concepts. For each text, the frequencies of words, i.e., the tf component in tf-idf formula ([1]), were smoothed by taking their square root. This modification was

² A standard stop word list from *openNLP* library was extended by the 100 most common words from DBpedia abstracts.

Table 1. List of ten most relevant DBpedia concepts for three exemplary documents, with degree of association included

(LTF-C): Architecture, Training Algorithm and Applications of New Neural Classifier		
[1]	9.19	"Neural_Lab"
[2]	9.17	"Echo_state_network"
[3]	8.75	"Auto-encoder"
[4]	8.30	"Interneuron"
[5]	8.09	"Oja's_rule"
[6]	8.08	"Multilayer_perceptron"
[7]	8.06	"Biological_neural_network"
[8]	8.06	"Artificial_neural_network"
[9]	8.00	"Artificial_neuron"
[10]	7.84	"Neuroevolution"
Judgment of satisfiability under incomplete information		
[1]	8.21	"Definable_set"
[2]	8.08	"Schaefer's_dichotomy_theorem"
[3]	7.96	"Formal_semantics_of_programming_languages"
[4]	7.85	"Empty_domain"
[5]	7.78	"Tautology_(logic)"
[6]	7.68	"Equisatisfiability"
[7]	7.54	"Method_of_analytic_tableaux"
[8]	7.38	"Conditional_quantifier"
[9]	7.36	"Model_checking"
[10]	7.32	"Satisfiability_and_validity"
Combination of Metric-Based and Rule-Based Classification		
[1]	8.92	"K-nearest_neighbor_algorithm"
[2]	6.19	"Backmarking"
[3]	6.08	"Wolfe_conditions"
[4]	5.90	"Evolutionary_data_mining"
[5]	5.66	"Event_condition_action"
[6]	5.64	"Transduction_(machine_learning)"
[7]	5.63	"Soft_independent_modelling_of_class_analogies"
[8]	5.63	"Ground_truth"
[9]	5.56	"Proximity_problems"
[10]	5.50	"Dominating_decision_rule"

dictated by a fact, that many of the documents which we use are of technical nature and as such contain many repetitions of specific terms (or single words). The strength of bounds between the concepts and the rough set articles was computed using the equation (2). Following the intuition, that it is meaningless to associate any document with a large number of specific concepts, we have restricted the number of concepts associated with each document. We have decided to use no more than 35 most related concepts for characterization of any given text. This number (35) was selected because it corresponds to around 1% of the average number of stems appearing in the single document in the corpus, which in turn gives more compact and comprehensive representation. Table 1

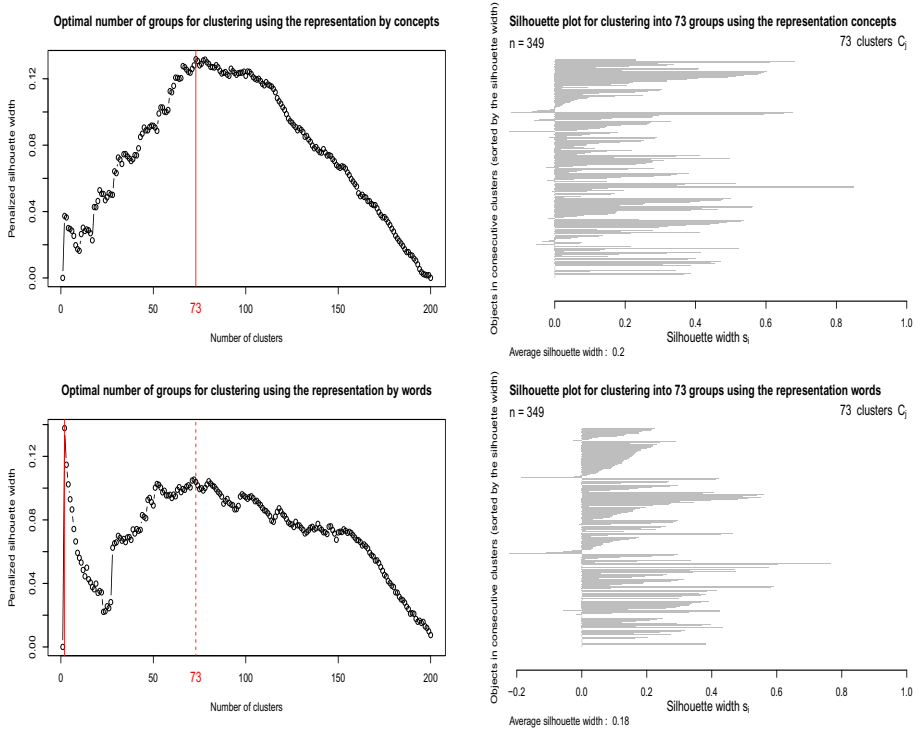


Fig. 2. The plot of silhouette coefficient values across clusters used to establish the optimal number of groups (on the left) and the silhouette coefficient values of individual documents for the selected clustering (on the right). The results obtained from the representation by concepts (on top) is compared to representation by words (at the bottom).

presents associations of top 10 concepts to three exemplary articles from the corpus.

The last step of our experiment involved computation of distances between documents from the rough set corpus which we then use for clustering. Due to an extremely sparse representation of our texts (only 35 non-zero values out of ≈ 2.5 million) the *cosine distance* was employed, which is a commonly used measure in high-dimensional information retrieval tasks ([3], [12]). For the clustering we utilized an agglomerative hierarchical approach with the “average” as a linking function (see [12]). The optimal number of groups (clusters) was decided using the *silhouette width coefficient* which was additionally penalized for selecting larger number of clusters. Figure 2 illustrates the values of average silhouette coefficient w.r.t. the growing number of clusters, along with the silhouette coefficient values of individual documents for the selected clustering. From this picture one can see that the highest cluster separability is achieved when we use 73 of them.

Figure 3 presents the clustering tree corresponding to the partition into 73 groups. Apart from using the silhouette coefficient, quality of the 73 clusters was also assessed with the aid of human experts. Mutual relatedness of documents from several groups has been evaluated. In order to gain another point of reference we have also performed clustering using the original bag-of-words³ representation of texts in the corpus (Figure 4).

The results of this comparison are encouraging. The consecutive partitions obtained using the bag-of-concepts representation yielded much more stable silhouette coefficients than those for the original word-vector (bag-of-words) one. The optimal number of groups for the latter is two, which corresponds to meaningless grouping of documents. This number is also not very stable as it may vary wildly between 2 and 157 if we alter the penalty for producing excessive clusters. Moreover, if with the bag-of-words representation we make the clustering algorithm produce 73 groups (optimal number for the bag-of-concepts), then brief analysis of this partition reveals a significant imbalance in the size of clusters (Figure 4). The largest cluster obtained in this way contained 60 papers and there were 29 singletons (clusters that contained only a single document). To make things worse, many of the larger groups constructed in this manner contain semantically unrelated documents and are very difficult to label. In contrast, size of the largest group resulting from utilization of the bag-of-concepts representation was 27 and there were only 19 singletons.

The observation, that employment of domain knowledge improves the quality of clustering was confirmed by domain experts. For instance, Table 2 shows members of three exemplary clusters taken from distinct branches in the clustering tree (Figure 3). Labels that briefly summarize contents of those groups were given by experts. Among 13 papers that belong to the cluster 21, 12 were recognized by experts as related to the notion of neural computing and artificial neural networks. The same subset of papers, partitioned based on the bag-of-words representation, was broken between three different clusters of which only one was semantically homogeneous and meaningful.

It is also worth mentioning that, even though information about authors and bibliography was removed from the corpus during the preprocessing phase, 12 out of 14 articles grouped in the cluster 39 were written by a single author (Anna Gomolińska). In those papers, the author considers a problem of partial satisfiability and validity of formulas (such as decision rules) under incomplete or uncertain information.

It seems that with the bag-of-concept representation, the clustering algorithm was able to conceptually discern them from other research topics of this particular author. The articles of the same author that belong to other research directions, the theory of approximation spaces, are located in another cluster. These articles (six of them) are placed in the cluster 36 (Figure 3). In comparison, when the representation by bag-of-words is used, almost all publications of Anna Gomolińska from our corpus (21 out of 22) are placed in a single group. That last fact, in our opinion, is probably due to usage of a characteristic and

³ For consistency, we used the smoothed tf-idf vector representation.

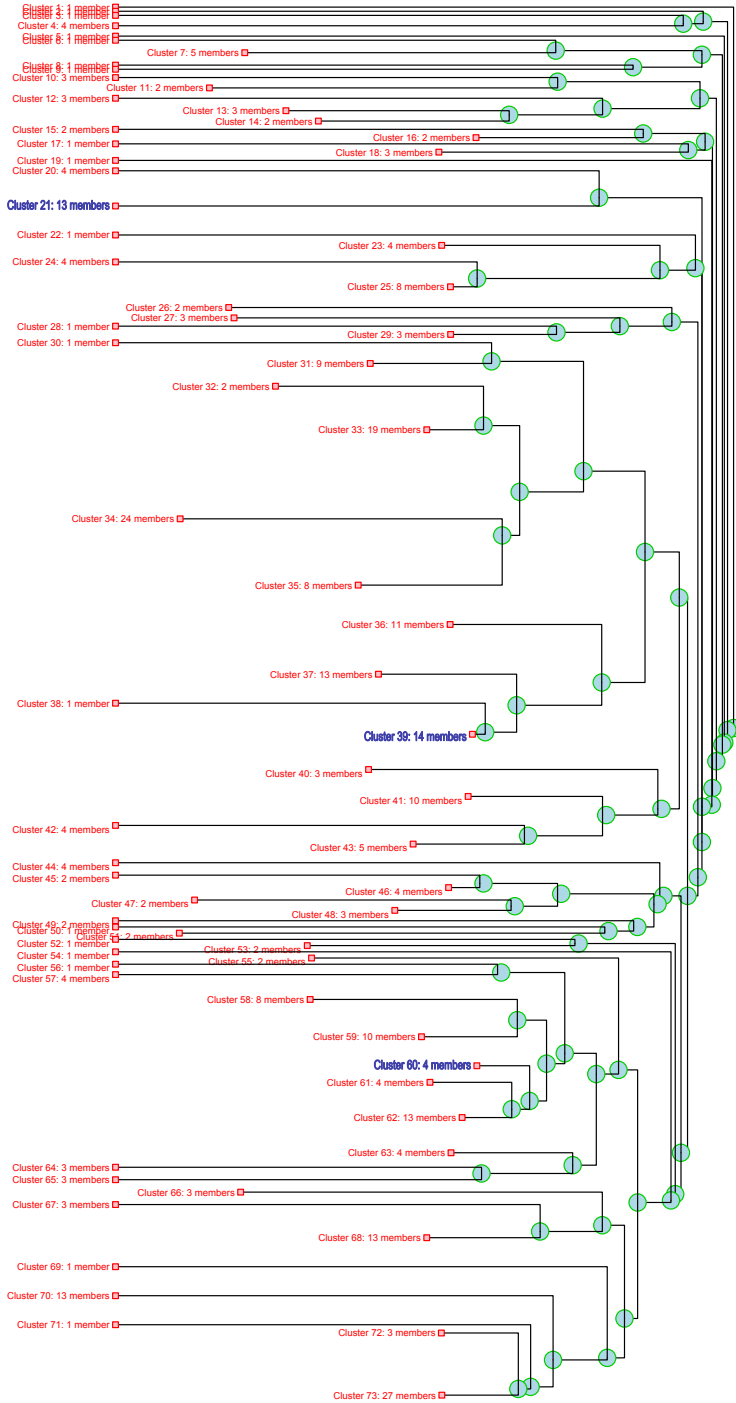


Fig. 3. Truncated tree of clusters (dendrogram) with 73 leafs, based on the bag-of-concepts representation. Clusters 21,39, and 60 that are detailed in Table 2 are marked with different font.

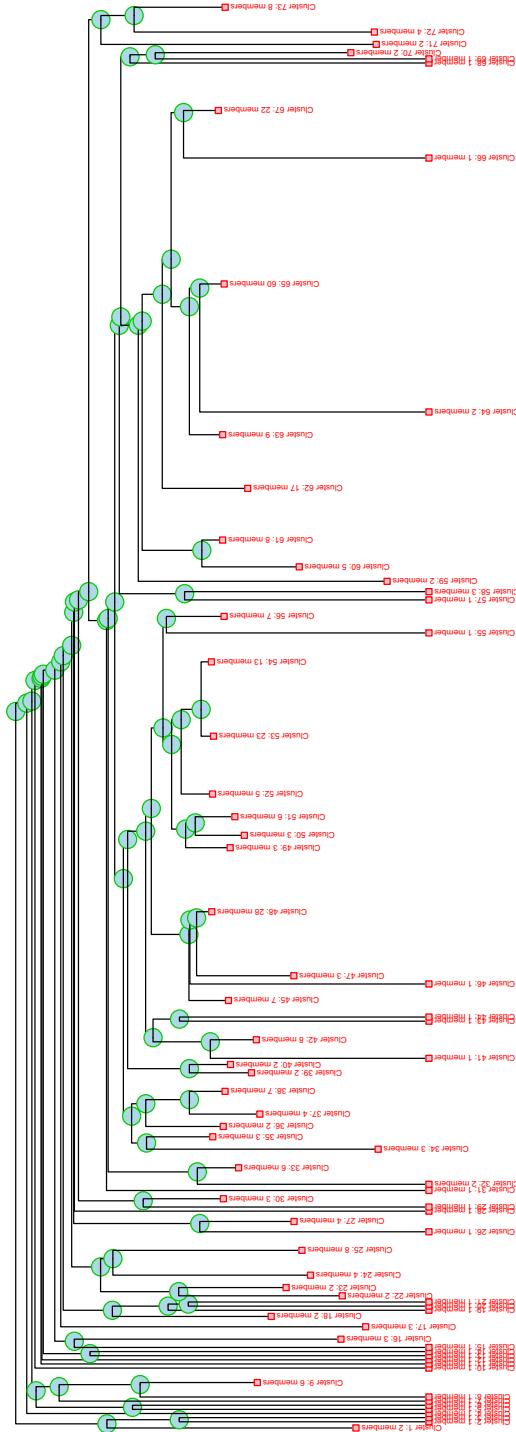


Fig. 4. A clustering tree based on the bag-of-words representation, truncated to 73 leaves

Table 2. Members of exemplary partitions, resulting from clustering with the bag-of-concepts representation. The IDs of branches from the clustering tree are given along with the labels assigned by domain experts and titles of corresponding documents.

Cluster 21: Neurocomputing and Artificial Neural Networks	
[1]	(LTF-C): Architecture, Training Algorithm and Applications of New Neural Classifier
[2]	Rough Neurons: Petri Net Models and Applications
[3]	Rough-Neural Computing: An Introduction
[4]	Toward Rough Neural Computing Based on Rough Membership Functions: Theory and Application
[5]	Rough Neurocomputing: A Survey of Basic Models of Neurocomputation
[6]	Design of rough neurons: Rough set foundation and Petri net model
[7]	Constructing Extensions of Bayesian Classifiers with use of Normalizing Neural Networks
[8]	Refining decision classes with neural networks
[9]	Harnessing Classifier Networks - Toward Hierarchical Concept Construction
[10]	Feedforward concept networks
[11]	Neural network design: Rough set approach to real-valued data
[12]	Hyperplane-based neural networks for real-valued decision tables
[13]	Rough Sets and Artificial Neural Networks
Cluster 39: Logical Satisfiability and Validity of Formulas	
[1]	Judgment of satisfiability under incomplete information
[2]	A graded applicability of rules
[3]	Toward rough applicability of rules
[4]	Satisfiability and meaning in approximation spaces
[5]	Satisfiability Judgment Under Incomplete Information
[6]	Reasoning Based on Information Changes in Information Maps
[7]	Rough validity, confidence, and coverage of rules in approximation spaces
[8]	Satisfiability and meaning of formulas and sets of formulas in approximation spaces
[9]	On rough judgment making by socio-cognitive agents
[10]	Rauszer's R-logic for multiagent systems
[11]	Rough rule-following by social agents
[12]	Satisfiability of formulas from the standpoint of object classification
[13]	Construction of rough information granules
[14]	Patterns in Information Maps
Cluster 60: Instance-based Learning	
[1]	Combination of Metric-Based and Rule-Based Classification
[2]	Rough Set Approach to CBR
[3]	Local Attribute Value Grouping for Lazy Rule Induction
[4]	Granulation in Analogy-based Classification

Table 3. Tags (concept labels) for three examples of clusters

Cluster 21: Neurocomputing and Artificial Neural Networks	
[1]	"ADALINE"
[2]	"Artificial_neural_network"
[3]	"Artificial_neuron"
[4]	"Auto-encoder"
[5]	"Delta_rule"
[6]	"Multilayer_perceptron"
[7]	"Universal_approximation_theorem"
[8]	"Echo_state_network"
[9]	"Neural_Lab"
Cluster 39: Logical Satisfiability and Validity of Formulas	
[1]	"Empty_domain"
[2]	"Formal_theorem"
[3]	"Limit-preserving_function_(order_theory)"
[4]	"Satisfiability_and_validity"
[5]	"Schaefer's_dichotomy_theorem"
[6]	"Tautology_(logic)"
[7]	"Well-definition"
Cluster 60: Instance-based Learning	
[1]	"Attribute_(computing)"
[2]	"Attribute_(network_management)"
[3]	"Integrity_constraints"
[4]	"K-nearest_neighbor_algorithm"
[5]	"Online_machine_learning"
[6]	"Relation_(database)"
[7]	"Structured_SVM"

highly specialized vocabulary that inadvertently biases the bag-of-words representation.

We have also investigated whether the bag-of-concepts representation may be used for the purpose of automated tagging (labeling) of clusters. For this purpose we associated each group (cluster) of articles with DBpedia concepts that appear in representations of at least 80% of its members. Table 3 presents these associations for the three exemplary clusters. From this example one can see that the selected concepts (cluster tags) are well in line with cluster labels assigned by the experts. Unfortunately, they seem to be too specific to express the semantic relatedness of the documents in the cluster by themselves. To overcome this issue, in the future we plan to employ knowledge about DBpedia categories and the structure of concepts to construct more general tags.

6 Conclusions and Further Work

The conclusions drawn from this case study, just like the motivations presented in Section 2, are of two kinds.

Firstly, we can draw conclusions regarding the structure and characteristics of the corpus of 349 rough set related documents that were used as the basis for the study. The experimental results confirm, that our text corpus is fairly uniform and focused. It is quite clear that our articles share a lot of common concepts at the same time being separable from other research areas. Within the area of rough sets, the papers can be arranged into groups (clusters) in a really meaningful manner.

The second conclusion is that the proposed approach to clustering, based on ESA approach, has a significant potential and shall be seriously considered as an element in the future studies. During the experiments it was possible to establish some ground knowledge about features of the method used. That gives us some confidence about the viability of this approach and its potential to become an element of the prototype software solution that we are eyeing in frame of our main (SYNAT) project.

As usual with this kind of experimental study, there is a plethora of things we can do next. In shorter perspective, next steps should include testing more clustering methods, playing with parameters, coefficients and so on, to obtain more optimal and versatile solution. Also, it would be very interesting to investigate whether including more knowledge from DBpedia, for instance structural information about categories, helps to improve results or not. Another natural next step is to extend the corpus of (currently 349) documents and check if our findings remain valid for larger data set. This step, however, requires the access to larger sources of PDF documents. It may be possible to have more documents from other research areas in the future and then perform the comparative study. We hope that with progress of the overall SYNAT project we will obtain more material (documents) from other co-operating partners.

Another possible direction of a continuation of this study may regard different methods for assessment of similarity between pairs of scientific documents. Currently, only the cosine distance is being used. That fact restrains our ability to detect semantically similar texts since it enforces potentially undesirable properties of a metric on the similarity measure. We believe, that in order to capture more semantic resemblance of articles, the similarity measure should be more dependent on a domain from which the documents come. One way to achieve that is through utilization of some similarity learning methods, such as the Rule-based Similarity model described in [6].

In a long run, the follow-up of this study should produce a software module that could serve as a part of a search/recommender system supporting development of information platform in the SYNAT project. For that to happen the tools that we have used to hand-craft the experiments in this study will have to be implemented as (semi-)automatic software modules that will be integrated with the prototype of main system. This will require not only, re-writing of some code but, quite possibly, re-designing some algorithms (e.g., calculation of reverse index) in order to make them computationally efficient even for processing very large number of documents at once. We are currently considering using

the SuperMatrix library and tools (cf. [2]) as a candidate for foundation of such implementation.

References

1. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia – a crystallization point for the web of data. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web* 7, 154–165 (2009)
2. Broda, B., Jaworski, D., Piasecki, M.: Parallel, massive processing in SuperMatrix - a general tool for distributional semantic analysis of corpus. In: *Proceedings of International Multiconference on Computer Science and Information Technology - IMCSIT 2010*, pp. 373–379 (2010)
3. Feldman, R., Sanger, J. (eds.): *The Text Mining Handbook*. Cambridge University Press (2007)
4. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In: *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pp. 6–12 (2007)
5. Grochowalski, P., Suraj, Z.: RSDS - the Rough Set Database System - a bibliographic database on wide aspects of rough sets (2009), <http://rsds.univ.rzeszow.pl/>
6. Janusz, A.: Utilization of dynamic reducts to improve performance of the rule-based similarity model for highly-dimensional data. In: *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and International Conference on Intelligent Agent Technology - Workshops*, pp. 432–435. IEEE (2010)
7. Jones, K.S., Willet, P.: *Readings in Information Retrieval*. Morgan Kaufmann, San Francisco (1997)
8. Maguitman, A.G., Menczer, F., Roinestad, H., Vespignani, A.: Algorithmic detection of semantic similarity. In: Ellis, A., Hagino, T. (eds.) *WWW*, pp. 107–116. ACM (2005)
9. Oleshchuk, V.A., Pedersen, A.: Ontology based semantic similarity comparison of documents. In: *DEXA Workshops*, pp. 735–738. IEEE Computer Society (2003)
10. R Development Core Team: *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria (2009), <http://www.R-project.org>
11. Shinyama, Y.: *PDFMiner: Python PDF parser and analyzer* (2010), <http://www.unixuser.org/~euske/python/pdfminer/>
12. Tan, P.N., Steinbach, M., Kumar, V.: *Introduction to Data Mining*. Addison Wesley, Boston (2006), <http://www-users.cs.umn.edu/~kumar/dmbook/index.php>
13. The DBpedia Community: *The DBpedia knowledge base* (2011), <http://DBpedia.org/>
14. Varelas, G., Voutsakis, E., Raftopoulou, P., Petrakis, E.G.M., Milios, E.E.: Semantic similarity methods in wordnet and their application to information retrieval on the web. In: Bonifati, A., Lee, D. (eds.) *WIDM*, pp. 10–16. ACM (2005)
15. Wikipedia Community: *Wikipedia - the free Encyclopedia* (2011), <http://en.wikipedia.org/>

Extended Document Representation for Search Result Clustering*

S. Hoa Nguyen, Wojciech Świeboda, and Grzegorz Jaskiewicz

Faculty of Mathematics, Informatics, and Mechanics, The University of Warsaw
Banacha 2, 02-097 Warsaw, Poland
hoa@mimuw.edu.pl, wswieb@mimuw.edu.pl, jaskiewicz@mimuw.edu.pl

Abstract. Organizing query results into clusters facilitates quick navigation through search results and helps users to specify their search intentions. Most meta-search engines group documents based on short fragments of source text called *snippets*. Such a model of data representation in many cases shows to be insufficient to reflect semantic correlation between documents. In this paper, we discuss a framework of document description extension which utilizes domain knowledge and semantic similarity. Our idea is based on application of Tolerance Rough Set Model, semantic information extracted from source text and domain ontology to approximate concepts associated with documents and to enrich the vector representation.

Keywords: Text mining, semantic clustering, DBpedia, document grouping, PubMed, bibliometric measure.

1 Introduction

Although the performance of search engines is improving every day, searching the Web can be a tedious and time-consuming task because (1) search engines can index only a part of the “indexable Web” due to its huge size and a highly dynamic nature, and (2) the user’s intention is not clearly expressed in general, short queries. In effect, a search engine may return as much as hundreds of thousands of relevant documents. One approach to manage the large number of results is clustering. The concept arises from document clustering in information retrieval domain: find a grouping for a set of documents so that documents belonging to the same cluster are similar and documents belonging to different clusters are dissimilar. Search results clustering can thus be defined as a process of automatic grouping of search results into thematic groups and discovering concise descriptions of these groups. Clustering of search results can help the

* The authors are supported by the grant N N516 077837 from the Ministry of Science and Higher Education of the Republic of Poland and by the National Centre for Research and Development (NCBiR) under Grant No. SP/I/1/77065/10 by the strategic scientific research and experimental development program: “Interdisciplinary System for Interactive Scientific and Scientific-Technical Information”.

user navigate through a large set of documents more efficiently and help users specify their intentions. By providing concise, accurate descriptions of clusters, it lets users localize interesting documents faster. Most notably, document clustering algorithms are designed to work on relatively large collections of full-text documents, as opposed to search results clustering algorithms, which are supposed to work on a set of short text descriptions (usually between 10 and 20 words).

The earliest works on clustering results were done by Hearst and Pedersen on scather/gather system [3], followed by an application to Web documents and search results by Zamir and Etzioni [16] to create groups based on novel algorithm suffix tree clustering. Inspired by their work, a Carrot framework was created by Weiss [2][14] to facilitate research on clustering search results. This has encouraged others to contribute new clustering algorithms under the Carrot framework like LINGO [6] and AHC [15]. The main problem occurred in all mentioned works is the fact that many snippets remain unrelated with a genuine content of documents because of their short representation.

Several works in the past have been devoted to the problem of document description enrichment. In [5] a method of snippet extension was investigated. The main idea was based on application of collocation similarity measure to enrich the vector representation. In [12], the author presented a method of associating terms in document representation with similar concepts drawn from domain ontology.

In this paper, we present a generalized scheme for the problem mentioned above. We investigate two levels of extensions. The first one is related to extending a concept space for data representation and the second one is related to associating terms in the concept space with semantically related concepts. The first extension level is performed by incorporating semantic information extracted from a document content (such as citations) or from document meta-data (like authors, conferences). The second one is achieved by application of Tolerance Rough Set Model [11] and domain ontology, in order to approximate concepts existing in document description and to enrich the vector representation of the document.

The paper is organized as follows. In the second section we present a framework of **Tolerance Rough Set Model** (TRSM). In Section 3 we discuss the application of *Generalized* TRSM to enriching document descriptions, whereas section 4 is devoted to experiments, followed by conclusions in section 5.

2 Generalized Approximation Space and TRSM

Rough set theory was originally developed [7] as a tool for data analysis and classification. It has been successfully applied in various tasks, such as feature selection/extraction, rule synthesis and classification [4]. In this chapter we will present fundamental concepts of rough sets with illustrative examples. Some extensions of the Rough Set model are described, concentrating on the use of rough sets to synthesize approximations of concepts from data.

Consider a non-empty set of object U called the universe. Suppose we want to define a concept over the universe of objects U . Let us assume that the concept can be represented as a subset X of U . The central point of Rough Set theory is the notion of set approximation: any set in U can be approximated by its *lower* and *upper approximation*.

2.1 Generalized Approximation Spaces

The classical Rough Set theory is based on equivalence relation that divides the universe of objects into disjoint classes. By definition, an equivalence relation $R \subseteq U \times U$ is required to be reflexive, symmetric, and transitive. Practically, in some applications, the requirement for equivalent relation has shown to be too strict. Concepts arising in many domains are by their nature imprecise and overlapping eachother. For example, let us consider a collection of scientific documents and keywords describing those documents. It is clear that each document can be assigned several keywords and a keyword can be associated with many documents. Thus, in the universe of documents, keywords can form overlapping classes.

Skowron [11] has introduced a generalized tolerance space by relaxing the relation R to a tolerance relation, where transitivity property is not required. Formally, the generalized approximation space is defined as a quadruple $\mathcal{A} = (U, I, \nu, P)$, where

1. U is a non-empty set of objects (*a universe*).
2. $I : U \rightarrow \mathcal{P}(U)$ is an *uncertainty function* ($\mathcal{P}(U)$ is a set of all subsets of U) satisfying conditions:

- $x \in I(x)$ for $x \in U$
- $y \in I(x) \iff x \in I(y)$, for any $x, y \in U$.

Thus, the relation $xRy \iff y \in I(x)$ is a *tolerance relation* (i.e. reflexive, symmetric) and $I(x)$ is a *tolerance class* of x .

3. $\nu : \mathcal{P}(U) \times \mathcal{P}(U) \rightarrow [0, 1]$ is a *vague inclusion function*.

Vague inclusion ν is a kind of membership function but extended to functions over $\mathcal{P}(U) \times \mathcal{P}(U)$ to measure the degree of inclusion between two sets. Vague inclusion must be *monotonic* with respect to the second argument, i.e., if $Y \subseteq Z$ then $\nu(X, Y) \leq \nu(X, Z)$ for $X, Y, Z \subseteq U$.

4. $P : I(U) \rightarrow \{0, 1\}$ is a *structurality function*.

The introduction of structurality function $P : I(U) \rightarrow \{0, 1\}$ ($I(U) = \{I(x) : x \in U\}$) allows us to enforce additional global conditions on sets $I(x)$ considered to be approximated. In generation of approximations, only sets $X \in I(U)$ for which $P(X) = 1$ (referred to as *P-structural element* in U) are considered. For example, a function $P_\alpha(X) = 1 \iff |X|/|U| > \alpha$ will discard all subsets that are relatively smaller than certain percentage (given by α) of U .

Together with uncertainty function I , vague inclusion function ν defines the *rough membership function* for $x \in U, X \subseteq U$:

$$\mu_{I, \nu}(x, X) = \nu(I(x), X)$$

Lower and upper approximations in \mathcal{A} of any $X \subseteq U$ are then defined as

$$\mathbf{L}_{\mathcal{A}}(X) = \{x \in U : P(I(x)) = 1 \wedge \nu(I(x), X) = 1\} \quad (1)$$

$$\mathbf{U}_{\mathcal{A}}(X) = \{x \in U : P(I(x)) = 1 \wedge \nu(I(x), X) > 0\} \quad (2)$$

With the definition given above, generalized approximation spaces can be used in any application where I , ν and P are appropriately determined.

2.2 Tolerance Rough Set Model

Tolerance Rough Set Model (TRSM) was developed [10,13] as a basis to model documents and terms in Information Retrieval, Text Mining, etc. With its ability to deal with vagueness and fuzziness, Tolerance Rough Set Model seems to be a promising tool to model relations between terms and documents. In many Information Retrieval problems, especially in document clustering, defining the relation (i.e. similarity or distance) between document-document, term-term or term-document is essential. In Vector Space Model, it has been noticed [13] that a single document is usually represented by relatively few terms¹. This results in zero-valued similarities which decreases quality of clustering. The application of TRSM in document clustering was proposed as a way to enrich document and cluster representation with the hope of increasing clustering performance.

The idea is to capture conceptually related index terms into classes. For this purpose, the tolerance relation R is determined as the co-occurrence of index terms in all documents from D . The choice of co-occurrence of index terms to define tolerance relation is motivated by its meaningful interpretation of the semantic relation in context of IR and its relatively simple and efficient computation.

Let $D = \{d_1, \dots, d_N\}$ be a set of documents and $T = \{t_1, \dots, t_M\}$ set of *index terms* for D . With the adoption of Vector Space Model [1], each document d_i is represented by a weight vector $[w_{i1}, \dots, w_{iM}]$ where w_{ij} denotes the weight of term t_j in document d_i . TRSM is an approximation space $\mathcal{R} = (T, I_\theta, \nu, P)$ determined over the set of terms T as follows:

- **Uncertainty Function:** The parameterized uncertainty function I_θ is defined as

$$I_\theta(t_i) = \{t_j \mid f_D(t_i, t_j) \geq \theta\} \cup \{t_i\}$$

where $f_D(t_i, t_j)$ denotes the number of documents in D that contain both terms t_i and t_j and θ is a parameter set by an expert.

The set $I_\theta(t_i)$ is called the *tolerance class* of index term t_i .

- **Vague Inclusion Function:** To measure degree of inclusion of one set in another, the vague inclusion function is defined as is defined as

$$\nu(X, Y) = \frac{|X \cap Y|}{|X|}$$

It is clear that this function is monotone with respect to the second argument.

¹ In other words, the number of non-zero values in document's vector is much smaller than vector's dimension – the number of all index terms.

Table 1. Tolerance classes of terms generated from 200 snippets return by Google search engine for a query “jaguar” with $\theta = 9$;

Term	Tolerance class	Document frequency
Atari	Atari, Jaguar	10
Mac	Mac, Jaguar, OS, X	12
onca	onca, Jaguar, Panthera	9
Jaguar	Atari, Mac, onca, Jaguar, club, Panthera, new, information, OS, site, Welcome, X, Cars	185
club	Jaguar, club	27
Panthera	onca, Jaguar, Panthera	9
new	Jaguar, new	29
information	Jaguar, information	9
OS	Mac, Jaguar, OS, X	15
site	Jaguar, site	19
Welcome	Jaguar, Welcome	21
X	Mac, Jaguar, OS, X	14
Cars	Jaguar, Cars	24

- **Structural Function:** All tolerance classes of terms are considered as structural subsets: $P(I_\theta(t_i)) = 1$ for all $t_i \in T$.

The membership function μ for $t_i \in T$, $X \subseteq T$ is then defined as $\mu(t_i, X) = \nu(I_\theta(t_i), X) = \frac{|I_\theta(t_i) \cap X|}{|I_\theta(t_i)|}$ and the lower and upper approximations of any subset $X \subseteq T$ can be determined – with the obtained tolerance $\mathcal{R} = (T, I, \nu, P)$ – in the standard way

$$\mathbf{L}_{\mathcal{R}}(X) = \{t_i \in T \mid \nu(I_\theta(t_i), X) = 1\}$$

$$\mathbf{U}_{\mathcal{R}}(X) = \{t_i \in T \mid \nu(I_\theta(t_i), X) > 0\}$$

2.3 Example

Consider a universe of unique terms extracted from a set of search result snippets returned from Google search engine for a “famous” query: **jaguar**, which is frequently used as a test query in information retrieval because it is a polysemy, i.e., a word that has several meanings, especially in the Web. The word jaguar can have the following meanings:

- jaguar as a cat (*panthera onca* - <http://dspace.dial.pipex.com/agarman/jaguar.htm>);
- jaguar as a car;
- jaguar was a name for a game console made by Atari - <http://www.atari-jaguar64.de>;
- it is also a codename for Apple’s newest operating system MacOS X - <http://www.apple.com/macosx>.

Tolerance classes are generated for threshold $\theta = 9$. It is interesting to observe (Table [11](#)) that generated classes reveal different meanings of the word “jaguar”: a cat, a car, a game console, an operating system and some more.

In the context of Information Retrieval, a tolerance class represents a concept that is characterized by terms it contains. By varying the threshold θ , one can control the degree of relatedness of words in tolerance classes (or the preciseness of the concept represented by a tolerance class).

One interpretation of given approximations is as follows: if we treat X as a concept described vaguely by index terms it contains, then $\mathbf{U}_{\mathcal{R}}(X)$ is the set of concepts that share some semantic meanings with X , while $\mathbf{L}_{\mathcal{R}}(X)$ is a “core” concept of X .

3 Applications of TRSM in Semantic Search

Tolerance Rough Set Model was applied to many text mining problems, including document clustering, search result clustering [\[5,10\]](#), and automatic syllabus generation. One can list the three most important applications of TRSM in text mining:

- Enriching document representation;
- Extended weighting scheme;
- TRSM based clustering algorithms.

In this paper we propose a new application of TRSM in Semantic Search, one actually developed within SYNAT project. The idea is to extend the representation of documents (or snippets) by additional semantic information extracted from text sources and/or from meta-data like citations, authors, publishers, publication years and semantic concepts related to the document. Three Tolerance Rough Set models will be discussed in this section: standard TRSM, extended TRSM by citations and extended TRSM by semantic concepts. We also present, how to apply these models to enrich documents’ representation.

3.1 Standard TRSM

Let $D = \{d_1, \dots, d_N\}$ be a set of documents and $T = \{t_1, \dots, t_M\}$ the set of *index terms* for D . The tolerance rough set model for term space was described in previous Section.

$$\mathcal{R}_0 = (T, I_\theta, \nu, P)$$

In this model a document, which is associated with a *bag of words/terms*, is represented by its upper approximation, i.e. the document $d_i \in D$ is represented by

$$\mathbf{U}_{\mathcal{R}}(d_i) = \{t_i \in T \mid \nu(I_\theta(t_i), d_i) > 0\}$$

The extended weighting scheme is inherited from the standard TF-IDF by:

$$w_{ij}^* = \begin{cases} (1 + \log f_{d_i}(t_j)) \log \frac{N}{f_D(t_j)} & \text{if } t_j \in d_i \\ 0 & \text{if } t_j \notin \mathbf{U}_{\mathcal{R}}(d_i) \\ \min_{t_k \in d_i} w_{ik} \frac{\log \frac{N}{f_D(t_j)}}{1 + \log \frac{N}{f_D(t_j)}} & \text{otherwise} \end{cases}$$

Table 2. Example snippet and its two vector representations in standard TRSM

Title: EconPapers: Rough sets bankruptcy prediction models versus auditor			
Description: Rough sets bankruptcy prediction models versus auditor signalling rates. Journal of Forecasting, 2003, vol. 22, issue 8, pages 569-586. Thomas E. McKee. ...			
Original vector		Enriched vector	
Term	Weight	Term	Weight
auditor	0.567	auditor	0.564
bankruptcy	0.4218	bankruptcy	0.4196
signalling	0.2835	signalling	0.282
EconPapers	0.2835	EconPapers	0.282
rates	0.2835	rates	0.282
versus	0.223	versus	0.2218
issue	0.223	issue	0.2218
Journal	0.223	Journal	0.2218
MODEL	0.223	MODEL	0.2218
prediction	0.1772	prediction	0.1762
Vol	0.1709	Vol	0.1699
		applications	0.0809
		Computing	0.0643

The extension ensures that each term occurring in the upper approximation of d_i but not in d_i itself has a weight smaller than the weight of any terms in d_i . Normalization by vector's length is then applied to all document vectors:

$$w_{ij}^{new} = \frac{w_{ij}^*}{\sqrt{\sum_{t_k \in d_i} (w_{ij}^*)^2}}$$

The example of standard TRSM is presented in Table 2.

3.2 Extended TRSM by Citation

Let $D = \{d_1, \dots, d_N\}$ be a set of documents and $T = \{t_1, \dots, t_M\}$ the set of *index terms* for D . Let $B = \{b_1, \dots, b_K\}$ be the set of bibliography items that are cited by documents from D .

The extended tolerance rough set model for terms and citations is a pair:

$$\mathcal{R}_1 = (\mathcal{R}_T, \mathcal{R}_B)$$

where

$\mathcal{R}_T = (T, I_{\theta_T}, \nu, P)$ and $\mathcal{R}_B = (B, I_{\theta_B}, \nu, P)$ are TRSM defined for term space T and bibliography item space B , respectively.

In the extended model, each document $d_i \in D$ is associated with a pair (T_i, B_i) , where T_i is the set of terms that occur in d_i and B_i is the set of

bibliography items cited by d_i . Each document can be represented by a pair of upper approximations, i.e.,

$$d_i \dashrightarrow (\mathbf{U}_{\mathcal{R}_T}(d_i), \mathbf{U}_{\mathcal{R}_B}(d_i))\}$$

where

$$\mathbf{U}_{\mathcal{R}_T}(d_i) = \{t_i \in T \mid \nu(I_{\theta_T}(t_i), d_i) > 0\}$$

$$\mathbf{U}_{\mathcal{R}_B}(d_i) = \{b_i \in B \mid \nu(I_{\theta_B}(b_i), d_i) > 0\}$$

Once can observe some properties of the extended model.

- if $B = \emptyset$, the extended model \mathcal{R}_1 is the standard TRSM.
- if $B = \emptyset$ and $\theta_T = \text{card}(D) + 1$, documents in D are represented by their original text without information about citations.
- if $T = \emptyset$ and $\theta_B = \text{card}(D) + 1$, documents in D are represented by bibliography items occurred in the documents without information about an original text.
- if $\theta_T = \theta_B = \text{card}(D) + 1$, each document $d_i \in D$ is represented by an original text and bibliography items cited by d_i .

In this paper we are investigating the last three cases and their influence to the search result clustering problem.

3.3 Extended TRSM by Semantic Concepts

Let $D = \{d_1, \dots, d_N\}$ be a set of documents and $T = \{t_1, \dots, t_M\}$ the set of *index terms* for D . Let $B = \{b_1, \dots, b_K\}$ be the set of bibliography items that are cited by documents from D and let C be the set of concepts from a given domain knowledge (e.g. the concepts from DBpedia).

The extended tolerance rough set model based on both citations and semantic concepts is a tuple:

$$\mathcal{R}_C = (\mathcal{R}_T, \mathcal{R}_B, \mathcal{R}_C, \alpha_n)$$

where

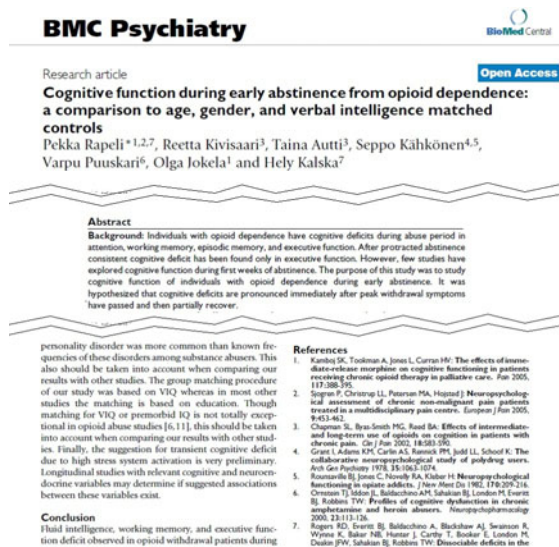
\mathcal{R}_T , \mathcal{R}_B and \mathcal{R}_C are tolerance spaces determined over the set of terms T , the set of bibliography items B and the set of concepts in the knowledge domain C , respectively.

$\alpha_n : \mathcal{P}(T) \longrightarrow \mathcal{P}(C)$ is called the *semantic association* for terms. For any $T_i \subset T$, $\alpha_n(T_i)$ is the set of top n most associated concepts for T_i , see [12].

In this model, each document $d_i \in D$ associated with a pair (T_i, B_i) is represented by a triple:

$$d_i \dashrightarrow (\mathbf{U}_{\mathcal{R}_T}(d_i), \mathbf{U}_{\mathcal{R}_B}(d_i), \alpha_n(T_i))\}$$

In Figure 1 we can see an example of a biomedical article and the list of concepts associated with the article.



Top 20 concepts:

"Opioid antagonist" "Neurocognitive" "Opiate replacement therapy" "Paced Auditory Serial Addition Test" "Withdrawal" "Opioid-induced hyperalgesia" "Methadone" "Benzodiazepine withdrawal syndrome" "Opioid" "7-Hydroxymitragynine" "Nalmefene" "DAMGO" "Cyprenorphine" "RB-101" "Meptazinol" "Post acute withdrawal syndrome" "IC-26" "Euphoriant" "Cognitive deficit" "Cognitive neuropsychology"

Fig. 1. An example of an article and the list of top 20 concepts that are related to the article

4 Case Study: Search Result Clustering of Biomedical Articles

The extended TRSM models described in the previous section are applied to enrich descriptions of biomedical articles from the PubMed Central database [9]. The purpose of this section is to evaluate document representation models. In experiments we investigate the following document representations:

- Abstract based representation.
- Citation based representation.
- Semantic concept based representation.
- Abstract enriched by citation.

These models are applied to a *search result clustering problem*. By analyzing cluster quality one can evaluate document representation models.

4.1 Data Sources

PubMed Central [9] (PMC) is a free online archive of journal articles in biomedicine and life sciences. A subset of this database, PMC Open Access subset, consists of articles available under Creative Commons license or similar, thus may be downloaded in bulk from PMC. This subset contains 200000 articles (roughly 10% of the PMC database) and provides a base text corpus for our experiments, further restricted by specific search queries. All articles are provided

along with rich metadata from MEDLINE database, and most articles in MEDLINE/PubMed database are indexed with MeSH (Medical Subject Headings), a controlled vocabulary. MeSH terms are assigned to texts by subject experts.

The second data source that we utilize in our experiments, one which provides an alternative document representation is DBpedia. The interested reader will find further details in [12], but for the purpose of this article it suffices to think of an abstract module which takes as the input an article and provides a list of DBpedia entries associated with this article, along with degrees of association.

4.2 Experiment Set-Up

The aim of our experiments is to explore clusterings induced by different document representations (lexical, semantic and structural). To cluster documents we adopt an algorithm LINGO in a Carrot clustering library [6]. The main idea of the algorithm is to apply a Singular Value Decomposition (SVD) method to document indexing. One of the advantages of LINGO is the ability to assign labels to clusters.

The diagram of our experiments is shown in Figure 2. An experiment path (from querying to search result clustering) consists of three stages:

- Search and filter documents matching to a query. Documents in a search result list are represented by *snippets* and/or *titles*.
- Extend representations of the documents by *citations* and/or *semantically similar concepts* from DBpedia.
- Cluster the document search results.

At this point, we limit our exposure in experimental section to a query: "*hippocampus cortisol*".

This query returned 989 papers from PubMed Central (PMC) search engine. 196 of these documents are available in PubMed Central Open Archive subset (as of this writing). We have further restricted this set to 134 journal articles which were suitable for our investigations (i.e. full text was available in Archiving and Interchange Tag Set format).

We will treat LINGO as a clustering engine. Conceptually, the algorithm takes as an input information about documents in terms of pairs (*title*, *snippet*).

While Carrot² workbench application is integrated with PubMed search engine, we use a Web based search application which directly queries PMC (rather than PubMed) database. For PubMed queries in Carrot, "snippet" field used by the system contains *document abstract* (along with basic document meta-data). Hence, a document representation which induces a grouping that serves as a natural benchmark is one which places document abstracts in "snippet" field. Please keep in mind that the set of PubMed documents clustered directly by Carrot² workbench and those clustered during our investigations differ, as we are limited to PMC Open Archive subset.

We have investigated various document representations: based on abstracts, based on citations (other articles referenced in bibliography as well as articles

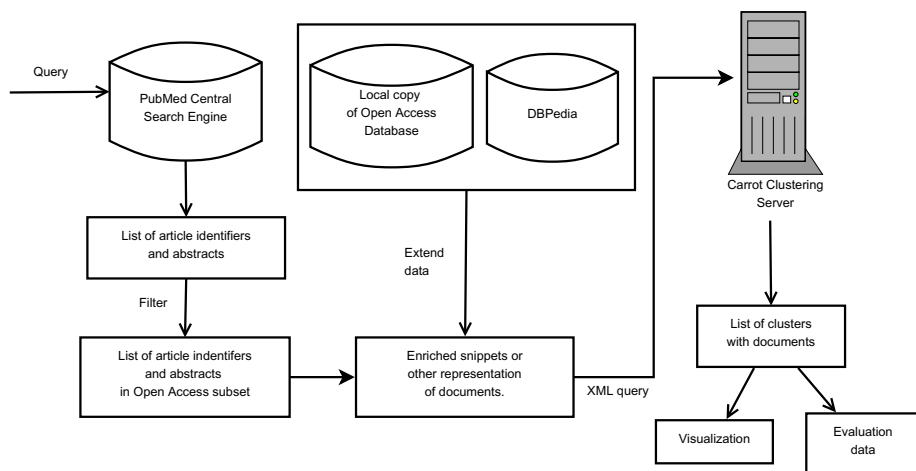


Fig. 2. Experiment diagram

that reference a given paper) and based on DBpedia entries semantically similar to a given document [12]. We have also considered mixed representations (e.g. abstracts augmented with citations) and experimented with varying number of DBpedia entries used in document representation.

Despite a slightly broader coverage of our work, on most graphs that follow, for the purpose of clarity we limit our exposure to selected representative document representations (and hence clusterings).

4.3 Results of Experiments

The subsequent subsections will focus on the following analyses:

- First, we will take a look at an example cluster that was only discovered when document representation was enriched with information about citations.
- Secondly, we will briefly look at stability of resulting clusters with respect to document representation. How does a clustering change if a baseline representation (based on abstracts) is enriched by including citations? How does it change when the representation is replaced with one based on DBpedia concepts?
- Next, we will see whether there is any common information in disjoint document representations.
- We will perform validation of clustering results using MeSH terms associated with articles.
- Finally, we will take a look at certain structural properties of resulting clusterings.

4.4 Example Cluster

Table 3 shows an example cluster (labeled “Body Weight”) discovered after extending baseline document representation with citation information. In

Table 3. A cluster labeled “Body Weight”, discovered after baseline document representation was extended with citation information. Column “Grouping (abstract)” shows original (baseline) groups assigned to each document (two of them were previously unassigned to any group), whereas the third column lists MeSH terms associated with each document (these terms were unavailable for the fourth document). We have emphasised concepts that seem (subjectively) similar to the group label.

Title	Grouping (abstracts)	MeSH keywords
Effects of antenatal dexamethasone treatment on glucocorticoid receptor and calcyon gene expression in the prefrontal cortex of neonatal and adult common marmoset monkeys.	Molecular; Dexamethasone	Age Factors; Animals; Animals, Newborn; Body Size; Body Weight ; Callithrix; Dexamethasone; Female; Glucocorticoids; Male; Membrane Proteins; Prefrontal Cortex; Pregnancy; Prenatal Exposure Delayed Effects; Receptors, Glucocorticoid; Receptors, Mineralocorticoid; RNA, Messenger
The body politic the relationship between stigma and obesity-associated disease.		Adiposity; Age Factors; Body Mass Index ; Electric Impedance; Female; Humans; Male; Obesity ; Prejudice; Risk Factors; Sex Factors; Stress, Psychological
Prenatal Stress or High-Fat Diet Increases Susceptibility to Diet-Induced Obesity in Rat Offspring.	High-fat Diet	Animals; Child; Diabetes Mellitus, Type 2; Dietary Fats; Energy Intake ; Female; Genetic Predisposition to Disease; Humans; Infant; Male; Obesity ; Pregnancy; Prenatal Exposure Delayed Effects; Rats; Rats, Sprague-Dawley
The TNF-System Functional Aspects in Depression, Narcolpdfy and Psychopharmacology.		

a nutshell, this example illustrates our core goal, which is to provide additional, meaningful clusters, which would guarantee high coverage (a small number of unassigned documents left in the result set). We will return to this problem in further analyses and show a trade-off between coverage and specificity of a clustering.

4.5 Clustering Stability

The general framework for comparing different clustering methods that underlies analysis in this section (and two subsequent subsections) is to consider similarity relations induced by these groupings. In other words, we work in the space of pairs of documents and label each pair as “similar” or “dissimilar” according to a given clustering. We interpret documents belonging to “Other topics” group as

dissimilar². If either clustering plays the role of a benchmark or a baseline, the similarity relation induced by this clustering may be considered a decision class, whereas the similarity relation induced by other clustering(s) may be interpreted as a classification model(s).

The goal of a clustering based on a different representation is never to reproduce the baseline grouping, but to provide an alternative one. Nevertheless, we can interpret “accuracy” in this context as a metric of stability of the clustering algorithm with respect to document representation. This metric, when applied to clustering comparison (in the space of pairs of documents) is called Rand Index [8].

We will reiterate two questions already mentioned earlier in this paper:

- How does a clustering change if a baseline representation (based on abstracts) is enriched by including citations?
- How does it change when the representation is replaced with one based on DBpedia concepts?

Graph [3] shows Rand Index calculated against the benchmark clustering. Most noticeable is an overall remarkably high Rand Index for most clusterings (around 0.9), while being significantly lower for clustering based on 300 DBpedia entries “closest” to a given article (we refer to this grouping as to “DBpedia-300” from now on, as this particular clustering will be highly illustrative further in the article).

This analysis shows that extended (or alternative) representations result in overall similar clusterings. The more DBpedia concepts used in document description, the less similar the resulting clustering to the baseline.

A care needs to be taken when interpreting the Rand Index, as this metric does not account for different proportions of similar to dissimilar document pairs (the issue is similar to using accuracy as an assessment of classification with rare classes). While most pairs of documents are dissimilar in most models, “DBpedia-300” induces a much “softer” notion of similarity, thus labeling many more pairs of documents as similar.

4.6 Do Different Document Representations Convey Common Information?

Our next analysis focuses on clustering based solely on bibliographical references. For each document, the set of bibliographical references is disjoint with its’ abstract, i.e. bibliographical references correspond to different coordinates in the vector space model than words. We compare similarity matches and mismatches of grouping based on this representation against the benchmark in a contingency table, and calculate Pearson’s Chi-squared test with Yates’ continuity correction. This procedure yields a p-value 3.42×10^{-06} , remarkably low,

² We stress that “Other topics” is an artificial group that consists of articles unassigned to any resulting cluster. One could argue that it conceptually corresponds to a set of singleton clusters.

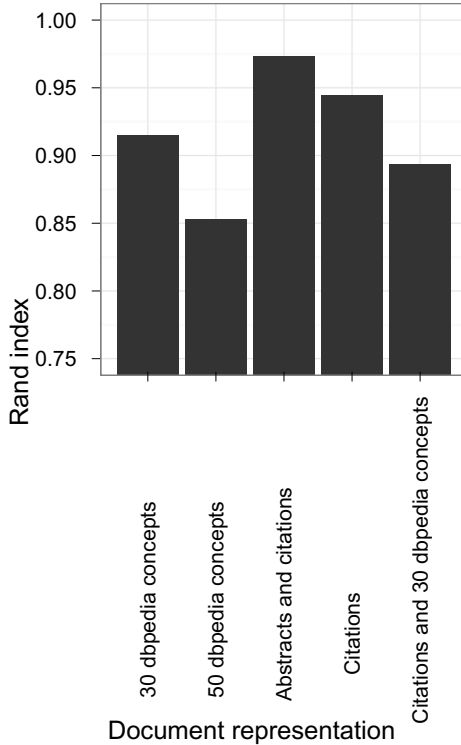


Fig. 3. Rand Index w.r.t. clustering based on abstracts (the benchmark)

despite the general tendency of Yates' continuity correction to underestimate statistical significance. Thus, we conclude that clusterings based on abstracts and those based on references/citations are not independent, with confidence far exceeding 99% ("far" ratio-wise, rather than in absolute terms). However trivial this observation may be, structural content of articles used in our experiment significantly overlaps with their lexical content.

4.7 Validation Using MeSH Vocabulary

Figure 4 shows the results of validation of clusterings using MeSH terms associated with each article. Since these tags were not used in either document representation, they provide a natural confirmatory source of information. Nevertheless, it is important to stress that these terms are not assigned to articles automatically, but by subject analysts.

For each pair of articles we define their distance as the fraction of MeSH terms associated with exactly one of these articles to the overall number of MeSH terms associated with either of these articles. Figure 4 shows the average distance between pairs of documents within the same clusters, with the average taken over all such pairs (rather than over clusters).

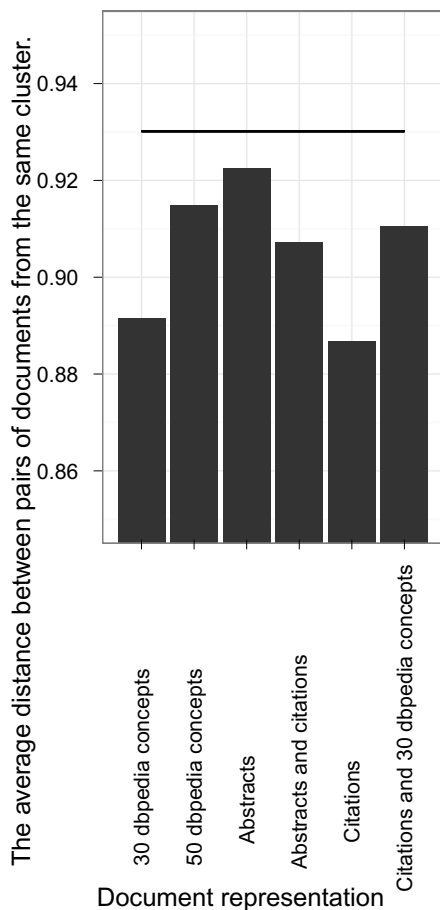


Fig. 4. Validation of clusterings based on different document representations using MeSH terms associated with articles. For each document representation (and hence clustering), we calculate the average distance between documents belonging to the same cluster. The average is taken over all such pairs rather than over all clusters. The vertical line corresponds to the average distance between two documents taken at random.

As the graph suggests, all document representations investigated in our analyses lead to plausible clusterings. However, in order to formulate conclusions of a comparative nature, we have yet to conduct experiments using different search queries. Moreover, structural information provided with MeSH can be used to define various distance metrics between these terms, and each such a metric can be extended to a distance between documents. Further yet, we plan to use other evaluation metrics, although care needs to be taken, as the clustering algorithm under consideration does not output disjoint clusters, neither are all documents assigned to a cluster.

4.8 Structural Properties of Clusterings

The left plot on figure 5 shows that “DBpedia-300” induces a very vague similarity between documents, as each document is assigned to approximately three different clusters on the average. Remaining clusterings are more conservative in this aspect, although not fully unambiguous either.

Figure 6 shows the trade-off between the “vagueness” of a clustering and the “coverage” (i.e. the number of documents with assigned proper clusters). Clusterings based on 30 (and 50) entries from DBpedia seem to balance this trade-off fairly well. Moreover, they are attractive for a different reason – they provide clusterings that are only indirectly based on document content (e.g. a cluster label may be an entry in DBpedia which is not directly worded in any paper, yet may be highly indicative of a general concept linking a group of articles).

It is worth stressing that a proper balance of these structural properties is crucial for grouping documents in the context of Document Retrieval, while not necessarily in the context of Web Search. When speaking of grouping documents in Web Search, we may restrict our discussion to non-navigational search queries. These queries (due to their vagueness) would usually return a very large number

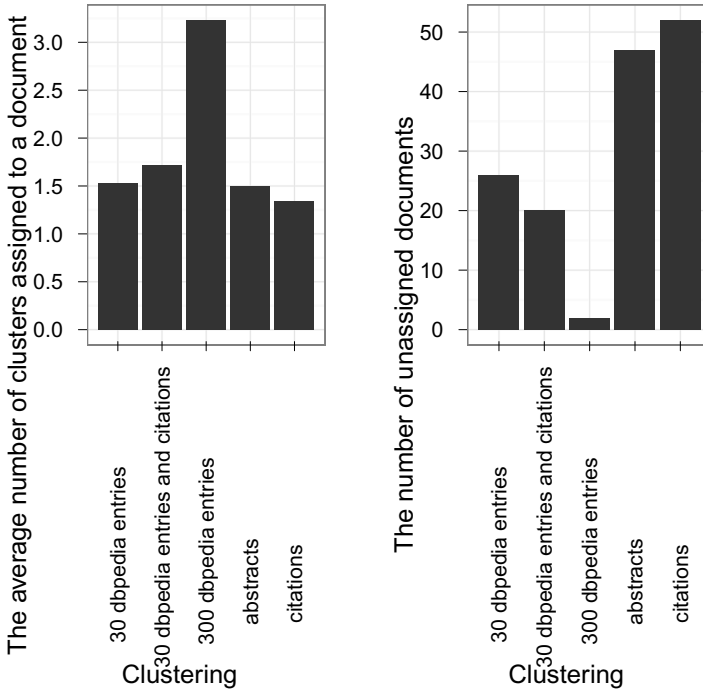


Fig. 5. The average number of clusters assigned to a document (left) and the number of unassigned documents (right) for selected clusterings

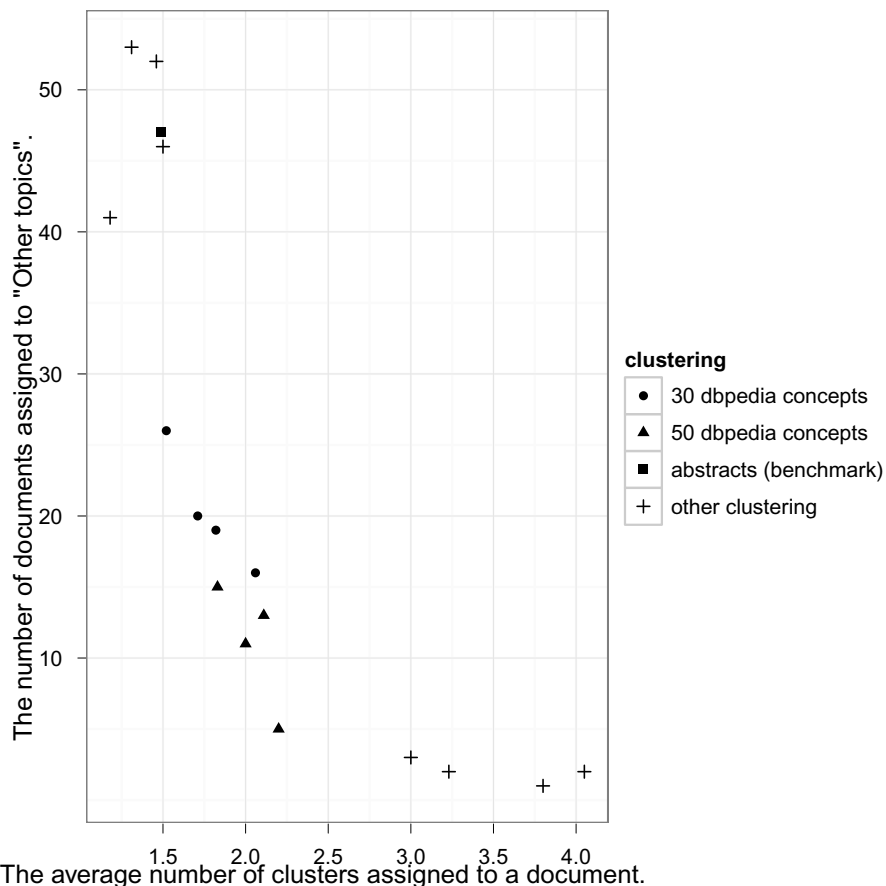


Fig. 6. The trade-off between clustering specificity and coverage. Points to the right correspond to clusterings based on the richest representations – those which include 300 DBpedia concepts.

of matching webpages. A clustering tool usually limits the number of processed snippets to a small subset (e.g. 200) and relies on the interaction with the user to further refine his query or navigate the result set by other means. On the other hand, in Document Retrieval we are always limited to a fixed (often small) number of matching documents, and our intent is to provide plausible grouping which preferably describes the vast majority of them.

5 Conclusions and Further Work

Our preliminary experiments lead to several promising insights, although a wider coverage of experiments need to be conducted in order to speak of definite

conclusions, which would translate to a wide range of queries (or rather, result sets) and text corpora. Nevertheless, our analyses suggest the following:

- Certain documents which would be naturally grouped together do not explicitly share much common lexical content in their abstracts, whereas other document representations convey such information, hence such representations may be used to supplement or provide alternative clusterings.
- The clustering algorithm used in our experiments (LINGO) is stable with respect to the input data. In other words, similarity relations induced by clusterings based on close document representations also yield a high degree of similarity. This suggests that a carefully selected extension method could in fact yield a natural refinement of a baseline clustering.
- Validation using MeSH terms suggests that all document representations investigated in this paper lead to plausible results, although our current analyses focused on clusters without regard to labels. Label evaluation is yet to be conducted.
- Varying richness of document representation displays a trade-off between important structural properties of resulting clusters, namely – the specificity of groups and the number of unassigned documents. Hence, results that confirm closer to a postulated balance may be found if we prepare appropriate document representations.

Our future plans are briefly outlined as follows:

- Use additional information from MeSH vocabulary (e.g. structural information about relationships between terms),
- analyse label quality of clusters resulting from different document representations,
- use MeSH for document representation or label assignment rather than merely for validation of results,
- conduct experiments using other extensions (e.g. citations along with their context; information about authors, institutions, fields of knowledge or time),
- grouping of objects of other types (e.g. authors, institutions, ...),
- visualization of clustering results.

References

1. Baeza-Yates, R., Ribeiro-Neto, B.: *Modern Information Retrieval*, 1st edn. Addison Wesley Longman Publishing Co. Inc. (May 1999)
2. Carpineto, C., Osiński, S., Romano, G., Weiss, D.: A survey of web clustering engines. *ACM Comput. Surv.* 41, 17:1–17:38 (2009)
3. Hearst, M.A., Pedersen, J.O.: Reexamining the cluster hypothesis: Scatter/gather on retrieval results. In: *Proceedings of SIGIR 1996, 19th ACM International Conference on Research and Development in Information Retrieval*, Zürich, CH, pp. 76–84 (1996)
4. Komorowski, J., Pawlak, Z., Polkowski, L., Skowron, A.: *Rough sets: a Tutorial* (1998)

5. Ngo, C.L., Nguyen, H.S.: A method of web search result clustering based on rough sets. In: Skowron, A., Agrawal, R., Luck, M., Yamaguchi, T., Morizet-Mahoudeaux, P., Liu, J., Zhong, N. (eds.) *Web Intelligence*, pp. 673–679. IEEE Computer Society (2005)
6. Osinski, S.: An algorithm for clustering of web search result. Master's thesis, Poznan University of Technology, Poland (June 2003)
7. Pawlak, Z.: *Rough Sets: Theoretical Aspects of Reasoning About Data*. Kluwer, Dordrecht (1991)
8. Rand, W.: Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association* 66(336), 846–850 (1971)
9. Roberts, R.J.: PubMed Central: The GenBank of the published literature. *Proceedings of the National Academy of Sciences of the United States of America* 98(2), 381–382 (2001)
10. Kawasaki, S., Nguyen, N.B., Ho, T.-B.: Hierarchical document clustering based on tolerance rough set model. In: Zighed, D.A., Komorowski, J., Żytkow, J.M. (eds.) *PKDD 2000. LNCS (LNAI)*, vol. 1910, pp. 458–463. Springer, Heidelberg (2000)
11. Skowron, A., Stepaniuk, J.: Tolerance approximation spaces. *Fundamenta Informaticae* 27(2-3), 245–253 (1996)
12. Szczuka, M., Janusz, A., Herba, K.: Clustering of rough set related documents with use of knowledge from dBpedia. In: Yao, J. (ed.) *RSKT 2011. LNCS*, vol. 6954, pp. 394–403. Springer, Heidelberg (2011)
13. Ho, T.B., Nguyen, N.B.: Nonhierarchical document clustering based on a tolerance rough set model. *International Journal of Intelligent Systems* 17(2), 199–212 (2002)
14. Weiss, D.: A clustering interface for web search results in polish and english. Master's thesis, Poznan University of Technology, Poland (June 2001)
15. Wroblewski, M.: A hierarchical www pages clustering algorithm based on the vector space model. Master's thesis, Poznan University of Technology, Poland (July 2003)
16. Zamir, O., Etzioni, O.: Grouper: a dynamic clustering interface to web search results. *Computer Networks* 31(11–16), 1361–1374 (1999)

Semantic Recognition of Digital Documents^{*}

Paweł Betliński, Paweł Gora, Kamil Herba,
Trung Tuan Nguyen, and Sebastian Stawicki

Faculty of Mathematics, Computer Science and Mechanics
University of Warsaw

Banacha 2, 02-097 Warsaw, Poland

pbetl@mimuw.edu.pl, pawelg@mimuw.edu.pl, k.herba@students.mimuw.edu.pl,
nttrung@pjwstk.edu.pl, stawicki@mimuw.edu.pl

Abstract. The paper presents methods developed by the Methods of Semantic Recognition of Scientific Documents group in the research within the scope of the SYNAT project. It describes document representation format together with a proof of concept system converting scientific articles in PDF format into this representation. Another topic presented in the article is an experiment with clustering documents by style.

Keywords: content representation, documents semantics, clustering.

1 Introduction

SYNAT is a large scientific project aiming to create the universal hosting and scientific content storage and sharing platform for academia, education and open knowledge society. The project is carried out by the research consortium comprising major Polish science institutions.

The goal of our research team is to develop methods for semantic recognition of digital documents, e.g.:

- extracting key information, e.g. title, names of authors, affiliations, references,
- splitting document content into sections,
- developing common format for document representation.

The rest of the paper is organized as follows: first we introduce our document representation which is an XML-based format developed by the National Library of Medicine. We will refer to this format as “NXML”. Particularly, we describe basic concepts from the tag suite and experiments conducted with the format in

^{*} The authors are supported by the grant N N516 077837 from the Ministry of Science and Higher Education of the Republic of Poland and by the National Centre for Research and Development (NCBiR) under Grant No. SP/I/1/77065/10 by the strategic scientific research and experimental development program: “Interdisciplinary System for Interactive Scientific and Scientific-Technical Information”.

the SYNAT project. In Section 2 we describe the NXML format and reference experiments (conducted by other research groups) which prove usefulness of the chosen representation. In the next section we present the process of generating NXML representation of the document from the PDF file.

In Section 4 we describe a procedure for clustering documents by styles. It is a preprocessing step which aims to facilitate generating document representation. Documents in each extracted cluster have similar layout, so in the further research it may be helpful in creating a domain knowledge base of document styles. We also point out how to use such base to improve the process of generating NXML representation.

Section 5 concludes the paper and describes plans for the future research.

2 Document Representation

2.1 Representation of the Documents

In the SYNAT project we have encountered a problem of data inconsistency. Our sources contain documents (books and articles) in various formats and this diversification is especially visible on two levels:

- File formats - image formats (jpg, tiff, bmp) representing bitmaps and pixels and documents exchange formats (ps, pdf) focused on presentation layer (used mainly to facilitate reading or navigating).
- Documents differ in layout (even within a given journal we can see layout evolution over time). Locating document meta information (such as title, abstract, keywords, authors or references) is a difficult task and depends on document “layout style”.

Despite having documents in a file-form it is not a trivial task to automatically process them. To solve the problem we have decided to use an abstraction layer of document representation. More precisely, we convert all documents (independently on their origin and initial format) into an intermediate representation - familiar to architects, designers and developers of project further layers. The following subsection describes it in more details.

2.2 NLM Journal Archiving and Interchange Tag Suite

The National Center for Biotechnology Information (NCBI) of the National Library of Medicine (NLM) has faced the problem of creating common format for exchanging textual content like articles or books. They prepared a tool (Journal Archiving and Interchange Tag Suite) which simplifies document structure schema preparation process. From a technical point of view the tag suite is a set of XML schema modules (DTD modules) which can be combined to create a specific (well-fitted to a given task) XML tag set (XML schema).

NCBI/NLM has prepared (by use of their own tag suite) and made available several tag sets for specific purposes. The one on which we focused and which

seems to meet our requirements is called “Journal Archiving and Interchange Tag Set”. It is the most liberal among the NLM/NCBI tag sets in a sense that it allows to express the highest spectrum of document structural notions and concepts. As to the description that appears on the tag suite web page [\[NXML\]](#) this tag set is “Created to enable an archive to capture as many of the structural and semantic components of existing printed and tagged journal material as conveniently as possible, with no effort made to model any particular sequence or textual format”. And as such it looks as a perfect match for SYNAT requirements for an intermediate representation which allows to convey documents structure and semantics. SYNAT codename for this format is NXML.

NXML documentation provides a brief description together with remarks on the usage of elements in the schema. Let us mention a few concepts from [\[NXML\]](#):

<article-title>

“The full title of a journal article or other journal component such as a book review.”

Remarks: “The <article-title> element is used in two contexts: as a part of a metadata concerning the article itself and as a part of a bibliographic reference metadata inside bibliographic citations (<element-citation> and <mixed-citation>).”

Best Practice: “Although this Tag Set cannot enforce either practice, retrieval performance will be enhanced if the subtitle is consistently placed within the <article-title> element (or the <source> element for book titles, proceedings titles, and other titles) for all cited material. Either with a <named-content> or as untagged text, the subtitle is easy to lose to searching.”

<abstract>

“Summarized description of the content of a journal article.”

<aff>

“Name of the institution or organization, such as university or corporation, which is the affiliation for a contributor such as the author or editor.”

<conf-name>

“The full name of the conference, including any qualifiers, such as ‘43rd Annual’.”

<contrib>

“Container element to contain the information (such as name and affiliation) about a single contributor to the article, for example, one author.”

<ref-list>

“List of references (citations) for an article, which is often called ‘References’, ‘Bibliography’, or ‘Additional Reading’.”

<sec>

“A headed group of material; the basic structural unit of the article.”

Remarks: “A very short article, or a simple article such as an editorial or an obituary, may contain nothing but paragraphs and other paragraph-level elements such as figures and tables. But most journal articles are divided into sections, each with a title that describes the content of the section, such as ‘Introduction’, ‘Methodology’, or ‘Conclusions’.”

2.3 Usefulness of the NXML Format

NXML conveys structural and meta information about the document which can be used by other teams in their work. Below we describe shortly two experiments conducted by other groups in the SYNAT project showing usefulness of the chosen representation.

Common Citations Experiment

The aim of the first experiment was to measure similarity between documents basing on common citations [ISMIS]. The general idea is that if documents have common references this might suggest that they treat about similar topics. Another variation of this concept is to look at papers which cite particular article (co-citations). In this experiment documents are used as nodes of a directed graph where references represent edges between them. NXML contains information about citations as particular elements defined in the XML schema which can be translated into the graph structure.

Semantic Clustering Experiment

Another experiment was a semantic clustering of scientific articles related to rough sets [RSKT]. Applied method grouped documents on the basis of their content and with assistance of DBpedia knowledge base [DBpedia]. In the experiment each document had to be converted into a vector representation. Some specific words occurring in certain parts of the document (page headers, footers, references, etc.) could have biased further analysis. Therefore only paragraphs, sections and their titles were extracted. Existence of separate elements in the NXML format representing these objects made this task relatively easy. We modified the program converting PDF files into the NXML format and added text output format containing only relevant parts of documents. Usage of selected kind of data improved the experiment results. They were better in comparison with case of using the whole document text as the input.

3 Retrieving Document Representation

In order to run a search engine or make further analysis and experiments on documents their content needs to be extracted and converted into a well structured format like NXML. We wrote a Python script based on the open-source library PDFMiner [PDFMiner] which is designed to process digital PDF files. Below we describe algorithm in more details.

3.1 Algorithm Steps

layout extraction - It is done by the PDFMiner library which obtains information about characters and their positions in a document. Basing on proximity in x- and y-axis characters are grouped into lines and blocks of text. In the third algorithm step text is classified as a semantic representation of the particular elements of a document.

validation - It is required because we cannot handle all encodings used in PDF files. Validation is done by couple heuristics involving statistics of alphanumeric characters in a document, etc.

layout conversion into the semantic representation - This is the main step in which layout elements are analyzed in order to obtain the actual content of the document. It involves analysis of properties of layout elements like common font, height and width of the line, vertical spaces between them, number of lines in a block of text, etc. Also some kind of analysis on the word level is done. For example words like 'introduction', 'references' or 'conclusion' are useful in detecting examples of headers and inferring the style of the font used in section headers. This knowledge can be applied to find the remaining headers.

conversion of the semantic representation into NXML - It is straightforward because in the third step we find parts of the text which have exact representation as the elements of our XML schema. This step can be replaced to produce some other output format as it was done in case of the experiment with classification of the rough set related documents (Section 2.3). In this particular case text representation was needed. It was easy to add functionality of producing text files containing only selected parts of the documents which were relevant for the experiment.

3.2 Layout Conversion

This step is most involved and it is split into small modules:

1. **Combining text lines which were accidentally split by the PDFMiner.**
2. **Determining number of columns in the text and their typical start and end positions** - This information is useful in finding the text flow on the page. In one of the next steps it allows us to find the paragraphs and order them correctly. Furthermore these statistics are used to determine which blocks of the text represent actual sections of the document. By that we can filter out (or leave for further processing) headers, titles, footers, etc. These unknown elements might be later analyzed by other modules to detect their semantic representation.
3. **Finding blocks of the same font** - It can be useful in finding emphasised elements and titles which are split into multiple lines, etc.
4. **Finding the article title** - We try to imitate human way of thinking. People usually do not have problem with detecting a title of the article. Our

heuristic is that it is usually on the first page before the actual start of the article text and is written with a “big“ font and is clearly separated from other parts of a page.

5. **Splitting article into sections and paragraphs** - Here we use statistics collected in the second step.
6. **Fixing encoding** - In this step we try to fix problems with unknown character encoding. Common example is converting ligatures (e.g., **fi** in “classification”) back to their original (textual) form. These problems are partly resolved with use of the English dictionary which makes it possible to guess the right encoding of some characters by determining whether words created after substitution of missing characters were proper English terms.

4 Clustering Documents by Style

Discovering rules identifying interesting elements of a document is easier for a set of similar (in a sense of similar layout) documents than for the whole corpus. Therefore clustering of the documents based on their styles is an important preprocessing step. The aim is to divide a given set of articles into disjoint groups representing different styles. Even if clustering method is not perfect and as a result we obtain a few styles in one cluster it is still much better than dealing with all documents. Number of rules describing position of particular elements for such cluster should be much smaller than for the whole documents set.

We have developed and tested our methods of clustering mostly on the basis of a small part of BazTech database [BazTech] - over 600 digital PDF documents divided a priori into 6 directories, each representing different style. We had to manually filter out the outliers. Based on this set we could easily check performance of the clustering methods by comparing result clusters with directories representing perfect style clustering. Number of articles in these 6 directories is respectively 88, 116, 72, 87, 6 and 269, so totally we have 638 articles.

Original PDF format is not very convenient to make an analysis of the content, in this case with purpose of clustering documents. Therefore we have decided to use a Python tool PDFMiner [PDFMiner]. We use it to transform PDF documents into an XML files which are the input for our clustering methods.

XML files produced by PDFMiner contain page characters together with their position in document and type of font grouped into text lines. Furthermore text lines are grouped into text boxes. Each text box is visually coherent and distinct from the rest of the document. PDFMiner obtains text boxes using heuristic, which of course does not work perfectly. However, it is still very good comparing to other similar tools which we have tested.

We have considered and tested two main ideas for clustering. The difference between them lies in a way of document representation (inferred from XML format of PDFMiner output).

The first idea is to represent a document by the first few text boxes, describing each of them as a sequence of numbers - sizes of fonts in the text box, written in order of appearance (change of a font is not considered inside a text line - each

text line is represented by the most frequent size of a font). Using this representation we have considered several distance functions and performed for each of them typical clustering methods, like k-medoids and hierarchical clusterization, we have also done visualization with CMS (Classical Multidimensional Scaling) and PCA (Principal Component Analysis). The results were not optimistic - some styles have big variance of their representation, so big that we were not able to clearly distinguish them from the others.

The second idea seems to be more promising. In contrast to the first approach, we try to find more general features of documents sufficient to recognize styles without looking at details which, as we have seen, are sometimes not regular. So far we have realized this idea in a very simple version, where document is represented by 6 attributes, describing: size of a page (width - attribute W , and height - attribute H) and position of a text on a page (coordinates of minimal rectangle covering the whole text on a page - attributes X_0 , Y_0 , X_1 and Y_1).

Figure 1 presents visualization of these attributes. Each picture corresponds to one attribute written below. X-axis represents sequence of all documents. They are written in order: documents from the first directory (first style), documents from the second directory (second style) and so on. Y-axis represents values of considered attribute for documents. Each point on these pictures is a document where X coordinate is a document ID and Y coordinate is an attribute value for this document. Additionally, color of a point corresponds to the directory to which it belongs. From these pictures we can see that chosen attributes should allow us to perform style clusterization of considered documents with a high accuracy.

For comparison, Figure 2 presents one more visualization for attribute expressing size of the most frequent font on a page - which we name F . Here we can see a reason why the first idea for clusterization was not successful. Indeed, it has turned out that for considered set of articles font attributes are significantly less useful (in the clustering process) than position attributes.

For each 'good' attribute: W , H , X_0 , Y_0 , X_1 , Y_1 we performed average-linkage agglomerative clustering, where the number of clusters was chosen using elbow criterion. Final clustering of documents was obtained by intersection of clustering results for each attribute. As the result there are 18 clusters, but 7 biggest represent perfectly desired partition with one exception - directory 1 is represented by 2 clusters (it is also important that remaining 11 clusters represent only 19 articles). Details of obtained partition are following: we have two clusters of size 29 and 56 for directory 1 and then the rest directories from 2 to 6 are represented by single clusters of size respectively 114, 60, 87, 6 and 267.

The results have turned out to be not only much better than for the first representation, but generally very close to the desired partition. However this method should be examined on a bigger set of documents. We expect that proposed document representation will be not sufficient. Therefore we will work on its improvement. For example, one of natural extensions may be considering page geometry in more details including not only general position of the whole text but also positions of text boxes.

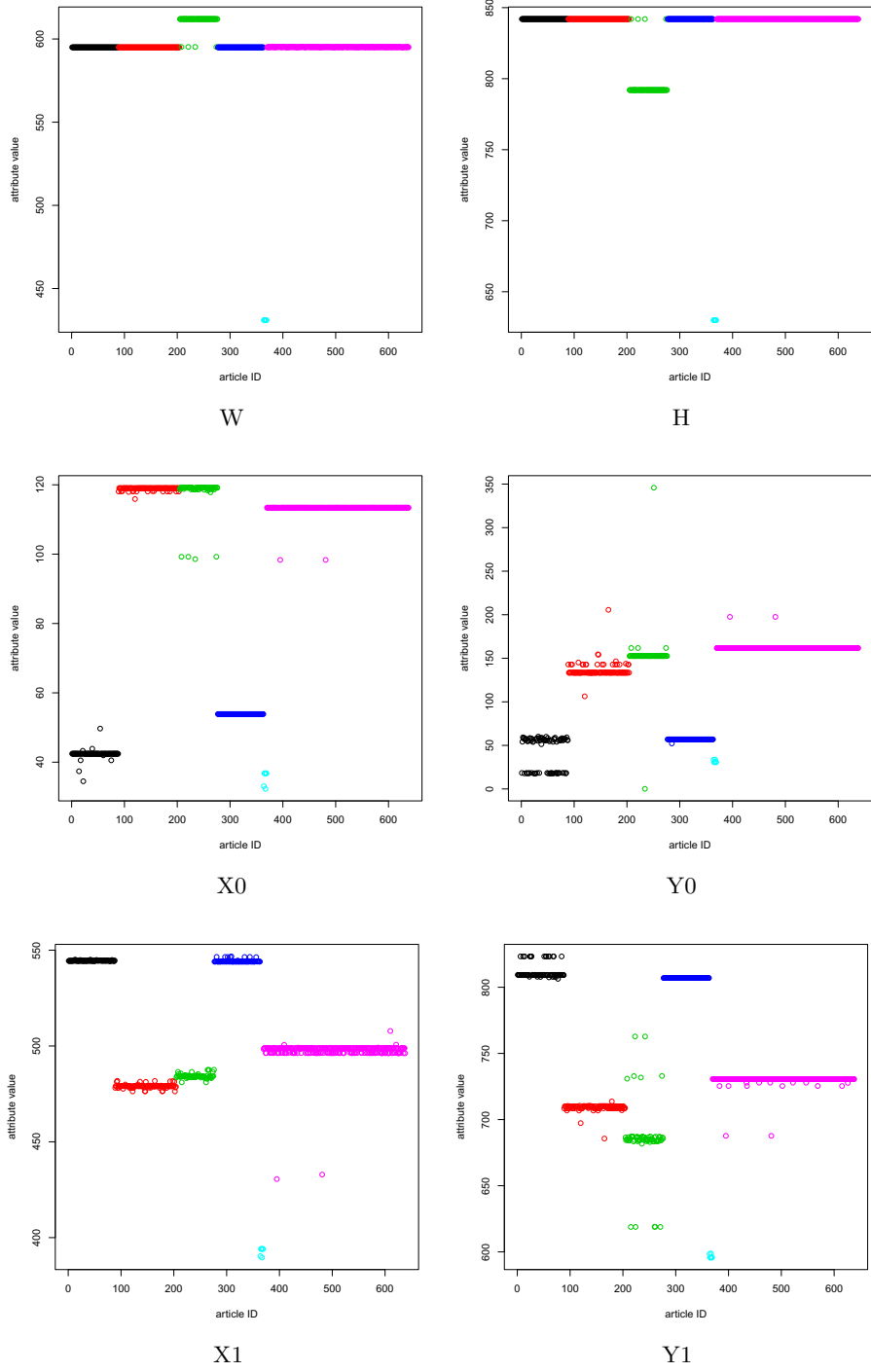


Fig. 1. Visualization of attributes used in clustering procedure

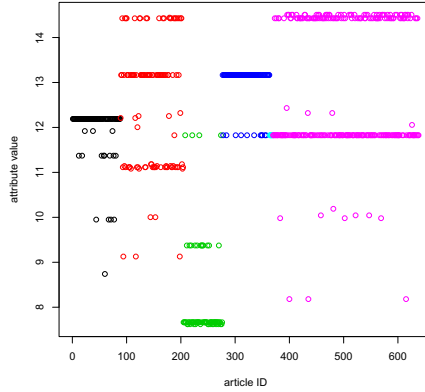


Fig. 2. Attribute F visualization

4.1 Usage of a Domain Knowledge

The extraction of content units from electronic documents can be facilitated by application of a domain knowledge. Many publications were formatted using professional systems such as TeX or other DTP packages. Most of such systems allow "styles" that help keeping publications by the same publisher in a common, uniform layout. We attempt to elicit external knowledge from these styles, whenever possible, in order to improve the performance of the content extraction process.

Typically, predefined styles would provide formatting data in the form of rules, e.g.

$$[attribute_1 = v_1] \wedge \dots \wedge [attribute_k = v_k] \implies [content = c],$$

for instance

$$[font_{size} = 14] \wedge [font_{bold} = TRUE] \wedge [centering = TRUE] \\ \implies [content = TITLE].$$

By analyzing the available style files, we can build a collection of such rules to be used later by the extraction/recognition process. Since documents of the same publisher tend to conform to a set of pre-defined styles, we can expect to attain higher accuracy with a larger set of documents with the application of such a priori rules. Such set of rules could significantly improve performance of the algorithm retrieving information about the document described in the previous chapter.

5 Conclusions and Further Work

Experiments conducted by other teams mentioned in Section 2.3 clearly show usefulness of the chosen document representation. It is not a simple task to convert a scientific article into NXML and the algorithm presented here is just a proof of concept. In the future we plan to extend retrieving system to extract more information about the document and its content.

Experiment with clustering documents by style described in Section 4 gave optimistic results. In a long run we plan to add a rule-based subsystem which will facilitate recognition process. Rules will be manually created and adjusted for the extracted clusters.

There are many possible ways of development of the content retrieving system. First of all we need to create an evaluation procedure. So far results were inspected only manually. PubMed Central [PubMed] is a large database of the medical documents already available in the NXML format. Therefore it is a good choice for a test dataset. Another possible way to improve the system is to use domain knowledge in detecting NXML notions. For this purpose we can use for example a database of authors or scientific institutions. Layout extraction algorithm implemented in PDFMiner is not optimal and in some cases it could be adjusted to our needs.

References

- [BazTech] Consortium BazTech: BazTech - Database of the Polish Technical Journal Contents (2011), <http://baztech.icm.edu.pl/>
- [DBPedia] The DBPedia Community: The DBPedia Knowledge Base (2011), <http://DBpedia.org>
- [PubMed] PubMed Central, <http://www.ncbi.nlm.nih.gov/pmc/>
- [ISMIS] S. Hoa Nguyen, Świeboda, W., Jaśkiewicz, G.: Extended document representation for search result clustering. In: Bembenik, R., Skonieczny, Ł., Rybiński, H., Niezgodka, M. (eds.) To be published in: Intelligent Tools for Building a Scientific Information Platform (2011)
- [NXML] Mulberry Technologies, Inc.: Journal Archiving and Interchange Tag Set Tag Library version 3.0 (2008), <http://dtd.nlm.nih.gov/archiving/tag-library>
- [PDFMiner] Shinyama, Y.: PDFMiner: Python PDF parser and analyzer (2010), <http://www.unixuser.org/~euske/python/pdfminer/>
- [RSKT] Szczuka, M., Janusz, A., Herba, K.: Clustering of rough set related documents with use of knowledge from dPedia. In: Yao, J. (ed.) RSKT 2011. LNCS, vol. 6954, pp. 394–403. Springer, Heidelberg (2011)

Towards Semantic Evaluation of Information Retrieval

Piotr Wasilewski

Faculty of Mathematics, Informatics and Mechanics,
University of Warsaw
Banacha 2, 02-097 Warsaw, Poland
piotr@mimuw.edu.pl

Abstract. The paper discuss fundamentals of semantic evaluation of information retrieval systems. Semantic evaluation is understood in two ways. Semantic evaluation *sensu stricto* consists of automatic global methods of information retrieval evaluation which are based on knowledge representation systems. Semantic evaluation *sensu largo* includes also evaluation of retrieved results presented using new methods and comparing them to previously used which evaluated unordered set of documents or lists of ranked documents. Semantic information retrieval methods can be treated as storing meaning of words which are basic building blocks of retrieved texts. In the paper, ontologies are taken as systems which represent knowledge and meaning. Ontologies serve as a basis for semantic modeling of information needs, which are modeled as families of concepts. Semantic modeling depends also on algorithmic methods of assigning concepts to documents. Some algebraic and partially ordered set methods in semantic modeling are proposed leading to different types of semantic modeling. Then semantic value of a document is discussed, it is relativized to a family of concepts and essentially depends on the used ontology. The paper focuses on semantic relevance of documents, both binary and graded, together with semantic ranking of documents. Various types of semantic value and semantic relevance are proposed and also some semantic versions of information retrieval evaluation measures are given.

Keywords: information retrieval, semantic information retrieval, semantic search engine, semantic evaluation, evaluation methodology, information need, semantic modeling of information need, semantic relevance, semantic value of document, semantic valuation measure.

1 Introduction

The research presented in this paper is aimed at laying foundations for semantic evaluation of effectiveness information retrieval methods. Semantic evaluation is understood in two ways. Semantic evaluation *sensu stricto* consists of automatic global methods of information retrieval evaluation which are based on knowledge representation systems. Semantic evaluation *sensu largo* includes also evaluation

of new methods of presentation of retrieved results and their comparison to previously used methods which evaluated unordered set of documents or lists of ranked documents.

In traditional evaluation methodologies human judges assessing relevance to documents on the basis of their knowledge and understanding of meaning of words appearing in judged texts. Knowledge and meaning are placed in judges' minds. Semantic evaluation methods are proposed in analogy to this situation: they are based on knowledge representation systems which are artificial stores of knowledge and meaning. In the paper, as in SYNAT project, ontologies are taken as knowledge representation systems. They can be view also as corresponding to conceptual hierarchies stored in human minds. On the theoretical basis of this correspondence, the paper introduce semantic modeling of user's information needs. Semantic modeling depends on ontologies as well as on algorithmic methods of assigning concepts to documents. Using algebraic and partially ordered set methods, various ways of semantic modeling are proposed. Semantic models of information needs serve as basis for introducing semantic value of documents. The paper proposes also semantic relevance of documents. It is a key notion of semantic evaluation of information retrieval. Semantic relevance is calculated on the basis of semantic values and is automatically assigned to retrieved texts. Semantic values of documents serves as a bridge between semantic modeling of information needs and semantic relevance of documents.

The paper has the following organization: Section 2 presents basic concepts and principles of traditional information retrieval evaluation. Section 3 discusses elements of semantic information retrieval. It presents briefly ontologies as knowledge representation systems and then introduce basic ideas of semantic evaluation of information retrieval. Section 4 discusses briefly semantic modeling of information needs and using algebraic methods proposes five types of semantic modeling. On the basis of semantic modeling, in Section 5 semantic value of documents is presented together with five types of semantic value. Section 6 discusses key concepts of semantic evaluation: semantic relevance (binary as well as graded) together with some remarks on semantic ranking of documents. Section 7 focuses on semantic evaluation of semantic information retrieval methods presenting tamped earth test (or duel test) as a way of comparing of such methods. This section is followed by Conclusions. Section 2 and first part of section 3 are reviews of a state of the art while the second part of the section 3 and the next sections introduce foundations for semantic evaluation of information retrieval.

2 Fundamentals of Information Retrieval Evaluation

Evaluation methods of information retrieval systems are aimed at reflecting how well results of searching meet user's information expectations. Currently, two broad classes of evaluation can be distinguished: system evaluation and user based evaluation (Vorhees, 2002). User based evaluation methods measure user's satisfaction with the system while system evaluation methods measure how well the system can rank documents (Vorhees, 2002).

Key notions in information retrieval evaluation are *information need* and *relevance of a document*. From the very beginning of the notion of information need, it was highlighted that information need has both conscious and unconscious components: it is a desire of an individual person or group of people to find and get information satisfying their conscious or unconscious needs (demands) (Taylor, 1967). In other words, IN is a topic on which a user would like to know more, and it is distinguished from the query - a data structure which is entered to a IR system by a user in order to communicate information need (Manning et al., 2008). *Relevance* indicates how well a document or set of documents satisfies the user's information needs (Cuadra and Katter, 1967). In other words, the document is *relevant* if it is perceived by the user as containing valuable information with regard to its information needs (Manning et al., 2008). Relevance is traditionally of binary nature: the document is relevant or irrelevant (Butcher et al., 2010; Manning et al., 2008), the vast majority of test collections assume this, however, in the first Cranfield experiments a five-point scale of relevance was used (Cleverdon, 1967; Vorhees, 2002; Voorhees and Harman, eds., 2005). Recently, *graded relevance* again become used in the evaluation experiments (Najork et al., 2007; Butcher et al., 2010).

For the standard way of ad hoc measuring the effectiveness of information retrieval systems test collections consisting of three components are used (Vorhees, 2002; Manning et al., 2008):

- A set of documents,
- The test kit of information needs, expressed as queries,
- A set of relevance propositions, usually binary assignments of labels *relevant* or *irrelevant* to each pair consisting of query and document.

Historically, the first proposed paradigm of this type was the Cranfield paradigm (Cleverdon, 1967; Vorhees, 2002; Voorhees and Harman, eds., 2005; Manning et al., 2008). Currently, more modern versions of this paradigm are used together with bigger test collections. They are discussed and developed at few conferences: the TREC conference (Text Retrieval Conference), organized in the U.S. and at two conferences dedicated to inter-language information retrieval NTCIR (NII Test Collections for IR Systems), which focuses on East-Asian languages and CLEF (Cross Language Evaluation Forum), which focuses on European languages (Manning et al., 2008).

This traditional way of information retrieval evaluation is based on two fundamental assumptions (Butcher et al., 2010):

- Having a given user's information need, represented by a query, each document in a given set of texts is relevant or irrelevant with regard to this information need.
- The relevance of the document d depends solely on the information need and the d itself, being independent from ranking of other documents in the collection by a search engine.

Methods of evaluating unordered sets of documents were historically first between information retrieval evaluation methods (van Rijsbergen, 1979). Then

methods evaluating ranked retrieved sets of documents appeared. The former include such classic measures as recall, precision, specificity, fallout, the latter include interpolated precision, mean average precision, reciprocal rank cumulative gain measures (Manning et al., 2008; Butcher et al., 2010).

3 Semantic Information Retrieval

Semantic retrieval is a new type of information retrieval. The semantic search engine, prepared under project SYNAT, will be one of the first systems of that type. The evaluation methods proposed up to now, are related to information retrieval systems, which can be described as linguistic/syntactic. In such systems searching is based on the presence of words in documents. In the semantic information retrieval the meaning of words are involved, whereas searching is done by looking at the knowledge contained in documents. Thus, semantic information retrieval must be based on the some way of knowledge representation. In the project SYNAT, for the purpose of knowledge representation ontologies are selected, they are presented as sets of concepts connected by various relations, mainly by the relation of subsumption (is-a relation), however being-a-part-of relation or other relations are also admissible (Breitman et al., 2007; Buitelaar and Cimi, eds., 2007; Colomb, 2007; Staab and Studer, eds., 2009). Additionally, we treat concepts from ontologies as meanings of words while the knowledge in ontologies is contained in relations, or also in the concepts, assuming that they are defined on the basis of attributes/slots¹. Therefore, methods for assessing the effectiveness of semantic information retrieval and semantic relevance of the documents should be based on ontologies. By this conclusion, we break the second assumption of the traditional information retrieval evaluation pointed out in Section 2:

semantic relevance of document d depends not only on information need α and the document itself but also on the ontology O_1 : when O_1 is changed to another ontology O_2 , document d in the context of query α may get a different semantic relevance.

Information retrieval systems are typical examples of human - computer interaction systems. In any information retrieval system, four elements can be distinguished:

- a user’s mind² being a source of information needs and formulated queries,
- user’s interface used for entering queries
- search engine operating with an inverted index and retrieving documents,
- data repository, storing all collected documents

¹ In taxonomies, being ontologies of the simplest form, one cannot claim that taxonomic topics contain knowledge, in this case knowledge is only contained in the subsumption relation.

² Understood following cognitive science as an information processing system.

In the semantic information retrieval system, a module of semantic searching is equipped with knowledge representation system, e.g. with a given ontology, while meanings are assigned to the words from document on the basis of this ontology. Therefore, in the semantic information retrieval system, meaning and further, knowledge are located in two modules of the system: in the user's mind in which they are components of information need and in the ontology incorporated in the system. In the SYNAT project it is also planned to develop user's interface to a dialogue model for user - search engine interactions, which will conduct a dialogue with the user aimed at specification of a query and driving the searching of documents or presentation of retrieved results. An important function of the module will be the translation of a query entered by the user and expressed in natural language, onto a query in an ontology based a descriptive logic language. In this translation, the ontology from a semantic search engine will be also involved.

Note that in the context of ontologies adopted in the project for knowledge representation, semantic evaluation of information retrieval should be distinguished from the evaluation of semantic information retrieval with respect to knowledge. Having two different semantic search engines W_1 and W_2 and basing them on the same semantic component, e.g., on the same ontology³, semantic search factor is controlled, therefore we can evaluate effectiveness of search engines W_1 and W_2 using classical nonsemantic methods. We can use such global methods even when two different search engines are supported by two different ontologies, but in this case it cannot be certain whether e.g. indexing algorithms or ontologies are responsible for the effectiveness of retrieval. Similarly, it is possible that having two nonsemantic search engines, we can evaluate their effectiveness using a semantic method of effectiveness evaluation based on given ontology O_1 .

It is worth noting that notions of evaluation of semantic information retrieval and semantic evaluation of information retrieval are independent, i.e. all four possibilities can hold. To two possibilities pointed above, one has to add the possibility practiced so far, i.e. nonsemantic evaluation of nonsemantic information retrieval and the forthcoming possibility of semantic evaluation of semantic information retrieval.

4 Semantic Modeling of Information Needs

An information need arising in the user's mind consist, inter alia, of concepts. However, an information retrieval system has no access to the conceptual frames in the user's mind. Communication between the mind and the information retrieval system is done through a query formulated and entered by the user expressing his/her information need. A query is a data structure usually consisting of words. In the sequel, we assume that words contained in the query and referring to concepts (terms) are mapped by the system to concepts included in the ontology of the system. Let us note that this mapping is in fact assigning to a

³ Semantic search is supported by the same ontology.

query its meaning in a given ontology and that the context of this ontology is essential: the same query in two different ontologies can have two different meanings. This reveals the nature of semantic modeling of queries: a given query is semantically modeled by a family of concepts interpreted as meanings of words contained in the query.

In the following considerations, we adopt simplifying assumption about the ontology: by an ontology we mean set of concepts O_1 partially ordered by subsumption relation \leq : $\langle O_1, \leq \rangle$. If it will not lead to confusion (a subsumption relation will be understood from the context), to ontology $\langle O_1, \leq \rangle$, as a partial order, we will refer also by O_1 .

Let $\langle O_1, \leq \rangle$ be an ontology used by a given semantic search engine. Hereafter we model the information need semantically as a set of concepts from ontology O_1 : $\{C_1, \dots, C_n\} \subseteq O_1$ determined in some way by query q expressing this information need. Such family we will call *a semantic model of query q* or *a semantic model of information need*. Because information need is always expressed in the form of a query, we will also briefly say that the family of concepts models semantically the query.

1. The simplest way of semantic modeling of the information need expressed by query q is to take concepts from the ontology: O_1 which are assigned by the system to terms contained in query q . Such family of concepts we will denote by $O_1(q)$.
2. Another way of modeling query q is to take additionally concepts from ontology O_1 which are placed between concepts from family $O_1(q)$ which are comparable with respect to subsumption relation \leq . Such family we will denote by $O_1[q]$, in other words:

$$O_1[q] := \{D \mid \exists A, B \in O_1(q); A \leq D \leq B\} \quad (1)$$

Let us note that family $O_1[q]$ can be empty even when $O_1(q)$ is nonempty, and this is when $O_1(q)$ is an anti-chain, i.e. any two concepts from $O_1(q)$ are not comparable with respect to subsumption relation \leq . Taking into account such possibility, we can introduce next ways of semantic modeling of information needs.

3. Family $O_1(q)$ can be taken as a set of generators of a complete lattice: we take family $O_1(q) \subseteq O_1$ as partially ordered set $\langle O_1(q), \leq_{|O_1(q)} \rangle$ and then we take the Dedekind-MacNeille completion of $\langle O_1(q), \leq_{|O_1(q)} \rangle$ which is a complete lattice⁴. For the family semantically modeling query q we take the universe of this lattice denoted by $L[O_1(q)]$.
4. Let us note that family $L[O_1(q)]$ not necessarily contains e.g. all upper bounds of family $O_1(q)$ in set $\langle O_1, \leq \rangle$ (upper bounds of family $O_1(q)$ are superconcepts of all concepts from family $O_1(q)$)⁵. In order to consider all

⁴ One of the methods of construction of the Dedekind-MacNeille completion is creating a concept lattice (Wille, 1982; Ganter and Wille, 1999) for a given partially ordered set (see Dedekind completion theorem in Ganter and Wille, 1999). Creating finite concept lattices has a computational character.

⁵ All lower bounds of family.

elements somehow generated from family $O_1(q)$ we can proceed in two ways. Firstly, the Dedekind-MacNeille completion of whole ontology O_1 is taken, denote the universe of this lattice by $L[O_1]$ (note that $O_1 \subseteq L[O_1]$). Then take family $O_1(q)$ as a set of complete generators and generate complete sublattice $Sg_{L[O_1]}(O_1(q))$ of the complete lattice $L[O_1]$. For the family semantically modeling query q we take family $Sg_{L[O_1]}(O_1(q))$.

5. Secondly, take the family of all concepts from ontology O_1 which are comparable by the subsumption relation with at least one concept from family O_1 , i.e. take the family of the form:

$$FI_{O_1(q)} = \bigcup_{A \in O_1} (A) \cup \bigcup_{A \in O_1} [A], \quad (2)$$

where (A) and $[A]$ are respectively a principal filter and a principal ideal determined by concept A in partially ordered set $\langle O_1, \leq \rangle$. Then take the Dedekind-MacNeille completion of partially ordered set $\langle FI_{O_1(q)}, \leq_{FI_{O_1(q)}} \rangle$, the universe of this complete lattice will be denoted by $L[FI_{O_1(q)}]$. For the family semantically modeling query q we take family $L[FI_{O_1(q)}]$.

Note that two last methods of modeling of information needs outlined above are different and have their own advantages and disadvantages. Lattices $Sg_{L[O_1]}(O_1(q))$ and $L[FI_{O_1}]$ do not have to be isomorphic. The first method is computationally expensive because it requires construction of lattice $L[O_1]$, however, only a half of the job should be done, since every ontology, as a tree, is a complete semi-lattice. In the second method, some elements of lattice $L[FI_{O_1}]$ do not have to belong to $L[O_1]$, thus they do not have to be related to concepts from ontology O_1 . On the other hand, the question of computational complexity of the first method is significant only in the case of dynamically changing ontology O_1 requiring to online computation of lattice $L[O_1]$ whereas elements from lattice $L[O_1]$ unrelated to concepts from ontology O_1 in some contexts may be regarded as an advantage rather than disadvantage, for example if such concept will appear in some documents this can be seen as a reason for adding it to ontology O_1 . It is worthy to note also that the above list is open and other methods of semantic modeling of information needs can be proposed.

Finally, semantic modeling of information needs can be used for constructing semantic information retrieval methods as well as semantic evaluation measures of retrieval effectiveness. In this paper we investigate the latter possibility but definitely the former is also worth of exploration.

5 Semantic Value of Documents

On the basis of semantic modeling of queries now we can move now to semantic characterization of documents. First notion of this kind is a semantic value of a document.

Having given family of concepts $\Phi = \{C_1, \dots, C_j$ of a given ontology O_1 we can determine a semantic value of document d . Let $C_\Phi(d, C_i) = 1$ for $1 \leq i \leq j$,

if C_i is contained in document d , otherwise $C_{\Phi}(d, C_i) = 0$. Semantic value of document d with respect to family Φ has the following form:

$$\sum_{i=1}^k C_{\Phi}(d, C_i). \quad (3)$$

Firstly, note that a semantic value can be calculated for a set of documents. In such case also an average semantic value of a set of documents can be calculated. Note also that family Φ does not have to be interpreted as a semantical model of an information need/query. For example, as family Φ can be taken the whole ontology, in this case a semantic value of documents can be used for characterization of this ontology on the basis of a given set of documents or as a basis for comparison of two different ontologies. It is also possible that semantic value is not an integer, it can hold in the case when an algorithm of mapping concepts to documents will describe particular concepts in the context of a given document by numbers other than 0 and 1. In the sequel, all considerations will admit this possibility.

Let us note that for family Φ different families of concepts can be taken representing different methods of semantic modeling of information needs, including the five methods outlined above. And so, keeping the way of enumerating of this list we get the following types of a semantic value of a given document d with respect to ontology O_1 :

$$SV_1(d) = \sum_{i=1}^k C_{O_1(q)}(d, C_i), \quad (4)$$

$$SV_2(d) = \sum_{i=1}^k C_{O_1[q]}(d, C_i), \quad (5)$$

$$SV_3(d) = \sum_{i=1}^k C_{L[O_1(q)]}(d, C_i), \quad (6)$$

$$SV_4(d) = \sum_{i=1}^k C_{Sg_{L[O_1]}(O_1(q))}(d, C_i), \quad (7)$$

$$SV_5(d) = \sum_{i=1}^k C_{L[FI_{O_1(q)}}(d, C_i). \quad (8)$$

Note also that family Φ can represent semantical modeling of many information needs at the same time, being simply set theoretical union of semantic models of particular information needs.

It is worthy to note that semantic value of documents essentially depends on an ontology taken as a basis for modeling of documents. Particularly, it is reflected by measures of semantic value from SV_2 to SV_5 . Consider now the fact that semantic value can be used to determine the semantic relevance of documents as

well as semantic ranking of documents. The first shows its usefulness in semantic evaluation of information retrieval, while the second can be applied both in semantic evaluation and semantic information retrieval.

6 Semantic Relevance and Semantic Ranking of Documents

A document is relevant if it is perceived by a user as containing valuable information with respect to of his/her personal information needs (Cuadra and Katter, 1967; Manning et al. 2008). Relevance indicates how well a document or set of documents satisfies the user's information needs (Cuadra and Katter, 1967). In order to be able to talk about semantic relevance, there must be some connection between the evaluation of semantic relevance of a document and a given information need. Semantic value of the document seems to give the basis for such a relationship, because through the semantic modeling it bounds information needs of users with the documents.

6.1 Binary Semantic Relevance

Having given document d , query q , family of concepts $\Phi = \{C_1, \dots, C_j$ and based on Φ a semantic value of d we can determine binary a *semantic relevance of d with respect of q* in the following way:

if $\sum_{i=1}^k C_{\Phi}(d, C_i) = k$, then d is semantically relevant with respect to q ,

if $\sum_{i=1}^k C_{\Phi}(d, C_i) < k$, then d is semantically irrelevant with respect to q
(d is not semantically relevant w.r.t. q).

In other words, the document d is relevant with respect to query q on the basis of family of concepts Φ , when every of the concepts from family Φ is contained in document d . Note that in conjunction with the five types of semantic value outlined above, we have at least five types of binary semantic relevance. It should be noted that, as in the case of methods of semantic modeling of information needs or types of semantic value of documents, a list of types of semantic relevance is open. It is also worth noting that the large cardinalities of family Φ of such approach to binary semantic relevance can be very restrictive.

Having the five types of binary semantic relevance, it is worth noting that we also have five semantic versions of each of the classical measures of the effectiveness of informational retrieval based on relevance of documents binary understood. For example, we consider the semantic version of the average precision (where $\Phi = \{C_1, \dots, C_j$ is a family of concepts which is a semantic model of a given information need):

$$AP^\Phi = \frac{1}{|Rel_\Phi|} \cdot \sum_{i=k}^{|Res|} relev(k) \cdot Pr@k_\Phi = \frac{1}{|Rel_\Phi|} \cdot \sum_{i=k}^{|Res|} relev(k) \cdot \frac{|Res[1, \dots, k] \cap Rel_\Phi|}{k}, \quad (9)$$

where Rel_Φ is a set of documents relevant with respect to a given information need on the basis of family of concepts Φ being the semantic modeling of this information need, Res is a set of all retrieved documents and $Res[1, \dots, k]$ consists of the top k documents ranked by the system, while $relev(k) = 1$ if k -document in Res is relevant, $relev(k) = 0$ otherwise.

Other example is semantic version of geometric mean average precision for n information needs:

$$GMAP(AP_1^\Phi, \dots, AP_n^\Phi) = \sqrt[n]{\prod_{i=1}^n (AP_i^\Phi + \varepsilon)} - \varepsilon, \quad (10)$$

where ε is a constant aimed at eliminating pathologies when one of the average semantic precisions, AP_i^Φ , is equal to 0.

6.2 Graded Semantic Relevance

The simplest way of introducing graded semantic relevance for document d , query q and family of concepts $\Phi = \{C_1, \dots, C_n\}$ semantically modeling query q is to identify semantic relevance with semantic value. Other way is to normalize graded semantic relevance to the unit $[0, 1]$:

$$IS_\Phi(d, q) = \frac{\sum_{i=1}^n C_\Phi(d, C_i)}{n}, \quad (11)$$

where $IS_\Phi(d, q)$ denotes semantic relevance of document d with respect to query q on the basis of family of concepts Φ . Note that $IS_\Phi(d, q) = 1$ if, and only if document d is semantically relevant with respect to query q by means of binary semantic relevance. A value of normalized semantic relevance can be average to given set of queries Q :

$$IS_\Phi(d) = \frac{1}{|Q|} \cdot \sum_{q \in Q} IS_\Phi(d, q). \quad (12)$$

Measure IS_Φ (for queries as well as sets of queries) we will call generally normalized semantic relevance.

For example, we present semantic versions of normalized discounted cumulative gain measure for graded relevance on the basis of family of concepts Φ : $nDCG_\Phi$. Let be given a list of ranked documents of which every has assigned normalized semantic relevance. For this list we create the semantic gain vector G_Φ composed of normalized semantic relevance of documents from the list, a value of relevance placed on i place of the vector G_Φ we denote by $G_\Phi[i]$. Therefore, $G_\Phi[i] = IS_\Phi(d_i)$. Then we calculate the cumulative semantic gain vector

CG_Φ of which value of k element is a sum of values of elements of the vector G_Φ from 1 to k :

$$CG_\Phi[k] = \sum_{i=1}^k G_\Phi[i]. \quad (13)$$

Then we calculate discounted semantic gain:

$$DCG_\Phi[k] = \sum_{i=1}^k \frac{G_\Phi[i]}{\log_2(1+i)}. \quad (14)$$

Then the perfect semantic gain vector G'_Φ is constructed, it consists of elements of the semantic gain vector G_Φ , where if $i \leq j$, then $G'_\Phi[i] \geq G'_\Phi[j]$. Then the cumulative perfect semantic gain vector CG'_Φ and the discounted cumulative perfect semantic gain vector DCG'_Φ are calculated. The last step is a normalization of the discounted cumulative semantic gain vector by the discounted cumulative perfect semantic gain vector:

$$nDCG_\Phi[k] = \frac{DCG_\Phi[k]}{DCG'_\Phi[k]}. \quad (15)$$

6.3 Semantic Ranking of Documents

Documents can be semantically ranked, e.g., by means of their semantic value. In this case we are dealing with at least five types of semantic ranking of documents. Note that the semantic ranking of documents can be naturally combined with the graded semantic relevance, e.g., with the normalized semantic relevance. Let us note that the combination of semantic ranking based on the semantic value and the normalized semantic relevance we always get the ideal semantic gain vector.

7 Tamped Earth Test or Duel of Two Search Engines

It has to be underlined that both semantic information retrieval and semantic evaluation of information retrieval (semantic as well as nonsemantic) significantly depend both on the algorithms for indexing documents and on concepts from ontologies (adopted in SYNAT project as a way of knowledge representation). This can be seen as disadvantage in the context of constructing semantic evaluation measures: such measures can prefer search engines using the same conceptual indexing algorithms. This problem can be partially solved in the future by establishing conventionally some standardized conceptual indexing algorithm/algorithms. However, it is still only a partial solution. Before further discussion, let us accept a notational convention:

retrieved results of search engine W_i for set of documents D_j we will denote by $W_i(D_j)$. A semantic evaluation measure M which is based on conceptual indexing algorithm p_s and ontology O_t will be denoted by $M(p_s, O_t)$. Search

engines with built in algorithms and/or using some ontologies we will denote multiplicatively, for example $W_i a_k O_t$ denotes search engine W_i with built in algorithm a_k and using ontology O_t , while $W_i a_k O_t(D)$ denotes its retrieved results on set of documents D .

Now, take into account the worst case from the perspective of the pointed above disadvantage. Assume that we have two search engines W_1 and W_2 which use two different conceptual indexing algorithms, respectively, p_1 and p_2 , and two different ontologies O_1 and O_2 . In such case, one can evaluate their retrieval results for a given set of documents by means of a semantic measure M using algorithms p_1 and p_2 and ontologies O_1 and O_2 in all possible arrangements, i.e. four versions of the appropriate measure of M : $M(p_1, O_1)$, $M(p_1, O_2)$, $M(p_2, O_1)$, $M(p_2, O_2)$. Note that each of these semantic versions of measure M is unjust and therefore, as a fair evaluation, the average value of all four measures can be taken. Note also that the following result would be particularly striking: namely, if one search engine defeat a second search engine using the opponent's weapon (the algorithm and ontology), for example, if search result $W_1(D)$ was better than $W_2(D)$ with respect to evaluation measure $M(p_2, O_2)$. In this case, W_1 would be particularly convincing winner of a duel - hence the name of the test. Also a nonsemantic version of the test is possible: evaluation of retrieved results of various search engines, e.g. $W_1 p_2 O_2$ vs $W_2 p_2 O_2$, by means of classical nonsemantic measures.

Let us also note that the tamped earth test has four types of variables:

- search engines,
- indexing algorithms,
- ontologies,
- evaluation measures.

Therefore, fixing of particular variables gives methods of testing of other elements. For example, fixing a search engine, indexing algorithm and evaluation measure, one can test various ontologies, e.g. O_1 and O_2 comparing $W_1 p_1 O_1(D)$ vs $W_1 p_1 O_2(D)$ with respect to appropriate versions of evaluation measure M : $M(p_1, O_1)$ and $M(p_1, O_2)$.

8 Conclusions

Semantic modeling of information needs and semantic values of documents introduced in this paper can be applied in semantic evaluation methods as well as in constructing new semantic retrieval methods.

To be applicable, methodology of evaluating the effectiveness of information retrieval must include the following elements (Butcher et al., 2010):

- characterization of the intended purpose of information retrieval method,
- measure, which quantitatively shows how well this goal is satisfied,
- precise, accurate and economical measurement technique,
- estimation of measurement error.

The estimation of measurement error is a problem solvable in a standard way for all evaluating methodologies including semantic ones (van Rijsbergen, 1979; Butcher et al., 2010). Therefore, research should be focused on the first three topics.

Some exemplary semantic evaluation measures are given in this paper, however, next semantic evaluation measures should be proposed in the future. Since methods of assessing of semantic relevance presented in this paper are automatic, it meets the third point. Implementation and testing of these methods will reveal their usefulness. They can be also compared to the traditional evaluation methods based on assessing made by human judges. Such comparison can show how far away semantic evaluation methods depart from them. Satisfaction of the first point involves a combination of further theoretical research with computational simulations of the proposed methods of semantic evaluations of information retrieval and this will be done in the future. Especially interesting are investigations into semantic evaluation of effectiveness of semantic information retrieval.

Let us note that generally semantic evaluation methods are not necessarily alternative to human judging methods. They can be used also to support assessing process made by human judges as it is done analogically in the case of manual indexing of documents from PubMed search engine, where an automated indexer indexes the title and abstract and supports the manual indexer by providing a list of potential MeSH ontology keywords (Berry and Browne, 2005).

Acknowledgements. Author would like to thank Andrzej Skowron, Hung Son Nguyen, Dominik Ślęzak, Wojciech Jaworski, Wojciech Świeboda and other colleagues from the SYNAT group for their critical comments and valuable discussions helping to improve the paper. Research was supported by the grant N N516 077837 from the Ministry of Science and Higher Education of the Republic of Poland and by the National Centre for Research and Development (NCBiR) under the grant SP/1/1/77065/10 by the Strategic scientific research and experimental development program: "Interdisciplinary System for Interactive Scientific and Scientific-Technical Information".

References

1. Berry, M.W., Browne, M.: *Understanding Search Engines*. Society for Industrial and Applied Mathematics (2005)
2. Breitman, K.K., Casanoca, M.A., Truszkowski, W.: *Semantic Web: Concepts, Technologies and Applications*. Springer, Heidelberg (2007)
3. Buitelaar, P., Cimino, P. (eds.): *Ontology Learning and Population: Bridging the gap between Text and Knowledge*. IOS Press (2008)
4. Butcher, S., Clarke, C.L.A., Cormack, G.V.: *Information Retrieval: Implementing and Evaluating Search Engines*. Massachusetts Institute of Technology Press (2010)
5. Cleverdon, C.W.: The Cranfield tests on index language devices. In: *Aslib Proceedings*, vol. 19, pp. 173–192 (1967); Reprinted in *Readings in Information Retrieval*, Sparck-Jones, K., Willett, P. (eds.) Morgan Kaufmann (1997)
6. Colomb, R.M.: *Ontology and the Semantic Web*. IOS Press (2007)

7. Cuadra, C.A., Katter, R.V.: Opening the black box of relevance. *Journal of Documentation* 231(4), 291–303 (1967)
8. Ganter, B., Wille, R.: *Formal Concept Analysis: Mathematical Foundations*. Springer, Heidelberg (1999)
9. Jarvelin, K., Kekalainen, J.: Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems* 20(4), 422–446 (2002)
10. Manning, C.D., Raghavan, P., Schütze, H.: *Introduction to Information Retrieval*. Cambridge University Press (2008)
11. Najork, M.A., Zaragoza, H., Taylor, M.J.: HITS on the Web: How does it compare. In: *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 471–478 (2007)
12. Robertson, S.: On GMAP and other transformations. In: *Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, pp. 78–83 (2006)
13. Staab, S., Studer, R. (eds.): *Handbook on Ontologies*. Springer, Heidelberg (2009)
14. Taylor, R.S.: Process of Asking Questions. *American Documentation* 13, 291–303 (1967)
15. van Rijsbergen, C.J.: *Information Retrieval*, 2nd edn. ch. 7. Butterworths (1979)
16. Voorhees, E.M.: The philosophy of information retrieval evaluation. In: Peters, C., Braschler, M., Gonzalo, J., Kluck, M. (eds.) *CLEF 2001*. LNCS, vol. 2406, pp. 355–370. Springer, Heidelberg (2002)
17. Voorhees, E.M., Harman, D.K. (eds.): *TREC. Experiment and Evaluation in Information Retrieval*. Massachusetts Institute of Technology Press (2005)
18. Wille, R.: Restructuring lattice theory. In: Rival, I. (ed.) *Ordered Sets*. Reidel (1982)

Methods and Tools for Ontology Building, Learning and Integration – Application in the SYNAT Project*

Anna Wróblewska¹, Teresa Podsiadły-Marczykowska², Robert Bembenik¹,
Grzegorz Protaziuk¹, and Henryk Rybiński¹

¹ Institute of Computer Science, Warsaw University of Technology, Nowowiejska 15/19,
00-665 Warszawa, Poland

{A.Wroblewska, R.Bembenik, G.Protaziuk, H.Rybinski}@ii.pw.edu.pl

² Institute of Biocybernetics and Bioengineering, Trojdena 4,
02-109 Warszawa, Poland

tpodsiadly@ibib.waw.pl

Abstract. One of the main goals of the SYNAT project is to equip scientific community with a knowledge-based infrastructure providing fast access to relevant scientific information. We have started building an experimental platform where different kinds of stored knowledge will be modeled with the use of ontologies, e.g. reference/system ontology, domain ontologies and auxiliary knowledge including lexical language ontology layers. In our platform we use system ontology defining “system domain” (a kind of meta knowledge) for the scientific community, covering concepts and activities related to the scientific life and domain ontologies dedicated to specific areas of science. Moreover the platform is supposed to include a wide range of tools for building and maintenance of ontologies throughout their life cycle as well as interoperation among the different introduced ontologies.

The paper makes a contribution to understanding semantically modeled knowledge and its incorporation into the SYNAT project. We present a review of ontology building, learning, and integration methods and their potential application in the project.

Keywords: Ontology building, ontology maintenance, ontology integration, semantic modeling, ontology-based systems.

1 Introduction

SYNAT is a large scientific project aiming at creating a universal hosting and scientific content storage and sharing platform for academia, education and open knowledge society. It is the national project funded by the National Centre for Research and Development. The project has started in 2010 and will be carried out

* This work is supported by the National Centre for Research and Development (NCBiR) under Grant No. SP/I/1/77065/10 by the Strategic scientific research and experimental development program: „Interdisciplinary System for Interactive Scientific and Scientific-Technical Information”.

until 2013 by the research consortium comprising major Polish science institutions. Its purpose is to develop a system being a heterogeneous repository of data from various structured and unstructured sources [1]. The system is supposed to be capable of automatic acquisition of knowledge from web sources, including universities resources, such as, *inter alia*, researchers' homepages, research projects, tutorials, conference and workshop information. The structuring of information in the system will be based on ontologies, and will be stored in the SYNAT Knowledge Base (KB).

The purpose of KB is to provide all potential SYNAT users with a fast access to relevant scientific information. SYNAT "actors" may vary significantly in their needs - they may be researchers, lecturers, reviewers, students of all levels, publishers, librarians etc. Their requirements are not static, and may vary dynamically in time. The final system should thus possess a flexible functionality being smoothly adjustable to the changing users' requirements. The functionality of KB will not be "hard coded" into the system, but instead, it will be "ontology driven".

To this end, we plan to build a system ontology and provide tools for building many needed domain ontologies. The system ontology describes kinds of information provided by the system (covering concepts and activities related to the scientific life) and relationships between the system-based concepts. The system ontology is also thought as a "semantic definition" of the system scope and possible functionality. It can help in finding meaning of queries and building a sequence of searches in KB, as well as suggesting user additional options in the searching process. In addition, a number of domain ontologies will be incorporated into KB in order to provide a semantic support within specific research domains.

The system will provide access to different scientific information resources in one place and integration with these resources. A very important aspect of the integration will be sharing. On one hand, we want our system to make Polish national resources available to other, global scientific databases. Global resources worth mentioning include Web of Science (WoS) [2], Scopus [3] and Google Scholar (GS) [4]. Each of these sources has its specifics when it comes to the scientific fields covered. WoS is the leader in classical areas such as Physics and Chemistry [5]. Scopus is efficient for fields like Health and GS takes a leader role in integrating all domains (in some domains, like computer science it is already one of the best). GS has some major advantages over the other scientific databases: it is freely available, and relatively easy to use. Because of that, we plan to open national resources to the global scientific information services, including also such free services like GS or Microsoft Academic Search [6]. On the other hand we would like to make use of free web resources for integrating scientific services with the main information providers¹.

System users should be able to access resources available in local (e.g. university databases), national, and global systems in a convenient way. As a matter of fact, in existing services, searching for documents in a given subject does not take into account specific needs of a user. For example, a student may request some basic tutorials, a Ph.D. student will probably be looking for break-through documents on a particular subject and publications presenting it in detail, whereas a matured researcher is probably interested in the latest documents on the topics from his/her domain.

¹ To the most possible extent from legal and technical point of view.

Because of that we plan to equip the system with semantically-grounded technologies. Semantics is considered to be capable of dealing with the heterogeneity, massive scale and dynamic nature of resources on the Web. Semantic technology means application of techniques that support and exploit semantics of information (as opposed to syntax and structure/schematic issues) to enhance existing information systems [7]. Accordingly the semantics used in the Web is supposed to provide well-defined meaning enabling cooperation between machines and people [8]. Currently in more practical terms, Semantic Web technology also implies the use of standards such as RDF/RDFS and OWL. The semantic standards provide a good framework to represent, share and use heterogeneous knowledge but also allow for discovering new information [7]. Nowadays ontology driven information systems are used for information retrieval, integration and analysis across many areas of applications. Such systems are utilized for example in public e-employment services [9] or to fuse knowledge automatically extracted from different media types [10]. More and more researchers make efforts towards the development of ontology-based knowledge bases including aspects from ontology design and population, using “semantic” data, to automated reasoning and semantic query answering.

In this paper we present a general idea of the SYNAT KB in the context of ontologies and their usage in the system. Also, we introduce a preliminary version of the system ontology, which is the core ontology of the KB. The rest of the paper is structured as follows: Section 2 outlines the proposed knowledge base and presents the first version of the system ontology. Sections 3, 4 and 5 give a review of methods for maintaining ontologies throughout their life-cycle, especially ontology building, ontology learning and ontology integration appropriately. Section 6 concludes the paper presenting the state of the art of ontology life-cycle tools and outlines research plans.

2 Ontologies in the SYNAT Project

The main functionality of the knowledge base can be divided into the two following categories: (i) searching and acquiring information, and (ii) building customized ontologies. An extensive usage of ontologies in the system is planned not only for better understanding users’ queries and obtained information but also for driving the information acquisition process.

2.1 General Ideas Concerning the Experimental SYNAT Knowledge Base

Potential Users and Usage Scenarios

The intended end-users of the knowledge base are people involved in research and scientific activities i.e. scientists coming from various fields of science, and with various levels of expertise. The users accessing the system may play various roles in the scientific life; they may be researchers, lecturers, reviewers, research organizers, administrators, etc.

One of the main aims of the system ontology is to support end-user dialog with the knowledge base and determine the system actions for answering the end-user queries.

The system ontology is also thought as a “semantic definition” of the system scope and possible functionality. It can help in resolving the query meaning and building a sequence of searches in the knowledge base as well as suggest to a user additional options in preparing the answer.

Searching in the Knowledge Base

The experimental user interface starts with a single Google-like field to be used by the end-user for specifying a query. The system applies the system ontology in the query analysis and tries to discover the “semantics” of a user query. If the process of understanding a query is positive, the system indicates a user possible ways of querying the knowledge base, and provides relevant answers. If the query parsing brings negative results, the system can still indicate to a user the most common ways of using the knowledge base. Additionally, the user’s profile will be used to help the system in recognizing the sense of the query and/or user needs.

In addition, the system starts collecting associated information from various resources, e.g. definitions from Wikipedia, information from local system resources, as well as external resources on the Internet. If there is a need, an intelligent query assistant can try to resolve query ambiguity on its own or, if necessary, interact with a user. The assistant can suggest information associated with a query or other related topics.

Scope

The SYNAT knowledge base is thought first and foremost as the source of scientific information concerning the community members, their research, documents (publications, reports, etc.), organizations (scientific, governmental, etc.), events (conferences, workshops, seminars, etc.) and useful data resources (databases, home pages of individuals or institutions, etc.) [11]. Another field of usage is connected with evaluation of this type of information.

Knowledge Domains

The experimental KB should cover the following categories of knowledge:

- Information about academic and scientific community, modeled by the system ontology.
- Research domains modeled *inter alia* by domain ontologies (describing particular details of the domains and also ontologies defining scientific domains and their new topics).
- Analytical knowledge related to evaluation measures of various objects. Such knowledge is acquired by applying various evaluation algorithms.
- Auxiliary knowledge base which includes different kinds of objects e.g. dictionaries with grammatical forms (inflection, conjugation, declension), language oriented tools (POS taggers, gazetteers, etc.), semantics analysis tools (dictionaries of synonyms, homonyms), ontological or semi-ontological lexico-linguistic layers of the above ontologies.

System Architecture Outline

A general idea of the system architecture is shown in Fig. 1. Three parts can be distinguished in the system: (1) the searching subsystem, (2) the ontology subsystem, and (3) repositories, dictionaries and thesauri subsystem.

The Searching Subsystem

The searching subsystem includes the following main components:

- **Query Analyzer** - the basic tasks realized by this component are: parsing – splitting text into terms (also dates, complex terms, etc.), lemmatization, recognition of query language, mapping terms to ontologies (using linguistic layers, taking into consideration also recognition of language, allowing or depicting any disambiguation), query enrichment based on ontologies and their linguistic layer (synonyms). The result of query analyzing process is a structured query prepared for interpretation.
- **Query Interpreter** is responsible for building a query execution plan; it includes: query disambiguation (choosing the most possible interpretation; asking a user about possible interpretations in the case of a strong ambiguity), generating additional prompts for a user and requesting for additional possible contexts. Query Interpreter uses the word sense disambiguation algorithms (based on statistical and data mining methods), thesauri and ontologies including WordNet.
- **Query Executor** searches in and communicates between knowledge base and the many available resources. The main task of Query Executor is to choose resources relevant for the given query, and send to them properly structured queries. After receiving answers, it should remove duplicates and refine obtained answers.
- **Results Evaluator** assesses results of a given query in terms of quality (e.g. similarity of the returned documents to the query) and quantity. Moreover it performs appropriate text mining algorithms (e.g. in order to group the results thematically, or classify them by requested type of information), and ranks the results, based on similarity to the query.

The Ontology Subsystem

The ontology subsystem is envisaged as a component, which provides the entire functionality needed for accessing, maintaining and developing ontologies. The main parts of this subsystem are:

- **Ontology Manager**, which is the main component of the ontology subsystem, and enables other components' access to ontologies. The functionality of this component includes, among others, searching and retrieving entities from ontologies, adding new entities, and extracting parts of ontologies.

Tools for ontology maintenance and enriching – it is a set of tools offering various functionalities helpful for maintaining and developing ontologies. It is envisaged that several methods for semi-automated ontology learning, and for integration of ontologies (e.g. comparison of ontologies at semantic level) will be available to the user. It covers discovering concepts or relations between concepts from text repositories, as well as language layers for the ontologies. This set also

includes tools supporting different types of ontology integrations (aligning, merging, mapping).

Besides functional components the ontology subsystem includes ontologies: the system ontology and domains ontologies.

Internal Resources

In the system we foresee using several types of internal resources. The main resources are repositories including text documents with their structured descriptions. Other types of internal resources are various kinds of dictionaries, thesauri, terminologies (e.g. dictionaries including proper names from a given domain, lemmas from a given language) which may be useful in methods for ontology integration and learning.

Descriptions of all accessible repositories (both internal and external) are stored in the repositories metadata component.

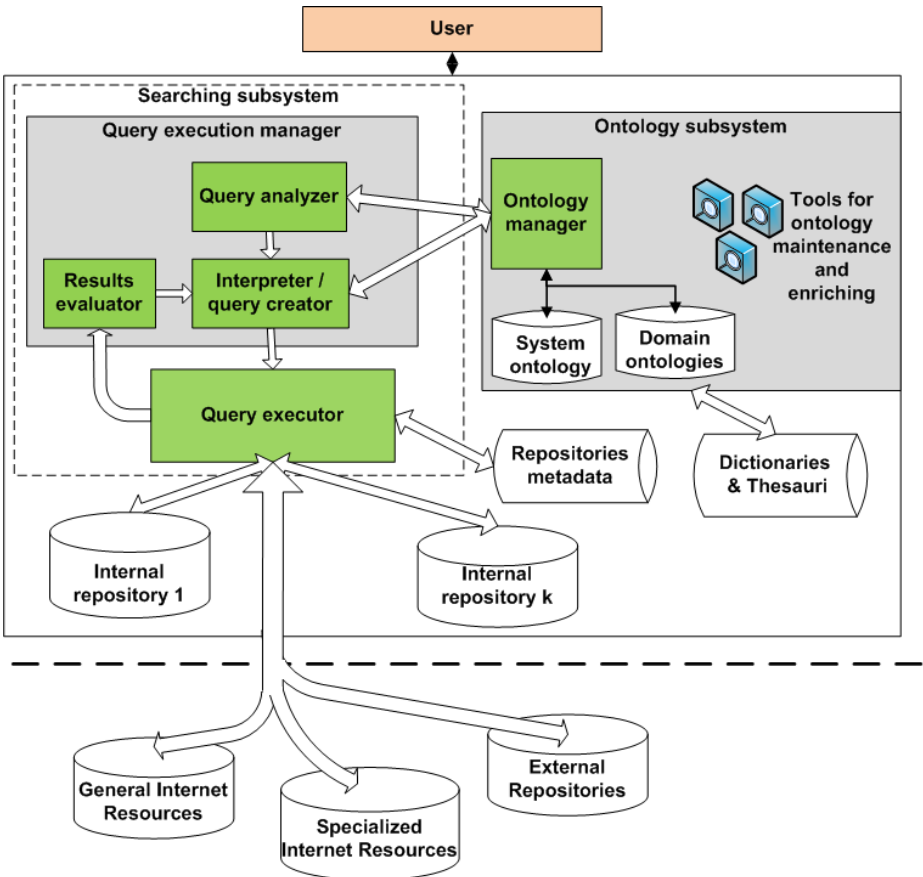


Fig. 1. A general idea of the knowledge base architecture

2.2 System Ontology

Construction of the System Ontology

The system ontology has been build using NeOn methodology [12], which decomposes the general problem of ontology construction into nine, more precisely stated, sub-problems, called scenarios [12], characterized in more detail in Section 3. Any combination of the scenarios is allowed for ontology building. All stages of the system ontology construction are a combination of four NeON scenarios, namely scenarios number 3, 4, 2 and 1.

The search for similar and potentially useful ontologies, performed using Watson plug-in from the NeON Toolkit, and searching for web resources using classical search engines (Google, Bing) revealed the existence of a large number of potentially useful ontologies (the detailed description of the chosen potentially useful ontologies is available in [11]). To this end, we decided to reuse existing ontological resources (scenario 3 in [12]).

The available ontological resources have been assessed as useful, but not exactly fitting our purpose. Therefore they were subject to modifications and reengineering (scenario 4 in [12]).

We also used non-ontological resources, in particular scientific domains from Google Scholar and Web resources characteristics elaborated by workers of Main Library of Warsaw University of Technology.

Finally, scenario 1 has been used to carry out *conceptualization*, *formalization*, and *implementation*. To formulate and validate the ontology domain and range, as well as ontology functional requirements, we formulated a set of competency questions (CQ) in the natural language. An ontology taxonomy has been constructed manually, based on the CQs. Concepts characteristics, and class definitions have been elaborated with the aim to enable CQ's answering and data control.

Ontology Formal Requirements

From the formal point of view, high-quality ontology should be: (1) consistent — all classes are consistent and can have instances; (2) semantically correct — should capture intuitions of domain experts, *inter alia*, clear and self-explanatory hierarchy of concepts and properties; (3) minimally redundant — no unintended synonyms should be present there; (4) sufficiently axiomatized — it should contain detailed descriptions, well-defined concepts (definitions with restrictions on properties, etc.), data and object properties defined with their domains and ranges [13].

Ontology Functional Requirements: Competency Questions

The core part of the system ontology is to characterize any scientist, giving general information about: personal and contact data, research interests and publications, education level, work positions, collaborators, activity in scientific events and teaching profile. The other aspects and information about academic life generally can be produced from the facts about scientists.

We collected about 50 competency questions for the ontology (the details are provided in [11]). They can be divided into the following groups:

- basic info about the *scientist* and the *research interest* (e.g. contact info, research areas);
- scientist's *education level* (e.g. achieved academic degrees, supervisors);
- *work activities* (e.g. affiliations, current and past work positions);
- *teaching profile* (e.g. supervisees, teaching activities, works reviewed, special lectures);
- participation in scientific events (e.g. taking part in scientific events, roles held at the events);
- *projects* (e.g. participation in the projects, kinds of projects, activities in projects);
- the authored *document(s)* (e.g. authored publications, patents, edited books);
- the *document profile* (e.g. publication data, publisher, referenced documents and citations);
- educational organization profile (e.g. education domains and research topics);
- additional possible CQs for *opinion* module (e.g. scientists working in a similar domain, similar projects).

The System Ontology – Current State

The image displays three overlapping windows from an ontology editor. The leftmost window, titled 'For Project: PassimOntology02-05-20', shows an 'Asserted Hierarchy' of classes under 'owl:Thing'. The hierarchy includes 'AidingModule', 'DataResource', 'foaf:Agent', 'foaf:Document', and 'swrc:Publication'. The 'DataResource' class is highlighted. The middle window, titled 'For Project: PassimOntology30-04-2011', is a 'PROPERTY BROWSER' showing 'Object properties' for 'hasAgentOPROPERTY'. The rightmost window, titled 'For Project: PassimOntology30-0...', is another 'PROPERTY BROWSER' showing 'Datatype Properties' for 'hasContactInfo'.

Fig. 2. The system ontology is composed of 5 main general modules: DataResource, Agent, Document, Project and Event (presented on the left panel). AidingModule contains notions used to define classes in the main ontology modules. The system core ontology counts: 175 classes (163 primitive, 12 defined) and 173 properties (objects and data types). Class definitions have been specified using universal, existential and cardinality restrictions.

The main goal of the system ontology² (Fig. 2) is to define basic metadata allowing to harvest repositories storing information about science communities from Web resources by specialized data mining algorithms. This goal dictates the scope of the ontology (basic facts about science and the academic community) and most of the modeling choices in the current version of the model.

The collected ontological and non-ontological knowledge sources are overlapping, but none of them covers overall the necessary scope of the system ontology. In the current model version we included notions from namespaces defined in the following ontologies: ESWC Conference ontology, FOAF, Science Ontology and SIOC. Other models have been used as a source of meaningful vocabulary.

The ontology has been specified in OWL-DL, edited with the Protégé-2000 editor [14], ver. 3.4.4. Documentation³ has been generated by the NeON toolkit [15]), its consistency has been verified using the Pellet reasoner.

The main classes in the system ontology correspond to the following five main general modules: DataResources, Agents (such as Person, Group or Organization), Documents resulting from scientific work, and finally scientific Projects and Events. The overall structure of the system ontology is presented on the left panel in Fig. 2. The first additional ontology module, AidingModule, groups *role* notions that characterize classes in the main ontology modules, for example educational level, scientist's achievements, or organization character and activity. Those notions are used as restrictions fillers in definitions of classes specifying formally their semantics.

The system ontology formally defines terminology for data mining algorithms and metadata for the design of the system platform. However, its current version does not contain class definitions and modeling options allowing for deeper reasoning or concept similarity assessment. The future version of the system ontology will not only increase the scope and granularity of the ontology concepts, but also change some of the preliminary, simplistic modeling choices. For example some notions such as geographical scope, countries, cities and languages will be modeled as classes, not data properties.

With almost all concepts in the designed ontology necessary conditions and extended ranges of object and data properties are associated. Personal roles are modeled in two different manners: roles with important additional facts (e.g. dates of holding them) are modeled as concepts (reified relations) and other roles with no particular additional information are designed as binary relations (e.g. participating in project, being an author of a publication).

3 Ontology Building

The ontology subsystem in the general idea of KB should provide appropriate workflows and possibility to design process sequences for ontology development and maintenance defined by specific methodologies. Additionally, the constructed system ontology for KB was carried out with the means of a suitable methodology. In this section, we introduce ideas referring to ontology building methodologies.

² Available at <http://wizzar.ii.pw.edu.pl/SYNAT-ontology/>, specified in OWL-DL, edited in Protégé-2000 editor ver. 3.4.5, documentation generated using NeON Toolkit ver. 2.4.2.

³ Glossary of terms for the ontology is available directly in the ontology documentation.

Ontology building is primarily a knowledge integration process. This means that albeit in theory handcrafting an entire ontology by hand is possible, in practice the only feasible way to build a reasonably complex ontology is *via* extracting information from other sources. Basically there are two types of information sources that might be considered for acquiring knowledge for ontology building: structured data sources (other ontologies, thesauri, dictionaries, semi-structured web content, text corpora), and unstructured data sources (unstructured web content, generic text archives and document repositories, generic web contents).

As mentioned, in practice the process of building and maintaining ontologies is always semi-automatic. Approaches usually used for this purpose include *inter alia* heuristics belonging to various areas (e.g. NLP, ontology resources reuse), statistics as well as data mining. So far many ontology building methodologies have been proposed. The presence of notions such as ontology development processes, activities and knowledge resources reuse, enable the division of ontology building methodologies into three main groups. TOVE [16], METHONOLOGY [17] and NeON methodology [12] are the most representative methodologies for the aforementioned groups.

TOVE contains the guidelines for general ontology building activities such as specification, conceptualization, formalization, implementation, documentation, and maintenance. It does not however accent the need for knowledge reuse. METHONOLOGY goes a step further, and defines the notions of ontology development process, maintenance activities and stresses the weight of knowledge reuse. Its disadvantage is the lack of explicit guidelines for knowledge reuse. The NeON methodology derives and extends the main ideas from its predecessors. Its main asset is that it formalizes the development processes by introducing ontology development scenarios and puts stress on reengineering of ontological and non-ontological knowledge resources. In particular it proposes Glossary of Processes and Activities, which identifies and defines the processes and activities carried out when ontology is collaboratively built by teams. To this end we plan to base our ontology building platform on NeOn⁴.

The NeON methodology is transparent and clear for users, because it defines each process or activity precisely, states clearly its purpose, inputs and outputs, and the set of methods, techniques and tools to be used. Above -described scenarios can be combined in different ways. Any combination of the scenarios should include Scenario 1, because this scenario is made up of the core activities that have to be performed in any ontology development.

One of the key elements in the NeON methodology is the set of nine general scenarios proposed for building ontologies and ontology networks[12]. In particular, on the one extreme NeON considers building an ontology from scratch (in [12] it is Scenario 1). We presume that this case does not happen even when dealing with a new research domain, as they usually emerge from existing areas. Another extreme is Scenario 9 (Localizing/nationalizing ontological resources). In this case, we presume that the concept layer of scientific domain ontologies is language independent. Therefore “localizing” ontology is limited to adding a specific language layer, and integrating it with the concept layer.

⁴ We used a combination of the NeON scenarios in constructing the system ontology outlined in Section 2.2.

We recall here the scenarios 4, 5, 6, and 8, which will play the crucial role in our approach.

Scenario 4: Reusing and re-engineering ontological resources

Ontological resources re-engineering process comprises the following activities: ontological resources reverse engineering, ontological resources restructuring and ontological resources forward engineering. After carrying out the ontological resources re-engineering process, its result should be incorporated into the corresponding activity: specification, conceptualization, and formalization.

Scenario 5: Reusing and merging ontological resources

This scenario is realized when several possibly overlapping ontological resources in the same domain are available. They can be merged, or only alignments among them can be established in order to create the ontology.

Scenario 6: Reusing, merging and re-engineering ontological resources

This scenario has the same activities as Scenario 5, but in this case ontology developers decide not to use the set of merged ontological resources as they are but to re-engineer it. Once the set of merged ontological resources is re-engineered, the result of such a process should be integrated with the corresponding activity of Scenario 1.

Scenario 8: Restructuring ontological resources

Ontological resources restructuring can be performed in one of the following four ways: (i) modularizing the ontology in different ontology modules, (ii) pruning the branches of the taxonomy not considered necessary, (iii) extending the ontology including new concepts and relations, (iv) specializing branches that require more granularity and including more specialized domain concepts and relations.

4 Ontology Learning

Specialized tools which support the task of ontology engineering are necessary to reduce the costs associated with the engineering and maintenance of ontologies [18]. As data in various forms (textual, structured, visual, etc.) is massively available, many researchers have developed methods aiming at supporting the engineering of ontologies by AI based techniques, (e.g. machine learning, NLP, but also text mining) which can support an ontology engineer in the task of modeling a domain. Such data-driven techniques supporting the task of engineering ontologies have become to be known as ontology learning. Ontology learning has the potential to reduce the cost of creating and, most importantly, maintaining an ontology. This is the reason why a plethora of ontology learning frameworks have been developed in the last years and integrated with standard ontology engineering tools.

There are three kinds of data to which ontology learning techniques can be applied: structured (such as databases), semi-structured (HTML or XML, for example), as well as unstructured (e.g. textual) documents. The methods applied are obviously dependent on the type of data used. While highly structured data, as found in databases, facilitate the application of pure machine learning techniques, such as

Inductive Logic Programming (ILP), semi-structured and unstructured data requires some preprocessing, which is typically performed by natural language processing methods.

Ontology learning will play an important role in the Ontology Subsystem of SYNAT Knowledge Base. Methods of ontology learning will be employed in the process of creating, expanding and updating system and domain ontologies.

4.1 Ontology Learning Architecture

In [18] a generic ontology learning architecture and its main components are presented. The architecture is given in Fig. 3.

The ontology learning architecture is composed of the following components: *ontology management*, *coordination*, *resource processing* and *algorithm library*. The individual components are described below.

Ontology Management Component

Ontology management component is used by an ontology engineer to manipulate ontologies. Ontology management tools typically facilitate the import, browsing, modification, versioning and evolution of ontologies. However, the main purpose of the ontology management component in the context of ontology learning is to provide an interface between the ontology and the learning algorithms. When learning new concepts, relations or axioms, the learning algorithms should add them into the ontology model accessing the API of the ontology management component. Thus, the ontology management API should contain methods for creating new concepts, relations, axioms, individuals, etc. Most of the available APIs fulfill this requirement.

Coordination Component

This component is used by an ontology engineer to interact with the ontology learning components for resource processing, as well as with the Algorithm Library. A comprehensive user interfaces should support the user in selecting relevant input data that are further exploited in the discovery process. Using the coordination component, the ontology engineer also chooses among a set of available resource processing methods, and a set of algorithms available in Algorithm Library. A central task of the coordination component is to sequentially arrange and apply the algorithms selected by the user, passing the results to each other. A neat way for arranging the text mining process has been implemented in TOM [19].

Resource Processing Component

This component encapsulates techniques for discovering, importing, analyzing and transforming relevant input data. An important subcomponent is the natural language processing system. The general task of the resource processing component is to generate a pre-processed data set as input for the algorithm library component.

Resource processing strategies differ depending on the type of input data made available. Semi-structured documents, like dictionaries, may be transformed into a predefined relational structure. HTML documents can be indexed and reduced to free text. For processing free text, the system must have access to language-specific

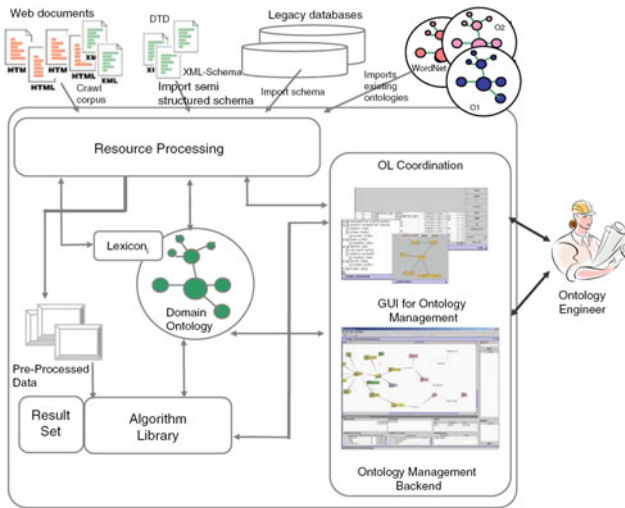


Fig. 3. Ontology learning generic architecture [18]

natural language processing systems. Some off-the-shelf frameworks, such as GATE [20], provide most of the functionality needed by ontology learning systems. The needed NLP components could be:

- A tokenizer and a sentence splitter to detect sentence and word boundaries.
- A morphological analyzer. For some languages a lemmatizer reducing words to their canonical form might suffice, whereas for languages with a richer morphology a component for structuring a word into its components (lemma, prefix, affix, etc.) is necessary. For most machine learning-based algorithms a simple stemming of the word might be sufficient.
- A part-of-speech (POS) tagger to annotate each word with its syntactic category in context, thus determining whether it is a noun, a verb, an adjective, etc.
- Regular expression matching allowing to define regular expressions and match these in the text.
- A chunker in order to identify larger syntactic constituents in a sentence. Chunkers are also called partial parsers.
- A syntactic parser determining the full syntactic structure of a sentence might be needed for some ontology learning algorithms.

Algorithm Library Component

Algorithm Library contains the algorithms applied to ontology learning. Depending on the knowledge discovery approach the library may contain machine learning algorithms (like in [18]), or text mining algorithms (like in [19]). There may also be quite efficient for some tasks algorithms looking for specific syntax patterns. In particular, typical tasks to be solved by the algorithms are as follows:

- association rule discovery – used to discover interesting associations between concepts;

- hierarchical clustering, used to cluster terms;
- classifiers, used to classify new concepts into an existing hierarchy.
- inductive logic programming – used to discover new concepts from extensional data;
- conceptual clustering – used to learn concepts and concept hierarchies.

4.2 Ontology Learning Algorithms

The tasks relevant in ontology learning have been organized in an *ontology learning layer cake* [21]. The ontology development process (manual or with automatic support) is primarily referring to the definitions of concepts and relations between them. It implies a need to acquire a linguistic knowledge about the terms that are used to refer to specific concepts in the texts, and their possible synonyms.

An important part of ontology building is discovering a taxonomy backbone (the relation *is-a*), as well as association relations (non-hierarchical ones). Finally, in order to derive facts that are not explicitly encoded by the ontology but could be derived, some rules should be defined (and if possible acquired). The layer cake presenting all the above aspects of ontology development is depicted in Fig. 4.

The layers build upon each other in the sense that results of the tasks at lower layers typically serve as input for the higher layers. For example, in order to extract relations between concepts, we should consider the underlying hierarchy to identify the right level of generalization for the domain and range of the relation. The two bottom layers correspond to the lexical level of ontology learning. The task in this part of the layer is to detect the relevant terminology as well as groups of synonymous terms. The extracted terms and synonym groups can then form the basis for the formation of concepts. Concepts differ from terms in that they are ontological entities and thus abstractions of human thought. According to the formalization, concepts are triples $c := \langle i(c), [c], Ref_c \rangle$ consisting of an intensional description $i(c)$, an extension $[c]$ and a reference function Ref_c representing how the concept is symbolically realized in a text corpus, an image, etc. At higher levels of the layer cake, we find the layers corresponding to the tasks of learning concept hierarchy, relations, a relation hierarchy, as well as, deriving arbitrary rules and axioms. The top two layers correspond to the most challenging task, as in principle there is no limit on the type and complexity of axioms and rules to be learned. In practice, however, as we commit to a specific knowledge representation language, the types of axioms allowed are usually restricted.

In the subsections to follow the learning layer cake will be presented in more detail (based on [18] and [22]).

Terms

Term extraction is a prerequisite for all aspects of ontology learning from text. Terms are linguistic realizations of domain-specific concepts and are therefore central to further, more complex tasks. There are many examples of term extraction methods that could be used as a first step in ontology learning from text. Most of them are based on information retrieval methods for term indexing, but many also take inspiration from text mining, as well as, terminology and NLP research.

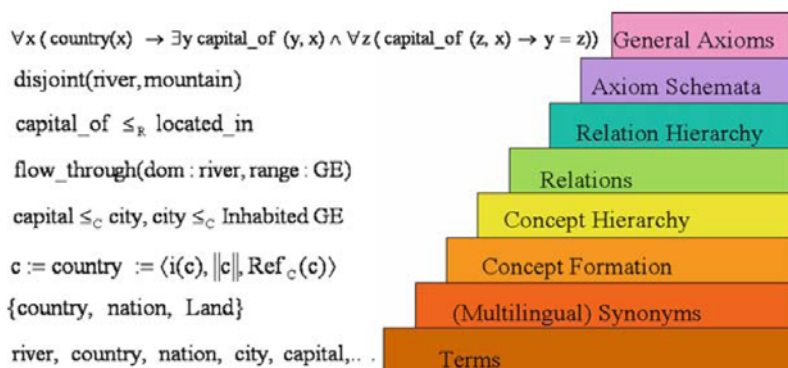


Fig. 4. Ontology learning layer cake [21]

Term extraction implies more or less advanced levels of linguistic processing, i.e. phrase analysis to identify complex noun phrases that may express terms and dependency structure analysis to identify their internal semantic structure. As such parsers are not always readily available, much of the research on this layer in ontology learning has remained rather restricted. The state-of-the-art is mostly to run a part-of-speech tagger over the domain corpus used for the ontology learning task and then to identify possible terms by manually constructing ad-hoc patterns, whereas more advanced approaches to term extraction for ontology learning build on deeper linguistic analysis. Additionally, and in order to identify only relevant term candidates, a statistical processing step may be included that compares the distribution of terms between corpora. A text mining approach (based on the T-GSP algorithm) presented in [23] has shown very good results for finding candidates for compound terms and proper names.

Synonyms

The synonym level addresses the acquisition of semantic term variants in a language, and between languages, where the latter in fact concerns the acquisition of term translations. Much of the work in this area has focused on the integration of WordNet for the acquisition of English synonyms, and EuroWordNet for bilingual and multilingual synonyms and term translations. An important aspect of this work is the identification of the appropriate (WordNet/EuroWordNet) sense of the term in question, which determines the set of synonyms that are to be extracted. This involves standard word sense disambiguation algorithms. However, specifically in the ontology learning context, researchers have exploited the fact that ambiguous terms have very specific meanings in particular domains allowing for an integrated approach to sense disambiguation and domain specific synonym extraction.

In contrast to using readily available synonym sets such as provided by WordNet and related lexical resources, researchers have also worked on algorithms for dynamic acquisition of synonyms by clustering and related techniques. On this basis much work has been done on synonym acquisition from text corpora that is based on distributional hypothesis that terms are similar in meaning to the extent in which they share syntactic contexts. Related work originates out of term indexing for information

retrieval, e.g. the family of Latent Semantic Indexing algorithms (LSI, LSA, PLSI and others). LSI and related approaches apply dimension reduction techniques to reveal inherent connections between words, thus leading to group formation. LSA/LSI-based techniques are interesting as they do not run into data sparseness problems such as approaches relying on raw data. In [24] a knowledge-poor text mining algorithm is presented. This algorithm shows interesting features and will be further upgraded.

Concepts

In [18], three paradigms of concept induction have been indicated:

- conceptual clustering;
- linguistic analysis;
- inductive methods.

Conceptual clustering approaches such as Formal Concept Analysis have been applied to form concepts and to order them hierarchically at the same time. Conceptual clustering approaches typically induce an intentional description for each concept in terms of the attributes that it shares with other concepts as well as those that distinguish it from other concepts.

Linguistic analysis techniques can be applied to derive an intentional description of a concept in the form of a natural language description. The definition of the term *knowledge management practices*: “*a kind of practice, knowledge of how something is customarily done, relating to the knowledge of management, the process of capturing value, knowledge and understanding of corporate information, using IT systems, in order to maintain, re-use and de-ploy that knowledge.*” is compositionally determined on the basis of the definitions of *knowledge management* and *practice*. For this purpose, disambiguation with respect to the different senses of a word with respect to its several meanings in a lexical database (such as WordNet) is required. Further, a set of rules is specified which drive the above compositional generation of definitions.

Finally, given a populated knowledge base, approaches based on *inductive learning* such as Inductive Logic Programming can be applied to derive rules describing a group of instances intentionally. Such an approach can for example be used to reorganize a taxonomy or to discover gaps in conceptual definitions.

Concept Hierarchy

Different methods have been applied to learn taxonomic relations from texts. Noteworthy approaches are based on matching lexico-syntactic patterns, clustering, phrase analysis as well as classification. Some text mining methods for finding close meaning pairs provide as a side effect the taxonomic relationships (e.g. [24]).

Relations

In order to discover arbitrary relations between words, different techniques from text mining, machine learning and statistical natural language processing community have found application in ontology learning. In order to discover ‘anonymous’ associations between words, one can look for a strong co-occurrence between words within a certain boundary, i.e., a window of words, a sentence or a paragraph. Association rule discovery algorithm can be utilized to represent co-occurrences of words within a sentence as transactions. This representation allows to calculate the support and

confidence for binary transactions and thus to detect anonymous binary associations between words. Good text mining results can be obtained with the algorithms mining for patterns. The algorithm T-GSP presented in [19], which enables searching for specific patterns expressed by regular expressions, can be efficiently applied for discovering associations relationships between concepts [25].

In the computational linguistics community, the task of discovering strong associations between words is typically called collocation discovery. The idea here is to discover words which co-occur beyond chance in a statistically significant manner. Statistical significance is typically checked using some test such as the Student's t-test or the χ^2 -test. Other researchers have aimed at learning labeled relations by relying on linguistic predicate argument dependencies. Typically, verb structures are considered for this purpose. When learning relations, a crucial issue is to find the right level of abstraction with respect to the concept hierarchy for the domain and range of the relation in question.

Rules

The success of OWL that allows for modeling far more expressive axioms has led to some advances in the direction of learning complex ontologies and rules.

A research towards generating formal class descriptions from extracted natural language definitions, e.g. from online glossaries and encyclopedias has been started recently. The implementation of this approach is essentially based on a syntactic transformation of natural language definitions into OWL DL axioms in line with previous work on lexico-syntactic patterns and lexical entailment.

One of the first methods for learning disjointness axioms relies on a statistical analysis of enumerations which has been implemented as part of the Text2Onto framework [26].

Evaluation

[27] and [18] argue there are not many gold standards that could be used for evaluation of ontology learning methods. The desired result of ontology learning is not a simple list with binary classifications, but a far more complicated structure. To make it even worse, there is no clear set of knowledge-to-be-acquired, not even for very specialized domains. There are several possibilities of conceptualizations for one domain that might differ in their usefulness for different groups of people, but not in their soundness and justification. So even if the outcome of an algorithm does not compare well with a manually built ontology, how can its quality be judged?

However, several approaches at approximating the appropriateness of ontology learning methods have been done. One of them [28] tries to measure the 'corpus fit' of the ontology by considering the frequency with which the terms in the ontology appear in the corpus. AEON framework [29] aims to automatize the application of the OntoClean methodology, hence ensuring the formal consistency of an ontology. [30] proposes integration of ontology learning and evaluation. It is an approach to exploiting contextual information, or confidence and relevance values for resolving logical inconsistencies in learned ontologies, and to optimize the outcome of the ontology learning process.

4.3 Ontology Learning Tools

Since building an ontology for a huge amount of data is a difficult and time-consuming task a number of tools have been developed in order to support the user in constructing ontologies from a given set of (textual) data. Some well-known and frequently cited tools include TextToOnto [31], Text2Onto [26], OntoLT [32] and OntoLearn [33] (TextToOnto was originally integrated into the KAON ontology engineering environment, OntoLT was integrated with Protégé and Text2Onto has been recently integrated with the NeOn Toolkit).

All these tools implement various methods. OntoLearn for example integrates a word sense disambiguation component to derive intentional descriptions of complex domain-specific terms, which are assumed to denote concepts, on the basis of WordNet glosses. In this sense, OntoLearn also induces intentionally defined domain concepts and ingeniously exploits the knowledge available in general resources for a specific domain. OntoLT, which is available as a plugin to the Protégé ontology editor, allows for term extraction using various measures such as *tf.idf* and extraction of taxonomic relations relying on interpreting modifiers (nominal or adjectival) as introducing subclasses.

TextToOnto is a framework containing various tools for ontology learning. It includes standard term extraction using a number of different measures, the algorithm for mining relations based on association rules as well as hierarchical clustering algorithms based on Formal Concept Analysis. Its successor, Text2Onto, besides implementing most of the algorithms available also in TextToOnto, abstracts from a specific knowledge representation language and stores the learned ontology primitives in the form of a meta-model called Possible Ontologies Model (POM), which can then be translated to any reasonably expressive knowledge representation language, in particular to OWL and RDFS. On the other hand, it implements a framework for data-driven and incremental learning in a sense that changes in the underlying corpus are propagated to the algorithms, thus leading to explicit changes to the POM. The advantage is that these changes can be easily traced back to the original corpus changes, which gives more control to the ontology engineer.

5 Ontology Integration

In the ontology integration area one can distinguish two main problems: building one ontology from two or more existing ontologies, and expressing entities from one ontology by means of entities from another ontology. Both problems may occur during the system usage, but the first one is especially important as it is expected that several domain ontologies will be created. Below, we describe the problem of ontology integration in detail and introduce solutions which can be adopted to the system.

5.1 Problem Outline

The integration of two or more ontologies generally refers to a process of identification and establishing correspondences between elements of these ontologies. However, ontologies may differ from each other in many aspects: language types in

which they are expressed, structure, domains, etc., which makes the integration process potentially very difficult and time-consuming. The ontology integration process is heterogeneous and various characteristics of it have been proposed in the literature. One can distinguish:

- *semantic integration* (focused on intended meaning of the concepts), *structural integration*, and *syntax integration* (focused on providing uniform formalism in which integrated ontologies are expressed);
- *global-centric approach*, where concepts of the global ontology are mapped into the local ontologies and *local-centric approach*, where concepts of the local ontologies are mapped into a global ontology;
- type of integration: mapping, merging, alignment, translation, etc.

Moreover there are several types of mismatches between integrated ontologies. The mismatches can refer to: *language* – formalisms used for expressing ontologies, *terminology* (synonyms, homonyms), *encoding* (different value formats e.g. grams vs. pounds), *paradigms* – different ways of representation of concepts (subclasses vs. attributes), *content level* – differences in the scope and the level of details in which a given domain is modeled in ontologies.

5.2 Types of Integration

In the literature several different types of integration of ontologies have been introduced, the most common are: merging, mapping/matching and alignment. Unfortunately, there is no common understanding of these notions and given definitions differ considerably. Below, the review of definitions and approaches to these three types of integration is provided.

Ontology Merging

The ontology merging process refers to creating a new ontology from two or more input ontologies. The new ontology should be a unification of original ontologies and should be capable of replacing them. This definition is quite common and used in many publications e.g. [34]. In the paper [38] a bit different approach has been proposed. The ontology merging has been defined as “the minimal union of vocabularies S_1 and S_2 and axioms A_1 and A_2 of two ontologies $O_1 = (S_1, A_1)$ to $O_2 = (S_2, A_2)$ that respects their articulation. Articulation of ontologies is a result of the mapping process. The exemplary approaches to ontology merging are presented below.

FCA-merge method for ontology merging has been proposed in [39]. It is the bottom-up approach in which techniques taken from Formal Concept Analysis (FCA) theory are applied. The merged ontology is generated from two input ontologies and a repository of text documents relevant to the input ontologies. The method consists of three steps: context generation, computing a concept lattice and final ontology generation. In the *context generation* step for each ontology a formal context is created. A formal context has been defined as $K = (G, M, I)$, where G - is a set of documents; M - set of concepts and I - binary relations, for $g \in G, m \in M$, a relation $(g, m) \in I$ means that document m contains an instance of m . The domain-specific lexicon is used for finding instances of concepts in documents. In the *computing a concept*

lattice step the pruned concept lattice is created by merging two formal contexts obtained in the previous step. In the *final ontology generation* step the merged ontology is generated from the pruned concept lattice. Several heuristics are applied in order to generate automatically candidate entries to the final ontology. However, in this step user interaction is required as several actions cannot be fully automated, e.g. possible conflicts or duplicates resolving.

The PROMPT algorithm for ontology merging and alignment has been introduced in [40]. The method supports the user in creation of one ontology from two input ontologies. In the process the following steps can be distinguished:

1. Creation of an initial list of matches based on names of classes. The appropriate similarity measure should be provided by the user.
2. Selection of an operation to be executed by the user. Each operation has been assigned one of the following: (1) changes that can be done automatically; (2) new suggestions; and (3) conflicts that the operation may introduce. The set of available ontology-merging operations includes among others: merge classes or slots, merge bindings between a slot and a class, or perform a shallow copy of a class.
3. Execution of the selected operation, automatic execution of additional changes based on the type of operation.
4. Generation of a list of suggestions for the user based on the structure of the current version of a created ontology.
5. Determination of conflicts that the last operation introduced in the ontology and finding possible solutions for those conflicts.

The steps 2-5 are realized in a cycle.

Ontology Mapping

An ontology mapping refers to the way in which one ontology may be expressed in entities from another ontology. The mapped ontologies are not changed, rather an additional specification (functions, set of axioms, queries) describing correspondence between entities is provided. Often mapping is done only in one direction e.g. the entities of one ontology are connected with entities of the second ontology but not vice versa. One of the main ontology mapping purposes is to allow collaboration between various systems which use various ontologies.

Various ideas of ontology mappings can be found in the literature. Here we present the definitions which seem to be the most representative. In [38] ontology mapping is defined as a morphism of ontological signatures. An ontology has been defined as a pair $O = (S, A)$ where S is the (ontological) signature describing the vocabulary, A is a set of (ontological) axioms, which indicates the intended meaning of the vocabulary. Based on this definition an ontology mapping from $O_1 = (S_1, A_1)$ to $O_2 = (S_2, A_2)$ is specified as morphism $f: S_1 \rightarrow S_2$ of ontological signatures such that $A_2 \models f(A_1)$. In [41] mapping is seen as a process of finding a semantic mapping between original ontologies. The authors concentrate here on finding one-to-one mapping between taxonomies. In [35] the following interpretation of mappings between two ontologies has been introduced: “for each entity (concept, relation, attribute, etc.) in one ontology we try to find a corresponding entity in the second ontology, with the same or the closest intended meaning; usually this correspondence is expressed by 1 to 1 functions.” In [36] mapping is defined as “a (declarative) specification of the semantic

overlap between two ontologies". Such representation of mapping indicates correspondences between entities from two ontologies and is not incorporated in definitions of these ontologies. According to the authors such correspondences are typically expressed as a set of axioms in a specific mapping language. The exemplary approaches to ontology mapping are presented below.

In [42] the problem of mapping between a global ontology and a local ontology is discussed. The authors stated that it is better to express mapping concepts from one ontology to concepts from another one as a view or query over the second ontology, rather than a certain match function. Such approach requires defining a query language, which should be included in the specification of a given ontology. An ontology has been defined as a pair $O = (L, A)$, where L is a theory in a logic and A stands for the alphabet of terms used in the ontology. The authors propose a formal framework for ontology integration systems. The framework is defined as a triple $(G, S, M_{G,S})$ where:

- $G = (L_G, A_G)$ – is a global ontology,
- S is a set of local ontologies – it consists of n ontologies denoted by S_1, \dots, S_n , $S_i = (L_{S_i}, A_{S_i})$;
- $M_{G,S}$ is a mapping between global ontology and the local ontologies.

The conceptual framework for ontology mapping process has been proposed in [43]. The framework consists of five horizontal modules reflecting the distinct phases in a mapping process. These modules are *Lift & Normalization*, *Similarity*, *Semantic Bridging*, *Execution* and *Post-processing*. The mapping between ontologies is represented by so-called semantic bridges. The semantic bridges are specified in the five following dimensions:

1. *Entity dimension* – indicates correspondences between ontology entities i.e. between concepts, relations, and attributes.
2. *Cardinality dimension* indicates the number of ontology entities at both sides of the semantic bridge (1:1, 1: n, n:m).
3. *Structural dimension* indicates the way how elementary bridges may be combined into more complex bridges. The following relations have been specified between bridges: specialization, alternatives, composition, and abstraction.
4. *Constraint dimension* is used for defining constraints concerning instances. The fulfillment of these constraints is required for executing transformation rule associated with a given bridge.
5. *Transformation dimension*: describes how instances from the source ontology are transformed during the mapping process.

The GLUE system has been described in [41]. In this system learning techniques are used for creating semi-automatically semantic mapping between ontologies, which are expressed in the same way i.e. the concepts are specified in a comparable manner. The system consists of three main components, namely: *Distribution Estimator*, *Similarity Estimator* and *Relaxation Labeler*. The *Distribution Estimator* module takes as input two ontologies O_1 and O_2 (also instances of concepts are included into input data) and for every pair of concepts $\langle a, b \rangle$ such that $a \in O_1$, $b \in O_2$, computes joint probability distribution. In the *Similarity Estimator* module the result obtained from Distributed Estimator component is used for calculating a similarity function.

The output of the module is a similarity matrix between the concepts in the two taxonomies. In the *Relaxation Labeler* module, based on the similarity matrix and additional to ontologies domain constraints and heuristic knowledge, generates mapping configuration which best satisfies the domain constraints.

In [44] the ontology mapping process has been defined based on information-flow theory proposed by Barwise and Seligman [45]. Authors analyzed the problem of mapping two ontologies: a reference ontology without instances and a local ontology populated with instances associated to concepts. The proposed method allows carrying out mapping between ontologies expressed in Horn logic. The mapping is defined in terms of logical infomorphisms between local logics. A local logic has been specified as quadruple $L = (I, T, \models, \perp)$ where I is a set of instances, T is a set of types, \models is a classification relation between elements of I and T , and \perp stands for a consequence relation between subsets of T . In the process every considered ontology has a local logic associated with it.

The H-match algorithm for ontology matching has been introduced in [46]. In the algorithm an ontology is seen as a set of concepts, properties, and semantic relations. In the method two sets of features are distinguished, namely linguistic and contextual. *Linguistic features* refer to names of ontology elements and their meaning. The meaning of names for ontology matching is determined based on a thesaurus of terms and weighted terminological relationships among them. In H-Match the thesaurus is automatically derived from the lexical system WordNet. *Contextual features* of a concept c refer both to the properties and to the concepts directly related to c through a semantic relation in an ontology. The context $Ctx(c)$ of a concept c defined as: $Ctx(c) = P(c) \cup C(c)$ where $P(c)$ is a set of properties of c and $C(c)$ is a set of concepts that participate in a semantic relation with c . In H-match the four following models of matching have been defined:

- *surface* - matching based only on names of ontology elements;
- *shallow* - matching based on names and concept properties;
- *deep* - matching based on names, concepts properties and semantic relations;
- *intensive* - matching based on names, concepts properties, semantic relations, and property values.

Ontology Alignment

The alignment of one ontology with another means that for each entity (concept, relations, instance, etc.) from the first ontology the corresponding object is found from the second one. The matched entities should have the same meaning. The equality alignment, one-to-one matching, is the most investigated type of alignment.

Here we provide selected definitions of ontology alignment; some others can be found in the literature. [38] defines ontology alignment as “the task of establishing a collection of binary relations between the vocabularies of two ontologies”. The authors introduce a common intermediate source ontology O_0 - articulation of two ontologies, and decomposed a binary relation into a pair of mappings from the intermediate ontology to aligned ontologies. In [35] ontology alignment is defined as “the process of bringing two or more ontologies into mutual agreement, making them consistent and coherent with one and another”. In such a process analyzed ontologies may be affected, e.g. contradictory relations may be removed. In [36] ontology

alignment is defined as a process of discovering similarities between two input ontologies, the result of which is a specification of similarities between these ontologies. In [37] the formal alignment function *align* has been defined for the *equal* relation, the appropriate formula is given below.

$$\text{align}: E \times O \times O \rightarrow E, \quad (1)$$

where: E - a set of all entities, O – a set of possible ontologies.

In practice objects may be aligned by using other than equal relations e.g. subsumption. The formal general function *gen_align* for alignment has been defined in [37] as:

$$\text{gen_align}: E \times O \times O \rightarrow E \times M, \quad (2)$$

where: E is a vocabulary, all terms $e \in E$; O is the set of possible ontologies, M is the set of possible alignment relations.

In general, the alignment of ontologies can be defined as a set of quadruples $\langle x, y, r, l \rangle$, where x and y stand for aligned entities, r is a relation between matched entities, l expresses a level of confidence of a given alignment (it is an additional element). The exemplary approaches for ontology alignment are presented below.

In [36], [34] the general process for ontology alignment has been proposed. The process is iterative and consists of the five following steps:

- In the *Feature Engineering* step the parts of ontologies definition are chosen for the needs of the description of entities to be aligned. The selected features which characterize entities depend on the structures and languages, on which the integrated ontologies are described.
- In the Search Step Selection the pairs of entities for comparison are selected. The typical methods for candidate pairs selection are: (1) Cartesian product of sets: E_1 – consisting of all entities from O_1 and E_2 – consisting of all entities from O_2 ; (2) all pairs of entities of the same type (concept, relation, instance).
- In the *Similarity Computation* step the similarity between entities in candidate pairs are determined. The similarity is calculated based on selected features which exist in both ontologies and an adequate similarity measure. More than one measure may be applied for calculating similarity between entities in one pair (e.g. one measure for one feature) and also similarity measures may vary for different types of compared entities.
- In the *Similarity Aggregation* step the single value indicating similarity between entities is calculated. Typically, the value is an average value for similarity measures used in the previous step or the weighted average value. The weight may be assigned manually or may be determined, e.g. by using data mining or machine learning algorithms on a training set.
- In the *Interpretation* step a decision whether two entities should be aligned is made based on calculated similarity measures. In the literature several methods for this purpose have been introduced. In the simplest methods, a threshold is used i.e. the alignment is made if the aggregate similarity measure is greater than a given threshold θ .

In [47] Integrated Learning In Alignment of Data and Schema (ILIADS) algorithm has been introduced. The method allows for aligning two ontologies written in OWL-lite language by means of sets of entities classes, properties, individuals, data values, triples and axioms. The proposed algorithm finds a set of axioms and facts that link entities in aligned ontologies. In the algorithm the following general steps can be distinguished:

1. Creation of an ontology O from input ontologies O_1, O_2 ; $O = O_1 \cup O_2$;
2. Finding candidate groups of similar entities in ontologies by using a hierarchical agglomerative clustering method;
3. Determination of an equivalence or subsumption relationships between the sets of entities;
4. Computing the logical consequences and logical inference similarity of the new relationships (a relationship models cluster created by merging two groups) based on the current state of O ;
5. Addition of the axiom of the highest logical similarity to a set of alignment and update ontology O .

5.3 Similarity Measures

Similarity is the key issue in many ontology integration methods. Typically comparison of ontologies is performed by comparison of their entities. In the comprehensive methods not only lexical or syntax level of entities should be taken into consideration but also their meaning and usage. In order to compare two ontologies various measures should be applied, adequate for given types of compared objects. The examples of measures for simple data type, set of entities and concepts are presented below.

Measures for Data Types

Equality

In some situations it is reasonable to assume that two entities are similar only if their data values are equal. In the simplest case a similarity function returns 1 if compared values are equal and 0 in other situations. In the context of ontology objects equality can be determined by using existing logical assertions. These assertions may be included in the definition of an ontology or may be determined manually or by using automated or semi-automated methods. The appropriate similarity function is presented below.

$$sim_{eq_object}(o_1, o_2) = \begin{cases} 1, & \text{if } \text{assert}(o_1, o_2) \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

String Values

The syntactic similarity measure for string values is defined by using equation 4. In this measure the edit distance of two strings is used.

$$sim_{syntactic}(v_1, v_2) = \max\left(0, \frac{\min(|v_1|, |v_2|) - ed(v_1, v_2)}{\min(|v_1|, |v_2|)}\right) \quad (4)$$

where: $ed(a, b)$ is the edit distance between two strings; v_1, v_2 are string values; $|v|$ is the number of characters in v .

Number from given limited range

$$sim_{diff}(n_1, n_2) = 1 - \left(\frac{|n_1 - n_2|}{mdiff} \right)^\gamma \quad (5)$$

where: n_1, n_2 – numbers from a given range; $mdiff$ – difference between the maximal and minimal value for considered data type; and γ - a number, $\gamma > 0$.

Functions for Sets

Dice Coefficient

$$sim_{dice}(A, B) = \frac{2 * |A \cap B|}{|A| + |B|} \quad (6)$$

where A and B are set of individuals.

In order to use the Dice coefficient for comparing two sets of entities it is necessary that entities from these two sets are identified by the same method, which makes it possible to determine whether a given entity belongs to one or both of these sets. In case of comparing ontologies this requirement may be difficult to fulfill.

Methods Based on Similarity between Individuals

Similarity between two sets of objects may be calculated by using similarities between individuals belonging to these sets. The methods of this kind rely on other measures which can be applied for calculating similarity between single entities. Three of them are presented below:

1. *Min linkage* - the minimum of similarity between individuals is used:

$$sim_{link_min}(A, B) = \min_{(a,b) \in A, b \in B} (sim(a, b)) \quad (7)$$

2. *Max linkage* - the maximum of similarity between individuals is used:

$$Max\ linkage: \quad sim_{link_max}(A, B) = \max_{(a,b) \in A, b \in B} (sim(a, b)) \quad (8)$$

3. *Average linkage* - the average similarity between individuals is applied:

$$Average\ Linkage: \quad sim_{link_avg}(A, B) = \frac{\sum_{\forall (a,b) \in A, b \in B} sim(a, b)}{|A| * |B|} \quad (9)$$

Similarity Measures for Entities

Label Similarity of Concepts

The basic property of concepts in ontologies is label. Labels are names of entities very often given by people and somehow indicating the meanings of concepts. For measuring the similarity of concepts based on their labels the syntactic similarity function may be used:

$$sim_{label}(c_1, c_2) = sim_{syntactic}(label(c_1), label(c_2)) \quad (10)$$

where: c_1, c_2 – concepts; $label(c)$ – label of concept c .

Taxonomic Similarity for Concepts

One of the most known generic measures of similarity of concepts in one concept hierarchy has been presented in [48]. The formula of this function is shown below.

$$sim_{taxonomic}(c_1, c_2) = \begin{cases} e^{-\alpha l} * \frac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}}, & \text{if } c_1 \neq c_2. \\ 1, & \text{otherwise.} \end{cases} \quad (11)$$

where: $\alpha \geq 0, \beta \geq 0$ are parameters used for scaling impact of respectively the shortest path of length l ; and the depth h in the taxonomy hierarchy.

Similarities Based on Instance

Several different similarity measures of concepts have been defined based on joint probability distribution between them. Joint probability is calculated with reference to the sets of instances of considered concepts. Joint probability between two concepts A and B consists of the four following probabilities:

- $P(c_1, c_2)$ – the probability that a randomly chosen instance belongs both to a concept c_1 and a concept c_2 .
- $P(c_1, \sim c_2)$ – the probability that a randomly chosen instance belongs to a concept c_1 but not to a concept c_2 .
- $P(\sim c_1, c_2)$ – the probability that a randomly chosen instance belongs to a concept c_2 but not to a concept c_1 .
- $P(\sim c_1, \sim c_2)$ – the probability that a randomly chosen instance does not belong to a concept c_1 nor to a concept c_2 .

Jaccard Coefficient for Concepts

In [41] Jaccard coefficient is used as “exact” similarity measure:

$$sim_{concept_jaccard}(c_1, c_2) = \frac{P(c_1, c_2)}{P(c_1, c_2) + P(c_1, \sim c_2) + P(\sim c_1, c_2)} \quad (12)$$

where c_1 and c_2 are concepts

The lowest value for this measure is 0 when concepts c_1 and c_2 are disjoint. The highest value is 1, when c_1 and c_2 are the same concept.

Concept Similarity of Instances

In [37] a similarity measure for instances of two concepts has been proposed. The measure is based on similarity of concepts, the proposed function is given below.

$$sim_{parent}(i_1, i_2) = sim_{object}(c_1, c_2) \quad (13)$$

where i_1, i_2 are instances and $i_1 \in instances(c_1), i_2 \in instances(c_2)$; $instances(c)$ is the set of instances of a concept c .

Similarity Based on Context

If the context of usage of two concepts from an ontology is available the similarity between these concepts may be expressed with respect to the context. In [37] the following function has been proposed:

$$sim_{use}(e_1, e_2) = sim_{diff}(usage(e_1, cnxt), usage(e_2, cnxt)) \quad (14)$$

where: e_1, e_2 are entities from the ontologies (e.g. concepts or instances of concept); $usage(e, cnxt)$ returns the frequency of usage of entity e in the context $cnxt$; sim_{diff} is the function defined in equation 5.

In [49] the Wordnet synsets have been used for measuring similarity of concepts. The proposed definitions are based on definitions of similarities between two coherent intensional meanings of concepts introduced in [50]. Assuming that: each sense in a Wordnet syntset indicates a concept; O_1, O_2 – ontologies; c_i, c_j – concepts, $c_i \in O_1, c_j \in O_1$; $S[c_n]$ – region of a synset including concept c_n the four following levels of similarity have been defined:

- *Disjoint*: concepts c_i, c_j are disjoint if the conjunction of their synsets definition implies false: $S[c_i] \wedge S[c_j] \equiv \text{false}$.
- *Specialized*: concept c_i is specialization of concept c_j if synset of c_j is an implication of the synset of c_i : $S[c_i] \wedge S[c_j] \equiv S[c_i] \Rightarrow c_i \leq c_j$.
- *Overlapping*: if the conjunction of synsets for c_i and c_j constitutes region for another concept c_k : $S[c_i] \wedge S[c_j] \equiv S[c_k]$.

6 Summary

In the paper we have presented a general idea of the role of ontologies in an experimental knowledge-base in the SYNAT project. In our system ontologies are considered to be pervasive, and utilized for modeling almost all aspects of the information stored in the Knowledge Base. Furthermore, we outlined details of the system ontology that is planned to be used as a reference for meta knowledge in the platform. We shortly introduced processes, methods, and tools for ontology maintenance, learning and integration which are assumed to be applied or adapted in our system especially in the ontology subsystem. It allows us to provide system which fulfills all users' requirements related to building, maintaining and integration of ontologies.

The planned research refers to the domain of ontology engineering, including constructing, and widely understood maintenance. It can be divided into the following three main topics: ontology learning, ontology constructing and applications, and ontology integration.

6.1 Ontology Learning

Research in the ontology learning area will be concentrated on automatic or semi-automatic methods that can be used for producing candidate entries for a given

ontology. Typically, such methods consist of two general phases: discovering terms and relations between terms and generation of candidate entries for a given ontology. In the former phase various techniques from natural language processing, statistics, and text mining areas are applied for exploration of text-document repositories in order to discover interesting terms and relations between terms. Here, the most common tasks include:

- compound term discovery;
- discovery of synonyms and homonyms;
- taxonomy discovery and/or validation;
- discovery of semantic relationship between terms for ontology building support;
- discovering relation signatures.

In the latter phase candidate entries are generated based on the results from the first phase. The entries may concern both lexical and concept layers (also instances), e.g.: a concept, a semantic relation between concepts (concept layer), and synonyms for a given term (lexical layer).

The goal of the planned research is to develop methods for discovering entries for a given ontology from text repositories. These methods will be based on text- and data-mining techniques, mostly with shallow grammatical analysis of texts and methods in which ontologies will be incorporated in algorithms used in both discovering and candidate generation phases. In particular, we plan to upgrade some methods worked out earlier by the team ([19], [23]) and to test new text mining-based approaches.

6.2 Ontology Constructing and Applications

The research concerning ontology building area will concern methods which may be applied for extending ontologies, especially the system ontology, and ways of application of it in the searching systems. The former task is concentrated on ways of extending and modeling notions on the basis of the SYNAT ontologies. It involves, among others, research on linking concepts between the system ontology and domain ontologies. The latter task is focused on applying ontologies for the evaluation of results in searching systems.

6.3 Ontology Integration

In the ontology integration area the planned research concerns both mentioned in the paper problems: building one ontology from two or more existing ontologies and expressing entities from one ontology by means of entities from another ontology. Particularly it will be focused on:

- methods in which the context associated with entries of ontologies is explored in order to obtain better correspondence between entries coming from merging ontologies;
- complex methods for measuring similarity between entries of ontologies based on all available layers of ontologies (e.g. concept and lexical layers).

References

- [1] Gawrysiak, P., Ryzko, D.: Acquisition of scientific information from the Internet: The PASSIM Project Concept. In: Proc. of ICAART 2011, Rome (2011)
- [2] Web of Science, http://thomsonreuters.com/products_services/science/science_products/a-z/web_of_science/
- [3] Scopus, <http://www.scopus.com/home.url>
- [4] Google Scholar, <http://scholar.google.com>
- [5] Etxebarría, G., Gomez-Uranga, M.: Use of Scopus and Google Scholar to measure social science production in four major Spanish universities. *Scientometrics* 82, 333–349 (2010)
- [6] Microsoft Academic Search, <http://academic.research.microsoft.com/>
- [7] [SR03] Sheth, A., Ramakrishnan, C.: Semantic (Web) Technology In Action: Ontology Driven Information Systems for Search, Integration and Analysis. *Bulletin of the Technical Committee on Data Engineering* 26(4), 40–48 (2003)
- [8] Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. *Scientific American* (May 2001)
- [9] Della Valle, E., Cerizza, D., Celino, I., Estublier, J., Vega, G., Kerrigan, M., Ramírez, J., Villazon, B., Guarrera, P., Zhao, G., Monteleone, G.: SEEMP: An Semantic Interoperability Infrastructure for e-Government Services in the Employment Sector. In: Franconi, E., Kifer, M., May, W. (eds.) *ESWC 2007*. LNCS, vol. 4519, pp. 220–234. Springer, Heidelberg (2007)
- [10] Project Boemie, financed by sixth framework programme UE (2006-2009), <http://www.boemie.org/>
- [11] State of the Art on Ontology And Vocabulary Building & Maintenance Research And Applications Solutions for System and Domain Ontologies. Technical Report B12 in SYNAT project, Institute of Computer Science, WUT (March 2011)
- [12] Suárez-Figueroa, M.C., et al.: Revision and Extension of the NeOn Methodology for Building Contextualized Ontology Networks. NeOn Deliverable D5.4.3, NeOn Project (January 2010), <http://www.neon-project.org>
- [13] Ye, J., Coyle, L., Dobson, S., Nixon, P.: Ontology-based models in pervasive computing systems. *The Knowledge Engineering Review* 22(04), 315–347 (2007)
- [14] Protégé Editor, <http://protege.stanford.edu>
- [15] NeOn Toolkit, <http://neon-toolkit.org>
- [16] Gruninger, M., Fox, M.S.: Methodology for the design and evaluation of ontologies. In: Proc. Int. Joint Conf. AI Workshop on Basic Ontological Issues in Knowledge Sharing, Montreal (1995)
- [17] Fernández-López, M., Gómez-Pérez, A., Jurysto, N.: METHONOLOGY: From Ontological Art Towards Ontological Engineering. In: Spring Symposium on Ontological Engineering of AAAI, Stanford University, California, pp. 33–40 (1997)
- [18] Cimiano, P., Maedche, A., Staab, S., Voelker, J.: Ontology learning, *Handbook on ontologies*. Springer, Heidelberg (2009)
- [19] Gawrysiak, P., Protaziuk, G., Rybiński, H., Delteil, A.: Text onto miner – A semi automated ontology building system. In: An, A., Matwin, S., Raś, Z.W., Ślęzak, D. (eds.) *Foundations of Intelligent Systems*. LNCS (LNAI), vol. 4994, pp. 563–573. Springer, Heidelberg (2008)

- [20] Cunningham, H., Humphreys, K., Gaizauskas, R.J., Wilks, Y.: GATE – A general architecture for text engineering. In: *Proceedings of Applied Natural Language Processing (ANLP)*, pp. 29–30 (1997)
- [21] Cimiano, P.: *Ontology Learning and Population from Text: Algorithms, Evaluation and Applications*. Springer, Heidelberg (2006)
- [22] Buitelaar, P., Cimiano, G., Magnini, B.: *Ontology Learning from Text: An Overview*. In: *Ontology learning from text: methods, evaluation and applications*. IOS Press (2005)
- [23] Protaziuk, G., Kryszkiewicz, M., Rybiński, H., Delteil, A.: Discovering compound and proper nouns. In: Kryszkiewicz, M., Peters, J.F., Rybiński, H., Skowron, A. (eds.) *RSEISP 2007. LNCS (LNAI)*, vol. 4585, pp. 505–515. Springer, Heidelberg (2007)
- [24] Rybiński, H., Kryszkiewicz, M., Protaziuk, G., Jakubowski, A., Delteil, A.: Discovering Synonyms Based on Frequent Termsets. In: Kryszkiewicz, M., Peters, J.F., Rybiński, H., Skowron, A. (eds.) *RSEISP 2007. LNCS (LNAI)*, vol. 4585, pp. 516–525. Springer, Heidelberg (2007)
- [25] Rybinski, H., et al.: *Text Mining Approach in Ontology Building and Maintenance Methodology – Project for FT, Final Report and Follow Up*, WUT, ICS Rep. Warsaw (2007)
- [26] Cimiano, P., Völker, J.: Text2Onto. In: Montoyo, A., Muñoz, R., Métais, E. (eds.) *NLDB 2005. LNCS*, vol. 3513, pp. 227–238. Springer, Heidelberg (2005)
- [27] Biemann, C.: *Ontology Learning from Text – a Survey of Methods*. In: *LDV-Forum*, vol. 20(2) (2005)
- [28] Brewster, C., Alani, H., Dasmahapatra, S., Wilks, Y.: Data-driven ontology evaluation. In: *Proc. of the 4th International Conference on Language Resources and Evaluation*, Lisbon (2004)
- [29] Völker, J., Vrandečić, D., Sure, Y.: Automatic evaluation of ontologies (AEON). In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) *ISWC 2005. LNCS*, vol. 3729, pp. 716–731. Springer, Heidelberg (2005)
- [30] Haase, P., Völker, J.: *Ontology learning and reasoning — dealing with uncertainty and inconsistency*. In: da Costa, P.C.G., d’Amato, C., Fanizzi, N., Laskey, K.B., Laskey, K.J., Lukasiewicz, T., Nickles, M., Pool, M. (eds.) *URSW 2005 - 2007. LNCS (LNAI)*, vol. 5327, pp. 366–384. Springer, Heidelberg (2008)
- [31] Maedche, A., Volz, R.: The Text-To-Onto ontology extraction and maintenance system. In: *Workshop on Integrating Data Mining and Knowledge Management, collocated with the 1st International Conference on Data Mining* (2001)
- [32] Buitelaar, P., Olejnik, D., Sintek, M.: A Protégé plug-in for ontology extraction from text based on linguistic analysis. In: Bussler, C.J., Davies, J., Fensel, D., Studer, R. (eds.) *ESWS 2004. LNCS*, vol. 3053, pp. 31–44. Springer, Heidelberg (2004)
- [33] Velardi, P., Navigli, R., Cuchiarelli, A., Neri, F.: Evaluation of OntoLearn, a methodology for automatic population of domain ontologies. In: Buitelaar, P., Cimiano, P., Magnini, B. (eds.) *Ontology Learning from Text: Methods, Applications and Evaluation, Frontiers in Artificial Intelligence and Applications*, vol. 123, pp. 92–106. IOS Press (2005)
- [34] Pinto, H.S., Gomez-Perez, A., Martins, J.P.: Some issues on ontology integration. In: *Proceedings of the IJCAI-1999 Workshop on Ontologies and Problem-Solving methods (KRR5)*, Stockholm, Sweden (1999)
- [35] INTEROP, *Ontology Interoperability, State of the Art Report*. WP8ST3 Deliverable (2004)
- [36] de Bruijn, J., Ehrig, M., Feier, C., Martin-Recuerda, F., Scharffe, F., Weiten, M.: *Ontology Mediation, Merging, and Aligning*. John Wiley & Sons, Ltd (2006)

- [37] Ehrig, M.: *Ontology Alignment Bridging the Semantic Gap*. Springer, Heidelberg (2007)
- [38] Kalfoglou, Y., Schorlemmer, M.: Ontology mapping: the state of the art. *The Knowledge Engineering Review Journal (KER)* 18(1), 1–31 (2003)
- [39] Stumme, G., Maedche, A.: *Ontology Merging for Federated Ontologies on the Semantic Web*. In: *Proceedings of the International Workshop for Foundations of Models for Information Integration (FMII-2001)*, Viterbo, Italy (September 2001)
- [40] Noy, N.F., Musen, M.: *PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment*. In: *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI 2000)*, Austin, TX, USA (2000)
- [41] Doan, A., Madhavan, J., Domingos, P., Halevy, A.: *Ontology matching: A machine learning approach*. In: Staab, S., Studer, R. (eds.) *Handbook on Ontologies in Information Systems*. Springer, Heidelberg (2004)
- [42] Calvanese, D., De Giacomo, G., Lenzerini, M.: *A framework for ontology integration*. In: *Proceedings of the 1st Internationally Semantic Web Working Symposium (SWWS)*, Stanford, CA, USA (2001)
- [43] Maedche, A., Motik, B., Silva, N., Volz, R.: *MAFRA – a Mapping fRamework for distributed ontologies*. In: Gómez-Pérez, A., Benjamins, V.R. (eds.) *EKAW 2002*. LNCS (LNAI), vol. 2473, pp. 235–250. Springer, Heidelberg (2002)
- [44] Kalfoglou, Y., Schorlemmer, M.: *IF-Map: an ontology mapping method based on information flow theory*. *Journal on Data Semantics* 1(1) (October 2003)
- [45] Barwise, J., Seligman, J.: *Information Flow: The Logic of Distributed Systems*. Cambridge University Press (1997)
- [46] Castano, S., Ferrara, A., Montanelli, S.: *Matching Ontologies in Open Networked Systems: Techniques and Applications*. In: Spaccapietra, S., Atzeni, P., Chu, W.W., Catarci, T., Sycara, K. (eds.) *Journal on Data Semantics V*. LNCS, vol. 3870, pp. 25–63. Springer, Heidelberg (2006)
- [47] Udrea, O., Getoor, L., Miller, R.J.: *Leveraging data and structure in ontology integration*. In: *Proceedings of the 2007 ACM SIGMOD International Conference on Management of data, SIGMOD 2007* (2007)
- [48] Rada, R., Mili, H., Bicknell, E., Blettner, M.: *Development and application of a metric on semantic nets*. *IEEE Transactions on Systems, Man and Cybernetics* (1989)
- [49] Cho, M., Kim, H., Kim, P.: *A new method for ontology merging based on concept using wordnet*. In: *The 8th International Conference on Advanced Communication Technology, ICACT 2006*, vol. 3 (2006)
- [50] Hakimpour, F., Geppert, A.: *Resolving Semantic Heterogeneity in Schema Integration: an Ontology Based Approach*. In: *Proc. of the 2nd Intl. Conf. on Formal Ontology in Information Systems*. ACM Press, New York (2001)

Transforming a Flat Metadata Schema to a Semantic Web Ontology: The Polish Digital Libraries Federation and CIDOC CRM Case Study

Cezary Mazurek, Krzysztof Sielski, Maciej Stroński,
Justyna Walkowska, Marcin Werla, and Jan Węglarz

Poznań Supercomputing and Networking Center, ul Z. Noskowskiego 12/14,
61-704 Poznań, Poland
{mazurek, sielski, stroins, ynka, mwerla, weglarz}@man.poznan.pl

Abstract. This paper describes the transformation of the metadata schema used by the Polish Digital Libraries Federation to the CIDOC CRM model implemented in OWL as Erlangen CRM. The need to transform the data from a flat schema to a full-fledged ontology arose during preliminary works in the Polish research project SYNAT. The Digital Libraries Federation site offers aggregated metadata of more than 600,000 publications that constitute the first portion of data introduced into the Integrated Knowledge System - one of the services developed in the SYNAT project. To be able to perform the desired functions, IKS needs heavily linked data that can be searched semantically. The issue is not only one of mapping one schema element to another, as the conceptualization of CIDOC is significantly different from that of the DLF schema. In the paper we identify a number of problems that are common to all such transformations and propose solutions. Finally, we present statistics concerning the mapping process and the resulting knowledge base.

Keywords: CIDOC CRM, digital libraries, metadata, ontologies, RDF repositories, semantic web, thesauri.

1 Introduction

SYNAT is a national research project aimed at the creation of universal open repository platform for hosting and communication of networked resources of knowledge for science, education, and open society of knowledge. It is funded by the Polish National Center for Research and Development (grant no SP/I/1/77065/10) and coordinated by ICM (University of Warsaw).

Poznań Supercomputing and Networking Center (PSNC) is one of the key SYNAT partners. One of the PSNC responsibilities in the project is the creation of a prototype of the Integrated Knowledge System (IKS). The IKS will become a part of a four-layer infrastructure of advanced network services: source data, distributed information services, knowledge integration and front-end services layers. The knowledge integration layer serves as middleware providing access to data from distributed information services, such as digital libraries, museums or scientific and technical information systems. To achieve this goal, a common representation of data is

necessary, to which the existing heterogeneous representations and schemas can be converted.

As the system is dedicated mostly to researchers in the field of humanities, and the first aggregated resources will be the Digital Libraries Federation data, the CIDOC Conceptual Reference Model [4] (implemented in OWL as Erlangen CRM / OWL [11]) has been chosen as the main description ontology. The obtained knowledge representation is stored in an RDF repository, hereinafter referred to as the *knowledge base*.

The remaining part of this paper describes the conversion process and highlights the most important research and technical issues and results obtained so far. We summarize the mission of the Polish Digital Libraries Federation and describe the metadata it aggregates. We discuss the applicability of the CIDOC CRM ontology and justify its choice as the main ontology describing the contents of the knowledge base. We also dedicate some space to describe the auxiliary ontologies and vocabularies that support CIDOC in the knowledge base. Later on we move to discussing the biggest challenges in the process of mapping a flat metadata schema to an event-centric ontology in order to create the knowledge base, such as a very different conceptualization, the need for nesting external type hierarchies (sometimes large, complicated, and even internally inconsistent) and for data enrichment. We propose a modular Knowledge Retrieval System architecture and describe the mapping process. In the last part of this paper we present the results and statistics, and also name some remaining problems and future tasks.

2 Digital Libraries Federation

The PIONIER Network Digital Libraries Federation (DLF, [15]) is the next stage of the development of an infrastructure of distributed digital libraries and repositories in Poland [17]. The name of the DLF corresponds to its nature - it is a set of advanced network services based on the resources available in Polish digital libraries and repositories deployed in the Polish NREN PIONIER. The resources are created by many institutions, such as universities, libraries or museums. The Digital Libraries Federation is maintained by the Poznań Supercomputing and Networking Center.

The mission of the DLF is to:

1. facilitate the use of resources from Polish digital libraries,
2. increase the visibility of Polish resources in the Internet,
3. provide advanced network services based on the digital libraries' content to Internet users and digital libraries creators.

Thanks to the aggregator features of DLF, digital libraries users have been given the possibility to search over the distributed repositories of all federated libraries from one website. The DLF does not store content - after choosing a resource of interest among the search results the user is redirected to the resource owner's website.

The DLF harvests data through the OAI-PMH protocol. The updates are performed nightly. The DLF publishes the harvested metadata further through the OAI-PMH protocol. The added functionalities include e.g. duplicate and similar resources

detection, system of persistent identifiers and a Shibboleth-based mechanism of networked readers' profile[8].

As of April 2011, the number of publication in DLF exceeds 600,000. 64 Polish digital libraries are connected in the DLF, holding content from hundreds of memory institutions. The majority of the content constitutes of newspapers and magazines, mostly historical. Most publications are in Polish, the second popular language is German.

Polish digital museums are also interested in joining the Federation. The first one which actually joined (in April 2011) was the National Museum in Warsaw.

The DLF's metadata schema is described in section 3. More information about the contents is given in section 7.

3 DLF Metadata

The Digital Libraries Federation metadata can be obtained via the OAI-PMH protocol. The basic metadata schema used by the Federation in the Dublin Core Metadata Element Set, but the DLF is now switching to a new, richer metadata schema called PLMET. This schema has been created in accordance with the demands and suggestions of creators digital libraries cooperating with the Federation. The mapping described here and performed by the prototype is prepared for the new schema, but also works on the current one (a subset of PLMET elements).

PLMET is comprised of a number of tags from different established namespaces and a few proprietary tags. The used namespaces are:

- Dublin Core Metadata Element Set, <http://purl.org/dc/elements/1.1/>, *dc*
- Dublin Core Metadata Terms, <http://purl.org/dc/terms/>, *dcterms*
- Electronic Thesis and Dissertation Metadata Standard, <http://www.ndltd.org/standards/metadata/etdms/1.0/>, *etdms*

The proprietary namespace is referred to as *plmet*.

Additionally, the DLF exposes data in the Europeana Semantic Elements schema (<http://www.europeana.eu/schemas/ese/>) which will be referred to as *ese* [3]. Besides a selection of Dublin Core Metadata Terms elements, *ese* includes also several proprietary elements whose values can be generated automatically, mostly on the basis of the *dc* elements. Those proprietary elements do not form part of the PLMET schema, but they are used in the mapping process as additional information.

The schema contains all *dc* elements, that is: *title*, *creator*, *contributor*, *subject*, *coverage*, *description*, *publisher*, *date*, *type*, *format*, *source*, *language*, *relation*, *rights*.

It also contains the following *dcterms* elements (so called qualifiers of elements from *dc*): *alternative* (title), *spatial*, *temporal* (coverage), *abstract*, *tableOfContents* (description), *available*, *created*, *dateAccepted*, *dateCopyrighted*, *dateSubmitted*, *issued*, *modified*, *valid* (dates), *extent*, *medium* (formats), *hasPart*, *isPartOf*, *hasVersion*, *isVersionOf*, *hasFormat*, *isFormatOf*, *references*, *isReferencedBy*, *replaces*, *isReplacedBy*, *requires*, *isRequiredBy*, *conformsTo* (relations), *bibliographicCitation* (identifier), *accessRights*, *license* (rights), *rightsHolder*, *provenance*.

The used elements from *etdms* [1] are: *degree*, *degree.name*, *degree.level*, *degree.discipline*, *degree.grantor* (*degree* is a part of the description)

The proprietary *plmet* elements are *userTag* (subject), *placeOfPublishing* (description), *callNumber* (identifier), *locationOfPhysicalObject* (provenance), *digitisation* and *digitisationSponsor*.

The *ese* elements that are not part of PLMET but are offered by the DLF and used in the mapping are *unstored*, *object*, *provider*, *type*, *rights*, *dataProvider*, *isShownBy*, *isShownAt*.

The semantics of the elements are discussed in more detail section 7.

4 CIDOC CRM as the Knowledge Base Target Ontology

To achieve the goals connected with Semantic Web, Linked Data and intelligent search and resource discovery, an ontology had to be chosen that would suffice to describe all types of data (including the DLF data described above) that might appear in the Integrated Knowledge System, dedicated to researchers in the field of humanities. There have been proposals to create a proprietary ontology, but finally CIDOC Conceptual Reference Model (CRM, [4]) was chosen due to the following reasons:

1. It is a widely recognized standard, so data from the IKS knowledge base will be understandable for wider audiences.
2. Substantial effort has been dedicated to the creation of CIDOC CRM, so the ontology is rather mature and stable.
3. CIDOC is an ontology for describing cultural heritage, and this kind of information will prevail in a humanity-research portal [20].
4. CIDOC is very expressive and is sometimes used as an intermediate representation while mapping between different metadata schemas [16],[10].

Attempts to map the Dublin Core Metadata Element Set (which is a subset of the DLF's PLMET schemata) to CIDOC CRM for museum collections have been described e.g. in [14], [7].

This combination of features allows us to use CIDOC as the main ontology for resource description in the knowledge base. This is a more convenient solution than trying to integrate different, possibly disjoint ontologies for different types of resources.

Still, some extensions (mostly subclasses and subproperties) of CIDOC have been proposed. Additional types of information are described with a number of other standard, smaller ontologies and vocabularies, described shortly in 4.3.

4.1 CIDOC CRM

CIDOC Conceptual Reference Model [4] is an ontology for describing the implicit and explicit concepts and relationships used in cultural heritage documentation. The

ontology defines 86 classes and 137 unique properties. It is event-centric: events such as creation or acquisition are crucial to the domain description.

CIDOC CRM is mostly used to describe museum collections, but as publications (both physical and digital) are also artifacts of cultural heritage, they can be described with the (slightly extended) ontology. Another reason of choosing CIDOC in the SYNAT project, and not a purely bibliographic ontology like <http://bibliontology.com/>, was the fact that both the data from digital libraries (or catalogues) and from digital museums have to be integrated in SYNAT, together with many other forms of resources.

By convention, class names in CIDOC CRM start with E (e.g. E21_Person), property names start with P (e.g. P25_moved). Inverse properties, if present, are marked with i (e.g. P25i_moved_by). If in any of the examples in this paper another letter appears after the number (e.g. E55b_Education_Level), it means that the class or property is a subclass or subproperty of a CIDOC class introduced for the purposes of the described knowledge base (in this case: E55_Type).

4.2 Erlangen CRM / OWL

The Erlangen CRM / OWL [11] is an OWL-DL 1.0 implementation of the CIDOC Conceptual Reference Model. A OWL2 (RL) version is planned soon.

It has been originally created at the Friedrich-Alexander-University of Erlangen-Nuremberg in cooperation with the Department of Museum Informatics at the Germanisches Nationalmuseum Nuremberg and the Department of Biodiversity Informatics at the Zoologisches Forschungsmuseum Alexander Koenig Bonn. It is currently maintained by Martin Scholz, Georg Hohmann and Mark Fichtner.

The Erlangen CRM is an interpretation of the CIDOC CRM in a logical framework attempting to be as close as possible to the text of the CIDOC specification. However, there are some CIDOC classes that are not present in the Erlangen implementation. These are mostly classes representing primitive types (e.g. E60_Number, E62_String). They are supposed to be replaced in OWL with typed (XSD) literals [12].

4.3 Other Ontologies and Vocabularies

Other ontologies and vocabularies used in the knowledge base are:

- the Geonames ontology (<http://www.geonames.org/ontology>),
- ISO 639 language codes (<http://downlode.org/rdf/iso-639/schema#>),
- Dublin Core Metadata Initiative Type Vocabulary (<http://dublincore.org/2010/10/11/dctype.rdf#>),
- WGS84 Geo Positioning (World Geodetic System, http://www.w3.org/2003/01/geo/wgs84_pos).
- OpenVocab (<http://open.vocab.org/>)

The Geonames ontology is used to represent place types (e.g. city, lake). Language codes are used to standardize language descriptions. The DCMI Type Vocabulary is a basis to describe types of works (books, images, audio files). WGS84 is used to hold information about geographic coordinates, which can be used to disambiguate locations (e.g. if there is a number of places with the same name it can be checked which one is closest to another place mentioned in the description) and also to perform advanced queries, as the RDF repository we use offers Geo-spatial Extensions. The method of connecting Erlangen CRM / OWL with WGS84 is presented in Fig. 1.

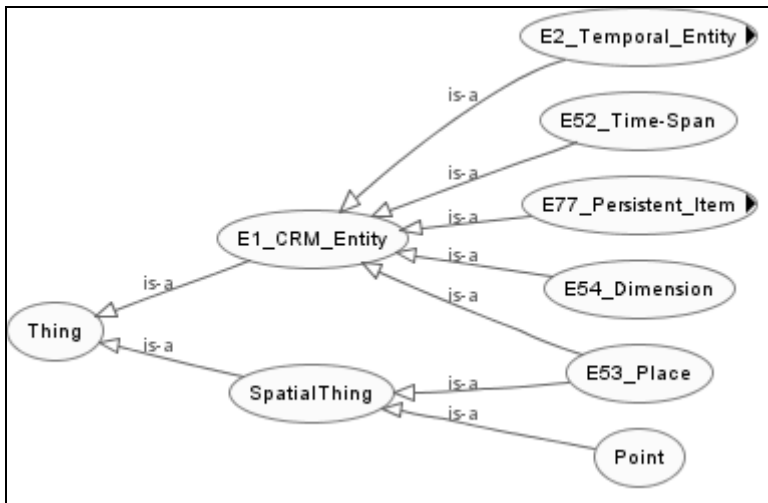


Fig. 1. Combining Erlangen CRM / OWL with WGS84

5 Knowledge Base Construction Steps

At the current stage of the prototype development, the CIDOC-conformant knowledge base is constructed from the DLF metadata (600,000 records mapped by the DLF to the PLMET schema). Other sources of data will be added in future. Knowledge is stored in an RDF repository and can be queried using RDF query languages such as SPARQL.

Fig. 2. illustrates the components participating in the process of knowledge base construction. The functional components are marked as circles and ellipses, the rectangles are data sources and databases. The dashed components are planned, but not yet implemented or connected as part of the prototype. The processing steps necessary to transform the Digital Libraries Federation’s metadata into CIDOC CRM and create a truly semantic knowledge base are described below.

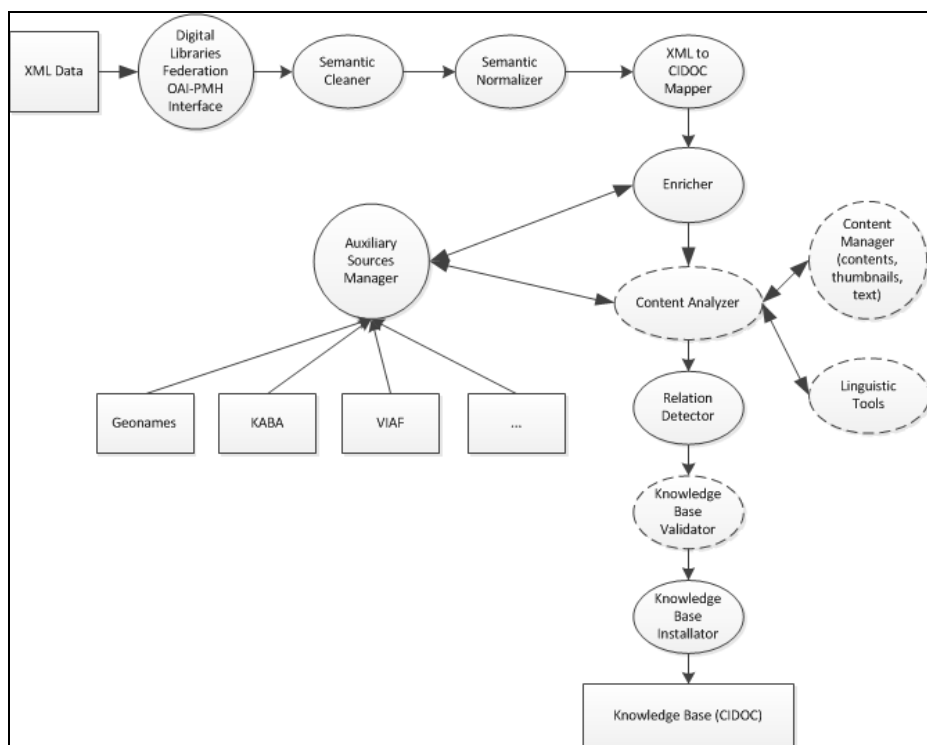


Fig. 2. Components participating in the knowledge base construction process

The processing steps are as follows:

1. The incoming metadata,¹ still in the flat XML source format, is semantically cleaned. Cleaning is done on the record level. As a result some of the record's elements may be deleted (if they contain irrelevant information, e.g. "no date" entry in the "publishing date" element), moved (if they have been assigned incorrectly by cataloguers), or copied to another element, if their meaning is wider.
2. The next step is semantic normalization. Normalization is done on the level of one element. Text elements representing entities such as dates or people are transformed into structures with established format (e.g. all dates are converted to structure representing range YYYY-MM-DD – YYYY-MM-DD).
3. The cleaned and normalized metadata are mapped to a so called *intermediate* CIDOC representation. The data is in CIDOC, but it has not yet been enriched or internally related. Because advanced reasoning has been removed from this stage, the mapping can be performed by one of the existing mapping tools, such as AnnoCultor (<http://annocultor.eu/>).

¹ Depending on the current state of the system and the knowledge base, the input constitutes of either the complete set of metadata from all sources, or just those records that have recently been added or updated.

4. The next stage is data enrichment. External sources, such as Geonames or VIAF, are called to see if they possess information about the entities referred to in the metadata. If so, two operations are performed:
 - a) the entity is linked to the external resource by a URI,
 - b) useful additional information (e.g. alternative names, geographic coordinates) is copied to the knowledge base.
5. The operations in step 4. are performed on information contained in the metadata. Similar operations should be performed on the contents of the resource (and also on longer natural language texts in some metadata elements) based on a number of linguistic tools. This step has not yet been implemented.
6. During the relation detection step, related resources (e.g. different editions of the same book) are connected to each other with CIDOC relations. This step may reduce the number of triples in the RDF repository, because some resources are discovered to represent the same entity. Because of this, up to this point the reasoning engine is turned off, as when it is turned on it is very costly to remove triples (the knowledge base contents have to be recalculated).
7. After the relations are detected, the reasoner is turned on and any possible contradictions are detected. Finding and fixing the reason for a contradiction is an advanced step that has not been implemented in the prototype.
8. Finally, the knowledge base is installed. In production environment it will be deployed on a cluster of nodes. The installer will be responsible for ensuring that during the installation process users of the system have access to at least one functional node.

6 Main Challenges in the Mapping Process

This section names some of the most difficult aspects of the transformation from the DLF metadata schema to the chosen ontology.

6.1 Conceptual Separation of Information Object and Information Carrier

One of the first challenges that need to be faced when transforming a flat digital libraries metadata schema to CIDOC is the conceptual separation between two classes: `E73_Information_Object` and `E84_Information_Carrier`. To quote [4]:

[The `E73_Information_Object`] class comprises identifiable immaterial items, such as a poems, jokes, data sets, images, texts (...) that have an objectively recognizable structure and are documented as single units. An `E73_Information_Object` does not depend on a specific physical carrier.

The `E84_Information_Carrier`, on the other hand:

(...) comprises all instances (...) that are explicitly designed to act as persistent physical carriers for instances of `E73_Information_Object`.

At first it might seem difficult to decide which metadata elements relate to the `E73` and which to the `E84`. However, careful analysis inevitably reveals that this kind of

separation is highly desired in the digital libraries world. Very often in the catalogs one can find information that a resource is *in the DjVu format, measures "15x10 cm", and was written by Jan Kowalski*. Obviously, the DjVu file has no spatial dimension, the original book is printed on paper and has nothing to do with the DjVu format, and the author created a composition that is independent from the carrier.

To conclude, what we have here is one **E73_Information_Object** (the work itself) and two **E84_Information_Carriers** (physical and digital). A straightforward consequence is that one **E73_Information_Object** can have a number of **E84_Information_Carriers**.

Fig. 3. presents the ideal outcome of transforming a number of metadata records describing related copies from digital libraries and catalogues.

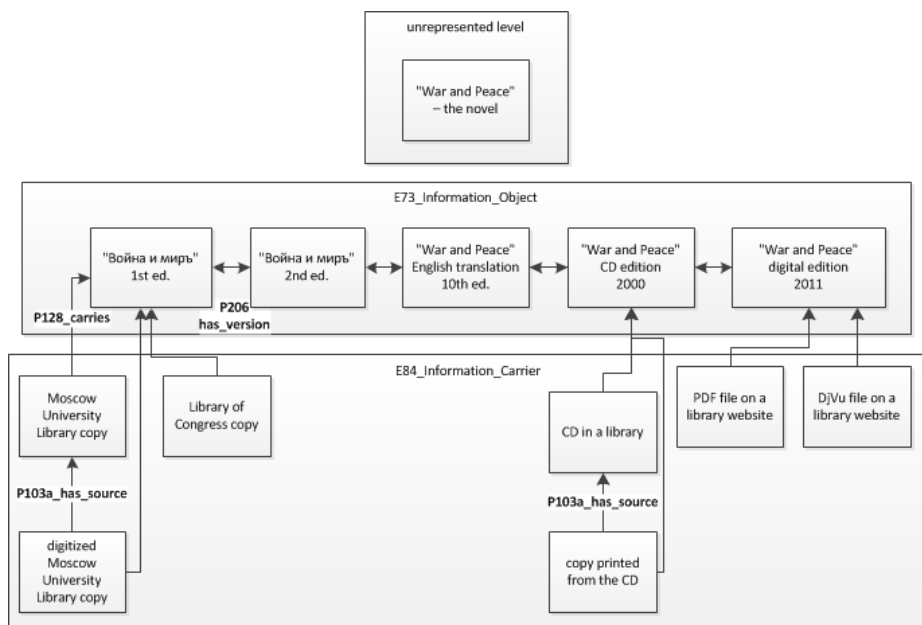


Fig. 3. Representation of different editions of the same work and different copies of the same editions in the knowledge base, based on CIDOC CRM

6.2 E55 Type Hierarchies

CIDOC's **E55_Type** class is an interface to domain specific ontologies and thesauri. External ontologies and classifications are represented as subtypes of **E55** (built-in subclasses include **E56_Language**, **E57_Material** and **E58_Measurement_Unit**). The instances of each subclass represent concepts (and not individuals!), and are connected by means of the **P127_has_broader_term** property.

The following external hierarchies (presented also in Fig. 4) are used in the described translation:

- E55a_Degree (to describe the degree associated with a thesis),
- E55b_Education_Level (as in the EU's Bologna declaration),
- E55c_Research_Discipline (the official hierarchy of Polish research disciplines),
- E55d_Resource_Type (resource types based on the DCMI type vocabulary),
- E55e_Subject (divided into E55_Subject_Hierarchy and E55f_User_Subject),
- E55g_Subject_Hierarchy (the resource subject hierarchy, based on KABA, a Polish thesaurus of subject headings, based on Library of Congress Subject Headings),
- E55f_User_Subject (subjects that are not from KABA and are not recognized as people, places or other entities)
- E55h_Place_Type (based on the Geonames classes).

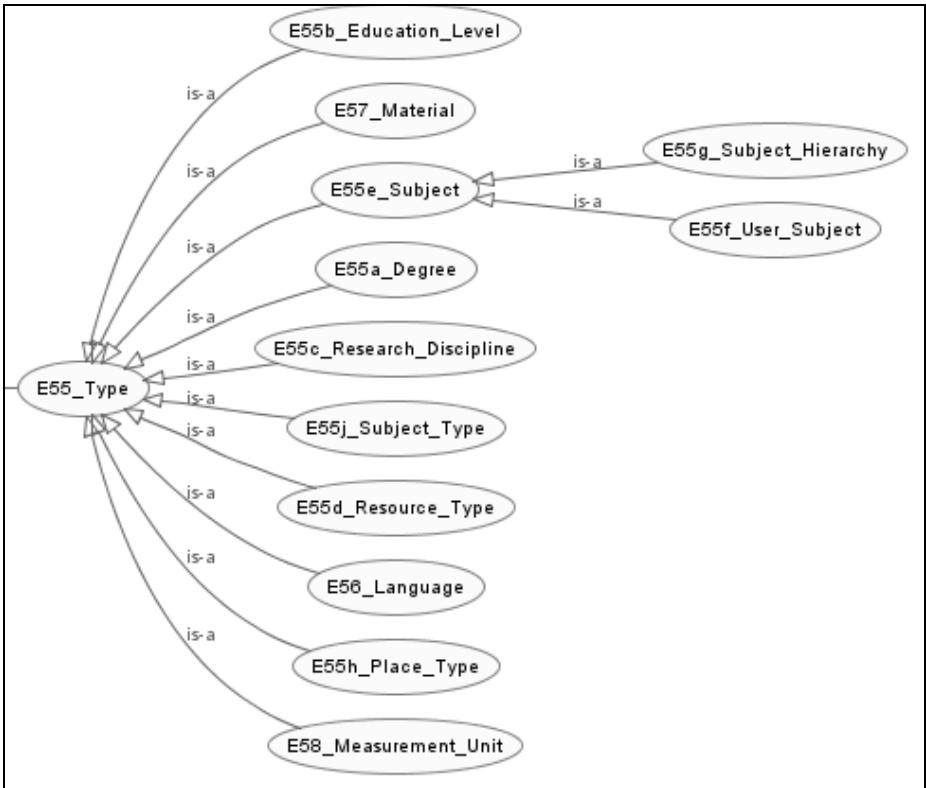


Fig. 4. The E55_Type hierarchies in the knowledge base

6.3 Identities and URIs

One of the goals of translating a flat schema to an ontology is to detect and show resource relations. One of the difficulties is that the related resources are often described in natural language. The text from the elements of a metadata record (the author, related publications, publication place etc.) itself represents knowledge base entities. These entities have to be recognized, created, enriched with external information, and merged with existing entities if they are already present in the knowledge base (e.g. other books of the same author are present).

6.4 Data Enrichment

To allow for more efficient searching and to offer users more valuable information, and also to facilitate disambiguation, the information obtained from the metadata records is enriched with data from auxiliary sources. The functional components of the Knowledge Retrieval System are shown in 5.

The auxiliary sources are heterogeneous and have to be accessed and processed in dedicated ways. Listed below are the auxiliary sources used in the prototype implementation together with their descriptions. With many of the sources the main difficulty is that there is more than one result with the given name, so some reasoning is necessary to choose the most probable option.

Geonames is a geographical database. It contains over 10 million geographical names and consists of 7.5 million unique features of 2.8 million populated places and 5.5 million alternate names. All features are categorized into one out of nine feature classes and further subcategorized into one out of 645 feature codes.

It can be accessed as a web service, but it is also possible to download the data and use it locally. The place descriptions contain information about name, alternate names, country, place type, population, elevation, time zone and geographic coordinates.

TERYT is the National Official Register of Territorial Division of Poland. It contains identifiers and names of units of territorial division, identifiers and names of localities, statistical regions and census enumeration areas, identification of addresses of streets, real estates, buildings and dwellings.

Most towns present in TERYT are also described in Geonames. However, Geonames do not offer information about territorial division, so it is useful to combine data from these sources. However, it is not a straightforward task to provide a one-to-one mapping between places in TERYT and in Geonames.

VIAF (Virtual International Authority File) is an international authority file. It is a joint project of several national libraries and operated by the Online Computer Library Center (OCLC). It gathers data from a number of national institutions and offers (via a web service) information about people and institutions.

NUKAT is one of the institutions participating in VIAF. It is the union catalogue of Polish scientific and academic libraries. It also maintains an authority file (also accessible through VIAF) and the KABA **subject headings system** described below. As NUKAT is a partner in the SYNAT project, we are able to access their resources directly.

6.5 KABA

KABA is used differently than other auxiliary sources in the way that it is completely incorporated into the knowledge base to allow reasoning.

In order to describe publications (*E73_Information_Objects*) with precise subjects we decided to incorporate the KABA Subject Headings into our knowledge base. KABA is a Polish subject indexing system built and maintained by NUKAT. Subject headings systems are used for cataloging of library collections by subjects. KABA is based on three such systems [18]: Library of Congress Subject Headings (LCSH), National Library of France Subject Headings (RAMEAU) and University of Laval in Quebec Subject Headings (RVM).

KABA consists of almost 900,000 records represented in the MARC21 authority format [19]. The MARC21 elements most significant from the Integrated Knowledge System's point of view include the heading (the 1XX fields in MARC21 format), alternative versions of the heading (4XX) and its relations with other records (5XX). The relations include: broader term, narrower term (these two can be regarded as hierarchical relations), earlier form of heading and later form of heading. Eleven types of records can be distinguished, such as: geographical name, person name, or corporate name. Some potentially significant information in record definitions had to be ignored as it is formulated in natural language. An example of it is field 360 with natural language explanation of complex relationships between records.

The idea was to convert KABA Subject Headings into a thesaurus compatible with CIDOC. A new subclass of *E55_Type* (see 6.2) has been introduced to represent this structure: *E55g_SubjectHierarchy*. Hierarchical relations were mapped as the *P127_has_broader_term* and *P127i_has_narrower_term* CIDOC relations. Other relations were mapped using the OpenVocab's (<http://open.vocab.org/>) *similarTo* symmetric property.

The first obtained version of the thesaurus lacked many hierarchical relations which are not explicitly defined in records, but can be inferred from the grammar of subject headings. Rules for inferring such relations have been proposed in [5]. We implemented all four rules described in the paper. The major rule is based on the fact that a subject heading may contain determiners (*subdivisions*) that bring more detail to the topic. A broader term can be built from a subject heading containing such subdivisions by removing some of them. For instance, the broader term *Malarstwo (Painting)* can be inferred from *Malarstwo -- techniki (Painting -- techniques)*. This rule has produced over 200,000 new relations in the knowledge base (KABA explicitly defines about 230,000 relations). The reasoning engine also induced a number of relations (implicit RDF triples) calculating the closure of symmetric and inverse relations.

Still, there are some concerns about the quality of the resulting thesaurus. KABA has been created manually and as such contains errors in record definitions: typographical errors, duplicates and specification incompatibilities. A similar experience of converting LCSH into a thesaurus revealed mistakes in hierarchy relation definitions which lead to apparently wrong conclusions, such as "*a doorbell is a mammal*" [22]. Such mistakes can result in structural inconsistencies in the hierarchy of subjects (i.e. cycles). The resulting thesaurus has been examined with the

Tarjan algorithm² to find all cycles. An example of such cycle is *Inżynieria Ropy Naftowej (Petroleum engineering)* and *Ropa naftowa -- produkcja i handel (Petroleum --industry and trade)* declared as broader terms of each other. 270 such cycles were discovered. The majority consist of two elements. In some cases a record is defined to be a broader term of itself.

The actual number of wrongly defined relations is unknown. Cycles form a special case that can be detected automatically, but the majority of errors and logical inconsistencies can be found only by human verification.

7 DLF Metadata Schema Mapping Description

This section describes the most important points and issues of the transformation, also addressing the non-standard way in which digital library editors use the metadata elements.

One of the most crucial decisions that have to be made while mapping a DLF metadata record is deciding which elements describe the E73_Information_Object and which the E84_Information_Carrier, and whether there is one E84 or two (i.e. the physical copy of the book and the digitized version).

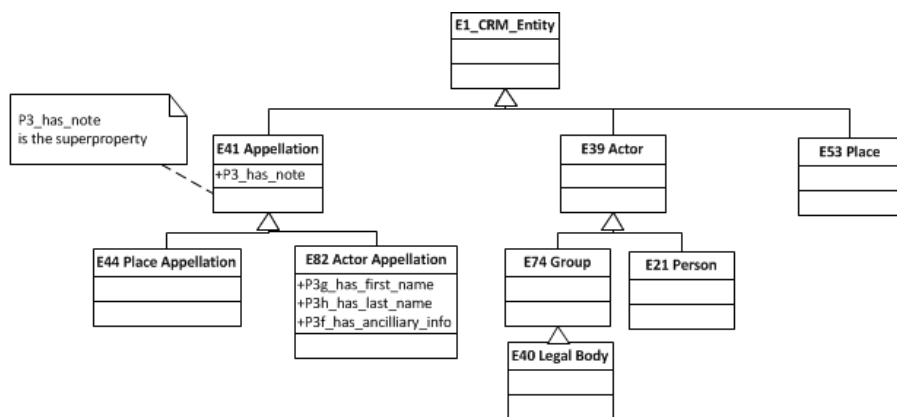


Fig. 5. The hierarchy of entities and their corresponding appellations

It is also important to realize that CIDOC treats names (*appellations*) in a special way. A person does not have a name and surname: a person has an appellation, or even a number of appellations of which one may be declared as preferred. Details and examples are illustrated in Fig. 5 and Fig 6. This is significantly different than what can be found in flat metadata schemas, but is fully justified. People, places and works of art may have different names in different languages, domains or historical periods.

² The Tarjan algorithm finds strongly connected components (such components contain at least one cycle) [23].

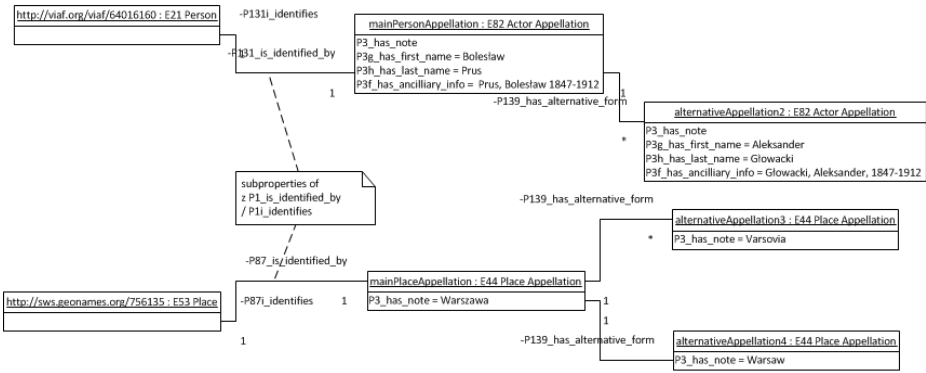


Fig. 6. Example of entities and their appellations

The names themselves (the appellations) may have a complicated structure, and grouping those structures in different appellation instances facilitates understanding and reasoning.

7.1 Titles

The *dc:title* is the title of the work, i.e. the `E73_Information_Object`. It can be easily mapped to CIDOC's `E35_Title` (both an `E33_Linguistic_Object` and `E41_Appellation`³) associated with the `E73` by means of the `P102_has_title` property. `P139_has_alternative_form` allows for connecting the *dcterms:alternative* title to the main title.

7.2 Creators

Both *dc:creator* and *dc:contributor* are instances of `E39_Actor`. Creator performs the `E65_Creation` event of the `E73_Information_Object`. An `E73` comes into existence through an `E65_Creation`, while an `E84_Information_Carrier` through an `E12_Production`.

The contents of the creator and contributor elements often consist of more than just the name string, for instance:

Kowalski, Jan (1950- ; historyk) Red.

Often, the name is followed by parentheses in which dates and other additional (e.g. domain of activity, especially when there are more people with the same name) information is given. Information about role in the creation process is usually given after the parentheses. This information is used in the mapping process to connect the actor instances not only with their respective appellations, but also with the `E67_Birth` and `E69_Death` events. The creator is connected to the creation event with the `P14i_performed` property. For the contributor, different roles (editor, advisor, ...) are proposed, represented by subproperties of `P14i`.

³ The OWL specification allows multiple inheritance.

7.3 Subjects and Coverage

The *dc:subject* is one of the trickiest elements, as a resource can be *about* anything (the CIDOC specification agrees, defining the range of the *P129_is_about* property as the most general *E1_CRM_Entity*). Fortunately, the majority of Polish librarians are accustomed to using the KABA subject headings. However, it is possible that, for instance, a resource is about another one. A possible option is to use the WordNet ([9],[6]) thesaurus to represent all possible concepts.

The current implementation tries to map the subjects to KABA (see 6.4) or resources of types found in VIAF (people, institutions) or Geonames (places). If it fails, a new user subject (see 6.2) is created.

One of the challenges addressed in the SYNAT project is automatic subject detection based on the contents of the resource.

The *plmet:userTag* is represented with a subproperty of *P129* with the assumption that *tags* are even more informal than user-created subjects. The tags in this metadata element come from the readers, i.e. the users of a digital library, and not from librarians or cataloguers (as is the case for the subjects forming the *E55e_Subject* hierarchies, see 6.2).

The *dc:coverage* and its subelements *dcterms:spatial* and *dcterms:temporal* are also connected to what the work is about, only they cover more specified types (time and place). It is important to remember that the coverage describes the subject of the resource and not aspects of its availability (e.g. when/where copies of particular book were distributed). Therefore in the mapping process coverage is related to the *E73_Information_Object* instance by means of the CIDOC *P67_refers_to* property. In case of spatial coverage the range (i.e. the object of the triple representing the relation) is an *E53_Place*, in case of temporal property it is *E52_Time-Span*.

In case of old publications the place instance may represent a historical place, significantly different from the modern one, although having the same name (e.g. compare borders of Poland before and after the Second World War).

7.4 Descriptions

The *dc:description*, *dcterms:abstract* and *dcterms:tableOfContents*, all related to the *E73*, constitute of plain text and at this point it would be difficult to propose a better mapping than just copying the contents to subproperties of CIDOC's datatype property⁴ *P3_has_note*. In some cases a more meaningful table of contents might be built based on different kinds of relations (*dc:relation* and its subelements, especially *dcterms:hasPart*). Future works allowing to achieve more meaningful mapping should include deep natural language processing.

7.5 Publishing

The *plmet:placeOfPublishing* element names the *E53_Place* in which the resource (this time this is the physical copy, i.e. the *E84_Information_Carrier*) was published.

⁴ A datatype property in OWL is a property whose range (object) is a literal and not a resource identified with a URI.

The publishing is an event represented by a subclass of `E12_Production` (as stated before, in CIDOC the carriers are produced and the information objects are created).

Another of the DLF's proprietary elements, *plmet:digitisationSponsor*, names the institution that paid for the digitization process (a subclass of `E12_Production`). The product of the digitization is a new `E84_Information_Carrier`, based on another `E84` instance representing the physical resource. The publications whose original form is digital are not subject to digitization. In their case the only `E84` instance represents the digital copy.

The *dc:publisher* is associated with the same publishing event as *plmet:placeOfPublishing*. The `E39_Actor` (usually an institution, i.e. `E40_Legal_Body`) is connected to the event by means of the `P14i_performed` property.

7.6 ETD-MS Elements

The translation of the ETDMS thesis elements is quite straightforward. It is based on the `E55_Type` hierarchies described in 6.2. The degree grantor is an institution, represented in CIDOC as an instance of `E40_Legal_Body`.

7.7 Dates

It is advised in the PLMET specification that the *dc:date* element be left empty, and the subelements be filled instead. Otherwise it is difficult to determine what the date represents. As CIDOC is an event-based ontology, the respective dates are not mapped to static values, but are instead translated to events (some of which are built-in CIDOC events, like `E65_Creation`, and some are new subclasses of `E7_Activity`). If *dc:date* is the only given date, depending on the type of the resource it is assumed to be either a creation (e.g. manuscripts) or publishing (e.g. old prints) date.

7.8 Types

The mapping of the *dc:type* to CIDOC and its consequences are described in [13].

Most resources in Polish digital libraries are periodicals and books. Periodicals have to be treated in a distinct manner. Typical library catalogues contain one record per periodical title, while in digital libraries each issue of the title is described. Often a periodical issue described in digital library metadata has two titles. One of them is the title of the periodical as a whole (e.g. *Kurier Warszawski*), the other contains the particular issue information (e.g. *Kurjer Warszawski / [red. L. A. Dmuszewski]. 1827, nr 121*). In the proposed CIDOC representation a new subtype of `E73` has been created to represent a periodical title, to which all issues are related (with `P148_has_component`⁵). Sometimes advanced parsing of the whole record has to

⁵ For some time we considered an alternative solution in which the periodical was not an instance of `E73_Information_Object`, but a type of `E78_Collection`. This solution proved to be inapplicable because the `E78_Collection` as specified in CIDOC aggregates physical resources.

be used to detect the part representing the title of the whole periodical, as librarians not always adhere to standards or create their personal standards.

7.9 Formats

The *dc:format* and its subelements (*dcterms:extent*, *dcterms:medium*) describe the dimensions and medium of the resource. It is important to realize that those are physical features (P45_consists_of and P43_has_dimension) of the E84_Information Carrier, and not of E73_Information Object (not to be confused with *dc:coverage*).

A unit should be associated with a dimension (P91_has_unit). In case of book copies there might be centimeters or the number of pages, in case of recordings these might be minutes.

7.10 Identifiers

The *dc:identifier* element should be a URI which can also be the URI of the relevant individual in the RDF repository. Often the resource has a number of URIs, e.g. different ones in its owner's and an aggregator's databases. The *plmet:callNumber* and *dcterms:bibliographicCitation* are also ids. One id should be set as preferred one – we propose the URI from the data provider. The *bibliographicCitation* is a subject of controversy. For one, large parts of it could potentially be generated automatically from the resource metadata (e.g. a paper from a conference volume). Another problem is that there is a number of different citation formats, although PLMET recommends to use PN-ISO 690 standard here.

7.11 Language

Representing the *dc:language* is straightforward, with one remark: the domain⁶ of the P72_has_language property is E33_Linguistic_Object, and neither the information object nor carrier are linguistic objects. The E33_Linguistic_Object has to be connected with the E73 by means of the P148_has_component property before the language property is set.

It is worth noting that E35_Title also is a subclass of E33. If the digital library metadata contains information about title language (for instance expressed as `xml:lang`) the information can be introduced to the knowledge base.

Languages in the knowledge base are described using one of the E55_Type hierarchies (see 6.2.).

7.12 Relations between Resources

The *dc:source*, *dc:relation* and its subelements represent relations between resources. New properties (and their inverse counterparts) had to be added to the ontology to represent some relations, some (e.g. P67_refers_to) were already present.

⁶ In OWL domain constrains the types that can be the subject of a triple in which the given property is the predicate.

The *dcterms:conformsTo* relation turned out to be controversial. It seems to be of a different kind than other relations - it relates a resource to a standard or norm, and not to another resource. This problem was solved with the decision to regard the standard as a new *E73_Information_Object*. At some point the actual text of the standard might be introduced into the repository as a regular resource.

During the mapping and knowledge retrieval process described in 5, a module called the Relation Detector is responsible for connecting related resources. The relations are most often described in plain text. The Mapper, if it is to be kept simple and possibly based on existing tools, should not be responsible for this task.

7.13 Rights

CIDOC is prepared for modeling rights information (license type, rights owner) with the *E30_Rights* class and the properties *P104_is_subject_to* and *P105_rights_held_by*. Additional information about rights may be extracted from Europeana elements, provided they are available (see 7.15).

7.14 Provenance

The *dcterms:provenance* element contains *a statement of any changes in ownership and custody of the resource since its creation that are significant for its authenticity, integrity, and interpretation*. There are two proprietary DLF subelements *plmet:locationOfPhysicalObject* and *plmet:digitization*. The digitization describes the person or organization who performed the digitization (which, as stated earlier, is a subclass of *E12_Production*). The location of physical object is either a place, in which case the *P55_has_current_location* property is used, or an institution, in which case we use *P50_has_current_keeper*.

If the provenance element contains a natural language description, at this point it is copied as-is.

7.15 Europeana Elements

The Digital Libraries Federations is one of the providers of data to Europeana [21], an aggregator portal that groups and displays information about digital resources of European museums, libraries, archives and audio-visual collections. Europeana is in the process of switching to a linked data schema, but for now the official data exchange schema is ESE, which is an extension (a profile) of the Dublin Core Metadata Terms schema.

The ESE proprietary tags are designed for automated processing and therefore quite strictly specified, so obtaining their contents from the DLF adds value to the proposed mapping. The kind of information that can be extracted from these tags and introduced into the knowledge base is summarized below. It is important to note that the Europeana tags concern mostly the *E84_Information_Carrier*:

- *ese:object* – this is the URI of a thumbnail of the resource. In the knowledge base the thumbnail is a subclass of the *E84* class.
- *ese:provider* – names the source from which Europeana obtained data (this might be an aggregator service).

- *ese:provider* – this is the original source, i.e. the keeper of data, mapped with the P50_has_current_keeper property.
- *ese:type* – the type of the resource, as in E55d_Resource_Type (Europeana’s list of allowed types is more limited). This, for consistency reasons, it mapped to the E73_Information_Object.
- *ese:rights* – valuable right information: the URI to a document describing the license. Mapped with P104_is_subject_to.
- *ese:isShownBy* – URI of a high quality presentation of the described resource (E84). This kind of information is useful to knowledge base clients.
- *ese:isShownAt* - URI of a presentation of the described resource in its full information context (E84).
- *ese:unstored* – this is additional information that cannot be otherwise assigned to the available tags. The best thing we can do with it is save it as-is with hope of applying linguistics tools in future.

8 Results and Statistics

The prototype implementation of the Knowledge Retrieval System is functional and has been tested on a large sample of the DLF metadata. The tests have been performed on a desktop computer with the following characteristics: Intel Core 2 Quad CPU, Q9650 3.00GHz, Windows 7 64b, 4GB RAM. In future the system is supposed to work on a cluster of dedicated servers with a significantly higher amount of RAM, so we expect shorter times even with larger amounts of data.

The RDF repository used to hold the knowledge base is Ontotext’s BigOWLIM (<http://www.ontotext.com/owlim/editions>) to which we have been granted a research license. BigOWLIM is both an RDF/OWL repository and a reasoning engine, one of the few that are able to perform the reasoning using their persistence mechanisms and not fully in the limited RAM memory, which is a critical condition if the reasoning engine is to be used in a system with millions or billions of triples.

This section presents statistics concerning the mapping and reasoning process, the contents of the resulting knowledge base, and some information about sample queries.

8.1 Knowledge Base Construction Process

Table 1 presents the times of the knowledge base creation with different amounts of starting DLF data. A *record* is a description of one publication in the DLF metadata schema. The first row of the table (*0 records*) contains data about the preparation processes in which all the ontologies and the KABA hierarchy are loaded. The results of this process are constant, hence it does not need to repeated every time the knowledge base is created. This is why the initialization time is not counted as part of the *processing time* in the subsequent rows.

The knowledge base creation process is described in section 5. The process is comprised of data cleaning and normalization, mapping from DLF metadata schema to CIDOC, relation detection, data enrichment, and performing reasoning (i.e. computing the full closure) on the triples.

Explicit triples are triples that have been introduced to the repository as a result of the mapping, enrichment, and relation detection operations. Implicit triples are triples generated by the reasoner (using BigOWLIM's owl-horst-optimized rule set, [2]). The total number of triples in the repository is the sum of explicit and implicit triples.

There are two different numbers of implicit triples in the 500,000 records case. The reason for this is that after close inspection of the KABA data in the repository we discovered that the `P127_has_broader_term` relation used to build `E55_Type` hierarchies was not declared transitive in Erlangen CRM. After our request this has been corrected in a newer Erlangen version, hence the higher number of triples (now all parts of the *broader term* chain are connected with implicit triples, there also has been a number of other similar changes).

Table 1. DLF records processing times and the number of resulting RDF triples

No. of Records	Processing Time	Explicit RDF triples	Implicit RDF triples
0	2.5 hours	6,059,665	30,710,518
10,000	7 minutes	6,443,246	32,907,960
50,000	30 minutes	7,928,790	40,849,942
100,000	70 minutes	9,808,073	51,177,846
500,000	6 hours	23,040,700	125,212,980
			181,209,531 ⁷

We find the processing times on the limited hardware quite satisfactory. However, we have detected two performance problems with BigOWLIM (or, possibly, reasoning engines in general). One is the unacceptable reasoning time for long chains of OWL properties that are both transitive and symmetric (especially when long chains of transitive properties with inverse properties work fine). The other is the fact that if we populate the repository with no reasoning at all (the `empty` rule set) and turn the reasoning on later, the processing time is significantly longer than with instant reasoning (e.g. 13 minutes for 10,000 records, when instant reasoning together with the knowledge base construction takes only 7 minutes).

This is an important issue for us, because after mapping the XML schema to Erlangen CRM we want to perform some operations on the repository with the reasoning turned off. If the reasoning is turned on, the triple deletion operation becomes very costly, as the full closure has to be recomputed and all triples inferred from the deleted one have to be deleted as well. Some triples have to be deleted in the relation detection process, e.g. when two different instances are discovered to in fact represent the same entity. At this point it seems that it would be more efficient to perform the operations with the reasoning turned off, serialize the triples, and load them with reasoning turned on than to simply turn reasoning on in a working repository.

⁷ This is the result for the newest version (2011-04-04) of Erlangen CRM in which on our request the `P127_has_broader_term` property has been declared transitive (which is in accordance with the CIDOC CRM specification).

8.2 Knowledge Base Contents

In accordance to the Semantic Web and Linked Data guidelines, in the knowledge base we are trying to reuse existing ontologies and vocabularies and connect our data (by means of URIs) to external recognized sources. In the prototype implementation we are enriching our data with information from VIAF (Virtual International Authority File), NUKAT (Polish National Union Catalogue) authority data, Geonames (a worldwide geographical database), TERYT (Central Statistical Office's administrative division register). We also try to map subjects to KABA, which is a subject headings language (and dictionary) based on the LCSH (Library of Congress Subject Headings).

We use the external sources not only to identify the entities in the knowledge base, but also to add new information that can be useful to its clients. For example, when we obtain location data from Geonames, we are able to unify all references to this location with different names (e.g. names in different languages), we download the coordinates information to allow for geographic proximity searching, and we remember all the alternative names, so that a query for the place using any one of them returns expected results.

Table 2. Contents of the knowledge base – links to external resources

No. of Records	VIAF Persons	NUKAT Persons	Other Persons	VIAF Institutions	NUKAT Institutions	Other Institutions	Geonames Places	TERYT Places	Other Plcs	KABA Subjects
10,000	518	37	860	139	10	304	422	10	3	805
50,000	1869	107	2,833	406	29	968	1.277	47	3	1,966
100,000	5,016	204	6,625	760	70	2.385	4.230	277	6	4,378
500,000	25,467	4,862	62,673	2,825	242	15.396	8.901	608	77	16,353

Table 2 shows the number of entities recognized in external data sources. A reason for optimism is that most people, places, institutions and subjects are repeated in the repository, which allows us to connect similar resources to each other. An interesting issue is the fact that there are some entities (people and institutions) that have been found in NUKAT and have not been found in VIAF which contains also the NUKAT information. One reason for this is that the copy of NUKAT data we received is newer than what the VIAF web service offers access to. Beside there are other two possible explanations: the web service API returns less good results than our local Lucene index query, or our local query should be more restrictive and exclude some results.

Locations are searched for both in Geonames and in TERYT, but Geonames have higher priority, as the service contains more information. The places that have been found in TERYT and not in Geonames are very small Polish villages.

Table 3 presents information about publications subjects. The *Total Publ.* column shows the number of distinct *E73_Information_Object* instances whose subject is of the given type (place, KABA subject, user subject). The two final columns show the

Table 3. Publication subjects distribution

Subject Type	Total Publ.	Subject Instances	Avg. Publ./Subj.
Place	249,129	8,742	28.5
KABA Subject	265,320	16,353	16.2
User Subject	328,389	75,844	4.3

number of used subject instances for each subject type and the average number of publications on one subject instance.

The number of KABA records used as a subject (Table 3) and the number of mapped KABA records (Table 2) are equal, because KABA records are only used as subjects. The number of mapped places and the number of places used as subjects are not equal, because places may play different roles in the knowledge base (they might for instance represent the place of publication).

8.3 Knowledge Base Queries

The knowledge base is to be used by the Integrated Knowledge System portal to allow for semantic (and more efficient) search of resources. The queries can be asked in SPARQL or SeRQL languages. Table 4 contains information about sample SeRQL queries to the knowledge base generated by processing 500,000 DLF records (over 200 million triples: 23 million explicit and 181 million inferred). The response times (which contain data retrieval times) are not fully satisfactory for a production system, but, as stated before, the impressive repository was held and queried on a desktop machine.

Table 4. Sample query times

Query	Time	Cached Time ⁸	No. of Results
All places with their names	15s	199ms	9663
Titles of books about Warsaw	771ms	7ms	1273
Works of a person with surname "Mozart"	17ms	3ms	221
All places whose names start with "Nowy"	886ms	24ms	27

As an example below is the SeRQL listing to perform the last query (*all places whose names start with "Nowy"*):

```
SELECT geold FROM
  {s} luc:luceneLiteralIndex {"nowy*"},
  {geold} rdf:type {cidoc:E53_Place} ; cidoc:P1_is_identified_by {}
cidoc:P3_has_note {s}
USING NAMESPACE
  cidoc = <http://erlangen-crm.org/current/>,
  luc = <http://www.ontotext.com/owlim/lucene#>
```

⁸ There are several components in OWLIM that make use of caching (e.g. full-text search indices, predicate list, tuple indices). When a query is asked for the first time some parts of the result are cached, so similar queries asked afterwards are completed in shorter times.

9 Conclusions and Future Work

The automatic conversion of 500,000⁹ metadata records described by the DLF schema (PLMET) to a Semantic Web ontology based on CIDOC CRM proved possible (within reasonable time) and brings promising results. It opens new possibilities for search and discovery of resources (e.g. displaying information about a person and all related resources, and not only textual fields for a given publication record), that will be used by the Integrated Knowledge System Portal. The knowledge base is enriched by additional information coming from external sources and contains links to other pieces of information that are not explicitly needed by the knowledge base or the portal, but can be obtained by users navigating in the Semantic Web environment. As the resources are identified by unique URIs, it is also easy for external services to link to the knowledge base.

There is still a large number of problems that should be solved to make the knowledge base truly useful. One problem is that external sources (e.g. Geonames) often return more than one possible result for a given string. At this point we use heuristics (e.g.: the most populated, but preferably a town rather than a region, etc.) but more context information should be included in the reasoning process (as it is possible that somebody is in fact referring Warsaw, Ohio, and not Warsaw, the capital of Poland).

Another issue is that of historical names of places. If a book from 1920 is about *Poland*, it may refer to towns that are now parts of another country. Works dedicated to the creation and management of a historical names database fall beyond the scope of the SYNAT project. However, the existence of such a database would be useful, so a separate project is being considered to handle this task.

An important (and challenging) position in our task list is the inclusion of linguistic tools (Fig. 2). One of the partners in the SYNAT project is Wrocław University of Technology, the creator of the Polish WordNet (plWordNet) [6] which would allow for further enrichment of resources and resource navigation possibilities.

Also, a very interesting task is the design of a convenient user interface to introduce data to the IKS portal. We want users to unambiguously describe the published resources (from their personal digital libraries) by means of an ontology (the extended CIDOC ontology). Instead of a very long metadata form, the users should be offered a dynamic “wizard” to introduce the data corresponding to the type of the resource being published. Linking to other resources (both internal and external to the IKS system) should be facilitated with innovative UI solutions.

Finally, the Digital Libraries Federation data is only the first step (although an important one) in acquiring and understanding cultural heritage information from heterogeneous sources. It is an inspiring challenge to create a knowledge base of interconnected information of different types: digital libraries, museums, archives, catalogues, etc.

⁹ There are now more than 600,000 publications in DLF, but the tests have been performed on data obtained in mid-February 2011.

References

1. Atkins, A., Fox, E., France, R., Suleman, H.: ETD-MS: an Interoperability Metadata Standard for Electronic Theses and Dissertations, 1.2 edn. (2008), <http://www.ndltd.org/standards/metadata/etd-ms-v1.00-rev2.html>
2. Bishop, B., Kiryakov, A., Ognyanoff, D., Peikov, I., Tashev, Z., Velkov, R.: OWLIM: A family of scalable semantic repositories. In: *Semantic Web – Interoperability, Usability, Applicability* (2010), <http://www.semantic-web-journal.net>
3. Clayphan, R. (ed.): *Europeana Semantic Elements Specification, Version 3.3.1*, January 24 (2011), https://version1.europeana.eu/c/document_library/get_file?uuid=a830cb84-9e71-41d6-9ca3-cc36415d16f8&groupId=10602
4. Crofts, N., Doerr, M., Gill, T., Stead, S., Stiff, M.: *Definition of the CIDOC Conceptual Reference Model, 5.0.2 edn.* (June 2005), http://www.cidoc-crm.org/docs/cidoc_crm_version_5.0.2.pdf
5. Daćko, D., Józefowska, J., Ławrynowicz, A.: An ontology based semantic library catalogue. In: *Proceeding of the 3rd Language & Technology Conference*, Poznań, pp. 109–113 (2007)
6. Derwojedowa, M., Piasecki, M., Szpakowicz, S., Zawisławska, M.: Polish WordNet on a Shoestring. In: *Proceedings of Biannual Conference of the Society for Computational Linguistics and Language Technology*, Universität Tübingen, Tübingen, April 11–13, pp. 169–178 (2007)
7. Doerr, M.: *Mapping of the Dublin Core Metadata Element Set to the CIDOC CRM. Technical Report, 274*, ICS-FORTH, Heraklion, Crete (July 2000), http://www.cidoc-crm.org/docs/dc_to_crm_mapping.pdf
8. Dudczak, A., Heliński, M., Mazurek, C., Mielnicki, M., Werla, M.: Extending the Shibboleth Identity Management Model with a Networked User Profile. In: *Proceedings of the 1st International Conference on Information Technology*, Gdańsk, May 18–21, pp. 179–182 (2008)
9. Fellbaum, C.: *WordNet. An Electronic Lexical Database*. MIT Press (1998)
10. Gill, T.: Building semantic bridges between museums, libraries and archives: The CIDOC Conceptual Reference Model. *First Monday* 9(5) (2004), <http://firstmonday.org>
11. Görz, G., Oischinger, M., Schiemann, B.: An Implementation of the CIDOC Conceptual Reference Model (4.2.4) in OWL-DL. In: *Proceedings of CIDOC 2008 — The Digital Curation of Cultural Heritage*. ICOM CIDOC, Athens (2008)
12. Hohmann, G., Scholz, M.: Recommendation for the representation of the primitive value classes of the CRM as data types in RDF/OWL implementations, <http://erlangen-crm.org/docs/crm-values-as-owl-datatypes.pdf>
13. Kakali, K., Doerr, M., Papatheodorou, C., Stasinopoulou, T.: *Wp5 - task 5.5 dc.type mapping to cidoc/crm*. Technical report. Department of Archives and Library Science. Ionian University (2007)
14. Koutsomitropoulos, D.A., Solomou, G.D., Papatheodorou, T.s.: Metadata and Semantics in Digital Object Collections: A Case-Study on CIDOC-CRM and Dublin Core and a Prototype Implementation. *Journal of Digital Information* 10(6) (2009), <http://journals.tdl.org/jodi/>

15. Lewandowska, A., Mazurek, C., Werla, M.: Enrichment of European Digital Resources by Federating Regional Digital Libraries in Poland. In: Christensen-Dalsgaard, B., Castelli, D., Ammitzbøll Jurik, B., Lippincott, J. (eds.) ECDL 2008. LNCS, vol. 5173, pp. 256–259. Springer, Heidelberg (2008)
16. Lourdi, I., Papatheodorou, C., Doerr, M.: Semantic Integration of Collection Description. Combining CIDOC/CRM and Dublin Core Collections Application Profile. *D-Lib Magazine* 15(7/8) (2009), <http://www.dlib.org/>
17. Mazurek, C., Stroiński, M., Węglarz, J., Werla, M.: Metadata harvesting in regional digital libraries in PIONIER Network. *Campus-Wide Information Systems* 23(4), 241–253 (2006)
18. NUKAT, the National Union Catalog, <http://www.nukat.edu.pl/>
19. Paluszkiewicz, A.: Format rekordu kartoteki haseł wzorcowych: zastosowanie w Centralnej Kartotece Haseł Wzorcowych NUKAT. In: *Formaty, Kartoteki, SBP*, Warszawa, vol. 17 (2009)
20. Pasi, M., Motta, E.: Ontological Requirements for Annotation and Navigation of Philosophical Resources. In: *Synthese*, September 28, pp. 1–33. Springer, Netherlands (2009)
21. Purday, J.: Think culture: Europeana.eu from concept to construction. *The Electronic Library* 27(6), 919–937 (2009)
22. Spero, S.: LCSH is to Thesaurus as Doorbell is to Mammal: Visualizing Structural Problems in the Library of Congress Subject Headings. In: *Proceedings of the 2008 International Conference on Dublin Core and Metadata Applications (DCMI 2008)*, Dublin Core Metadata Initiative, pp. 203–203 (2008)
23. Tarjan, R.E.: Depth-First Search and Linear Graph Algorithms. *SIAM Journal on Computing* 1(2), 146–160 (1972)

Modularized Knowledge Bases Using Contexts, Conglomerates and a Query Language*

Krzysztof Goczyła, Aleksander Waloszek, Wojciech Waloszek, and Teresa Zawadzka

Gdansk University of Technology, Poland
{kris,alwal,wowal,tegra}@eti.pg.gda.pl

Abstract. The paper presents a novel approach to design and development of a modularized knowledge base. It is assumed that the knowledge base consists of a terminology (axioms) and a world description (assertions), both formulated in a Description Logics (DL) dialect. The approach is oriented towards decomposition of a knowledge base into logical components called contexts and further into semantic components called conglomerates. Both notions were elaborated separately elsewhere. The paper shows how contexts and conglomerates concepts can work in harmony to create a maintainable knowledge base. An architecture of a system that conforms to this approach, which additionally uses a query language called KQL (Knowledge Query Language), is presented. The approach is intended to be used to build a prototypical system that aims at integrating knowledge on cultural heritage coming from digital libraries, including user-defined libraries. The thorough discussion of related work is also given.

Keywords: ontology, knowledge base, modularization, contexts, conglomerates, knowledge integration.

1 Introduction

In this paper we present a novel approach to design and implementation of large and scalable knowledge bases that conform to Description Logics [1] dialects expressiveness (according to W3C standards: that conform to OWL2 [2] with SWRL [3] extensions). This approach is based on three major concepts: contexts, conglomerates and a query language. Contexts are to logically situate a knowledge based in different frameworks so that logical statements are interpreted in a proper way, according to a given “situation”. Conglomerates are to divide a complex, large knowledge base into components, like a relational database is divided into relations. Conglomerates can be operated upon by algebraic operations, like relations in a relational database. The operations can be performed via Knowledge Query Language

* This work is partially supported by the Polish National Centre for Research and Development (NCBiR) under Grant No. SP/I/1/77065/10 by the strategic scientific research and experimental development program: „Interdisciplinary System for Interactive Scientific and Scientific-Technical Information”.

(KQL) statements that structurally remind that of SQL statements, which make them more easily understood by systems engineers.

The paper is organized as follows. In the Section 2 we present the notion of contexts. In Section 3 we present the notion of conglomerates. In Section 4 the KQL language is presented. In Section 5 the architecture of the system that incorporates the three components is presented. The system is to be implemented as a part of the SyNat system intended to make the cultural and science Polish heritage available worldwide. Section 6 presents the related work. In Section 7 we conclude the paper.

2 Contexts

Contextualizing knowledge bases makes the process of their deployment and use more effective. Contextualized knowledge bases, if contextualization is properly conducted, are easier to design and maintain. Moreover, with use of contextualization, the efficiency of reasoning process can be greatly improved. In this section we describe our approach to modularization, *Structured-Interpretation Model* (SIM), an approach that has several distinguishing features, the most important of which is the relative straightforwardness of adapting the structure of knowledge the base to evolving needs of the user.

As a basis we have taken Description Logic formalism and we adapted to it the notions of generalization and context instance. Let us recall that a DL ontology consists of a terminology (TBox) and a world description (ABox). The Formal definition of contextualized DL ontology is as follows:

Definition 2.1. A contextualized TBox $T = (\{T_i\}_{i \in I}, \preceq)$ consists of a set of TBoxes whose elements are called contexts, and a generalization relation $\preceq \subseteq I \times I$ which is a partial order established over the set of indexes I . The relation is acyclic and contains the only least element m . We also introduce the following notions:

T_m called the root context of the contextualized TBox T ,

T_i generalizes T_j iff $i \preceq j$,

T_i specializes T_j iff $j \preceq i$. □

The idea behind introducing such hierarchical arrangement of contexts was to allow for constrained interactions between parts of terminology. The general rule here is that more specialized terminologies may “see” more general ones, but more general terminologies may be unaware of the existence of more specialized ones.

Introduced contexts encompass only terminology. To deal with assertional part of the knowledge base we allow for creation of many ABoxes for one terminology. We call these ABoxes *context instances*.

Definition 2.2. A contextualized ABox $A = (\{A_j\}_{j \in J}, \text{inst}, \ll)$ of contextualized TBox $T = (\{T_i\}_{i \in I}, \preceq)$ is a triple consisting of:

1. A set of ABoxes $\{A_j\}_{j \in J}$, each of which is called an instance of context,
2. The function $\text{inst}: J \rightarrow I$ relating each ABox from $\{A_j\}_{j \in J}$ with TBox from $\{T_i\}_{i \in I}$,
3. The aggregation relation $\ll \subseteq J \times J$, which is a partial order established over the set of indexes J . We require that:

- a. \ll is acyclic and contains the least element n ,
- b. $\text{inst}(n) = m$, where m is the least element of the relation \preceq ,
- c. for each $j \ll k$ such that $j \neq k$ holds $\text{inst}(j) \preceq \text{inst}(k)$ and $\text{inst}(j) \neq \text{inst}(k)$.

We also say that:

- A_n is called the root context instance of the contextualized ABox A ,
- A_j is an instance of the context T_i iff $\text{inst}(j) = i$,
- A_j aggregates A_k iff $j \ll k$,
- A_j is aggregated by A_k iff $k \ll j$,
- A_j is an aggregating context instance iff $\exists k: j \ll k$ □

Definition 2.3. A contextualized knowledge base $K = (T, A)$ consists of a contextualized TBox T and a contextualized ABox A of T . □

Contextualized knowledge base is given the interpretation in a specific way: each context instance is given its own interpretation. Such an approach gives some level of locality within context instances.

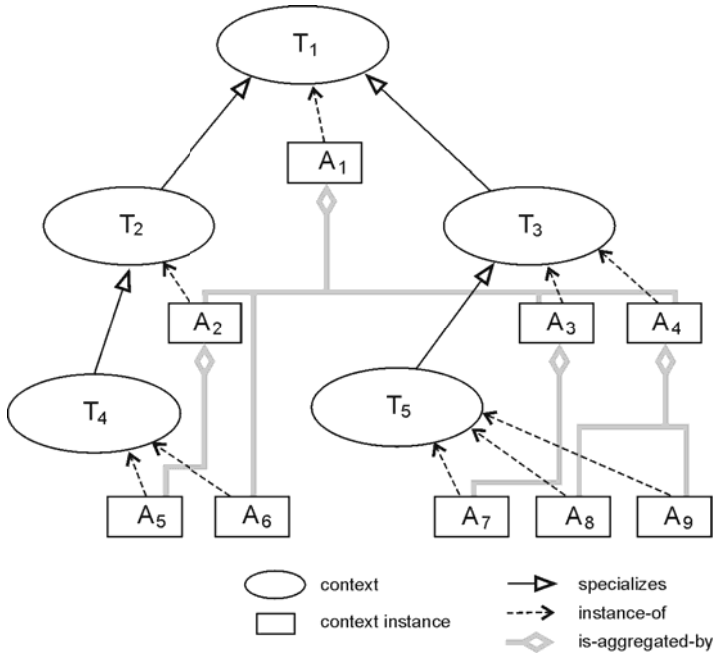


Fig. 1. An example of contextualized knowledge base

The general schema of a contextualized DL-ontology (a DL-knowledge base) is presented in Fig. 1. In this figure relationships between context instances and between context instances and contexts are depicted in the form of graph, e.g. the instance A_4 aggregates instances A_8 and A_9 . For the sake of clarity only the transitive reductions of generalization and aggregation relations have been depicted.

The example of the aggregation conformance of denotation is presented in Fig. 2. Here we have three contexts: T_1 that describes general notions of WOMAN and MAN, T_2 that specializes T_1 towards description of voices in a choir, and T_3 that also specializes T_1 but towards description of social relations. Context instance A_1 aggregates context instances A_2 and A_3 . Although ABox of A_1 is empty, interpretation \mathcal{I}_1 in order to be a model of the knowledge base has to assign Mary to the concept WOMAN (i.e. $Mary^{\mathcal{I}_1} \in WOMAN^{\mathcal{I}_1}$). As a consequence of this, the same rule enforces that in the interpretation \mathcal{I}_3 Mary is assigned to the concept WIFE (i.e. $Mary^{\mathcal{I}_3} \in WIFE^{\mathcal{I}_3}$), as the information about Mary being a woman “flows” down the aggregation relationships.

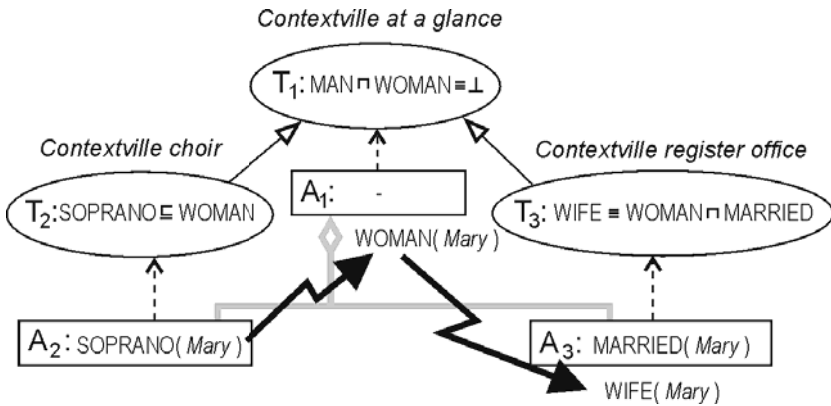


Fig. 2. An example of the aggregation conformance of denotation

Other aspects of contexts and contextualized knowledge bases can be found in [4].

3 Conglomerates

Conglomerates are used to modularize large knowledge bases into manageable pieces like relations in a relational database. The main motivation behind conglomerates is the hope to reach the maturity of the collaborative ontology development and reuse comparable to the one achieved by software engineering methods in the case of software modules.

In the following we introduce basic notions of conglomerates algebra and present examples of its use.

3.1 Basics

An assumption we take is that a user is interested in conclusions that can be drawn from an ontology rather than in particular axioms and assertions. So, we define an ontology module strictly semantically, focusing only on its interpretations.

Definition 3.1. (conglomerate). A conglomerate $M = (S, W)$ is a pair of a signature S (called a *signature* of the conglomerate) and a class W of S -interpretations (called *models* of the conglomerate). The two parts of M are denoted as $S(M)$ and $W(M)$, respectively. Each S -interpretation from W we call a *model* of M . \square

According to this definition, each conglomerate simply consists of all its models. We say that a conglomerate satisfies a particular sentence α , denoted $M \models \alpha$, iff $\forall \mathcal{I} \in W(M): \mathcal{I} \models \alpha$.

We think that creators of an ontology are not able to foresee all its future uses. By necessity, the creators have to focus on a small set of chosen contexts of use of particular modules, and the contexts of their choice may not be adequate for a particular application of a knowledge base with this ontology. So, the conglomerate algebra puts stress on various methods of manipulation for them; that in general allow for changing and combining signature and models.

3.2 Operators

We assume that description logic \mathcal{L} and domain set Δ are chosen and fixed. We denote a set of all modules as M , a set of all signatures as Σ , and a set of all S -interpretations as $\mathcal{I}(S)$. We also use the notion of a *projection* $\mathcal{I}' = \mathcal{I}|S'$ of an S -interpretation \mathcal{I} to a signature S' ($S' \subseteq S$). \mathcal{I}' is an S' -interpretation for which the following holds: $\Delta^{\mathcal{I}'} = \Delta^{\mathcal{I}}$ and $X^{\mathcal{I}'} = X^{\mathcal{I}}$ for every $X \in S'$.

Extend

$$\begin{aligned} \varepsilon_S: M &\rightarrow M, S \in \Sigma; \\ \varepsilon_S(M) &= (S(M) \cup S, \{\mathcal{I} \in \mathcal{I}(S(M) \cup S): \mathcal{I}|S(M) \in W(M)\}). \end{aligned} \quad \square$$

Extension extends a signature of a given module M by names from a given signature S . The allowed set of interpretations of each original name is preserved, and so are the relationships between original concepts, roles, and individuals (e.g. if $M \models \alpha$, $\alpha \in \mathcal{L}(S(M))$, then also $\varepsilon_S(M) \models \alpha$).

Project

$$\begin{aligned} \pi_S: M &\rightarrow M, S \in \Sigma, S \subseteq S(M); \\ \pi_S(M) &= (S, \{\mathcal{I}|S: \mathcal{I} \in W(M)\}). \end{aligned} \quad \square$$

Projection reduces a signature of a given module. However, relationships between original concepts, roles, and individuals whose names remain in the signature are preserved (e.g. if $M \models \alpha$, $\alpha \in \mathcal{L}(S)$, then also $\pi_S(M) \models \alpha$).

Rename

$$\begin{aligned} \rho_\gamma: M &\rightarrow M, \gamma \text{ is a signature mapping;} \\ \rho_\gamma(M) &= (\gamma(S), \gamma(W)). \end{aligned} \quad \square$$

Renaming uses the notion of a *signature mapping*. Signature mapping γ is a triple: $(\gamma_C, \gamma_R, \gamma_I)$, each of them being a bijection from \mathcal{N} to \mathcal{N} . By $\gamma(S)$ we mean $\gamma_C(C(S)) \uplus \gamma_R(R(S)) \uplus \gamma_I(I(S))$, and by $\gamma(\mathcal{I})$, where \mathcal{I} is an S-interpretation, we mean an $\gamma(S)$ -interpretation \mathcal{I}' such that $\Delta^{\mathcal{I}'} = \Delta^{\mathcal{I}}$ and $\gamma(X)^{\mathcal{I}'} = X^{\mathcal{I}}$ for every $X \in S$. Rename preserves relationships between concepts, roles, and individuals, however with respect to their name changes (e.g. if $M \models \alpha$, $\alpha \in \mathcal{L}(S(M))$, then $\rho_\gamma(M) \models \gamma(\alpha)$, where $\gamma(\alpha)$ is α transformed in such a way that all names in α have been systematically changed according to γ).

Select

$$\sigma_\alpha: M \rightarrow M, \alpha \in \mathcal{L}(S(M));$$

$$\sigma_\alpha(M) = (S, \{\mathcal{I} \in \mathcal{W}(M): \mathcal{I} \models \alpha\}). \quad \square$$

Selection leaves only these interpretations that are models of a sentence α . Obviously $\sigma_\alpha(M) \models \alpha$.

Union

$$\cup: M \times M \rightarrow M, S(M_1) = S(M_2);$$

$$M_1 \cup M_2 = (S(M_1), \mathcal{W}(M_1) \cup \mathcal{W}(M_2)). \quad \square$$

Union performs a set-theoretic union of sets of models of conglomerates. The condition that $S(M_1) = S(M_2)$ is not very restrictive because we can easily upgrade this operation to a *generalized union* $\cup_g: M_1 \cup_g M_2 = \varepsilon_{S(M_2)}(M_1) \cup \varepsilon_{S(M_1)}(M_2)$.

Intersection

$$\cap: M \times M \rightarrow M, S(M_1) = S(M_2);$$

$$M_1 \cap M_2 = (S(M_1), \mathcal{W}(M_1) \cap \mathcal{W}(M_2)). \quad \square$$

Difference

$$-: M \times M \rightarrow M, S(M_1) = S(M_2);$$

$$M_1 - M_2 = (S(M_1), \mathcal{W}(M_1) - \mathcal{W}(M_2)). \quad \square$$

Intersection and difference are analogous to the union, and can be generalized in the similar way to the case when signatures of the operands differ.

Union and difference are non-linguistic (strictly semantic), i.e. their use may lead to generation of a conglomerate M for which there does not exist any corresponding set of sentences S in \mathcal{L} . This issue will be elaborated on later in the paper.

I-Join (intersecting join)

$$\times: M \times M \rightarrow M;$$

$$M_1 \times M_2 = (S', \mathcal{W}');$$

$$S' = \gamma_1(S(M_1)) \cup \gamma_2(S(M_2));$$

$$\mathcal{W}' = \{\mathcal{I} \in \mathcal{I}(S'): \mathcal{I}\gamma_1(S(M_1)) \in \gamma_1(\mathcal{W}(M_1)) \wedge$$

$$\mathcal{I}\gamma_2(S(M_2)) \in \gamma_2(\mathcal{W}(M_2))\}. \quad \square$$

I-Join is an operation on two conglomerates. It uses two signature mappings γ_1, γ_2 , each of them preceding every terminological name in a signature with a unique prefix. Thus, I-Join helps to solve potential naming conflict between conglomerates. I-Join preserves relationships between original concepts and roles in both modules (if $M_1 \models \alpha, \alpha \in \mathcal{L}(S(M_1))$, then $M' \models \gamma_1(\alpha)$, analogically for M_2).

I-Join is a non-primitive operation, as $M_1 \times M_2$ can be expressed as $\rho_{\gamma_1}(M_1) \cap_g \rho_{\gamma_2}(M_2)$. This derivation justifies the name “intersecting join” as we may perceive this join as a “safe” way of intersecting modules.

U-Join (union join)

$$\begin{aligned}
 \bowtie: M \times M &\rightarrow M; \\
 M_1 \bowtie M_2 &= (S', W'); \\
 S' &= \gamma_1(S(M_1)) \cup \gamma_2(S(M_2)); \\
 W' &= \{I \in \mathcal{I}(S') : \mathcal{I}\gamma_1(S(M_1)) \in \gamma_1(W(M_1)) \vee \\
 &\quad \mathcal{I}\gamma_2(S(M_2)) \in \gamma_2(W(M_2))\}. \quad \square
 \end{aligned}$$

U-Join is a counterpart of I-Join. γ_1, γ_2 have the same meaning as above. U-Join offers the “safe” way of performing union. Naturally, $M_1 \bowtie M_2 = \rho_{\gamma_1}(M_1) \cup_g \rho_{\gamma_2}(M_2)$.

Put-Under

$$\begin{aligned}
 \nu_C: M \times M &\rightarrow M, C \in \mathcal{L}_C(S(M_2)); \\
 M_1 \nu_C M_2 &= (S', W'); \\
 S' &= S(M_1) \cup S(M_2); \\
 W' &= \{I \in \mathcal{I}(S') : \mathcal{I}S(M_2) \in W(M_2) \wedge \\
 &\quad (\mathcal{I}S(M_1) \cap C^{\mathcal{I}}) \in W(M_1)\}. \quad \square
 \end{aligned}$$

Put-Under correlates the domains of two conglomerates. We use here a *restriction* of an S-interpretation \mathcal{I} : by $\mathcal{I} \cap \Delta'$ we mean an interpretation $\mathcal{I}' = (\Delta', \cdot^{\mathcal{I}'})$ such that $X^{\mathcal{I}'} = X^{\mathcal{I}} \cap \Delta'$ for every $X \in S$. As each conglomerate induces a set of laws that enforce certain relationships between concepts, roles, and individuals, then Put-Under can be perceived as a restriction of the scope of these laws to a fragment of a larger domain.

3.3 Examples

Example 3.1. (intersecting conglomerates). Consider two conglomerates: M_1 describes human resources and M_2 the structure of a hospital.

$$M_1 = M(\{\exists isManagerIn.HTBusinessUnit \sqsubseteq Expert, \\
 Expert \sqsubseteq Employee\})$$

$$M_2 = M(\{leadsDepartment(johnSmith, neurosurgery), Department(neurosurgery)\}).$$

To merge the information from the two conglomerates in order to infer that *johnSmith* is an expert, we first create an intersection of the conglomerates: $M' = M_1 \cap_g M_2$, and then restrict the set of models by introducing additional “bridge” axioms: $M'' = M' \cap_g$

$M(\{leadsDepartment \sqsubseteq isManagerIn, Department \sqsubseteq HTBusinessUnit\})$. The last step can also be done by double selection. \square

In the example we did not encounter any name conflict between conglomerates being merged. In general, such a conflict may occur and I-Join operator should be used. Below we show how to align two conglomerates in which the same set of terms is used to express different meanings.

Example 3.2. (joining conglomerates). Consider two conglomerates: M_1 and M_2 . They contain assessment of several rooms for rent, and use the same categorization and signature $S = \{HSRoom, ASRoom, LSRoom\}$, where the concepts denote high, average and low standard rooms. But in M_1 and M_2 different criteria were used for categorization, as in the first case we were looking for a room to spend just one day and in the second case to stay for a longer period of time. We “import” the assessment from M_1 to M_2 performing necessary translation of classification between the conglomerates.

1. In the first step we simply I-Join the conglomerates (modules). As a result we obtain a conglomerate $M' = M_1 \times M_2$. The concepts have been renamed, so $S(M) = \{1:HSRoom, 2:HSRoom, 1:ASRoom, \dots\}$.
2. Next, we make the criteria of assessment explicit. In this example we use only one criterion: a bathroom. So, we extend the signature of M' appropriately: $M'' = \varepsilon_{\{RoomWithBathroom\}}(M')$.
3. Afterwards, we bind the criteria with the assessment. In M_1 rooms with bathrooms were automatically considered high standard. According to the criteria used in M_2 no room with bathroom can be considered more than low standard.

$$M''' = M'' \cap_g M(\{RoomWithBathroom \sqsubseteq 1:HSRoom, \\ \neg RoomWithBathroom \sqsubseteq 2:LSRoom\}).$$

Naturally, the second axiom is valid only if the domain consists of only rooms (which is assumed).

4. Finally we remove unwanted terms from the module signature:

$$M = \pi_{\{2:HSRoom, 2:ASRoom, 2:LSRoom\}}(M''').$$

In these steps all the translations possible to perform were done. All average or low standard rooms from M_1 were considered low standard in accordance with criteria from M_2 . \square

Some other aspects of using conglomerates can be found in [5].

4 Knowledge Query Language

The basis of our work on a query language for a DL knowledge base is our opinion that the development of a universal language for accessing knowledge bases must be based on assumptions similar to those adopted and practically proven in the case of SQL – beside the others, theoretical mathematical basis, language closure and availability of commands to create, manipulate and control the knowledge. Thus, we have based the KQL language on theoretical backgrounds presented in Sections 2 and 3, that is on contexts and conglomerates.

4.1 Basic Assumptions for KQL

The main statement for manipulating a conglomerate (called in KQL a conglomeration) is the SELECT statement of the following structure:

```
SELECT concept_list, role_list, attributes_list
      ADD axiom_list
      FROM conglomeration_expression
      WHERE concept_expression
      HAVING concept_expression
```

With respect to the conglomerates algebra, the `SELECT` clause corresponds to the projection—the choice of terms for a newly created conglomerate, the `ADD` clause corresponds to the selection—the contraction of the set of allowed interpretations, the `WHERE` and `HAVING` clauses correspond to the projection with respect to individual names. The difference between the `WHERE` and `HAVING` clauses lies in the fact that the `WHERE` clause selects those individuals that belong to a specified concept of the original conglomerate (as defined in the `FROM` clause), while the `HAVING` clause selects those individuals that belong to a specified concept of the target conglomerate. The query is conceptually executed in the following steps: (1) Determining the basic conglomerate on the basis of the `FROM` clause. (2) Reducing the individual names on the basis of the `WHERE` clause. (3) Extending the alphabet with new concepts / roles / attributes / individuals that occur in the statements contained in the `ADD` clause. (4) "Adding" (i.e. extending the ontology) to the conglomerate the statements from the `ADD` clause. (5) Projection of the alphabet only to the concepts / roles / attributes contained in the `SELECT` clause. (6) Reducing the individuals names basing on the `HAVING` clause.

A direct consequence of KQL closure is ability of query nesting. Every time a conglomerate is needed, one can use named conglomerates defined in knowledge base or conglomerates created on the fly by KQL expressions with the use of conglomerates algebra operators (such as `INTERSECT`, `JOIN`, `UJOIN`) or the `SELECT` clause. As a consequence of KQL closure and the ability of query nesting, the uniformity of modification and definition language has been achieved: one can use query nesting also in conglomerate creation statements:

```
CREATE CONGLOMERATION conglomeration_name
...
FROM conglomeration_expression
```

In KQL it is assumed that terminological queries are issued against a metaontology or a meta conglomerate. A metaontology is nothing but a single conglomerate knowledge base that stores information about conglomerates and rules. A meta conglomerate is also a single conglomerate knowledge base that stores information about exactly one conglomerate.

As a consequence of using the metaontology it is possible to issue terminological queries in a similar way how the assertional queries are issued; the only difference is that a query is directed to the metaontology rather than to the knowledge base itself. This follows from the fact that conglomerates, concepts, roles, attributes, and individuals from a knowledge base are reified to individuals in the metaontology and the relationships between them are reified to the corresponding binary relationships between individuals.

4.2 Examples of KQL Usage

The examples below have already been defined in terms of the conglomerate algebra.

Example 4.1. (simple import). We consider two conglomerates: M_1 describes human resources, and M_2 describes a structure of a hospital.

```
CREATE CONGLOMERATION M1
    ADD CONCEPT HTBusinessUnit
    ADD CONCEPT Expert
    ADD CONCEPT Employee
    ADD ROLE isManagerIn

CONSTRAIN M1
    EXIST isManagerIn HTBusinessUnit ISSUB Expert
    Expert ISSUB Employee

CREATE CONGLOMERATION M2
    ADD CONCEPT Department
    ADD ROLE leadsDepartment
    ADD INDIVIDUAL johnSmith
    ADD INDIVIDUAL neurosurgery

CONSTRAIN M2
    (johnSmith, neurosurgery) IS leadsDepartment
    neurosurgery IS Department
```

In KQL each conglomerate is created in two steps. In the first step the conglomerate signature is created (`CREATE CONGLOMERATION` statement). In the second step the constraints (sets of statements that must be satisfied) are added (`CONSTRAIN` statement). In the example above, the first conglomerate defines an employee and an expert. We assume that each expert is an employee, and anyone who manages at least one business unit is an expert. The second conglomerate describes John Smith who leads the department of neurosurgery. We want to ask whether `johnSmith` is the expert.

```

SELECT Expert
  ADD leadsDepartment ISSUB isManagerIn
  ADD Department ISSUB HTBusinessUnit
FROM M1 INTERSECT M2
WHERE {johnSmith}

```

To merge the information from the two conglomerates in order to check whether `johnSmith` is an expert we first create an intersection of the conglomerates (`FROM` statement), and then restrict the set of model by introducing additional “bridge” axioms (`ADD` statement). The result of the query is a new conglomerate whose signature consists of two terms: `>` and `Expert`. In our example `johnSmith` is an instance of `Expert` concept. \square

Example 4.2. (different versions and what-if problems) This example illustrates use of union and negation in QQL. Let us consider a conglomerate *M*:

```

CREATE CONGLOMERATION M
  ADD CONCEPT TrustedWitness    ADD CONCEPT CrimeScene
  ADD ROLE murdered
  ADD ROLE accuses
  ADD ROLE presentAt
  ADD INDIVIDUAL victim
CONSTRAIN CONGLOMERATION M
  Top ISSUB <= 1 INV (murdered) {victim}
  EXIST INV(accuses) TrustedWitness
    ISSUB EXIST INV(murdered) {victim}
  TrustedWitness ISSUB EXIST presentAt CrimeScene

```

The *M* conglomerate describes a world that assumes the only one murderer who is accused by a trusted witness (who has been present at the crime scene). We consider two (mutually exclusive) versions of facts (e.g. collected by two investigating agents: John Shady and Henry Brilliant). To achieve that we define two new conglomerates (one for each agent).

```

CREATE CONGLOMERATION M1
FROM(
  SELECT *
  ADD johnShady IS TrustedWitness
  ADD (johnShady, tedInnocent) IS accuses
FROM M
)

```

```

CREATE CONGLOMERATION M2
FROM(
  SELECT *
  ADD henryBrillant IS TrustedWitness

```

```

        ADD (henryBrillant, markGuilty) IS accuses
        FROM M
    )

```

Having such defined conglomerates M_1 i M_2 we would like to analyze different scenarios. Therefore we create a new conglomerate $M_0 = M_1 \text{ UNION } M_2$.

We may assume that `henryBrillant` is not a trusted witness:

```

SELECT Murderer
    ADD NOT TrustedWitness(henryBrillant)
    ADD Murderer EQ EXIST murdered {victim}
    FROM M1 UNION M2

```

We ask for a murderer. Therefore we define `Murderer` as a person who murdered a `victim`. The resulting conglomerate will return exactly one individual (`tedInnocent`) as an instance of `Murderer` concept.

We may assume that `markGuilty` is a murderer

```

SELECT TrustedWitness, NotTrustedWitness
    ADD (markGuilty, victim) IS murdered
    ADD NotTrustedWitness EQ NOT TrustedWitness
    FROM M1 UNION M2

```

In this case we can conclude that `henryBrillant` is a trusted witness (but this does not mean that `johnShady` is not a trusted witness). The conglomerate created within the query defines `NotTrustedWitness` concept (the complement of `TrustedWitness` concept defined in conglomerates M_1 and M_2). The conglomerate which is the result for this query will return a single instance of `TrustedWitness` concept and will return no instance of `NotTrustedWitness` concept.

We may assume that `johnShady` was not present at the crime scene

```

SELECT TrustedWitness, Murderer
    ADD johnShady IS EXIST presentAt CrimeScene
    ADD Murderer EQ EXIST murdered {victim}
    FROM M1 UNION M2

```

In this case we can conclude that `johnShady` is not a trusted witness and `markGuilty` is the murderer. To do this we define `Murderer` concept similarly to the second example. □

More examples of KQL usage can be found in [6].

5 The Architecture

The techniques described above can be composed to create a comprehensive and uniform approach to creating large federated knowledge bases. In such a base every component is treated as a sovereign entity (conglomerate), so that it is possible to exploit its contents in its full variety. While the entities are sovereign, they are not independent, but organized in hierarchical structure of contexts.

The SyNat project gives an opportunity to create such a federated base in a systematic way, and thus to validate the proposed approach. The notions of contexts, conglomerates, and the query language presented above are used in the following knowledge integration system (see Fig. 3). Data (pieces of knowledge) are stored in data (knowledge) sources¹ (at the left-hand side of the figure). They are “pushed” (pre-loaded statically) or “pulled” (loaded dynamically, on demand) into conglomerates (ontology modules) of a knowledge base. Contents of each source are perceived by the rest of the system as autonomous conglomerates.

The knowledge base is managed by a knowledge base management system called RKaSeA, the system developed at Gdansk University of Technology. The system, in its present stage of development, is equipped with the ability of handling the most important algebraic operations (intersection, projection, and selection), with the option of extending the range of operators by use of a specialized subsystem. The specialized subsystem takes advantage of \mathcal{L} -(2)-representation of conglomerates (described in details in [28]; in a nutshell it gives the possibility of representing conglomerates in the form of sets of sentences), which enables use of standard inference engines for selected reasoning tasks. Another subsystem of RKaSeA is Knowledge Layer [29], fully integrated (by use of MCA—Maximum Coverage Algorithm—for reasoning from both internally stored ABoxes and external knowledge) with the reasoning mechanism and allowing for attaching external knowledge sources as separate conglomerates. The external knowledge sources may also be NORs (non-ontological resources).

In the proposed architecture the pieces of knowledge are enriched with additional semantics. This enrichment is done by combining the knowledge from external modules with the selected fragments of integrated and contextualized Ontology of Science (Fig. 3). This name might be a bit misleading as we intend to include in this ontology also a choice of upper-level, commonly shared ontologies, primarily CIDOC [7] CRM. Such organization allows us to conduct an individualized method of integration of knowledge into the contextualized framework for each of the external conglomerates. It is also worth stressing that the system is flexible enough to easily accommodate new knowledge. Ontology of Science may contain “private” entities, added in personalized way by each user, such that use of them may improve the process of integration and tailor it for fulfillment of special needs of individual users and working groups of scientists.

¹ During the first phase of development knowledge sources from the Polish Federation of Digital Libraries (<http://fbc.pionier.net.pl>) will be exploited.

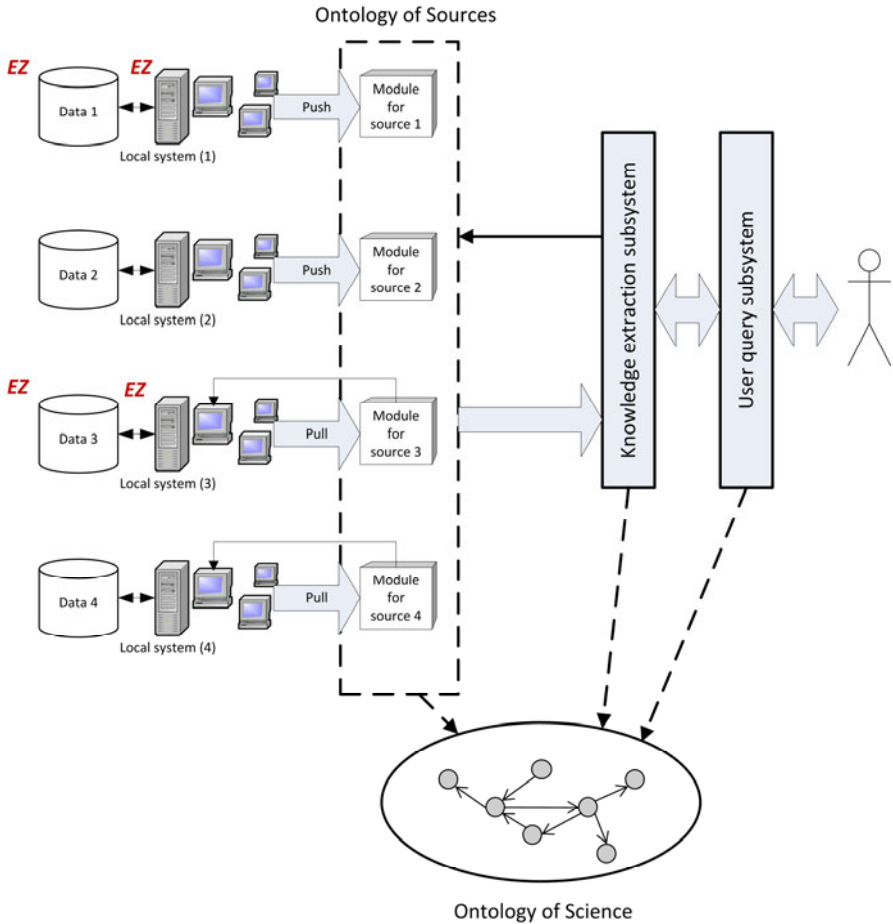


Fig. 3. The architecture of a knowledge integration system with contextualized knowledge bases divided into conglomerates and queried using KQL

Both external conglomerates and modules of Ontology of Science may be accessed by RKaSeA [8], that is able to interpret KQL queries issued by a user. The exact form of queries issued by the user is still subject to discussions. It is assumed though that the User Query Subsystem produces enough information to formulate KQL requests. Since the process of composing a query might be burdensome and difficult, we purport that it might be desirable to use some selected reasoning mechanisms to support it. This kind of reasoning is done over Ontology of Sources which holds meta-knowledge about the structure of the knowledge base.

The proposed general architecture forms a basis for two more detailed scenarios of use of the system. In the first scenario we assume that the contents of external modules is well-known and standardized to conform to selected upper-level

ontologies and standards. In this scenario the primary role of the system focuses on selection of appropriate fragments of knowledge sources and on interpretation of queries issued by the users. In the second scenario we may not assume any initial level of conformance of the external source. In this scenario the system focuses on managing sources (with use of Ontology of Sources), selecting methods and procedures of integration, and maintaining uniform contextualized structure of the collection of managed external and internal conglomerates.

6 Related Work

In this section we present some related work concerning issues connected with integration of knowledge sources. In this section we describe two major approaches to modularization, and we also present some languages aimed at communication with knowledge bases.

We can distinguish between two main approaches in this field: an *inference approach* and an *algebraic approach*. The former addresses the question of how knowledge collected in one source affects logical consequences inferred from another source. The latter proposes algebraic methods to separate needed fragments from given sources and then join them in an appropriate way in order to find a proper answer for a given question.

The inference approach was firstly connected with the research field called *ontology merging* and then embraced issues addressed to the subject of *modularization* or *contextualization* of ontologies.

Modularization (contextualization) of ontologies is recently a domain of intensive research. It was the subject of two large European projects: Knowledge Web (<http://knowledgeweb.semanticweb.org/semanticportal/sewView/frames.html>) and NeOn (<http://www.neon-project.org/>). There exist a lot of motivations to apply a contextual approach to representing knowledge. The most significant are (according to the NeOn deliverable D 3.1.1 [10]):

1. Supporting different viewpoints.
2. Dealing with temporal information.
3. Dealing with inconsistent information.
4. Personalization.
5. Situation awareness in pervasive computing.
6. Scalability.
7. Ontology adaptation and views on ontologies.
8. Matching pairs (groups) of ontologies.

In the field of integration of knowledge from different and heterogeneous sources the results achieved during research on contextualization are very useful. Particularly the work on contextualization of logics is worth of mentioning. The examples of useful formalisms are, among others, *modal logic* ([11]), *the logic of demonstratives* ([12]), and *context logics* ([13]). Obviously, all of them comply to the inference approach. In the environment of Semantic Web the most valuable is work connected with

MultiContext Systems/Local Model Semantics ([14][15]). The main idea of MCS/LMS relies on restricting the set of allowed interpretations of an ontology (called a *target ontology*) by a subset of conclusions drawn from another ontology (called a *source ontology*). Those conclusions are enabled via *bridge rules* which are the kind of inference rules defined particularly for the target ontology. MCS/LMS became a start point for the family of modularization methods that allow to create *distributed knowledge bases* consisting of a number of pairwise linked ontologies.

One of the most recognized methods is *Distributed Description Logic* (DDL) ([16]). In DDL, the idea of MCS/LMS is adopted to DL. In this idea the most important notion is a distributed interpretation—a set of local interpretations for the modules that are independent ontologies. The information is imported from a source module to a target module indirectly through bridge rules and individual correspondences, which are special bridge rules created for individual names. For a source ontology \mathcal{K}_i and a target ontology \mathcal{K}_j the rules have the form:

$$\begin{array}{lll} i:C & \xrightarrow{\sqsubseteq} & j:D & \text{into bridge rule} \\ i:C & \xrightarrow{\supseteq} & j:D & \text{onto bridge rule} \\ i:a & \xrightarrow{\quad} & j:b & \text{individual correspondence} \end{array}$$

where $i:C$ and $i:a$ is respectively a concept or an individual defined in \mathcal{K}_i , and $j:D$ and $j:b$ is respectively a concept or an individual defined in \mathcal{K}_j .

Every module has its own local domain. In order to describe properly the semantics of the rules the relation between the two domains should be defined. The relation is called a domain relation and relates the two domains $r_{ij} \subseteq \Delta_i \times \Delta_j$, where Δ_i is the domain of \mathcal{K}_i , and Δ_j is the domain of \mathcal{K}_j . Usage of the domain relation r_{ij} allows to describe conditions under which a distributed interpretation satisfies a distributed knowledge base. If we denote the distributed interpretation $\mathcal{I} = \{\mathcal{I}_i\}_{i \in I}$ and the distributed knowledge base $\mathcal{K} = \{\mathcal{K}_i\}_{i \in I}$, we can say that \mathcal{I} is a model of \mathcal{K} iff:

$$\begin{array}{l} \mathcal{I}_i \text{ satisfies } \mathcal{K}_i \\ r_{ij}(C^{\mathcal{I}_i}) \subseteq D^{\mathcal{I}_i} \text{ for all } i:C \xrightarrow{\sqsubseteq} j:D \\ r_{ij}(C^{\mathcal{I}_i}) \supseteq D^{\mathcal{I}_i} \text{ for all } i:C \xrightarrow{\supseteq} j:D \\ b^{\mathcal{I}_j} \in r_{ij}(a^{\mathcal{I}_i}) \text{ for all } i:a \xrightarrow{\quad} j:b \end{array}$$

where $C^{\mathcal{I}_i}$ is a subset of Δ_i representing the concept C in the local interpretation \mathcal{I}_i , and $r_{ij}(C^{\mathcal{I}_i})$ is a shortcut for $\{x \mid x \in \Delta_j \wedge \exists y \in C^{\mathcal{I}_i} \wedge (x, y) \in r_{ij}\}$. Analogously, $r_{ij}(a^{\mathcal{I}_i})$ means $\{x \mid x \in \Delta_j \wedge \exists y \in \Delta_i \wedge (x, y) \in r_{ij}\}$.

During reasoning over the target ontology the source ontology and the bridge rules should be taken into account. If, for example, the following bridge rules are defined:

$$\begin{array}{ll} i:A & \xrightarrow{\sqsubseteq} j:G \\ i:B & \xrightarrow{\supseteq} j:H \end{array}$$

and in \mathcal{K}_i the axiom $B \sqsubseteq A$ exists, then, in \mathcal{K}_j , the axiom $H \sqsubseteq G$ holds.

Another technique is \mathcal{E} -Connections ([17]). The main semantic difference comparing to DDL is that domains of modules are disjoint, so a relation between them is not needed. The technique allows to define roles connecting individuals from different domains. These special roles are called *link relations*. All link relations are gathered in the set \mathcal{E} consisting of sets \mathcal{E}_{ij} of relations linking ontologies i and j . \mathcal{E} -Connections also defines the notion of distributed interpretation:

$$\mathcal{M} = \langle (\Delta_i)_{1 \leq i \leq n}, (\bullet^{M_i})_{1 \leq i \leq n}, (\bullet^{M_{ij}})_{1 \leq i, j \leq n} \rangle$$

where Δ_i is the domain of i -th ontology, for $i \neq j$ $\Delta_i \cap \Delta_j = \emptyset$, and interpretation functions maps each concept A of i -th ontology into $A^{M_i} \subseteq \Delta_i$, each role R into $R^{M_i} \subseteq \Delta_i \times \Delta_i$, each individual a into $a^{M_i} \in \Delta_i$, and each link relation $p \in \mathcal{E}_{ij}$ into $p^{M_{ij}} \subseteq \Delta_i \times \Delta_j$.

During creation of an ontology containing link relations it is possible to use special concept constructs $\exists p.Z$, $\forall p.Z$, $\leq np.Z$ or $\geq np.Z$, Z is a concept from j -th ontology, and $n \in \mathbb{N}$. The semantics of such concepts is defined as follows:

$$\begin{aligned} (\exists p.Z)^{M_i} &= \{a \in \Delta_i: \exists b \in \Delta_j ((a, b) \in p^{M_{ij}} \wedge b \in Z^{M_j})\}, \\ (\forall p.Z)^{M_i} &= \{a \in \Delta_i: \forall b \in \Delta_j ((a, b) \in p^{M_{ij}} \rightarrow b \in Z^{M_j})\}, \\ (\leq np.Z)^{M_i} &= \{a \in \Delta_i: |\{b: ((a, b) \in p^{M_{ij}} \wedge b \in Z^{M_j})\}| \leq n\}, \\ (\geq np.Z)^{M_i} &= \{a \in \Delta_i: |\{b: ((a, b) \in p^{M_{ij}} \wedge b \in Z^{M_j})\}| \geq n\}. \end{aligned}$$

Usage of these concepts allows for reasoning in the terms of i -th ontology within j -th ontology. If, for example, in j -th ontology the axiom $D \sqsubseteq C$ is defined, and in i -th ontology there exist axioms $\exists p.C \sqsubseteq G$ and $H \sqsubseteq \exists p.D$, where $p \in \mathcal{E}_{ij}$, then in i -th ontology the axiom $H \sqsubseteq G$ holds.

The both presented methods try to solve the problems with contextual reasoning, i.e. how should interpretations of modules affect each other, how to preserve decidability, soundness and completeness. Although very important, results of this work do not help with the problem that we call the *problem of contextual designing*. They offer no tools that could allow to express why contexts were created and what is their purpose. In contrast to them, our proposal, the SIM method, is an example of a framework for designing *context-semantic knowledge bases*, i.e. knowledge bases where contexts are explicit elements of conceptualization and affect the semantics of another members of the model.

The second approach, the algebraic one, is based on the idea that all possible theories constitute a structure with the relationships between the elements describable by algebraic operation. In this approach we can distinguish two kinds of structures: *syntactic* and *semantic* ones.

A prominent example of an algebra for syntactic structures is the method described by Mitra in [18] (also by Mitra and Wiederhold in [19]). An ontology module is

defined here as a set of sentences, namely RDF triples, so that every algebraic operation gives such a set as a result.

The simplest use case for the algebra is integration of knowledge from two ontologies. To show how this algebra works in practice, we follow here a slightly modified example from [19] and assume that the two ontologies being integrated, O_1 and O_2 , describe respectively the domains of maritime vessels and cars (Fig. 4).

To enable conduction of any algebraic operation on the two ontologies, we firstly have to define special rules, called *articulation rules*, that explain the relationships between terms in the ontologies. For our example we define two such rules:

- *subClassOf*(O_1 .Motorboat, CombustionVehicle)
- *subClassOf*(O_2 .Car, CombustionVehicle)

These rules state that both a motorboat and a car are vehicles with a combustion engine. After defining the rules it is possible to perform binary operations on O_1 and O_2 . There are three kinds of such operations: union, intersection and difference. Outcome of each operation is a set of sentences, defined as follows. We assume that $O_{1,2}$ denotes the RDF graph obtained by “executing” the *articulation rules* (treated as production rules). Then (see Fig. 4):

- the union of O_1 and O_2 is the union of graphs O_1 , O_2 and $O_{1,2}$,
- the intersection of O_1 and O_2 is the graph $O_{1,2}$,
- the difference between O_1 and O_2 is the graph O_1 from which all nodes present in $O_{1,2}$ have been removed.

The described algebra is relatively straightforward and convenient for describing different points of view (or even contexts, as Mitra and Wiederhold argue in [19]). Nevertheless, its commitment to syntax can be a source of major difficulties with its use for practical, large ontologies. The outcome of algebraic operations relies heavily on the exact form of the sentences used in the modules being integrated. If the same semantic information is expressed with two different sentences, the result can easily convey the intention of the user. Moreover, to use the algebra one has to define a set of articulation rules. These sets constitute a new kind of extraontological objects that have their own syntax and particularities and need to be managed and maintained by a user, which can be cumbersome.

The aforementioned issues are not characteristic for the algebra by Mitra; on the contrary, they seem to be inherent for every algebra whose universe is based on syntactic form of sentences. This strongly suggests that semantic approach might be much more convenient for combining knowledge from different sources.

No approach known to the authors of this paper goes as deeply into semantic structures as conglomerate algebra. However, in the algebra proposed by participants of the NeOn project (deliverables D1.1.3 [20] and D1.1.4 [21]), a module is treated essentially (when we neglect its interfaces and version information) as a theory (i.e. a set of sentences augmented with all conclusions). This allows us to treat this method as a semantic one, because a theory can be identified with an appropriate set of models.

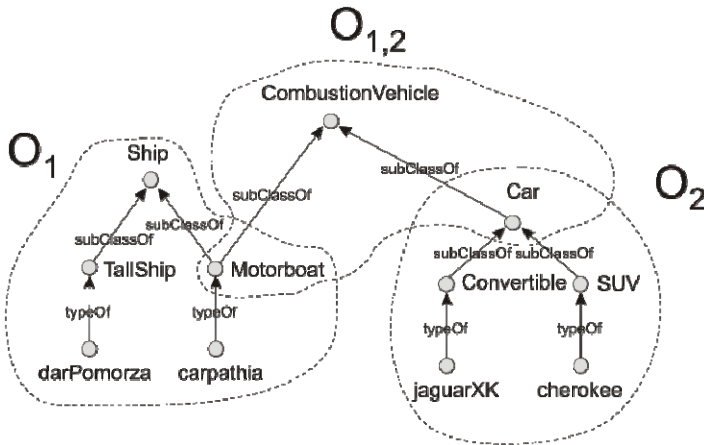


Fig. 4. An example of the aggregation conformance of denotation

In NeOn algebra there is no need for defining any special bridge axioms or rules. Instead, the algebra introduces several operations characterized in the terms of (semantic) entailment. Among them, there are the operations of union, intersection and difference (\cup , \cap , $-$, resp.) defined as follows (α is any standard DL sentence):

$$\begin{aligned}
 M_1 \cup M_2 = M & \text{ iff } & M \models \alpha & \leftrightarrow M_1 \models \alpha \vee M_2 \models \alpha \\
 M_1 \cap M_2 = M & \text{ iff } & M \models \alpha & \leftrightarrow M_1 \models \alpha \wedge M_2 \models \alpha \\
 M_1 - M_2 = M & \text{ iff } & M \models \alpha & \leftrightarrow M_1 \models \alpha \wedge \neg (M_2 \models \alpha)
 \end{aligned}$$

There is strong analogy between the above definitions and the conglomerate algebra. The union operation in NeOn and the intersection of conglomerates give the same outcome. Nevertheless, there are also important differences. The most notable one concerns difference between modules. NeOn definition seems to be faulty, because no module can possibly be a difference, which can be easily shown by putting for α any tautology.

While the definition of difference can probably be fixed, there is a discrepancy between NeOn and conglomerate algebra that is much deeper than a simple dissimilarity of definitions. Since NeOn module is a theory, it cannot exceed the expressivity of the used language. In conglomerate algebra it is not the case, as e.g. for union of $M(\{A \sqsubseteq B\})$ and $M(\{B \sqsubseteq A\})$ there is no corresponding set of sentences that expresses the result. While possibly this may be considered a drawback in some situations, this fact is also foundational for conglomerate algebra, as only through such assumption we may obtain an extension of Boolean algebra for semantic modules. Consequently, through this assumption, conglomerate algebra may constitute a basis for a query language that is equivalently strong as the relational algebra is for SQL.

In the last part of this section we refer to some languages created to communicate with knowledge bases. The most known and most widely applied is SPARQL [22], recommended by W3C. This is a query language designed for RDF repositories, thus it is not strictly addressed to ontologies. But, as OWL is built on top of RDF, it is

often used for ontological purposes. As SPARQL is a query language, the syntax does not allow to enter changes into knowledge base.

There are many languages designed for ontologies. DIG ([23], [24]), proposed in University of Manchester, is a language defined for maintaining ontologies defined in DL. OWL-QL ([25]), designed for Stanford Knowledge Systems Laboratory is a continuation of DQL—DAML Query Language. nRQL [26] is the query language for RacerPro. SAIQL [27] is developed in University of Koblenz-Landau and University of Aberdeen during the works on the NeOn project. None of these languages is able to deal with modular knowledge bases. A query is always addressed to one ontology. None of them is closed: queries cannot be nested in themselves, although the last one—SAIQL—returns as an answer a set of sentences forming a fully working OWL DL ontology rather than just substitutions of variables.

7 Conclusions

This paper presents the architecture and methods for building large federated knowledge bases. As the primary criteria for selecting methods of knowledge representation in such knowledge bases an ontology organization, capabilities of inference for a selected representation and a set of characteristics of an internal query language were selected.

With respect to the ontology organization, we assumed that the adequate method of knowledge representation must provide:

1. An appropriate structure of ontological modules
 - with the ability to add new source modules and update *ontology of science*,
 - with the ability to organize knowledge at various levels of detail,
 - with the ability to dynamically create ontological modules when answering queries,
 - with the ability to dynamically locate a set of modules containing information that can affect the answer to the query.
2. Expressiveness of CIDOC CRM (*SHIN(D)*, version *Erlangen CRM*).
3. Capability of retrieving data from external sources using “push” and “pull” methods.
4. Capability of adding sources incompatible with CIDOC CRM (changing perspective) in a way that preserves possibility of metadata conversion and unification to CIDOC CRM-compatible form.

With respect to the inference capabilities, we assumed that the method of knowledge representation must provide:

5. Capability of inference from combined modules (including these combined *ad hoc*) with a well-defined semantics (*global semantics*) while still maintaining the semantics of each module (*local semantics*).
6. Capability of sound and complete inference from ontologies defined in description logic with minimal expressiveness of $\mathcal{ALCHI}(D)$.
7. Capability of *Open World Assumption* aware inference.

With respect to the internal query language, we assumed that an adequate language must provide:

8. Capability of processing information expressed in description logic with minimal expressiveness of $\mathcal{ALCHI}(D)$.
9. Capability of binding information contained in different modules of varying granularity.
10. Capability of transformation of assertional knowledge (ABox) to terminological knowledge (TBox) and vice versa.
11. Capability of nesting queries (i.e. using modules created ad hoc in further processing).

These requirements imply that the basic feature of the presented solution is its modularity, and a non-modular methods are only a theoretical basis for the modular methods. One of the key distinguishing factors is the division of modular methods to *semantic* and *non-semantic* ones. The non-semantic methods emphasizes on locality and compatibility problems associated strongly with the inference. Although these problems are not irrelevant with respect to the proposed solution (see requirement 5), the emphasis here is put on the dynamic and efficient management of modules from the point of view of contextual features, especially of detail and scope (requirements 1 and 5), but also perspective (requirement 4). This analysis led to exploiting in the presented solution the SIM method that allows automatic conclusions flow between modules (instances of contexts). In this way, knowledge bases built on the basis of the SIM method solve the detail and scope problem in a natural and effective way.

The conglomerate-based method used in our approach combines the features of semantic and non-semantic approach. With the latter it has in common the great emphasis put on the management of the global semantics of modules. However, the space of modules in the conglomerate-based method has an algebraic nature. The actual semantics of this space must be imposed by a designer of a knowledge base by different means of expressiveness, outside the scope of conglomerate-based method. Hence, the proposed combination of the SIM and conglomerate-based methods work smoothly together.

Although selecting a knowledge representation language is somehow independent on the choice of knowledge representation methods, it is easily seen that the choice of conglomerate knowledge base is tightly connected with KQL as the internal query language. KQL, as based on closed algebraic structure, is the only one that provides nesting queries in a natural way (requirement 11).

In conclusion we can state that none of the methods of knowledge representation is mature enough to meet all the requirements defined above. Using SIM and conglomerate-based methods together seems to simultaneously fulfill the requirements for the management of modules scope and details as well as perspective and offer a very flexible way to create a new ontological modules (by means of conglomerate algebra and KQL). So, next research and prototype implementation will be conducted in this direction.

References

1. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F.: *Description Logic Handbook*. Cambridge University Press (2002)
2. OWL 2 Web Ontology Language Document Overview, W3C Recommendation, October 27 (2009), <http://www.w3.org/TR/owl2-overview/>
3. SWRL: A Semantic Web Rule Language Combining OWL and RuleML, W3C Member Submission, May 21 (2004), <http://www.w3.org/Submission/SWRL/>
4. Goczyła, K., Waloszek, A., Waloszek, W.: Contextualization of a DL knowledge base. In: *Proceedings of the 20th International Workshop on Description DL 2007*, Brixen/Bressanone, Italy, pp. 291–299 (2007)
5. Goczyła, K., Waloszek, A., Waloszek, W.: Algebra of ontology modules for semantic agents. In: Nguyen, N.T., Kowalczyk, R., Chen, S.-M. (eds.) *ICCCI 2009. LNCS (LNAI)*, vol. 5796, pp. 492–503. Springer, Heidelberg (2009)
6. Goczyła, K., Piotrowski, P., Waloszek, A., Waloszek, W., Zawadzka, T.: Terminological and Assertional Queries in KQL Knowledge Access Language. In: Pan, J.-S., Chen, S.-M., Nguyen, N.T. (eds.) *ICCCI 2010. LNCS*, vol. 6423, pp. 102–111. Springer, Heidelberg (2010)
7. Goczyła, K., Grabowska, T., Waloszek, W., Zawadzki, M.: The Knowledge Cartography – A New Approach to Reasoning over Description Logics Ontologies. In: Wiedermann, J., Tel, G., Pokorný, J., Bieliková, M., Štuller, J. (eds.) *SOFSEM 2006. LNCS*, vol. 3831, pp. 293–302. Springer, Heidelberg (2006)
8. CIDOC Conceptual Reference Model (CRM), <http://www.cidoc-crm.org/>
9. Dublin Core Metadata Element Set, Version 1.1, <http://dublincore.org/documents/dces/>
10. Haase, P., Hitzler, P., Rudolph, S., Qi, G.: D3.1.1 Context Languages—State of the Art. The NeOn project deliverable (2006)
11. Garson, J.W.: *Modal Logic for Philosophers*. Cambridge University Press (2006)
12. Kaplan, D.: On the Logic of Demonstratives. *Journal of Philosophical Logic* (8), 81–98 (1978)
13. McCarthy, J.: Notes on Formalizing Context. In: *Proceedings of IJCAI 1993*, pp. 562–555. Morgan Kaufmann (1993)
14. Giunchiglia, F., Serafini, L.: Multilanguage hierarchical logics, or: how we can do without modal logics. *Artificial Intelligence* 65(1), 29–70 (1994)
15. Ghidini, C., Giunchiglia, F.: Local model semantics, or contextual reasoning = locality + compatibility. *Artificial Intelligence* 127(2), 221–259 (2001)
16. Borgida, A., Serafini, L.: Distributed description logics: Assimilating information from peer sources. In: Spaccapietra, S., March, S., Aberer, K. (eds.) *Journal on Data Semantics I. LNCS*, vol. 2800, pp. 153–184. Springer, Heidelberg (2003)
17. Kutz, O., Wolter, F., Zakharyashev, M.: Connecting abstract description systems. In: *Proceedings of the Twelfth International Conference on the Principles of Knowledge Representation and Reasoning 2002*, pp. 215–226. Morgan Kaufmann (2002)
18. Mitra, P.: *An Algebraic Framework for the Interoperation of Ontologies*. PhD thesis, Stanford (2004)
19. Mitra, P., Wiedehold, G.: An Ontology-Composition Algebra. In: Staab, S., Studer, R. (eds.) *Handbook on Ontologies*, pp. 93–113. Springer, Heidelberg (2004)
20. d’Aquin, M.: D1.1.3 NeOn Formalisms for Modularization: Syntax, Semantics, Algebra. The NeOn project deliverable (2008)

21. d'Aquin, M.: D1.1.4 NeOn Formalisms for Modularization: Implementation and Evaluation. The NeOn project deliverable (2008)
22. Prud'hommeaux, E.: SPARQL Query Language for RDF, <http://www.w3.org/TR/rdf-sparql-query/>
23. Bechhofer, S.: The DIG Description Logic Interface: DIG/1.1. University of Manchester (2003)
24. DIG 2.0: The DIG Description Logic Interface. DIG Working Group Note (September 2006), <http://dig.cs.manchester.ac.uk/>
25. Fikes, R., Hayes, P., Horrocks, I.: OWL-QL—A Language for Deductive Query Answering on the Semantic Web. Knowledge Systems Laboratory, Stanford University, Stanford (2003)
26. RacerPro User's Guide, <http://www.racer-systems.com/products/racerpro/manual.phtml>
27. Kubias, A., Schenk, S., Staab, S.: SAIQL Query Engine - Querying OWL Theories for Ontology Extraction. In: OWLED 2007 OWL: Experiences and Directions Third International Workshop (2007)
28. Goczyła, K., Waloszek, A., Waloszek, W.: A Semantic Algebra for Modularized Description Logics Knowledge Bases. In: Grau, B.C. (ed.) Proceedings of the 22nd International Workshop on Description Logics (DL 2009), Oxford (2009)
29. Goczyła, K., Zawadzka, T., Zawadzki, M.: Managing Data from Heterogeneous data Sources using Knowledge Layer, Software Engineering Techniques: Design for Quality. In: Sacha, K. (red.) IFIP International Federation for Information Processing. 227, pp. 300–312. Springer, Boston (2006)

FPGA Implementation of the Selected Parts of the Fast Image Segmentation

Maciej Wielgosz, Ernest Jamro, Dominik Żurek, and Kazimierz Wiatr

AGH University of Science and Technology,
al. Mickiewicza 30, 30-059 Krakow
ACK Cyfronet AGH
ul. Nawojki 11, 30-950 Krakow
{wielgosz,jamro,wiatr}@agh.edu.pl,
dominik.zurek1102@gmail.com

Abstract. This paper presents preliminary implementation results of the SVM (Support Vector Machine) algorithm. SVM is a dedicated mathematical formula which allows us to extract selective objects from a picture and assign them to an appropriate class. Consequently, a black and white images reflecting an occurrence of the desired feature is derived from an original picture fed into the classifier. This work is primarily focused on the FPGA implementation aspects of the algorithm as well as on comparison of the hardware and software performance. A human skin classifier was used as an example and implemented both on AMD AthlonII P320 Dual-Core2.10 GHz and Xilinx Spartan 6 FPGA. It is worth emphasizing that the critical hardware components were designed using HDL (Hardware Description Language), whereas the less demanding or standard ones such as communication interfaces, FIFO, FSMs were implemented in HLL (High Level Language). Such an approach allowed us both to cut a design time and preserve a high performance of the hardware classification module.

Keywords: Picture segmentation, FPGA, reconfigurable logic, SVM.

1 Introduction

This work is part of the Synat project embracing several initiatives aiming to create a repository of images which are assigned a descriptive name according to their contents. Such a database of tagged images will significantly reduce search time since only picture tags will be processed instead of images so the process will involve simple string operations rather than image recognition. It is worth noting that such a database may also provide a mean to combat Internet threats and crimes through an access to the large well classified repository of visual information.

The project is a huge challenge due to an immense volume of data collected over the past years denoted today as the Internet resources. Therefore the core part of the undertaking is to design and implement a classification system which

should be both reliable and fast. In order to achieve the high performance of a search engine the most computationally intensive operations are to be ported to hardware. Thus FPGAs due to their strongly parallel structure, huge logic resources and growing processing speed [1] seem to be the best choice.

Image segmentation is a process which aims to separate a picture into several regions based on objects or features of interest. A single SVM system may embrace several modules trained to recognize different features so the unit as a whole is capable of tracing multidimensional objects in terms of a number of features.

It is worth emphasizing that a segmentation may also be regarded as a form of data compression, the classifier accepts images and yields information regarding objects which usually occupies much less memory resources than corresponding original data.

There are plentiful image segmentation algorithms [2,3,4] and their number is still growing to meet constantly rising demands of data analysis systems. However, reliability and data processing speed are the factors which are at a premium when it comes to a real life application of a given algorithm. SVM meets both those criteria and therefore was chosen as a classification algorithm for the project.

2 SVM Classifiers

Support vector machines were originally devised and described by Vapnik [5,6]. They are used for binary classification which means that there are exactly two classes of objects (e.g. black and white rectangles) and a classification formula is found in a training process of the classifier.

The SVM algorithm can be envisioned as a process of creating a hyperplane which separates data in an n -dimensional space. It is conducted in an iterative manner in which a selected plane is gradually adjusted to provide the optimal so-called generalization margin. The following cases may occur:

- Input data is linearly separable and the SVM method guarantees that at least one plane of the best separation margin exists and will be adopted
- A dimension incrementation is used to bring data to a space of more dimensions space so a separation plane can be found

A feature space for 2D can be modeled as a sphere with its center and radius (see Fig. [1]).

The sphere is built upon a set of supportive vectors which constitute its structure. A classification process of a priorly trained SVM maybe perceived as probing whether a given point (input data) belongs to the sphere or it's located outside of it. In the first case a point is positively classified whereas in the second one it's considered to be an outlier.

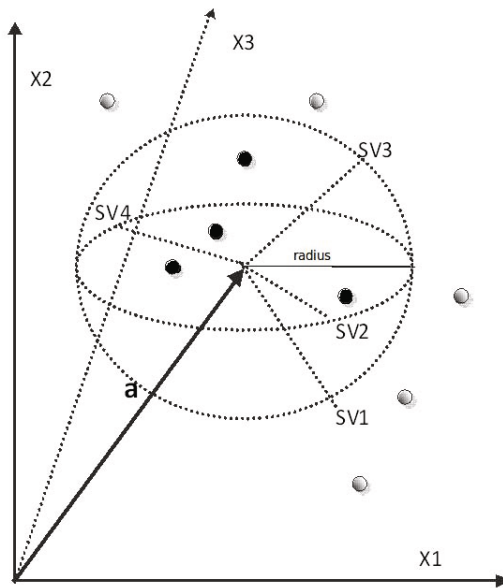


Fig. 1. N-sphere

3 A Choice of a Hardware Platform

It is very important to choose a proper architecture and appropriate data transfer protocol since it affects the overall performance of the computational system. It also may balk an effort invested in the development of the hardware algorithm. Therefore the authors decided to review available FPGA platforms.

FPGAs have been developed since late 1980s, and have a lot of advantages over processors. The most important ones are: massive parallel architecture, reconfigurability, low energy consumption, ability to shape freely its internal architecture.

Design and effective use of computing system based on FPGA is a difficult task, as evidenced by the long history of such trials. The father of the concept of using reconfigurable logic in computational systems is considered Erstina Gerald, who first introduced a vision of a machine consisting of a number of reprogrammable matrices in the 1960s. But the idea had not been fulfilled until the late 80 century, when it first appeared as a reconfigurable computer Algotronix CHS2x4, built with 8 CAL1024 units. Since then the continuing development of modules, which in various ways integrate FPGAs in a computing system.

Existing HPRC (High Performance Reconfigurable Computing) solutions can be classified based on their integration with other computing nodes in the system.

1. Solutions based on FPGAs as the processor directly attached to a computational system. Usually such modules communicate over PCI or PCI Express or Ethernet bus. Such an arrangement is most common in small systems consisting of several nodes. A system bus bandwidth is often the limiting

performance parameter of such solutions. Alpha-Data [7], Nallatech [8], Pico Computing [9] cards can be given as example.

2. System with a dedicated point-to-point connection between FPGAs. This solution has a great potential for parallelization and scaling algorithm. A communication with the nearest neighbor is particularly effective. An example of such a system may be a computer Maxwell [10]
3. Solution based on two-domain approach which divide FPGA part of the system from CPU section. The advantage of this configuration is the ability to communicate across of the all computational units within a system. It is worth taking into account that overloading of a communication bus results in a sudden drop in system performance. Cray XD1 computer can be stated as an example of such a solution.
4. A system based on non uniform memory access. This approach allows for virtually any data exchange between computing nodes in the system and creates a challenge in the form of memory coherence. The best-known commercially available solution of this type is SGI RASC [11], which utilizes the NUMalink (Non Uniform Memory Access link) bus to access the global memory. The downside of this solution is unpredictability of data access time, what could potentially create difficulties for algorithms for which this parameter is critical.
5. The idea of populating CPU sockets with FPGAs. This approach can be considered as a modification of the architecture described in section 2. An example of such a system is the topology of the DRC coprocessor system Accelium (Xilinx Virtex-5) [12]. This architecture allows for equal access of processor and FPGA to the system resources. However, this has a significant drawback, the occurrence of any error in the processing FPGA automatically affects the stability of the whole system, in particular shared memory read and write operations.

The project described in this paper will be launch on the 1, 4 and 5 architecture since those platforms are available at ACC Cyfronet AGH.

The main research objectives from the hardware perspective are:

1. Implement a large part of the SVM algorithm in FPGA. Design a set of hardware modules which constitute the SVM classifier.
2. Design of effective mechanisms for the inter-module data transfer.
3. Research on the possibility of an effective integration of the FPGA modules within a single computational system using MPI, OpenMP or OpenCL. Such an approach would significantly facilitate integrating of the building modules within a system.

4 System Overview

A human skin classifier OC-SVM (One Class Supportive Vector Machine) was implemented as a preliminary project which allows us to estimate performance

and resource consumption for other classifiers. As a result of an experiment a black-and-white image is generated which reflect human skin location in the original picture which was fed into the classifier. A complete computational procedure is composed of several steps:

- SVM vectors and τ generation (training of the classifier)
- Input image fetch (the step is different for hardware and software implementation)
- Image resize and normalization
- Classification
- Noise and skin-like objects filtration

4.1 The Classification Algorithm

The classification algorithm is given by the following formula:

$$\sum_i \alpha_i K(X_x, X_i) \geq \sum_i \alpha_i K(X_s, X_i) = \tau \quad (1)$$

where τ is the sphere radius, X_s and α_i are supportive vectors derived in a training process, X_x is an input pixel.

Regardless of a choice of vectors in right side of the equation (1) the result is constant and equals τ . Each pixel fed into the classifier is compared against all the support vectors in order to determine if it is located inside the sphere (see Fig. 2).

In this implementation a Gaussian computational kernel was used:

$$K = e^{-\gamma \|X_i - X_j\|^2} \quad (2)$$

where γ is a spread of the kernel.

If the (1) is met a given point is classified as belonging to the desired class. For SVM classifies the best results are achieved when input data is normalized (i.e. fall in the range [-1;1]).

4.2 Architecture of the Hardware Module

The computationally intensive routines were ported to hardware to offload the GPP (General Purpose Processor) and to accelerate the computations. It was possible due to several features of the algorithm which makes it well suited for the FPGA implementation such as: fixed-point arithmetic, parallel structure (easy to pipeline), narrow-range input argument. Consequently a series of hardware units were designed which constitute the internal structure of computational module as presented in Fig. 3. All the modules are parameterized and pipelined blocks which process a single input vector X every clock cycle. For a sake of the software compatibility the base data format employed in the application is 32 bit fixed-point (16 bits of both fractional and integer part) but it can be adjusted to meet different precision requirements in the future. Each module is equipped

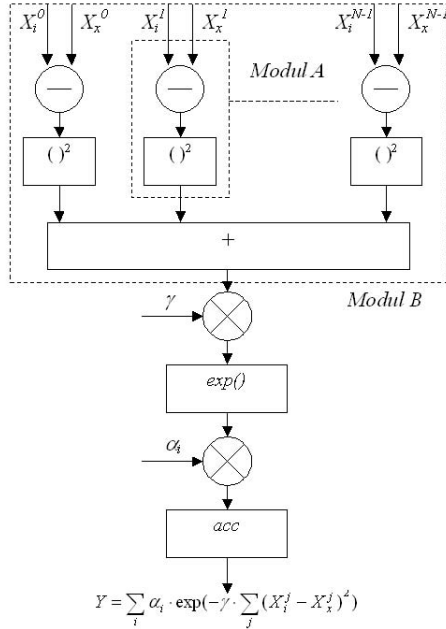


Fig. 2. Block diagram of the classification module

with the overflow signal which propagates across all the units composing the classification module. Such an approach allows us to avoid corruptions of the result just by simply examining the overflow output.

It is possible to connect several classification modules to form a parallel structure as it is depicted in Fig. 3. Furthermore, it is worth noting that supportive vectors (denoted as SV in Fig. 3) are fetched from an external memory only once for the whole computations and therefore can be stored in the internal memory for all the computation. Moreover, the number of the supportive vectors as well as α is not large and usually not exceed tens, thus internal BRAM memory suffice to accommodate those coefficients (e.g. for the human skin classification only 17 supportive vectors are used). Increase of a number of supportive vectors improves the classifier accuracy at the expense of the accumulator throughput decrease (see Fig. 2) which in turn affects an overall system performance.

The classifier (presented in Fig. 2) yields one bit results which reflects an occurrence of a feature of interest within an image. Therefore in order to take a full advantage of an external bus throughput, classification results are compacted into 32 bit bundles and sent to a host processor as such. Thereafter the GPP transforms those binary values into pixels to form a black-and-white image depicting the features of interest.

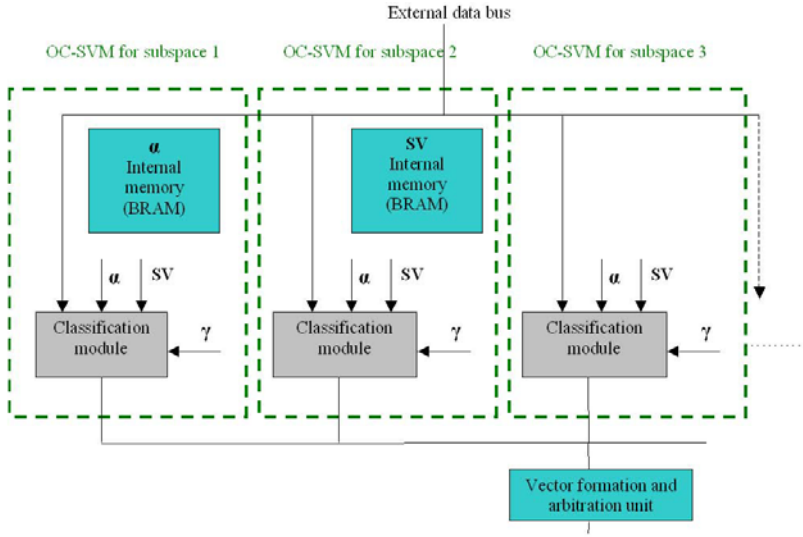


Fig. 3. Block diagram of the multimodule structure

4.3 Exp Module

The $\exp()$ function described in this paper was designed in VHDL from scratch. According to authors knowledge there are only few available libraries of fixed point modules including $\exp()$ e.g. [13] but they do not meet design requirements such as a parameterization range. Therefore the authors implemented their own $\exp()$ module.

The \exp hardware algorithm is composed of two steps: reduction of a calculation range and approximation of the function within a priori defined range. It is based on a similar algorithm for floating point numbers [14] and the block diagram of the module was presented in Fig.4

After applying the following formula, the input argument is separated to integer X_i and fractional X_f part. It should be noted that $2 \times X_i$ can be easily calculated employing a barrel shifter. Therefore the main problem is calculation of $\exp(X_f)$.

$$e^x = e^{X/\ln(2)} \tag{3}$$

A fractional part $X_f = X - (\lfloor X/\ln(2) \rfloor \times \ln(2))$ is obtained according to Fig.4. It should be noted that integer part is calculated employing base 2 (thus barrel shifter can be used) but the fractional part is calculated employing base e (thus a simply Taylor expansion can be used). It is worth emphasizing that the $\exp()$ is a strongly declining function (the argument is always a negative number) consequently a relatively small input value (absolute value) yields roughly zero as an $\exp()$ result. For instance an argument larger than $\ln(2^{16}) = 11.09$ (which integer part can be mapped on 4 bits) gives zero as the fractional part of the $\exp()$ result is represented only on 16-bits. Therefore calculation of $\lfloor X/\ln(2) \rfloor \times$

$\ln(2)$ can be conducted with the limited bit-width, which significantly reduces hardware requirements of the $1/\ln(2)$ and $\ln(2)$ multipliers. The fractional part X_f is then bit separated into -the most X_{msb} and less X_{lsb} significant part. The X_{msb} is fed to the LUT (Look-Up Table) memory. And the X_{lsb} is calculated employing Taylor expansion.

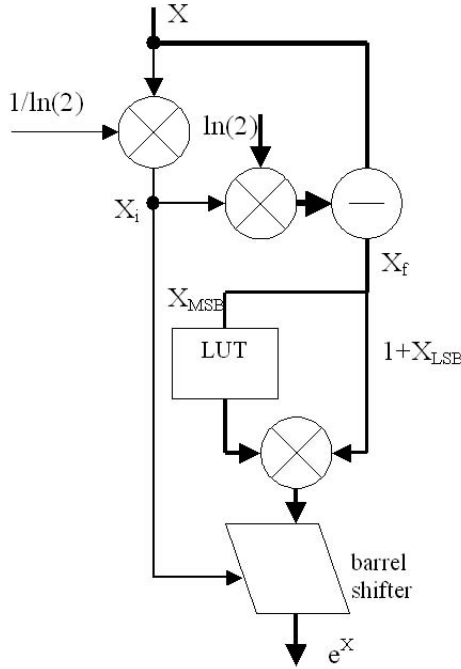


Fig. 4. Block diagram of the exp() module

The complete exp() algorithm is given as follows:

$$e^x = e^{x/\ln(2)} = 2^{x_i} \cdot e^{(x-x_i)/\ln(2)} \tag{4}$$

$$e^{X_f} = e^{X_{MSB}+X_{LSB}} = e^{X_{MSB}} \cdot e^{X_{LSB}} = LUT(X_{MSB}) \cdot (1 + X_{LSB}) \tag{5}$$

The $e^{X_{LSB}}$ evaluation is performed as a LUT operation which coefficients are stored in a internal FPGA block memory denoted as a BRAM. The second part of the equation - $1+X_{LSB}$ was implemented as a short Taylor expansion. It is achievable because X_{LSB} is so small that the next term of the Taylor series, $x^2/2$, does not influence the result. The presented exp() implementation is fast and consumes insignificant resources. Consequently a single FPGA can accommodate several of such modules along with the other units.

5 Implementation Results

The classification algorithm was initially implemented on GPP in C++ and the OpenCV library was used. A set of 17 support vectors was generated which described a human skin, each of which are 32bit RGB colors.

The figures below represent experimental results for the randomly chosen images. It can be noticed that the system wrongly classified some parts of the image. Unfortunately the system often confuses bright objects with a human skin. One way to improve the accuracy is increasing the contrast between an object and a background. Similar result of accuracy improvement may be achieved when a larger number of supportive vectors is employed but it is done at a expense of a loss of classifier's generalization feature.



Fig. 5. Original image (before segmentation)

Time required to execute the following algorithm on GPP: AMD Athlon II P320 Dual-Core 2.10 GHz (on a single core) for an image of 480x480 pixels takes roughly 0.015s. Hardware implementation according to the formula (1) (assuming that no input data fetch delay is introduced) can be calculated as follows: $17(SVM) \times 480(\text{pixels}) \times 480(\text{pixels}) \cong 4 \times 10^6$ clock cycles. Consequently theoretical processing time for 200 MHz equals 0.02s. Due to a low resources consumption a single FPGA can accommodate several modules which boost a performance several times.



Fig. 6. Results of human skin segmentation with the proposed method



Fig. 7. Original image (before segmentation)



Fig. 8. Results of human skin segmentation with the proposed method

The implementation results of the module on Xilinx Spartan 6 (XC6SLX75) FPGA were presented in Tab. 1 and Tab. 2.

Table 1. Implementation results of the module building blocks (see Fig. 2)

Module	# 4-input LUT	# flip-flops
Square module - A	660 (1%)	220 (1%)
Module - B	1953 (4%)	760 (1%)
exp() module	312 (1%)	277 (1%)

It is worth noting that a number of coefficients has a large impact on the resources occupation. In this particular implementation the number of the coefficient is three (R, G, B).

The classification module is a fully pipelined structure and it is capable of working at the frequency of 200 MHz. Each module generates a single result every n clock cycles where n denotes number of the support vectors employed. This processing speed constrain results from the internal structure of the accumulator (see Fig. 2) which requires n clock cycles to generated a single result.

Table 2. Implementation results of the classifier module for different number of $[j]$ vector coefficients

Module	# 4-input LUT	# flip-flops
2	2689 (5%)	1048 (1%)
3	3377 (7%)	1311 (1%)
4	6319 (13%)	1834 (2%)

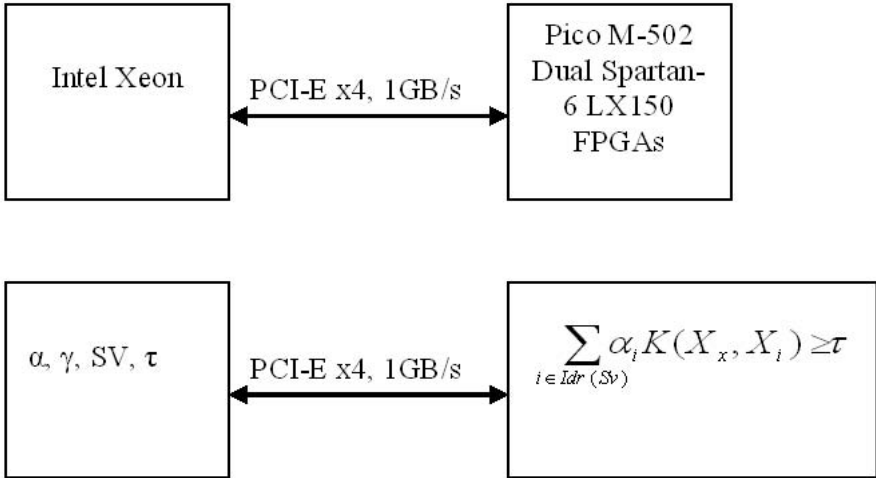


Fig. 9. Data transfer scheme between FPGA and the GPP

Initially the platform of choice was Pico M-502 equipped with two Spartan-6 LX150 FPGA and 512 MB DDR3 of local memory but as the calculations below show the data bandwidth is too low. The board is connected with GPP via PCI-express x4 interconnect which allows for 1 GB/s data transfer in each directions.

Support vectors along with α, γ, τ are generated on the host side (by GPP) and are sent to the FPGA only once for the whole computations (see Fig. 9). The module implemented in the FPGA performs the classification for all the X vectors and sends the results back to the host processor.

The data bus is 32 bits width and each vector consist of 32×3 bits. However, taking into account the range in which X vectors fall i.e. $[-1;1]$, it can be noticed that only 16 LSB bits are occupied. Therefore the amount of data to be transfer over the data bus is reduced twice and it results in 16×3 bits. Nevertheless still more than a single bus cycle is required to transfer a single x vector on Pico platform.

Time essential to process a single x vector is given by the following formula (FPGA clocked at 200 MHz):

$$n \cdot 5ns/NoM \quad (6)$$

where NoM stands for the Number of Modules (classification modules) working in parallel and n - denotes a number of support vectors.

Back-of-the-envelope calculations for the human skin classifier (17 support vectors) show that in order to provide compatibility with HD (High Definition) 1080p ($1920 \times 1080 \times 25$) standard FPGA should process each pixel every 25 ns. It means that at least 4 parallel classification modules should be used ($17 \times 5ns/25ns = 3,4$) which in turn requires an external bus throughout of 4,8 GB/s ($16b \times 3 \times 4 \times 200MHz = 6B \times 4 \times 200MHz$).

It turns out that in the case of Pico platform data throughput is a bottleneck. Therefore the DRC AC2020 [12] will be used to implement the classifier. The DRC platform is capable of achieving 9.6 GB/s of aggregated HT bus bandwidth.

6 Summary

In this paper preliminary implementation results of the selected parts of the fast image segmentation were presented along with some performance analysis. Both software and hardware approached were discussed with their critical aspects such as bandwidth limitations and data precision. Furthermore several HPRC platforms were described with a special focus of their architecture and inter-module data exchange capabilities.

As a future work presented segmentation algorithm will be extended with some additional functionality and implemented on the DRC platform [12].

Acknowledgments. The work presented in this paper was financed through the research program - *Synat*.

References

1. Mueller, R., Teubner, J., Alonso, G.: Data Processing on FPGAs, Systems Group, Department of Computer Science, ETH Zurich, Switzerland, VLDB 2009, August 24-28, Lyon, France (2009)
2. Jun, T.: A color image segmentation algorithm based on region growing. In: 2010 2nd International Conference on Computer Engineering and Technology (ICET), April 16-18, vol. 6, pp. V6-634-V6-637 (2010)
3. Farmer, M.E., Jain, A.K.: A wrapper-based approach to image segmentation and classification. IEEE Transactions on Image Processing 14(12), 2060-2072 (2005)
4. Lan, Y., Li, C., Zhang, Y., Zhao, X.: A novel image segmentation method based on random walk. In: Asia-Pacific Conference on Computational Intelligence and Industrial Applications, PACIIA 2009, November 28-29, vol. 1, pp. 207-210 (2009)
5. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer, Heidelberg (2000)

6. Ben-Hur, A., Horn, D., Siegelmann, H.T., Vapnik, V.: A support vector clustering method. In: Proceedings. 15th International Conference on Pattern Recognition 2000, vol. 2, pp. 724–727 (2000)
7. <http://www.alpha-data.com/>
8. <http://www.nallatech.com/>
9. <http://www.picocomputing.com/>
10. Baxter, R., Booth, S., Bull, M., Cawood, G., Perry, J., Parsons, M., Simpson, A., Trew, A., McCormick, A., Smart, G., Smart, R., Cantle, A., Chamberlain, R., Genest, G.: Maxwell - a 64 FPGA Supercomputer. In: Second NASA/ESA Conference on Adaptive Hardware and Systems (AHS 2007), pp. 287–294 (2007)
11. http://www.silicongraphics.ru/pdf/rasc_data.pdf
12. http://www.drccomputer.com/pdfs/DRC_Accelium_Overview.pdf
13. <http://www.vhdl.org/fphdl/>
14. Wielgosz, M., Jamro, E., Wiatr, K.: Hardware implementation of the exponent based computational core for an exchange-correlation potential matrix generation. In: Wyrzykowski, R., Dongarra, J., Karczewski, K., Wasniewski, J. (eds.) PPAM 2009. LNCS, vol. 6067, pp. 115–124. Springer, Heidelberg (2010)

Evaluation of Image Descriptors for Retrieval of Similar Images

Rafał Frączek and Bogusław Cyganek

AGH University of Science and Technology
Al. Mickiewicza 30, Kraków 30-059
Academic Computer Center Cyfronet AGH
Ul. Nawojki 11, Kraków 30-950

Abstract. The paper addresses the issue of searching for similar images in an information repository. The contained images are annotated with the help of sparse descriptors. In the presented research different color and edge histogram descriptors were used. To measure distances among images the sets of their descriptors are compared. For this purpose different similarity measures were employed. Results of these experiments, as well as discussion of the advantages and limitations of different combinations of methods for retrieval of similar images are presented.

Keywords: color descriptors, edge histogram, graph matching, query-by-example.

1 Introduction

Many applications in industry, entertainment and digital library require access and query of image data sets. This access can be performed using a text based queries (e.g. find images with a given tag). Another possibility is to access the image data base by using a reference image. In this approach, the user may want to find an image that is the same or similar in some way to the reference image (QbE – Query by Example). This type of systems is based, in most cases, on features extracted from the media. Sets of features are generally referred to as “descriptors” and their instances are called “descriptor values”. The descriptor values are the meta-data of the media. Over the past years several methods of QbE have been presented. For instance, the VICTORY project [1] is a good example of advanced research in the area of QbE – in this case focusing on search for 3D objects. Another example of software for images retrieval is a program called Picture Finder [2] which is a fast image retrieval library for image-to-image matching. Similarity is measured based on spatial distribution of color and texture features. It is especially optimized for fast matching within large data-sets.

When developing a system for similar images retrieval, a question arises: which descriptors and which distance measures should be applied for the most accurate results. In this paper we address this question as well and try to discuss results of our research in the area of QbE. We examine several color descriptors, especially most

common variants of the SIFT (Scale Invariant Feature Transform) [3] descriptor as well as mpeg-7 [4] descriptors. Another aspect of content based images retrieval is choice of the most appropriate distance (similarity) measure [5]. This is a big challenge in the case of SIFT descriptor because it is composed of variable number of key points with assigned feature vectors. This representation enforces non-trivial approaches for descriptors matching. To address this problem, two possible methods of similarity measure for SIFT descriptors were examined.

The rest of the paper is organized as follows: section 2 presents the image descriptors used in experiments, section 3 describes the similarity measures that have been applied, section 4 presents obtained results and also provides more information on possible further work. The last section provides conclusion with possible applications and insight into the further work on the topic.

2 Image Descriptors

This section presents a short description of image descriptors used in the experiments. These are: SIFT, OpponentSIFT, C-SIFT, Edge Histogram and Dominant Color.

2.1 SIFT Descriptor

SIFT is a descriptor that extracts so called key points in an image. Each key point has assigned a vector of features describing the distribution of local gradients in the nearest neighborhood of a given key point. The descriptor was invented by David G. Lowe [3] and is widely applied in digital image processing. The extraction procedure runs as described in the following steps.

Constructing a Scale Space. It is a well known fact that real objects are meaningful at certain scale. This is why the SIFT descriptor employs the multi-scale approach for features extraction. It is important that when details are removed from an image, no false details are added to the image. This is achieved by using two-dimensional Gaussian filter, defined as follows:

$$L(x, y, k \sigma) = I(x, y) * G(x, y, k \sigma) \quad (1)$$

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(\frac{-x^2 + y^2}{2\sigma^2}\right) \quad (2)$$

Scale space representation is created by taking the source image and then creating progressively blurred out images. There are five blurred images generated in this way. In the next step the source image is resized to half size and blurred images are generated again. Images of the same size form an octave. The SIFT algorithm uses by default 4 octaves.

Laplacian of Gaussian Approximation. In the previous step, the scale space representation of the source image was created. In this step, the previously generated images are used to create another set of images, namely the Difference of Gaussians (DoG). In order to find good candidates for key points, it is necessary to extract

corner points form the source image. It is usually achieved by calculating the Laplacian of Gaussian (LoG) operator which is an estimate of the second order derivatives. However, calculating second order derivatives is computationally intensive. Therefore, to speed up the calculations, an approximation of the LoG operator is used. To generate LoG images, the difference between two consecutive scales is calculated (DoG). The difference of Gaussian images are approximately equivalent to the Laplacian of Gaussian.

Extracting Key Points. In the previous two steps, a scale space representation was generated and the difference of Gaussians images were calculated. In this stage, the DoG images are used to find key points. The procedure of finding possible key points consists of two steps: localizing extremes (minimum and maximum) in DoG images and calculating sub-pixel location of these key points. The first step is performed by iterating through all pixels and checking their neighbors. In order to verify if a given pixel represents a local minimum or maximum, a comparison is done within the current image, and also the one above and below it (26 checks). A pixel is labeled as a key point if its value is less, or alternatively greater, of all values of its 26 neighbors. Key points are not calculated in the topmost and lowermost scales. The found pixels are approximated extrema because the real extrema usually do not lie on pixel exactly. Therefore, it is necessary to do sub-pixel approximation. This is achieved by applying the Taylor expansion of the image around the extreme point. Mathematically, it is described by the following equation:

$$D(x) = D + \frac{\delta D^T}{\delta x} x + \frac{1}{2} x^T \frac{\delta^2 D}{\delta x^2} x \tag{3}$$

where D denotes the sub-image around the extreme point.

Eliminating Low Contrast and Edge Points. Extreme points found in the previous stage generate many key points. Some of them lie in a region of low contrast or lie along an edge. Both these types of points are not meaningful as features. So, they are filtered out. For low contrast pixels, if the intensity of a pixel is below a certain level then this pixel is discarded. For edge points, the idea is to calculate two perpendicular gradients at the given key point. If both gradients are big enough (a corner point), the candidate pixel is marked as a key point. Otherwise, it is rejected.

Assigning Key Point Orientation. In the previous stage, the legitimate key points have been found. In this step, an orientation is assigned to each key point. It is done by collecting gradients and their magnitudes around each key point. Then, the most significant gradients are found and they are assigned to the key point. Gradients are calculated using the following formulas:

$$m = \sqrt{(L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1))^2} \tag{4}$$

$$\theta(x, y) = \arctan \left(\frac{L(x, y+1) - L(x, y-1)}{L(x+1, y) - L(x-1, y)} \right) \tag{5}$$

where $L(x, y)$ stands for pixel value.

All further calculations are done relative to this orientation. This way rotation invariance is ensured.

Having calculated gradients around the key point, a histogram is created. The histogram consists of 36 bins (each 10 degrees). Each gradient adds value proportional to its magnitude to the respective bin. A key point is assigned the orientation corresponding to the bin with the greatest value.

Generating SIFT Features. In this stage a feature vector is assigned to each key point. To do this, a 16x16 window around the key point is used. This 16x16 window is broken into sixteen 4x4 windows. Within each 4x4 window, gradients are calculated. These orientations are used to generate an 8 bin histogram. Since there are 16 sub-windows, the overall descriptor has 16x8=128 numbers. These numbers are normalized and form a feature vector.

2.2 OpponentSIFT

The OpponentSIFT descriptor works in the same way as standard SIFT descriptor. The only difference is that before the calculations, the source image is transformed to the so called opponent color space. The mathematical formula is as follows:

$$(O_1, O_2, O_3) = \left(\frac{R-G}{\sqrt{2}}, \frac{R-G-2B}{\sqrt{6}}, \frac{R+G+B}{\sqrt{3}} \right) \quad (6)$$

The O_3 channel represents the intensity information. The other channels represent the color information in the image. The opponent color space ensures both intensity change and intensity shift invariance.

2.3 C-SIFT

In the opponent color space, the O_1 and O_2 components still carry some intensity information. In order to overcome this drawback, the color invariants [6] were suggested as the input color space for the SIFT algorithm. The C-SIFT [7] uses the normalized opponent color space O_1 / O_3 and O_2 / O_3 . As the result, the C-SIFT is invariant with respect to the light intensity.

2.4 Edge Histogram

Edge Histogram [8] is a descriptor that is a part of the mpeg-7 standard. This descriptor calculates spatial distribution of all edges in the image. The descriptor defines five types of edges: horizontal, vertical, 45 degrees, 135 degrees and non-directional. The extraction procedure starts with partitioning the source image into 16 square blocks. For each of these 16 blocks, the local histogram of edges is generated. The histogram is composed of five bins corresponding to the five types of edges. The overall descriptor is composed of 16*5=80 values. In order to generate local histograms, each of 16 blocks is divided into smaller blocks. The size of these small block depends on the source image resolution. To determine the edge type, each small

block is treated as if it were 2x2 pixel block. Then standard five directional edge detectors are applied. The block is assigned an edge type corresponding to the detector with the greatest response.

2.5 Dominant Color

Dominant Color is a compact descriptor that calculates representative colors and their percentages. The extraction procedure is based on the Generalized Lloyd Algorithm (GLA) [9] which is used to quantize colors with clusters merging. The descriptor is defined as $F = \{ \{c_i, p_i\}, i=1, \dots, N \}$, where c_i is a 3D dominant color vector, p_i is the percentage of each dominant color and N is the total number of dominant colors in an image.

3 Similarity Measures

If we want to compare two images, we need to calculate a distance between two descriptors. In the reported experiments we have employed and tested two approaches for calculating the distance between two SIFT descriptors.

Simple Graph Matching. Suppose we have the first SIFT descriptor of N key points and the second SIFT descriptor of M key points. The distance between these two descriptors is calculated in the following way. For each feature vector in the first descriptor the nearest (D_1) and the second nearest (D_2) feature vectors in the second descriptors are found. The distance between two feature vectors is calculated as the Euclidean distance. If $D_1/D_2 > 1.5$ then this match is acceptable. Otherwise, it is ambiguously matched and rejected as a correspondence. If the match is acceptable, then the distance between two feature vectors is added to the global distance between two images (descriptors). To ensure that the calculated distance has symmetric property ($d(x,a) = d(a,x)$), all feature vectors in the second descriptor are matched in the same way to feature vectors in the first descriptor. Finally, the global distance between descriptors is normalized by the number of matches.

Histogram Based Distance. In this case the distance (similarity) between two images is measured by using the distance between histograms of the horizontal and vertical projections of the positions of the key points. The idea is depicted in figure 1.

The distance between two images is the sum of distances of corresponding histogram for both images. For calculating the distance between two histograms (A and B), the Bhattacharyya [10] formula is used.

$$D_B = \sum_{i=1}^N \sqrt{P(a_i)P(b_i)}, \quad P(a_i) = \frac{a_i}{\sum_{k=1}^N a_k}, \quad P(b_i) = \frac{b_i}{\sum_{k=1}^N b_k} \tag{7}$$

where a_i and b_i are the i -th elements of the A and B histograms respectively.

This formula was chosen because it exhibits many advantages over other metrics [11] (e. g. this distance can be used regardless of the type of distributions).

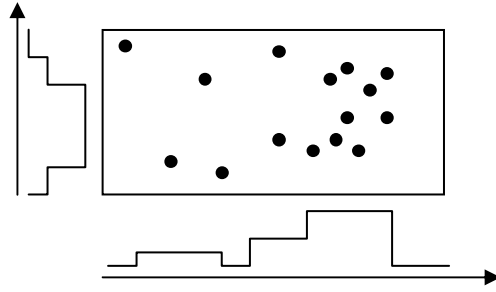


Fig. 1. Histograms of key points localization

Since SIFT descriptors provide not only location of key points but also corresponding scales, the modified distance was also used. The modification consists in taking into account only key points with the same scale parameter. This way, for a single image several histogram pairs are computed. Each pair corresponds to key points on a given scale.

Edge Histogram Distance. In case of the Edge Histogram, the distance between two descriptors is calculated according to the following formula [12]:

$$D = \sum_{i=0}^{79} |h_A(i) - h_B(i)| + 5 * \sum_{i=0}^4 |h_{Ag}(i) - h_{Bg}(i)| + \sum_{i=0}^{64} |h_{As}(i) - h_{Bs}(i)| \tag{8}$$

where h_A and h_B represent histogram values of image A and B respectively, h_{Ag} and h_{Bg} represent global edge histograms, h_{As} and h_{Bs} represent semi-global edge histograms. Both global and semi-global histograms are obtained from the local histograms.

Dominant Color Distance. In this case, the distance between two image descriptors $F_1 = \{ \{c_i, p_i\}, i=1 \dots N_1 \}$ and $F_2 = \{ \{b_j, q_j\}, j=1 \dots N_2 \}$ is defined [13]:

$$D(F_1, F_2) = 1 - \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} a_{i,j} S_{i,j}, \quad S_{i,j} = [1 - |p_i - q_j|] \times \min(p_i, q_j) \tag{9}$$

where

$$a_{i,j} = \begin{cases} 1 - d_{i,j} / d_{max} & d_{i,j} \leq T_d \\ 0 & d_{i,j} > T_d \end{cases} \quad d_{i,j} = \|c_i - b_j\| \tag{10}$$

The T_d is the maximum distance used to determine whether two clusters are similar.







4 Tests

In order to verify the effectiveness of the presented descriptors as well as the similarity measures, the following experiment was carried out. First, the data base was

created. It consists of 1200 test images downloaded from the Flickr website with the accompanying user-generated keywords. All images in the database were divided into semantic categories, for example: cars, offices, flowers, scenery, sunset and cell phone. Each category comprises 200 images.

For all images in the database, SIFT, C-SIFT and OpponentSIFT descriptors were calculated with default settings using the software provided by the authors of [14]. Edge Histogram and Dominant Color descriptors were calculated by using reference software provided by the mpeg-7 organization. In addition, a special program in C++ was developed which loads descriptors saved in files and performs the analysis by searching the most similar image to the given reference one. In order to determine which descriptors and which distances are most suitable for finding similar images, we have conducted an experiment whose goal was to measure the number of correctly identified similar images.. We have adopted the QbE approach which means that the most similar image is found on the basis of the input reference image rather than the description in a natural language. The result of searching the similar image is considered correct if the most similar image belongs to the same semantic category as the reference one. The obtained results are presented in tables 2 and 3.

Table 1. Examples of images categories

<p style="text-align: center;">flowers</p> 	<p style="text-align: center;">office</p> 	<p style="text-align: center;">scenery</p> 
<p style="text-align: center;">sunset</p> 	<p style="text-align: center;">car</p> 	<p style="text-align: center;">cell phone</p> 

In the case of histogram based similarity measure, it is necessary to determine the optimal number of bins. The idea is to measure the number of correct results with respect to the number of histogram bins. For this purpose, an experiment was carried out using key points calculated with the SIFT descriptor. The results are depicted in the figure 2.

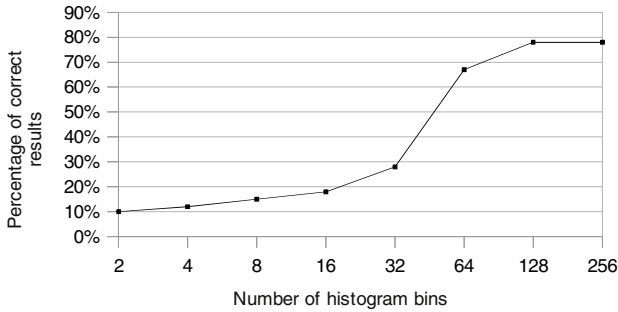


Fig. 2. Percentage of correct results with respect to number o histogram bins

Table 2. Results for color descriptors

Descriptor/similarity measure	Graph matching		Histogram based	
	Precision	Recall	Precision	Recall
SIFT	85,00%	83,50%	71,00%	68,40%
OpponentSIFT	90,50%	90,00%	75,30%	69,70%
C-SIFT	88,00%	86,50%	75,00%	68,90%

Table 3. Results for Edge Histogram and Dominant Color

Descriptor	Precision	Recall
Edge Histogram	90,80%	87,60%
Dominant Color	61,10%	59,30%

Table 2 and 3 contain averaged results. Two metrics (precision and recall) have been used in order to analyze the performance of the descriptors. The best results were obtained using the OpponentSIFT descriptor with graph matching as the similarity measure. This measure yields better results than histogram matching in case of all examined descriptors. Both OpponentSIFT and C-SIFT tend to outperform the standard SIFT. The reason is that C-SIFT and OpponentSIFT are invariant to some deviations of the image parameters (e. g. lightness). Edge Histogram achieves similar performance as the OpponentSIFT. The downside of the OpponentSIFT is that it is much more computationally expensive than the Edge Histogram. Another problem with the OpponentSIFT is that graph matching measure is also very constitutionally expensive in comparison to the distance formula used for the Edge Histogram descriptor. Dominant Color is characterized by the poorest performance because this descriptor extracts only dominant colors without their spatial distribution. This result confirms that for the efficient similar images retrieval it is necessary to calculate the

spatial distribution of edges or key points. Although the Dominant Color yields poor results on its own, it can be used in the planned development as a part of the hybrid method which is supposed to combine information extracted by various descriptors. The further work will be focused on improving graph matching by employing more sophisticated algorithms. The whole system can also be optimized in terms of speed execution by employing parallel programming techniques.

5 Conclusions

The paper addresses the issue of finding similar images using different sparse image descriptors as well as various measures of their similarity. At the beginning the color and edge descriptors used for features extraction as well as the methods of their comparison were described. The second goal was to determine the best algorithms as well as to select the ones especially pertinent for hardware acceleration. Finally, the description of the test and results are presented. The obtained results indicate that the Edge Histogram with the OpponentSIFT descriptor, combined with the graph distance, are the most effective for selecting the similar images.

The further work will be focused on implementing and testing the hybrid descriptor as well as hardware implementation of the SIFT descriptors. The hybrid method will aggregate information from different descriptors (e. g. OpponentSIFT, Dominant Color etc.). This method is expected to offer the end-user more options which can enable more customized search. Another aspect of the further development will include employing the hardware acceleration. It is especially important because calculating SIFT descriptors on a standard CPU can take a few minutes. Therefore, utilizing NVIDIA CUDA technology is planned to significantly reduce the descriptors calculation time. Another aspect is the execution speed of the graph matching algorithm. It is computationally expensive because it executes many floating point comparisons. However, this comparisons are independent and therefore the algorithm will be parallelized in the next updates.

Acknowledgment. This research was supported from the Polish funds for scientific research in the years 2010/2011 under the SYNAT project.

References

- [1] Mademlis, A., Daras, P., Tzovaras, D., Strintzis, M.G.: 3D volume watermarking using 3d krawtchouk moments. In: International Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (2007)
- [2] Hermes, T., Miene, A., Herzog, O.: Graphical Search for Images by Picture-Finder. Multimedia Tools and Applications. Special Issue on Multimedia Retrieval Algorithmics (2005)
- [3] Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2), 91–110 (2004)
- [4] Martinez, J.M., Koenen, R., Pereira, F.: MPEG-7: the generic multimedia content description standard. IEEE Multimedia 9(2), 78–87 (2002)

- [5] Frączek, R., Grega, M., Liebau, N., Leszczuk, M., Luedtke, A., Janowski, L., Papir, Z.: Ground-Truth-Less Comparison of Selected Content-Based Image Retrieval Measures. *LNICST*, vol. 40, pp. 101–108 (2010)
- [6] Geusebroek, J.M., Boomgaard, R., Smeulders, A.W.M., Geerts, H.: Color invariance. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 23(12), 1338–1350 (2001)
- [7] Burghouts, G.J., Geusebroek, J.M.: Performance evaluation of local color invariants. *Computer Vision and Image Understanding* 113, 48–62 (2009)
- [8] Manjunath, B.S., Salembier, P., Sikora, T.: *Introduction to MPEG-7: Multimedia Content Description Interface*. John Wiley and Sons Ltd., Chichester (2002)
- [9] Gersho, A., Gray, R.M.: *Vector Quantization and Signal Compression*. Kluwer Academic Publishers (1992)
- [10] Cyganek, B.: *Methods and Algorithms of Objects Recognition in Digital Images*. AGH University of Science and Technology Press, Krakow (2009)
- [11] Aherne, F.J., Thacker, N.A., Rockett, P.I.: The Bhattacharyya Metric as an Absolute Similarity Measure for Frequency Coded Data. *Kybernetika* 34(4), 363–368 (1998)
- [12] Chee, S.W., Dong, K.P., Soo-Jun, P.: Efficient Use of mpeg-7 Edge Histogram Descriptor. *ETRI Journal* 24(1) (2002)
- [13] Yang, N.C., Kuo, C.M., Chang, W.H., Lee, T.H.: A Fast Method for Dominant Color Descriptor with New Similarity Measure. *Journal of Visual Communication and Image Representation* 19(2), 92–105 (2008)
- [14] van de Sande, K.E.A., Gevers, T., Snoek, C.G.M.: Evaluating Color Descriptors for Object and Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32(9), 1582–1596 (2010)

Online Sound Restoration for Digital Library Applications

Andrzej Czyżewski, Adam Kupryjanow, and Bożena Kostek

Gdansk University of Technology, Multimedia Systems Department,
Narutowicza 11/12, 80-233 Gdansk, Poland
{andcz, adamq, bozenka}@sound.eti.pg.gda.pl

Abstract. In this paper, a sound restoration system conceived and engineered at the Multimedia Systems Department of the Gdansk University of Technology is discussed with regard to the principles of its design, features of operation and the achieved results. The system has been designed so that: no special sound restoration software is needed to perform audio restoration; no skills in digital signal processing are required from the user; the process of online restoration employs automatic reduction of noise, wow and impulse distortions.

Keywords: Automatic audio restoration, noise reduction, wow and flutter, impulsive distortions.

1 Introduction

There is a widely supported claim that audio archives need to be explored faster, restored more easily and preserved with greater efficiency. Moreover, currently recorded or transmitted sound is quite often so much degraded that instant improvement to this negative aspect is seen by many digital library users as a forthcoming must. However, new methods of automatic sound restoration based on artificial intelligence and some selected techniques known from digital communication enabled to create an audio restoration system which addresses all the limitations of today's solutions. In particular, these systems expose users to the following disadvantages:

- users need a licensed software package installed locally on their computers,
- all audio material to be restored must be copied to a local hard disk,
- sound restoration usually is a burdensome and time-consuming process,
- effective sound restoration requires knowledge, experience and skills from the users.

The proposed system overcomes the above drawbacks by offering the following features:

- the cleaning software necessary to perform audio restoration resides on a master server, and thus:
 - restoration may take place from almost any localization in the world,

- restoration takes the same time, no matter whether the audio material is copied to a local or to a remote hard disk; both versions of the signal – the original and the restored one – are archived independently;
- there are two modes of restoration:
 - fully automatic *online* reduction of noise, “wow” (parasite frequency modulation) and impulse distortions (no feedback monitoring necessary),
 - semi-automatic mode – requiring some input from the user – usually through setting up a few virtual controls to the user’s subjective preferences (feedback monitoring needed);
- no skills are required from the user in the fully automatic mode, and limited knowledge on digital signal processing principles is sufficient in the semi-automatic mode.

Due to their present limitations, sound acquisition, transmission, restoration and archiving systems have required an innovative response from researchers, professional sound engineers, musicians, radio and TV broadcasters and private persons. It may be expected that the demand for a straightforward access to sound restoration and archives enhances along with the European integration. In turn, as regards the material outcomes or tangible results, the system could provide means for a trans-European information infrastructure supporting collection, searching and navigation, processing and publishing the relevant information on audio.

2 Online Sound Restoration System Concept and Principles

The on-line sound restoration system that is being developed at the Multimedia Systems Department (MSD) provides a combination of a typical digital library software extended with the automatic sound restoration algorithms [1]. Owing to the website service, it is possible to upload and to restore or just to watch/download the archival audio remotely via the Internet. Consequently, the system introduces the possibility of uploading of end-users’ audio and accompanying information, and also offers a remote restoration of those uploaded recordings according to the “on-demand audio restoration” concept. It is worth noticing in this context that the archive resources of the online database include also speech recordings whereas the digital archives proposed so far are oriented mostly at storing music. This is because radio and TV stations and other audio publishers are envisaged among the end-users, so that reportage, political speeches, reports, drama performances etc. will be as much important as the musical content of the audio archive. A variety of sound sources were studied in context of audio restoration, namely old records, optical film and magnetic tracks, magnetic tapes and life microphone recordings, vinyl records.

3 Speech Signal Processing Algorithms

Many archive recordings contain only speech signal e.g. speeches of politicians, actors, musicians or movies dialogues. Since the speech frequency characteristics

are highly different from the music characteristics, especially designed algorithms should be used for the purposes of speech reconstruction. In this Section some basic speech signal restoration algorithms were presented. Those are among others: noise whitening and spectral expander. All of them modify the signal spectra in order to reduce the noise and other distortions influence on the recorded speech intelligibility. The presented algorithms could work in the automatic or semi-automatic mode.

3.1 Noise Whitening (Speech Restoration Application)

The acoustic noise, and other additive distortions usually have non-flat amplitude spectrum. The noise whitening algorithm equalizes the spectrum of the signal, making it similar to the white noise spectrum. Noise whitening algorithm works similarly to the automatic filter that enhances low level spectral components and attenuates high level ones. Additionally, the so-called de-emphasis operation is applied after whitening. This means that high frequencies are attenuated according to a rule, so that spectrum of processed signal is similar to the spectrum of speech signal, which has low energy in the high frequency range. Whitening and de-emphasis enhance quality of the speech signal. This algorithm is especially useful if constant “whistling” noise is present in the recording.

The algorithm starts with finding the part of the recording that contains only noise and no speech signal. This part is segmented and the spectrum is calculated for each segment. The noise estimate is calculated as a smoothed (low-pass filtered) noise spectrum averaged in all segments. This estimate is inversed and used in restoration of the speech signal. The block diagram of the whitening algorithm is shown in Fig. 1.

The averaging of the noise spectrum is described by Equation 1:

$$S_{xx}(\omega) = \sum_{m=-\infty}^{\infty} R_{xx}(m)e^{-j\omega m} \tag{1}$$

Equation 1 represents the power spectrum estimate. The power spectrum of the stationary random signal $x(n)$ is related to the autocorrelation by the discrete Fourier transform. The autocorrelation of the signal can be calculated as the inverted discrete Fourier transform expressed as:

$$R_{xx}(\omega) = \int_{-\pi}^{\pi} \frac{S_{xx}(\omega)e^{-j\omega m}}{2\pi} d\omega \tag{2}$$

Average signal power $x(n)$, limited by the Nyquist frequency, is given by:

$$R_{xx}(0) = \int_{-\pi}^{\pi} \frac{S_{xx}(\omega)}{2\pi} d\omega \tag{3}$$

where the power spectral density (PSD) of the $x(n)$ is defined as:

$$P_{xx}(\omega) = \frac{S_{xx}(\omega)}{2\pi} \tag{4}$$

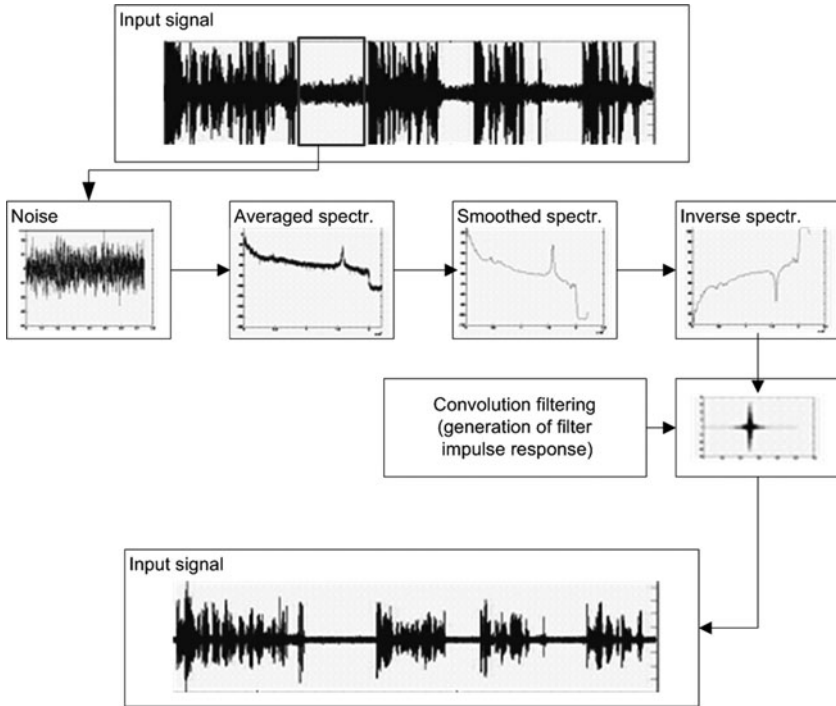


Fig. 1. Block diagram of the whitening algorithm

Practically, the averaged spectrum is obtained by calculating the spectra of the segmented signal, and segments may overlap and they may be time-weighted using windows. The averaged signal spectrum is the average spectral energy calculated from separate segments. The spectrum of each segment is calculated using the discrete cosine transform (DCT). The averaged spectrum is additionally smoothed using the moving average filter in the cepstrum domain (cepstrum liftering) in order to avoid extreme variations of the spectrum magnitude.

3.2 Spectral Expander

The spectral expander algorithm enhances the signal intelligibility by increasing the difference between the speech signal level and the level of noise and distortions. It is assumed here that the noise is stationary. The algorithm works by comparing the values of the observed noise level (in the noisy recordings) and the reference level (in the noise pattern) in a number of frequency bands.

The concept of this algorithm is similar to the spectral subtraction method [23]. The average spectrum of the signal is used to obtain the correction function. Two expansion points that control the shape of the correction function provide the input parameters of the algorithm: the 0 Hz point and the Nyquist frequency point. The threshold function has the following form:

$$f(n) = \frac{(N - n) \cdot a + n \cdot b}{N} \cdot y(n) \quad (5)$$

where $y(n)$ is the average noise spectrum, N denotes the sample index for the Nyquist frequency, a defines the expansion coefficient for 0 Hz, and b – the expansion coefficient for the Nyquist frequency.

The spectral expansion algorithm is based on the assumption that the speech signal level is much greater than the noise level and it is possible to determine the threshold level that will enable noise reduction. The power spectrum of the signal is computed using the FFT algorithm. Spectral components having level surpassing the threshold value are not modified while those of magnitudes remaining below the threshold are processed using the expansion transform expressed by Eqs. 6 and 7:

$$\operatorname{Re}\{v(n)\} = \frac{|x(n)|}{|f(n)|} \cdot \operatorname{Re}\{x(n)\}, \quad \operatorname{Im}\{v(n)\} = \frac{|x(n)|}{|f(n)|} \cdot \operatorname{Im}\{x(n)\}, \quad (6)$$

for $|x(n)| < |f(n)|$

$$\operatorname{Re}\{v(n)\} = \operatorname{Re}\{x(n)\}, \quad \operatorname{Im}\{v(n)\} = \operatorname{Im}\{x(n)\} \quad \text{for } |x(n)| \geq |f(n)| \quad (7)$$

where: $x(n)$ – source signal, $v(n)$ – processed signal, $f(n)$ – threshold function.

If the spectral components with level below the threshold were simply removed, this would result in the discontinuous spectrum. The expansion procedure described above ensures the continuity of spectrum. If the second-order threshold function is used, the dynamic range is doubled (e.g. increase from 6 dB to 12 dB after the expansion is achieved).

The average noise spectrum is by default computed from the noise pattern which should be selected from the recording by the user. The expansion coefficients could be set from range 0–100%. Optionally, a linear threshold function may be used instead of the noise pattern, provided the reconstruction procedure is to be done in the full automatic mode. The expansion coefficient value which is equal to 0% means no signal modification, larger values mean that more distortions are removed from the signal, but at the same time the signal quality may be deteriorated. The choice of expansion coefficient values should make a compromise between noise level and speech intelligibility.

4 Audio Signal Processing Algorithms

Typical distortions that occur in many music archival recordings are: noise, impulsive distortions (clicking), parasitic frequency modulation (the “wow” distortion) and signal clipping. In this Section some selected algorithms allowing automatic reduction of those distortions are presented.

4.1 Automatic Noise Reduction

The automatic noise reduction (ANR) algorithm (Fig. 2) described here is composed of three sub-algorithms: noise detection, whitening and spectral

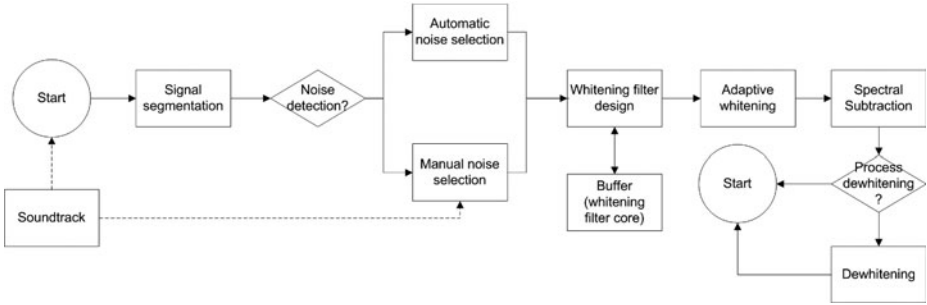


Fig. 2. Scheme of automatic noise reduction algorithm

subtraction [4,5,6,7,8]. The following Section presents the idea of the ANR and describes its steps.

Noise Detection

The noise detection algorithm is based on frequency domain criteria: average energy, concentration of spectrum energy and relative flatness of the spectrum in consecutive segments.

Energy Criterion

The criterion is based on a comparison of the modified average energy (*MAE*) of the spectrum of each segment in a frame (Eq. 8):

$$MAE_i = \begin{cases} \frac{2}{3}AE_i + \frac{1}{3}AE_{i+1}, & i = 0 \\ 0.5AE_i + 0.25(AE_{i-1} + AE_{i+1}), & i = 2, \dots, l-2 \\ \frac{2}{3}AE_i + \frac{1}{3}AE_{i-1}, & i = l-1 \end{cases} \quad (8)$$

where: $AE_i = \frac{1}{K} \sum_{k=0}^{K-1} |X_i[k]|^2$.

Segments, the values of which are lower than the threshold, defined as:

$$Th_{MAE} = \min(MAE) + \sigma(MAE) \quad (9)$$

are selected as representing the noise-print for further evaluation. Fig. 3 shows an example of noise (Fig. 3a) and of sound (Fig. 3b) segments along with the threshold (dotted line) for a single frame.

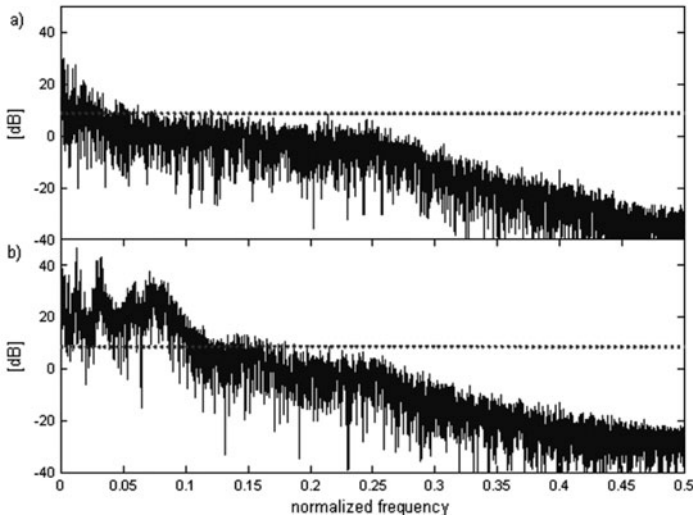


Fig. 3. Energy criteria evaluation: the noise segment spectrum (a) and sound segment spectrum (b). Th_{MAE} threshold marked as a dotted line

Concentration of Spectral Energy

The evaluation of the spectral energy concentration is a step parallel to the energy evaluation criterion calculating. The average magnitude of each segment spectrum AM_i is calculated according to the formula [10]:

$$AM_i = \sum_{k=0}^{K-1} |X_i[k]| \tag{10}$$

then the mean value of it is computed. The spectrum concentration coefficient (SCC) is equal to the number of spectral lines whose magnitude value is not lower than mean value of AM . Segments whose SCC values are lower than the average are considered to represent noise segments.

Flatness of Spectrum

Both algorithms discussed previously can detect noise segments with acceptable correctness, however there are no standard values of thresholds qualifying a segment as noise/sound and the presence of noise segment in each signal frame is no guaranteed. Therefore, the third algorithm checking the flatness of segment spectrum is used. The spectrum flatness measure (SFM) is defined as follows:

$$SFM_i = 10 \log \left(\frac{\left(\prod_{k=0}^{K/2-1} P_i[k] \right)^{2/k}}{\frac{2}{K} \sum_{k=0}^{K/2-1} P_i[k]} \right) \tag{11}$$

where $P_i[k]$ is the power spectral density of the i -th segment. As the *SFM* tells only as how flat the spectrum is, the higher value of *SFM*, the flatter spectrum is. Therefore a relative *SFM* (*RSFM*) was used as a measure of difference between sound and noise segments. The *RSFM* is given by:

$$RSFM_i = SFM_i - REFSMPF \quad (12)$$

where *REFSMFM* is the reference spectrum flatness measure, which is equal to *SFM* of averaged spectra of segments whose *SFM* is higher than the average *MAE*. The default threshold, experimentally determined, deciding whether the segment represents the noise-print or not, is set to 2.5 [dB].

Adaptive Whitening Algorithm

In the ideal case designing the noise whitening filter means finding the filter inversed to the coloring one. It is assumed that the received signal passed the coloring filter before it is restored. In practice, it is only possible to estimate the coloring filter characteristics using a fragment of signal which fulfills the following condition:

$$x[n] \equiv 0, \quad n = i, \dots, j, \quad \text{where } i, j \in \mathbf{Z}, \quad i < j \quad (13)$$

In effect the signal containing only noise, called the noise-print – $NP[n]$, is obtained. As the ideal magnitude characteristic of white noise is equal to $|S_w(\omega)| = A_0 1(\omega)$ where A_0 is a magnitude of white noise, $1(\omega)$ is a function constant for all ω , and the filtering operation in the frequency domain is defined as:

$$|S_c(\omega)| = |S_w(\omega)| |H_c(\omega)| \quad (14)$$

$H_c(\omega)$ is the frequency response of the coloring filter. It can be assumed that the magnitude characteristic of the noise-print is equal to the magnitude characteristics of coloring filter. In practice, as the operation is performed on the finite signal and the spectrum of white noise is not constant for the full range of ω (it has frequency-dependent $\sigma(\omega)$ deviation), the characteristic of coloring filter has to be estimated in order to design a whitening filter.

In the whitening filter design algorithm the two sub-algorithms estimating the magnitude characteristic of the filter were used. The first one is based on averaging of the noise segment characteristics. The algorithm uses the Chebyshev window of 80 dB attenuation and the fixed length $L = 51$ for averaging and the Savitzky-Golay filter for additional smoothing. The second algorithm is an iterative Empirical Mode Decomposition (EMD) algorithm. The algorithm is generally used for decomposition of the signal in the time domain into high and low speed components, but here it proves its usefulness for estimating characteristic of the filter. The last stage of the filter design is regulation of the filter magnification. The gain of the filter is defined as:

$$A_0 = \frac{\sum_{k=0}^{K-1} |NP[k]|^2}{\sum_{k=0}^{K-1} |H_w[k] NP[k]|^2} \quad (15)$$

where $NP[k]$ is the DFT of the noise-print and $H_w[k]$ is DFT of the whitening filter.

Two basic scenarios can be defined when using the whitening algorithm for audio signals. In the first case it is assumed that the signal and the additive white noise are colored with the same filter, in the second scenario, differently to the previous assumption, it is considered that the signal is distorted with additive color noise. In both scenarios the process of noise whitening can be considered as the application of the whitening filter $h_w[n]$ which is, in the ideal case, the inversion of the coloring filter $h_c[n]$.

$$h_c[n] * h_w[n] = \delta[n] \tag{16}$$

The idea of using the spectral subtraction algorithm for signal denoising was introduced in the literature [2],[3]. The proposed algorithm is based on the standard denoising algorithm, but the spectral subtraction is preceded by the whitening operation. The spectral subtraction provides inefficient method for the denoising purpose, especially if white noise spectra (their magnitudes) differ considerably from frame to frame. To increase efficiency of spectral subtraction a modification of the whitening algorithm was introduced to match the spectra as much as possible and simultaneously avoid adding distortions into signal. The whitening algorithm is presented in Fig. 4.

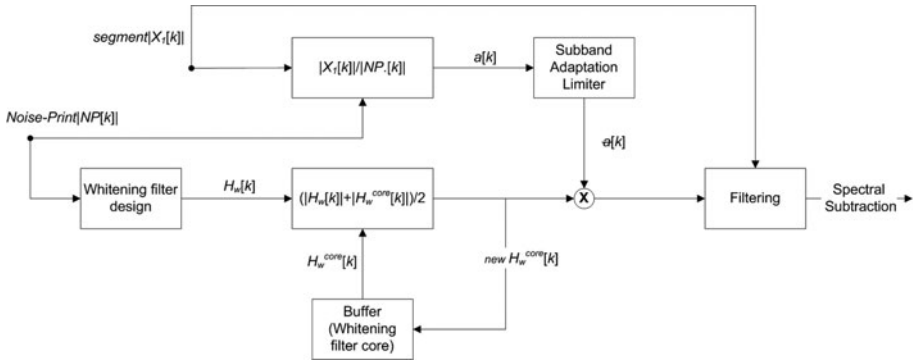


Fig. 4. Adaptive whitening algorithm

The adaptation of whitening filter is performed in two steps. The first one is modification of existing whitening filter core in the case the noise-print in the current frame is detected, the second modifies the filter characteristics. The spectral subtraction algorithm is widely used for removing noise from the signal. The basic idea of the spectral subtraction operation is that using the noise-print allows for reducing the level of noise in every segment of the signal. The operation of spectral subtraction can be defined as follows:

$$\forall_{i \in \mathfrak{S}} Y_i[k] = (|X_i[k]| - |NP_i[k]|) e^{j\varphi[k]} \quad (17)$$

where $\varphi[k] = \arg(X[k])$.

As the spectral magnitudes of components cannot be negative, several approaches are used to eliminate a negative result of spectral subtraction. The most common approach sets value of spectral line to 0 if the difference comes out negative. The spectral subtraction algorithm that provides a part of the adaptive whitening algorithm was implemented in the way described above.

4.2 Impulse Distortions Removing

Removing impulse distortions, also called declicking of the recorded sound, is performed in two steps. In the first step impulse distortions – clicks – are detected within a signal, which are going to be removed from the signal in the second step [9]. Both stages are performed using artificial neural networks. The click detector searches the signal for clicks and marks up distorted fragments. For this task a neural dichotomizer was used. There are only two output classes for the classifier; one for undistorted and one for distorted signal. Therefore, the network output layer can consist of only one unit. For the purpose of training a set of samples was selected. According to the presented assumption an error measure can be defined as follows:

$$e = \sum_{i=1}^m (y_i - z_1)^2 + \sum_{j=1}^n (y_j - z_2)^2 \quad (18)$$

where: e – mean-square error, m – number of non-distorted samples (class 1), n – number of distorted samples (class 2), y_i – output value for the i -th non-distorted sample, y_j – output value for the j -th distorted sample, z_1 – expected output value for non-distorted samples (-1), z_2 – expected output value for distorted samples ($+1$). The error measure is minimized during training.

For the purpose of the distorted sample reconstruction a non-linear neural predictor was used. The predictor processes a signal in both directions: forward and backward. A predicted sample is added to the signal and the prediction is repeated until the whole range of distorted signal is reconstructed. The neural predictor was trained using a wide variety of undistorted music and speech samples. During the training network minimizes error measure defined as follows:

$$e = \sum_{i=1}^n (y_i - p_i)^2 \quad (19)$$

where: e – mean-square error, n – number of examples, y_i – output value for the i -th example, p_i – training sample value i -th example. During reconstruction neural predictor generates two series of estimated samples using forward and backward prediction. This process can be described as the prediction function $x_i = NP^+ \{x, r\}$, non-linearly predicting sample x_i using r previous samples of indexes: $i-1, i-2, \dots, i-r$, where r is the prediction order equal to the number

of network input units. Therefore sample series x_n as a result of a forward prediction can be denoted as follows:

$$x_n = NP^+ \{x, r\} \tag{20}$$

A result of the backward prediction, sample series y_n , can be denoted as follows:

$$y_n = NP^- \{x, r\} \tag{21}$$

where: $n = j, j + 1, j + 2, \dots, k, j, k -$ markers of a distorted range. Thereafter, both series are weighted and summarized according to the following formula:

$$z_n = 0.5 \left[\left(1 + \cos \left(\frac{n-1}{k-j} \pi \right) x_n \right) + \left(1 - \cos \left(\frac{n-j}{k-j} \pi \right) y_n \right) \right] \tag{22}$$

where: $n = j, j+1, j+2, \dots, k$.

The neural predictor can also be used as a click detector. Detection of clicks is performed by tracking the generated prediction error signal. The neural predictor was trained using only undistorted samples, therefore while tracking distorted signals an error increases significantly on distorted samples. Obtained results prove, that developed declicking methods can successfully be applied as a sound restoration tool.

4.3 Wow Distortion Reduction

Wow distortion often appears in many archival recordings. It is perceived as an undesired frequency modulation (FM). The range of FM for wow includes between 0.5 to 6 Hz [10,11,12]. The distortion could be introduced to the recording by the irregular velocity of the analog medium (tape, vinyl, wax cylinder) or shrinkage of the tape (e.g. movie tape). Typically a wow reduction algorithm is a combination of the wow characteristic determination algorithm and non-uniform audio signal resampling. Distortion characteristic is described by the pitch variation curve (*PVC*) as:

$$PVC(t_{org}) = \frac{d[f_w(t_{org})]}{dt_{org}} \tag{23}$$

where $f_w(t_{org})$ is the time warping function which transforms original time axis t_{org} to the distorted time t_{wow} axis. If the pitch is constant, then the $PVC(t_{org}) = 1$. Other values of the *PVC* signify wow modulation.

Many techniques enable *PVC* determination were proposed in the literature [13,14,15,16,17,18,19,20]. The idea of those algorithms is to analyze changes of the audio signal component frequency. Signal components that are used in the tracking process are: pilot tones (high frequency bias), power line hum, and audio signal tonal components. It is assumed that the frequency of those components was constant in the original recording, so that all detected changes are caused by the wow distortion. Other group of algorithms were designed in order to work

with the movie tape images. These methods assumed that the size of the movie tape elements such as frame height, perforation hole size and distance between successive perforation holes, are constant and the changes of the size detected in the current image defines the *PVC* [21,22].

At the Multimedia Systems Department many innovative algorithms for the *PVC* determination were developed [23,24,25,26,27,28], but not all of them allow for automatic audio signal restoration. The algorithm that is almost fully automatic and could be used for nearly every recording is the power line hum tracker. Therefore, the hum tracker could be used in the online sound restoration digital library subsystem. In order to overcome DFT constraints in the analysis time resolution, the algorithm was based on the autoregressive method. In the Fig. 5 a block diagram of the hum tracking algorithm is presented. Since the algorithm has to be resilient to noise and other distortions, thus at the beginning of the algorithm the preprocessing step is carried out. To minimize the computational complexity and the order of the autoregressive model the input signal is down-sampled. Downsampling causes also elimination of non-hum tonal components which could generate false detection errors. In the next step, in order to reduce noise level, the signal is bandpass filtered. After the preprocessing the hum frequency is tracked using an autoregressive model. This step is described in details in the earlier paper [29]. At the end the *PVC* is smoothed using the median and the moving average filters.

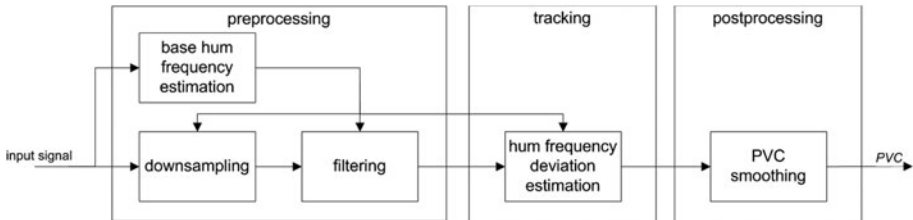


Fig. 5. Block schema of the hum tracking algorithm

5 User Interface

To facilitate the automatic recording restoration, a special Web service was designed. It enables a user to find recordings uploaded by other users (Fig. 6), and to upload their own audio recordings (Fig. 7). A quality assessment (distortions detection) and reconstruction process will be performed using the algorithms described in Sections 3 and 4 during the upload process. The idea of the automatic recording technical quality assessment and reconstruction is shown in the block diagram, presented in Fig. 8. Since the restoration algorithms integration with the website is not fully completed yet, therefore currently only sample original recordings could be found using the website.

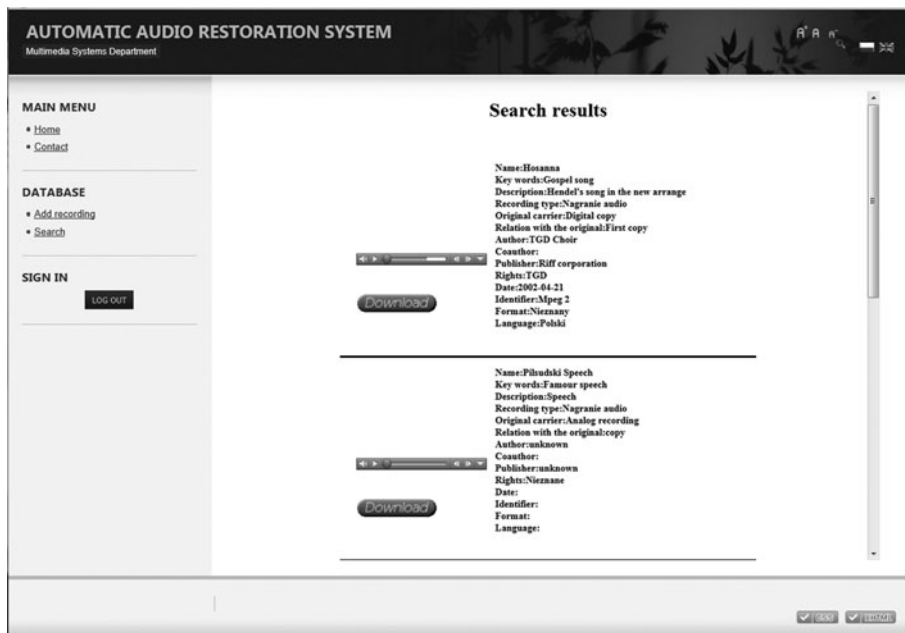


Fig. 6. Web service GUI – results of audio searching in the online repository

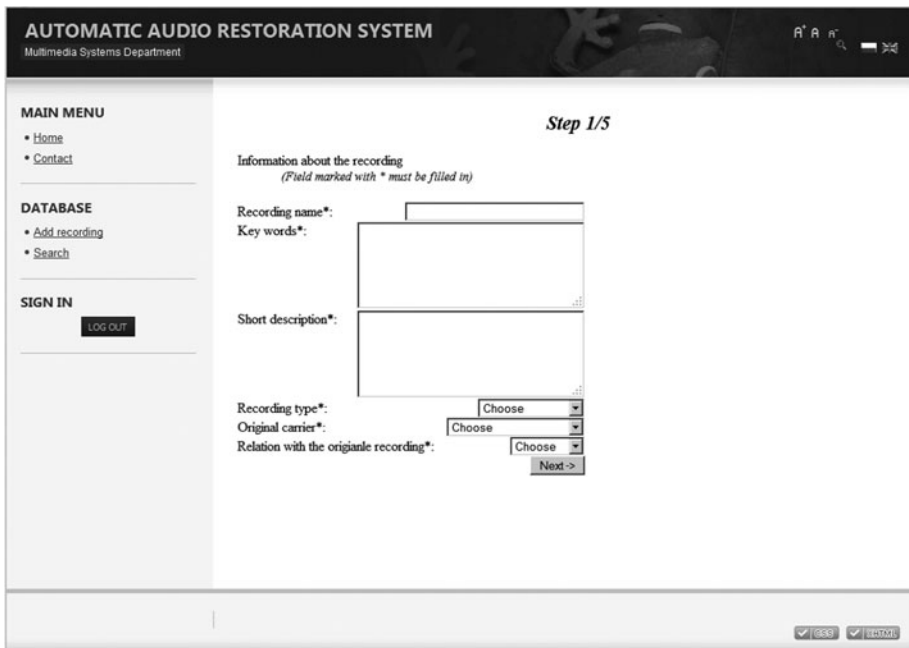


Fig. 7. Web service GUI – form for the archival recording upload

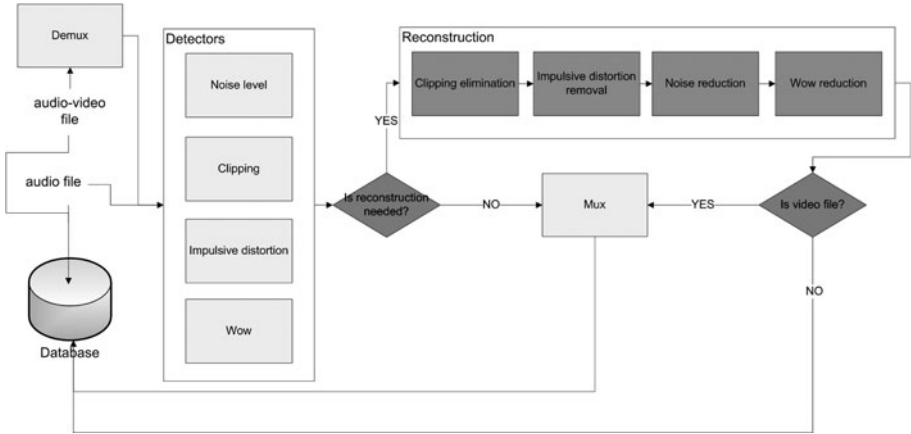


Fig. 8. Block diagram of the digital online library

As is seen in Fig. 8, in the first step of processing, all original files are saved into a database. Those files are available for future use e.g. when more algorithms for quality assessment will be added or in case the process of the distortion reduction will not be of satisfactory quality.

In the next step distortion detectors are used to assess quality of the recording. The detectors analyze noise level, clipping occurrence, impulsive distortion location and they estimate the *PVC*. The restoration is carried out according to the distortion assessment and the distortion thresholds set by the user. Every type of distortion has its own measure threshold initiating the restoration process. The restoration is performed in the following order: clipping elimination, impulsive distortion removal, noise reduction, wow reduction.

6 Conclusions

The presented online sound restoration digital library subsystem is still under construction. However, all presented algorithms were tested and high quality of the restoration results was obtained with their application. Consequently, it is expected that the final effect of this work will result in satisfactory quality, easy to use and universally accessible tool for multimedia digital libraries.

Acknowledgments. The research is supported within the project No. SP/I/1/77065/10 entitled: “Creation of universal, open, repository platform for hosting and communication of networked resources of knowledge for science, education and open society of knowledge”, being a part of Strategic Research Programme “Interdisciplinary system of interactive scientific and technical information” funded by the National Centre for Research and Development (NCBiR, Poland).

References

1. Kupryjanow, A., Czyżewski, A.: Automatic system for audio-video material reconstruction and archiving; NTAV/SPA, Poznań, Polska, pp. 173–176 (2008) (in Polish)
2. Kamath, S.D., Loizou, P.C.: A Multi-Band Spectral Subtraction Method For Enhancing Speech Corrupted By Colored Noise. In: Proceedings of ICASSP-2002, Orlando, FL (May 2002)
3. Yektaeian, M., Amirfattahi, R.: Comparison of Spectral Subtraction Methods used in Noise Suppression Algorithms. In: ICICS (2007)
4. Czyżewski, A., Królikowski, R.: Neuro-Rough Control of Masking Thresholds for Audio Signal Enhancement. *Journal of Neurocomputing* 36, 5–27 (2001)
5. Czyżewski, A., Królikowski, R.: Noise Reduction in Audio Signals Based on the Perceptual Coding Approach. In: Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, New Paltz, NY, USA, October 17–20, pp. 147–150
6. Czyżewski, A., Królikowski, R.: Noise Reduction in Audio Employing Auditory Masking Approach. In: Proc. 106th Audio Engineering Society Convention
7. Królikowski, R., Czyżewski, A.: Noise reduction in telecommunication channels using rough sets and neural networks. In: Zhong, N., Skowron, A., Ohsuga, S. (eds.) RSPFDGrC 1999. LNCS (LNAI), vol. 1711, pp. 100–108. Springer, Heidelberg (1999)
8. Królikowski, R.: Exploitation of Self-Organising Maps for the Reduction of Non-Stationary Noise in Speech Signals. In: CD-ROM Proceedings of ICSC (International Computer Science Conventions) Neural Computation 2000, May 23–26, Berlin, Germany (2000)
9. Czyżewski, A.: Learning Algorithms for Audio Signal Enhancement - part I: Neural Networks Implementation for the removal of Impulse Distortions. *Journal of Audio Engineering Society* (1997)
10. Daniel, E.D., Mee, C.D., Clark, M.H.: *Magnetic Recording: The First 100 Years*. IEEE Press, New York (1999)
11. Read, P., Meyer, M.P.: *Restoration of Motion Picture Film*. Butterworth Heinemann, Oxford (2000)
12. Busby, E.S.: Analog Tape Recording. In: Whitaker, J., Benson, B.K. (eds.) *Standard Handbook of Audio and Radio Engineering*, pp. 6–21–6–42. McGraw-Hill, New York (2003)
13. Gerzon, M.: *Don't Destroy the Archives!* (1992) (unpublished), http://cec.concordia.ca/education/archive/10_x/gerzon_archive.html (accessed April 2011)
14. Godsill, S.J., Rayner, P.: The Restoration of Pitch Variation Defects in Gramophone Recordings. In: Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, Mohonk, NY, October 17–20, pp. 148–151 (1993)
15. Godsill, S.J.: *The Restoration of Degraded Audio Signals*. Ph.D. dissertation, Cambridge University, Cambridge, UK (1993)
16. Godsill, S.J.: Recursive Restoration of Pitch Variation Defects in Musical Recordings. In: Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing 2, Adelaide, Australia, April 19–22, pp. 233–236 (1994)
17. Godsill, S.J., Rayner, P.J.W.: Restoration of Pitch Variation Defects. In: *Digital Audio Restoration—A Statistical Model-Based Approach*, pp. 171–190. Springer, London (1998)
18. Godsill, S.J., Reid, G., Hicks, C.: Restoration of Smooth Pitch Variations over Long Timescales. In: *Joint Technical Symposium*, Toronto, Canada, June 24–26 (2004)

19. Nichols, J.: An Interactive Pitch Defect Correction System for Archival Audio. In: Proc. AES 20th Int. Conf., Budapest, Hungary, October 5–7 (2001); paper 1941
20. Nichols, J.: High Performance, Low-Cost Wax Cylinder Transcription System. In: Proc. AES 20th Int. Conf., Budapest, Hungary, October 5–7 (2001); paper 1937
21. Maziewski, P., Kupryjanow, A., Czyzewski, A.: Drift, wow and flutter measurement and reduction in shrunken movie soundtracks. In: 24th AES Convention Amsterdam, Netherlands, Amsterdam, May 17-20 (2008)
22. Czyzewski, A., Maziewski, P., Kupryjanow, A.: Reduction of Parasitic Pitch Variations in Archival Musical Recordings. In: Signal Processing. Special Issue of Signal Processing: Ethnic Music Restoration, April 2000, vol. 90(4), pp. 981–990 (2010)
23. Czyzewski, A., Maziewski, P., Dziubinski, M., Kaczmarek, A., Kostek, B.: Wow Detection and Compensation Employing Spectral Processing of Audio, 117 Convention of the Audio Engineering Society. J. Audio Eng. Soc. 53, 90 (2005); convention paper 6212
24. Czyzewski, A., Dziubinski, M., Ciarkowski, A., Kulesza, M., Maziewski, P., Kotus, J.: New Algorithms for Wow and Flutter Detection and Compensation in Audio, 118 Convention of the Audio Engineering Society. J. Audio Eng. Soc. 53, 669 (2005); convention paper 6353
25. Czyzewski, A., Maziewski, P., Dziubinski, M., Kaczmarek, A., Kulesza, M., Ciarkowski, A.: Methods for Detection and Removal of Parasitic Frequency Modulation in Audio Recordings. In: Proc. AES 26th Int. Conf., Denver, CO, July 7–9 (2005); paper 3-3
26. Litwic, L., Maziewski, P.: Evaluation of Wow Defects Based on Tonal Components Detection and Tracking. In: Proc. XI Int. AES Symp. of Sound Engineering and Tonmeistering, Polish Section, Krakow, Poland, June 23–25, pp. 145–150 (2005)
27. Maziewski, P.: Wow Defect Reduction Based on Interpolation Techniques. In: Proc. 4th Nat. Electronics Conf., Darlowko Wschodnie, Poland, June 12–15, vol. 1/2, pp. 481–486 (2005)
28. Maziewski, P., Litwic, L., Czyzewski, A.: Accidental Wow Evaluation Based on Sinusoidal Modeling and Neural Nets Prediction, 120 Convention of the Audio Engineering Society. J. Audio Eng. Soc. 54, 709 (2006); Convention paper 6769
29. Czyzewski, A., Ciarkowski, A., Kaczmarek, A., Kotus, J., Kulesza, M., Maziewski, P.: DSP Techniques for Determining “Wow” Distortion. Journal of the Audio Engineering Society 55(4), 266–284 (2007)

Connectionist Language Model for Polish

Lukasz Brocki, Krzysztof Marasek, and Danijel Koržinek

Polish-Japanese Institute of Information Technology
Koszykowa 86, 02-008 Warszawa, Poland

Abstract. This article describes a connectionist language model, which may be used as an alternative to the well known n-gram models. A comparison experiment between n-gram and connectionist language models is performed on a Polish text corpus. Statistical language modeling is based on estimating a joint probability function of a sequence of words in a given language. This task is made problematic due to a phenomenon known commonly as the “curse of dimensionality”. This occurs because the sequence of words used to test the model is most likely going to be different from anything present in the training data. Classic solutions to this problem are successfully achieved by using n-grams which generalize the data by concatenating short overlapping word sequences gathered from the training data. Connections models, however, can accomplish this by learning a distributed representation for words. They can simultaneously learn both the distributed representation for each word in the dictionary as well as the synaptic weights used for modeling the joint probability of word sequences. Generalization can be obtained thanks to the fact that if a sequence is made up of words that were already seen, it will receive a higher probability than an unseen sequence of words. In the experiments, perplexity is used as measure of language model quality.

Keywords: statistical language model, neural networks, multilayer perceptron.

1 Introduction

Statistical language modeling is especially important for automatic speech recognition and statistical machine translation where a language model is a crucial component for searching in the excessively large hypothesis space. The literature on the topic and the following paper describe a novel, connectionist language model which, in case of English, achieved better results than traditional, statistical n-gram models. The purpose of language modeling is to reduce the search space of the speech recognizer (decoder) by allowing only syntactically and semantically correct word sequences to be decoded. This is commonly achieved by two methods: formal grammars and statistical language models.

Grammars explicitly define which sequences are allowed. Algorithmically, they are represented by a Finite State Automaton, which is defined using a formal description often written in the Bachus-Naur Form (BNF) or some of its derivatives. Statistical language models offer greater flexibility than formal grammars, but at an increased uncertainty of the outcome.

The task of the language model is to assign a probability to all the words in a dictionary depending on the previous words in a sequence. Instead of defining exactly which word sequences are allowed, they estimate the probability of a word sequence using a model trained on a large textual corpus. Usually only a short historical context is used, i.e. the model estimates only $P(w_t|w_{t-1})$ for bigram and $P(w_t|w_{t-1}, w_{t-2})$ for trigram language models. Given the sheer amount of words in a dictionary for any reasonable application of this model (at least several tens of thousands of words), even with such a small historical context, these models are usually very large. Because of this, the corpora used for training has to be unfeasibly large to make the model statistically significant and to deal with this problem many heuristics and tricks have to be used to make the model work well [9].

The goal of the work is to compare if the results with regards to connectionist language models shown in the literature apply equally well to the Polish language. Using a Polish language corpus, n-gram models are compared to the new connectionist approach described in sections 3 and 4. Section 5 contains the description and the results of the experiment comparing the two approaches. Section 6 contains the conclusions described by the achieved results and future research plans.

2 Dimensionality Curse

The language model must theoretically assign probabilities to all possible word combinations. The amount of these combinations increases exponentially with the amount of modeled words and length of the language model history. This problem is known as the curse of dimensionality. The amount of combinations is so large that the corpora, having even billions of words, contain only a small subset of all the possible word combinations in a given language. The language model trained on such data wouldn't work so well without extra treatment, because it would assign zero probability or an otherwise imprecise estimate to all the word sequences not present in the corpus.

The maximum likelihood method of training depends on the probabilities of word sequence (uni-gram, bi-gram, tri-gram, etc.) occurrences as measured from training data. Brown et al. [12] show that for the language model used in their experiments the maximum likelihood estimates will be 0 for 14.7 percent of the 3-grams and for 2.2 percent of the 2-grams in a new sample of English text. Generally, as n increases, the accuracy of an n-gram model increases, but the reliability parameter estimates, drawn from a limited training text, decreases.

The most often used solution to this problem is the reduction of accuracy of prediction by decreasing the context length. Jelinek and Mercer [13] propose interpolated estimation method that combines the estimates of several language models so as to use the estimates of the more accurate models where they are reliable and, where they are unreliable, to fall back on the more reliable estimates of less accurate models. Other often used methods are deleted interpolation

2 and backing-off 3. The use of these algorithms guarantees that each word sequence is going to have non-zero probability.

3 Connectionist Language Model

The connectionist models use a distributed representation of the items in the history and make much better use of contexts than interpolated or back-off n-gram models. They are much better in fighting the data sparseness problem, but also have the advantage that the model size grows only quasi-linearly when the context length is increased 14. The general outline of the neural network performance when used for language modeling is following: the neural network calculates the contextual probabilities of words:

$$P(w_t|w_{t-1}, w_{t-2}, \dots, w_{t-n}) \tag{1}$$

using a distributed representation for each word in the dictionary. In distributed representation, each word is represented by a real number vector of certain length, generally several dozen dimensions. The information about the word sequences is distributed over several independent components. One could say that the neural network is a set of functions that transform the real valued vectors representing the previous word sequence, into probabilities of the occurrence of all the words in the dictionary. The probabilities of word occurrences in context calculated by the neural network depend on the feature vectors given to the input layer of the neural network. During the neural network training phase, both the network weights and values for the features of individual words in the dictionary are simultaneously set. The words that exist in the similar contexts will have similar features. The features are not a result of clustering analysis, but derived from automatic learning of dependencies between words using a gradient descent algorithm 11. Such distributed representation of words allows for significant reduction in undesirable phenomena such as overtraining and overfit of the model to the data 16,17,18.

4 Formal Description of the Connectionist Language Model

The probability of the word sequence can be calculated when individual word frequencies in the context of previous words are known:

$$P(w_1, w_2, \dots, w_{t-1}, w_t) = P(w_1)P(w_2|w_1)P(w_3|w_1, w_2) \dots \tag{2}$$

$$\dots P(w_t|w_1, w_2, \dots, w_{t-1}, w_t) \tag{3}$$

Feed forward neural networks use a context of size $n - 1$, exactly as in the case of traditional n-grams:

$$P(w_t|w_{t-n+1}, \dots, w_{t-1}) \tag{4}$$

Its worth noting that when n equals 1, the model is known as unigram and the probability of individual words is independent from context. A connectionist language model uses a dictionary where each word is assigned a feature vector C consisting of d real numbers representing the features of the given word. To calculate the probabilities of words next in the sequence, the neural network is fed a feature vector x consisting of a concatenation of the vectors:

$$C(w_t) \tag{5}$$

where w_t denotes a word number t in the sequence. The formula:

$$x = C(w_{t-n}), C(w_{t-n+1}), C(w_{t-n+2}), \dots, C(w_{t-n+(n-1)}) \tag{6}$$

describes the connection of n final feature vectors in the word sequence - context of length n . The next step is to calculate the excitation of the feed forward neural network. The output layer consists of as many neurons as there are words in the dictionary plus one extra neuron which model the remaining words (i.e OOV or Out-Of-Vocabulary words). The output layer uses a softmax activation function [7]. The softmax function, described in formula [7], guarantees that the excitation of each neuron will lie in the range $[0..1]$ and that the sum of all excitations will be equal to 1. This property allows interpreting the result of the neural network computation as the probabilities of individual words. The probability of the word represented by the neuron of index k equals:

$$P(w_t = k | w_{t-n}, w_{t-n+1}, \dots, w_{t-n+(n-1)}) = \frac{e^{net\ k}}{\sum_{i=1}^N e^{net\ i}} \tag{7}$$

where net is the weighted sum of signals reaching the neuron, i.e. net value. Its worth noting that calculating the probability of any word requires the calculation of the weighted sum of signals for all the neurons of the network (the denominator of the equation above) and multiple use of a computationally expensive exponential function. In case of large vocabularies, such calculations may turn out to be considerably slower than their n-gram counterpart. The neural network is trained using a gradient descent algorithm in such a way as to minimize the log-likelihood of the word sequence:

$$L(\theta) = \frac{1}{T} \sum_t \log f(w_t, w_{t-1}, \dots, w_{t-n+1}; \theta) \tag{8}$$

where θ are the parameters of the language model, that is the weights of the neural network and word features. The gradient:

$$\frac{\delta L(\theta)}{\delta \theta} \tag{9}$$

can be calculated using the commonly known back propagation algorithm [6,7,10,15] for the training of a feed-forward multilayer perceptron (MLP) neural network. The algorithm needs to be modified, however, to calculate the gradient not only in the context of the synaptic weights, but also in the C matrix

containing the feature vectors representing the individual words. The model can be trained online, or using batch learning, e.g. by using 100 samples at a time. It's worth noting that unlike the synaptic weights, during an individual training iteration the matrix C will have most of its features' errors not calculated. Only after a long training and many iterations will all the words have their features modified.

5 Experiment

The fundamental problem of researchers endeavoring in the field of language modeling for the Polish language is the lack of substantial amounts of text corpora. The authors therefore decided to use the only corpus available, which is the IPI PAN corpus [11] in his experiments, which is a substantially large, morphosyntactically annotated corpus of general modern Polish language. The corpus was created by Zespół Inżynierii Lingwistycznej w Instytucie Podstaw Informatyki PAN as a part of projects of Komitet Badań Naukowych and statutory research of IPI PAN. The IPI PAN corpus is the first publicly accessible polish language corpus in the true meaning of the word corpus: it is a large, containing 250 million segments, collection of polish texts, morphosyntactically labeled, created according to modern standards and practices of large corpora development.

The experiments were performed using the transcripts of the fourth term session of the Polish parliament, which are a part of the IPI PAN corpus. Due to a long time needed to train and test the model, the transcripts that were randomly selected from the corpus of parliamentary meetings contained exactly 1005569 words. These were split into training (around 800 thousand words) and validation and test sets (around a 100 thousand each). The division of the data was also done randomly. In order to keep the dictionary size at an acceptable level and to have a reasonable statistical presence of all the words in the dictionary, the words that occurred in the corpus less than 9 times were not added to the dictionary (this value was established empirically). The final dictionary consisted of 10570 words. Two experiments were performed: the first used a feed forward neural network and the second was based on a traditional language model created using SRILM [5]. SRILM is de facto standard for n-gram modeling and contains a set of programs which support language model calculation and evaluation. It supports various smoothing and discounting techniques (e.g. Knesser-Ney [48], Witten-Bell, Good-Turing with individually adjustable parameters), large text corpora, class-based and cache models, etc. Model quality evaluation is perplexity based and can be computed counting all tokens in the test corpus or excluding end-of-sentence tags. Language model training using SRILM toolkit is very fast - for the experiment it took less than 1 minute on Pentium dual core processor. The first experiment was based on a feed forward neural network that had 250 inputs (5 words with 50 features each), 100 neurons in the hidden layer and 10508 outputs. It's worth nothing that the amount of outputs is larger by one than the size of the dictionary. That is because the model requires an extra neuron to

represent the words not in the dictionary - OOV. Each word in the dictionary had a unique vector of 50 real numbers representing the feature vector fed to the neural network. The initial values of the features were set to the range -0.01 to 0.01 using a random Gaussian distribution. The output layer used a softmax activation function. The model was trained using a gradient descent algorithm with parameter update momentum. The learning rate was set to 10^{-17} and the momentum to 0.99. These parameters were set empirically. Each time the parameters of the model were changed based on the calculated gradient, the change from the last iteration is multiplied by the momentum parameter and also added to the parameters. This is a known technique used to speed up the teaching process when the error gradient is stable in the subsequent iterations. Additionally, the usage of this technique increases the probability of escaping the local minima of the error function, as claimed by the researchers [7,10]. The training of the neural network based model took 16 days. The neural network achieved a perplexity of 164, which is around 20% better than the traditional n-gram model built using SRILM. Perplexity is a standard method for estimating the amount of fit the model has with the data in the test corpus. The lower the value, the better the model represents the underlying data. It has also been shown by many researchers that lower values of perplexity mean higher accuracy rates in speech recognition systems (e.g. [2]). Perplexity is calculated using the following formula:

$$Perplexity = 2^{-\sum_{i=1}^N \frac{1}{N} \log_2 p(x_i)} \quad (10)$$

The table below shows the results achieved in our experiments. It is worth noting that the context length was chosen as the optimum for their respective models: a context of 5 words for the MLP based model and 3 for the n-gram model. For n-gram models, larger contexts would require much more data than was available. Models based on MLPs require, however, a slightly larger context to function optimally. Similar results have been obtained by other researchers in the field [1]. N-gram model has been computed for several discounting and smoothing techniques giving similar results - perplexity on test data ranges from 193 to 194.

Table 1. Experimental results for the language model trained on the fourth term session of the Polish parliament

Model	Context	Perplexity
SRILM	3	193
MLP	5	164

6 Conclusions

This paper showed a connectionist language model trained on the task of Polish language. The results achieved by the authors for the Polish language are consistent with the results for the English language described in the literature. The

paper contains a general description of the connectionist language models. The experiments performed by the authors show that connectionist language models work better than traditional n-gram based models on the IPI PAN corpus.

Modeling of the context using a simple feed-forward network, also known as the multilayer perceptron, is less elegant than using recurrent neural networks. Feed forward networks have to observe the entire word context (i.e. several words at once) at each time interval. This approach is not as efficient and elegant because the increase of context also greatly increases the size of the network, which makes it lose the generalization potential and slows down its performance. In future works, the authors intend to use recurrent neural networks to eliminate this problem.

Acknowledgments. This work is a part of the SYNAT project financed by the Narodowe Centrum Badań i Rozwoju, contract no. SP/I/1/77065/10.

References

1. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A Neural Probabilistic Language Model. In: NIPS 2000, vol. 13, pp. 933–938; revised in *J. Machine Learning Research* 3, 1137–1155 (2003)
2. Jelinek, F.: *Statistical Methods for Speech Recognition*. MIT Press, Cambridge (1998)
3. Katz, S.M.: Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing* 3, 400–401 (1987)
4. Kneser, R., Ney, H.: Improved backing-off for m-gram language modeling. In: *International Conference on Acoustics, Speech and Signal Processing*, pp. 181–184 (1995)
5. Stolcke, A.: SRILM - An Extensible Language Modeling Toolkit. In: *Proc. Intl. Conf. Spoken Language Processing*, Denver, Colorado (2002)
6. Duch, W., Korbowicz, J., Rutkowski, L., Tadeusiewicz, R.: *Biocybernetyka i inżynieria biomedyczna, tom 6, Sieci neuronowe*. Akademicka Oficyna Wydawnicza EXIT, Warszawa (2000)
7. Bishop, C.M.: *Neural networks for pattern recognition*. Oxford University Press (1995)
8. Ney, H., Kneser, R.: Improved clustering techniques for class-based statistical language modeling. In: *European Conference on Speech Communication and Technology (Eurospeech)*, Berlin, pp. 973–976 (1993)
9. Chen, S.F., Goodman, J.T.: An empirical study of smoothing techniques for language modeling. *Computer, Speech and Language* 13(4), 359–393 (1999)
10. Tadeusiewicz, R.: *Sieci neuronowe*. Akademicka Oficyna Wydawnicza RM, Warszawa (1993)
11. Przepiórkowski, A.: *Korpus IPI PAN. Wersja wstępna / The IPI PAN Corpus: Preliminary version*. IPI PAN, Warszawa (2004)
12. Brown, P.F., DeSouza, P.V., Mercer, R.L., Della Pietra, V.J., Lai, J.C.: Class-based n-gram Models of Natural Language. *Computational Linguistics* 18, 467–479 (1992)
13. Jelinek, E., Mercer, R.L.: Interpolated estimation of Markov source parameters from sparse data. In: *Proceedings, Workshop on Pattern Recognition in Practice*, Amsterdam, The Netherlands, pp. 381–397 (1980)

14. Xu, P., Emami, A., Jelinek, F.: Training connectionist models for the structured language model. In: Proceedings of The 2003 Conference On Empirical Methods In Natural Language Processing (EMNLP 2003), pp. 160–167. Association for Computational Linguistics, Stroudsburg (2003)
15. Tan, C.N.W., Wittig, G.E.: A study of the parameters of a backpropagation stock price prediction model. In: Proceedings, First New Zealand International Artificial Neural Networks and Expert Systems (1993)
16. Niesler, T.R., Whittaker, E.W.D., Woodland, P.C.: Comparison of part-of-speech and automatically derived category-based language models for speech recognition. In: International Conference on Acoustics, Speech and Signal Processing, pp. 177–180 (1998)
17. Hinton, G.E.: Learning Distributed Representations of Concepts. In: Proceedings of the Eighth Annual Conference of the Cognitive Science Society, pp. 1–12 (1986)
18. Hinton, G.E.: Connectionist Learning Procedures. *Artificial Intelligence Journal* 40, 185–234 (1989)

Commonly Accessible Web Service Platform — Wiki-WS

Henryk Krawczyk and Marek Downar

Faculty of Electronics, Telecommunications and Informatics,
Gdansk University of Technology,
11/12 Gabriela Narutowicza Street, 80-233 Gdansk-Wrzeszcz, Poland

Abstract. Web service technology on the basis had to supply complete and reliable system components. Nowadays this technology is commonly used by companies providing results of their work to end users and hiding implementation details. This paper presents a SOA-enabled platform — Wiki-WS — that empowers users to deploy, modify, discover and invoke web services. Main concept of the Wiki-WS platform is searching and invocation of web services written by different workgroups in different technologies deployed on different servers. Wiki-based web service code modification allows engineers from any place to construct and to implement components, which are mature and ready to use. Components deployed in the platform cover different functionalities from various knowledge domains. Service categorization done by users and by the platform classification engine in the early deployment and every modification time tunes up future searching and usage decision processes. Fundamental architecture components and user categories characterization are described in this paper. There is also included presentation of sample scenarios of Wiki-WS usage and advantages derived from its deployment.

1 Concept of a Service Platform

The assumption of web services technology was to provide IT engineers with ready-made program components. Currently, this technology is commonly used by a sector, which makes it accessible to clients against payment, but it does not disclose the implementation details. Such services are created by hermetic environments, which results in their imperfect character, and common problems with integration and their reliability [1]. The study offers a platform, due to which not only the web services technology will become popular, but also the access to services will be more convenient as all of them will be collected in one place.

SOA (*Service Oriented Architecture*) based development is one of the emerging approach in building scallable and flexible applications using reusable components. Wiki-WS platform represents a base web services market. Services developed on Wiki-WS are ready-to-use application building components. Due to large granularity of function modules one of the Wiki-WS concept is to reach

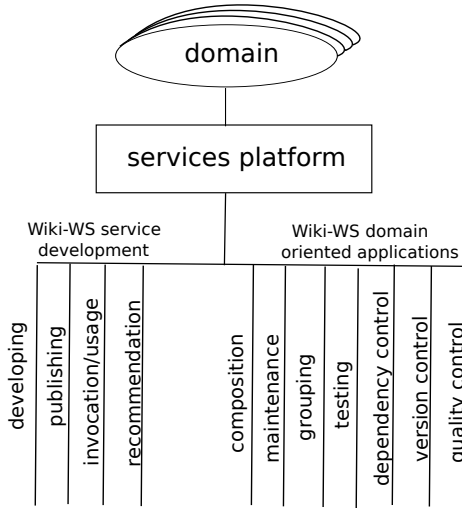


Fig. 1. The concept of Wiki-WS platform

full free-form of services composition and easy components swapping. Published service code and availability of modification would allow developers to execute constant in time maintenance of simple components. In that way there will be reduced effort that has to be put in the implementation of each individual sub-modules.

Figure 1 illustrates the assumed functions which will be made accessible to the user of Wiki-WS platform. Functions are divided into two subgroups: Wiki-WS service development and Wiki-WS domain oriented applications. The first submodule allows users to work with single services published at Wiki-WS platform. The second submodule is domain oriented, and is responsible for application composition, testing, grouping and management functionalities.

The suggested platform is similar to the management system used by Wikipedia, enriched with automatic classification of information. It is believed, that the designed platform enables implementation and realization of web services. The described platform was named Wiki-WS (prefix Wiki — from the engine, on which the richest generally accessible source of knowledge repository was based — Wikipedia), and it would enable user to freely provide the network with the self-created fragments of code in a form of web services. Users, as in the case of articles in Wikipedia, apart from possibility of providing self-created service, can get the access to codes of different services, as well as they can modify them. In the case of web services, introducing a code in a form of text itself would be pointless if it could not be called. For this purpose, the service during its introduction into the system is automatically implemented on the application server operating in the programming language chosen by a developer (C#, Java, PHP). After deployment of the service it is possible to modify it by the designer and other system users by editing its code. The platform of that type will not only

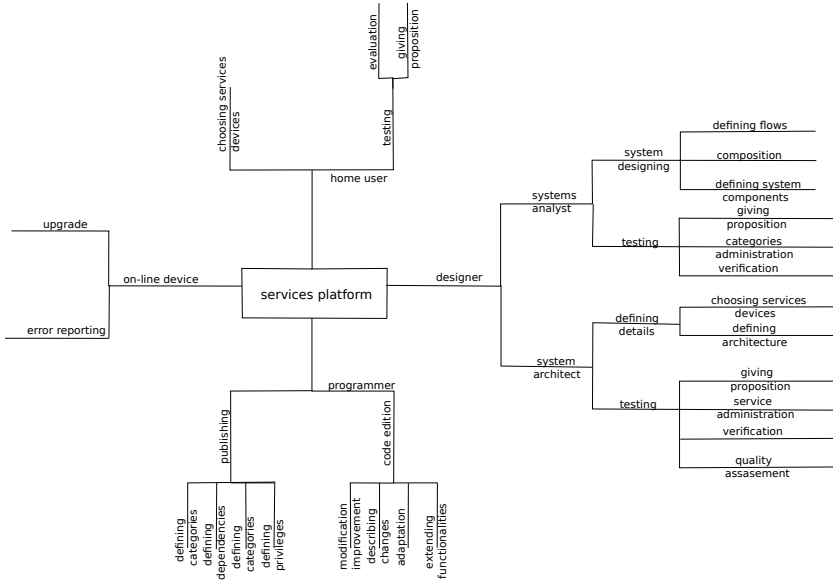


Fig. 2. Division of activities according to system users

be useful for programmers. Figure 2 presents activities that can be realized by different users. This picture distinguishes the role of the mentioned programmer, analyst and system designer, domestic user and online devices.

The collection of a big number of web services will enable to generate clusters, or categories of web services by means of artificial intelligence algorithms. Such classification of services would be useful for system analysts enabling the construction of complex systems with the use of ready-made components. Therefore, when creating the framework of a system, system analysts define components of which it should be made. Additionally, they determine interaction occurring between those components. Then system designers would have to choose devices and individual services according to their qualitative measures, such as efficiency, reliability, source and error frequency, illustrated with a use of indicator such as MTTR (*Mean Time To Repair*), MTBF (*Mean Time Between Failures*). On the basis of those measures an architect would decide on the method of user access management methods, e.g. for systems requiring high level of security the services must be provided by qualified working teams and based on devices with high level of reliability. On the contrary, for basic use, it is satisfactory to select a cheaper solution, with lower level of adopted qualitative matrices.

2 Wiki-WS Architecture

Figure 3 shows the architecture of the presented platform. It consists of components responsible for communication with a user (services management, administration and verification, scenarios composition and calling) and of components

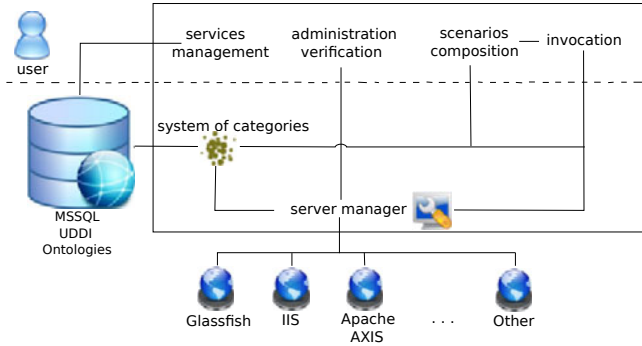


Fig. 3. Architecture of Wiki-WS platform

which are a “core” of the platform (administration of servers, system collecting information about services, and a system for services classification).

The platform itself enables users to implement, edit and control web services introduced in any language of programming. The collected services are a subject to cluster algorithms in order to create coherent groups of services belonging to the same categories, therefore, realizing a similar functionality. Inference relating to the belonging of a service to a certain group is based on the analysis of formal and informal description of an individual service. The formal description is a service code with comments, WSDL (*Web Service Description Language*) code, OWL-S (*Web Ontology Language for Services*). The informal description should be understood as a detailed description presented in a form of sentence equivalents, and in a form of free description without details. The detailed description additionally contains information in a form of tables relating to a device subject to a service. The combination of formal and informal service description, after rejection of redundant and insignificant information, creates a “reference space” characteristic to a service, on the basis of which the belonging of a service to a certain group is determined (Figure 4).

Such data set would enable to engage the most of known grouping algorithms described in [5]. As an experiment, one would be able to choose algorithms and descriptions allowing to achieve best search results. The services are additionally evaluated in the scope of meeting qualitative criteria. The evaluation, after selecting adequate weights, will constitute the basis for the created service quality tree. This evaluation will be provided by programmers participating in designing and modification process, as well as by the users. However, the target is that they will be also a subject to automatic evaluation made on the basis of analyses of a code, designers and the scope of detailed and accurate nature of the description.

It is worth emphasizing here that originally implemented service is marked with certain key phrases, e.g.:

- [analyze],[clustering],[GAAC],
- [retrieve],[documents],[PDF],
- [organize],[ontology],

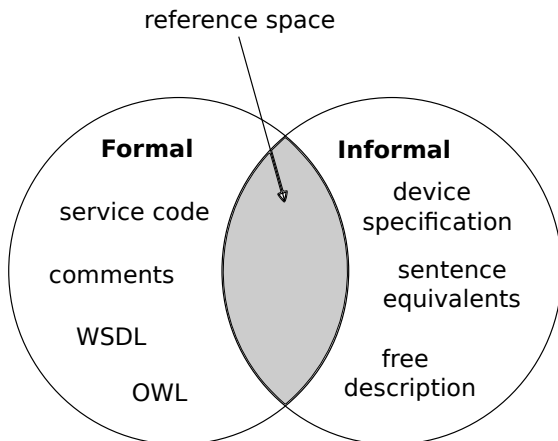


Fig. 4. Reference space characteristic to a service

- [access],[user],[management],
- [access],[interface],[query],[visual],
- [translate],[SPARQL],[SQL].

enabling to initially assign it to a certain category. It facilitates grouping of services and their selection for a certain system.

2.1 Service Development

The platform is initially dedicated to programmers of all kinds (low and high level) who wish to share and make the collected knowledge available in a form of web services. System users can be divided to:

- base users – implementing, improving services, providing description and initial weights,
- verifying users – performing a function of moderators, verifying the validity of services and descriptions,
- using users – e.g. system analysts,
- “info” type users – acquiring information about updates (e.g. home users or on-line devices).

Depending on the type of a user, they are provided with certain functions of the platform.

The life cycle of services provided by Wiki-WS service development module is presented in Figure 5. Work organisation in this environment is similar to organisation commonly used in Wikipedia.

Implementation of a Web Service. We assume that users have a code of a web service implemented in a free language of programming and they are willing

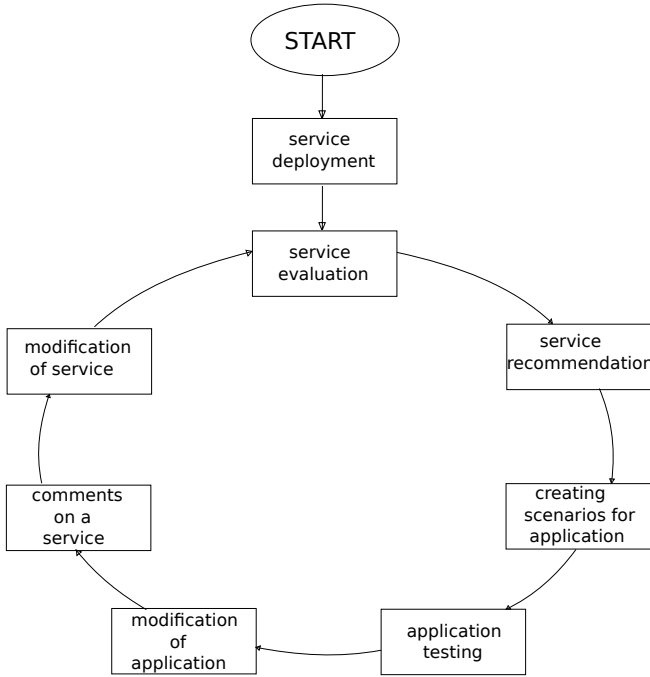


Fig. 5. Life cycle of a service on Wiki-WS platform

to share it within the frames of Wiki-WS system. In order to introduce a service, a user provides the service code through web interface of Wiki-WS platform. The user also defines authorization of other users, the default setting is that everyone can edit the service code. Another assumption is that the users, similarly to those of Wikipedia, will act in a reliable and honest way, but there is predicted a subsystem that will support services reliability control by implementing review policies.

The service is then automatically deployed on one of the servers, depending on technology and possible preferences of a user. After service introduction a user provides its description in a formal and informal form. The scheme of grouping process is presented in figure 6. A user provides service description by means of informal language. This description is forwarded to a component responsible for token extraction (key phrases). Those tokens, together with a formal description, are used to create an ontological description of an individual web service. The tokens and the whole of description in words is memorized and kept in database. The database also stores all of the formal description in a form of WSDL code and a source code. The internal UDDI (*ang. Universal Description Discovery and Integration*) registry aims to help detection and integration of services implemented into a platform. Information supplied by UDDI, the engine of database and ontological description, will constitute input data in a form of reference space for grouping algorithms, presented in figure 4. As a

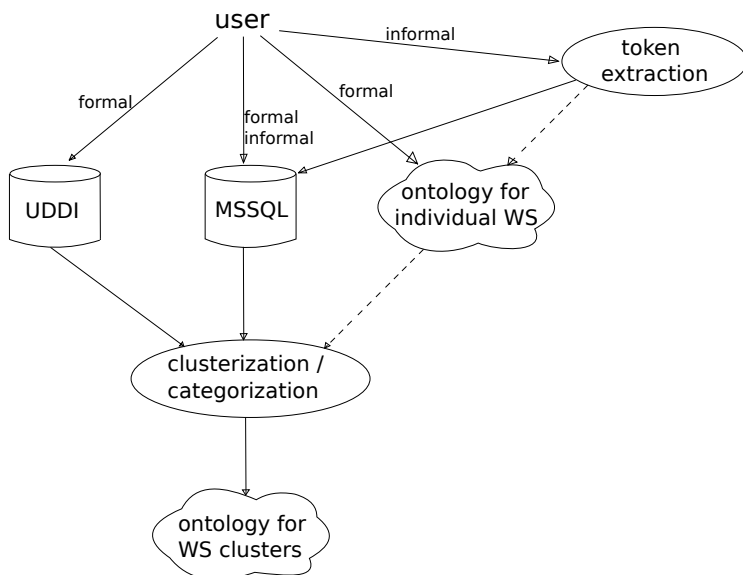


Fig. 6. Process of service grouping

result of a whole process, we will acquire information on the group in which a certain service should be included.

Edition and Evaluation. One of the key elements of the system is the ability to edit a service code. Without it, the system would be nothing else than an ordinary repository making hermetic, and often imperfect components available. Due to the possibility of modification, the implemented services would evolve to become mature and error resistant components ready to be used in target systems. Edition is limited to making modifications of the source code, as it is done in the case of articles introduced into Wikipedia, where there is a creator of an article, editor and people verifying the article's content. By assumption, the service code can be edited by anyone, and by default it is implemented on the basis of Open Source license. However, it is possible to limit the license by limiting the accessibility to the source code to a certain group of people, in order to derive possible financial benefits from it in the future, from distributed working units. In this way the groups of programmers would be created (distributed companies) realizing a project in Wiki-WS, a team system working on a program unit in order to sell it later. Wiki-WS system would constitute a perfect distribution environment of such product. Coming back to the merits of the realized platform, a user can edit service code and once again introduce it into a server. The possibility of going back to the previous code version is necessary, a server makes server svn or git available. If source changes, service is automatically deployed with proper version number. Previous version of service is still available, so applications that are already using it act in the same way. Additionally, each

service has an individual entry in work and bug tracking system (bugzilla, trac). Therefore, reporting errors and their repair by distributed teams of programmers is simpler.

Composition. Composition of services into scenarios will be done by a target user, i.e. analyst or home user, by means of a web graphical interface. It will be done at two levels. In the first phase the service categories created on the basis of algorithm classification will be used (e.g. service responsible for allowing the entry, service responsible for changing illumination). A user at higher level of abstraction would be able to model the system by creating scheme similar to the one presented in Figure 8. After determination of such system “mock-up”, a user will be able to make system more detailed by choosing certain service implementations.

2.2 Domain Applications

The life cycle of applications developed using Wiki-WS domain oriented applications development module is presented in Figure 7.

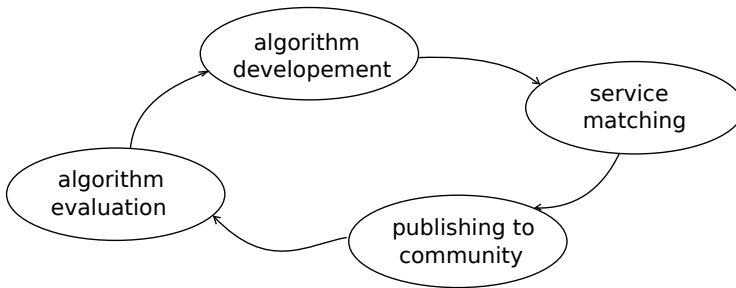


Fig. 7. Life cycle of an algorithm in Wiki-WS platform

Wiki-WS application development module allows to design SOA based applications. User designs scenario in which algorithm modules should cooperate. After generation of a algorithm framework the only problem left would be to transform the communication flow defined by the analyst into a certain service flow defined by means of languages used for description of scenarios. The description of existing languages (BPEL4WS, WF, WS-CDL, UML-S, Q) and their selection for realization of individual undertakings is presented in article 3 4. Generated algorithms with matched web services are provided to the open usage of community. After algorithm usage clients gives opinions and recommendations which are taken into account in future algorithms improvements.

3 Area of Platform Applications

Wiki-WS can be used in applications of any domain. This chapter presents Wiki-WS exemplary usage in digital libraries and in context oriented environments.

3.1 Searching Engine for Digital Library

Wiki-WS platform is designed to provide easy way to share technical experiences between software experts. Allows software development based on services that are constantly improved by different distributed workgroups. Granularity of components (web services) lead to free-form algorithm composition and fully automated processes of building scenarios from domain base components. For digital libraries there is a lot of algorithms realizing different tasks that can be put into Wiki-WS platform in order to be constantly improved, e.g. algorithms responsible for:

- searching for documents,
- documents translation,
- searching for relevant information from document set,
- obtaining document description and metadata creation,
- document recommendation.

An exemplary searching platform framework which is a result of system analyst's work is presented in Figure 8. It is assumed that simple digital library would be divided into 5 submodules. Data retriever module is responsible for analyzing information provided in various form of digital documents (e.g. finding key phrases and constructing metadata of each document). Data organizer module defines structures and storage platforms for data provided by data retriever module. Data analyzer module using previously organized data prepares suitable structures for searching algorithms. Data miner module contains set of algorithms and low level query interfaces (e.g. ontology query languages (RDQL [9], SPARQL [11], SeRQL [12]), database query languages like SQL) that are able to retrieve data from data prepared by data analyzer module. Also contains high-to-low level query translators (e.g. MDDQL [8] SPARQL-TO-SQL query translator [10]). Access module provides interfaces for high level query languages supporting users in easy-to-learn querying (e.g. VOQL [7]). Also contains modules responsible for user and access management. The main advantage of such modular approach is that these modules may have many submodules that could be used interchangeably in order to provide best searching systems dedicated for many kinds of users.

3.2 Ubiquitous Application

In the case of home users, it is assumed to combine the Wiki-WS system with ubiquitous computing. An exemplary application can be as follows:

- a consumer buys a set of home ubiquitous computing, which consists of a server, set of sensors and actuators,

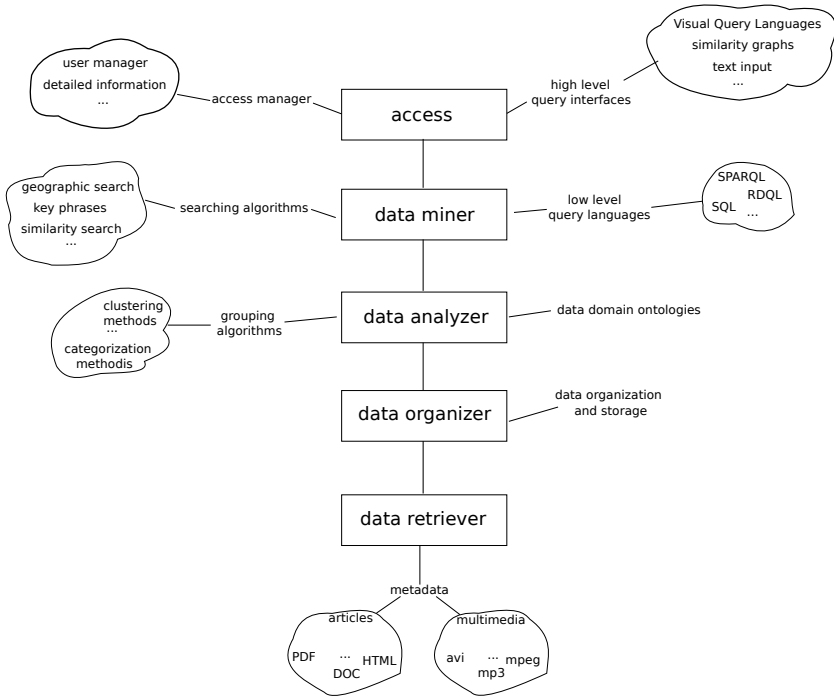


Fig. 8. Exemplary scheme of search engine platform

- a consumer installs and configures a purchased product at home,
- a programmer implements a service providing interface of web service for a new device,
- a programmer places service code in Wiki-WS system, apart from its description they provide a detailed description of a device, with which this service interacts,
- a programmer determines authorizations and possible cost of offline service usage,
- a consumer who wants to expand home system of ubiquitous computing choses a service according to categories (e.g. services responsible for allowing an entry),
- a consumer after choosing a service interesting for them buys a device (e.g. RFID card reader), for which the service was implemented,
- a consumer graphically defines (similarly to system analyst) the usage scenario for a new component in their ubiquitous computing system.

An exemplary system framework which is a result of system analyst’s work is presented in Figure 9. The components used on this scheme would be a generalization generated by means of clusterization of certain services. An exemplary component could be the one responsible for enabling a person to enter a room.

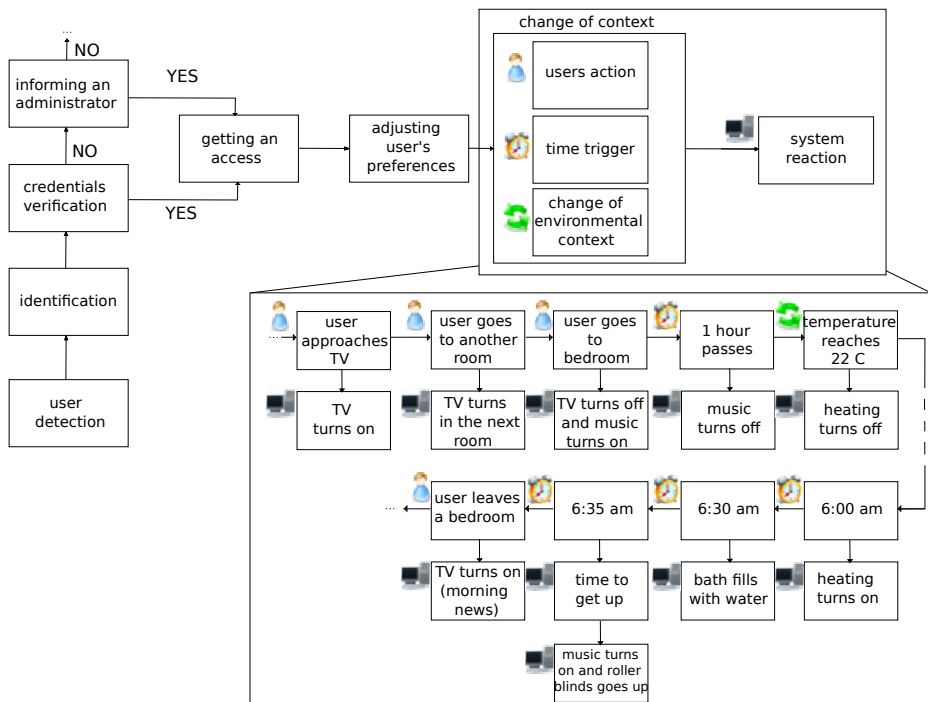


Fig. 9. Exemplary system scheme designed by a system analyst

By its assumptions the system should remain an Open Source one, therefore enabling a free online usage of web services. However, a programmer can claim financial benefits for offline usage of their service, only under condition that other designers will not have the access to the code. The usage of the service, e.g. in home ubiquitous computing system operating in offline mode, should ensure higher system security, as it would not require constant access to the Internet [2].

4 Future Work

The project provides the following research problems:

- selection of algorithms which dynamically classify different services,
- selection of factors which enable construction of service quality tree,
- selection of languages and methods of service description, both formal and informal,
- graphical composition of scenarios from ready-made components on the basis of their e.g. interfaces,
- performing scenarios and making automatic scenarios testing,

Table 1. Comparison between SOA and Wiki-WS

	SOA	Wiki-WS
target	better connection of a business side of organisation with its IT resources	real opening of web services market
business application	realizing operation on remote calculation units	connecting technology of web services with ubiquitous computing
concentration	defining a service that fulfils user's requirements	making efficient and reliable web services in a form of components available
implementation details	hiding implementation details from a user	introducing levels of accessibility to implementation details (programmers, analysts, "info" type users)
aggregation	within the scope of working units	in an open implementation environment
classification	none	with the use of artificial intelligence algorithms
scenarios	set of algorithms	selection of the best algorithm
realization	remote	making consolidation of a code to an offline form possible
evaluation	none	evaluation made automatically or by users (programmers, analysts, "info" type users)
reporting demand for a product	none	possible
service description	for automatic tools	both for automatic tools and ordinary users
implementation of service on user's unit	impossible	possible

- consolidation of web services scenarios to applications working offline – used e.g. in the home system of ubiquitous computing, mentioned before,
- ensuring efficiency, reliability and system safety,
- integration with other external systems, e.g. BeesyCluster [6]
- ensuring interoperability of web services generated by working groups (consisting of the same programmers, of some of the same programmers, or of different programmers),
- automatic generation of web services interfaces (in the case of selection of an integrating component that should constitute a coherent whole together with already existing services),
- ensuring universality of web service interface in order to intensify interoperability,
- other problems relating to testing, describing, and effective system implementation.

5 Conclusions

Common accessibility to such platform will contribute to the growth and popularization of the market of open source web services. The programmers will make their applications or their fragments available to a wide range of other programmers, so they could be improved (e.g. on the basis of Open Source license). The idea of Wiki-WS system is based on SOA, but it improves and enriches it by additional aspects, providing a new trend of web services which are “friendly” to users of components with an open code. Differences between SOA and Wiki-WS are presented in table [11](#).

Acknowledgements. This work has been supported by the National Centre for Research and Development (NCBiR) under research Grant No. SP/I/1/77065/1 SYNAT: “Establishment of the universal, open, hosting and communication, repository platform for network resources of knowledge to be used by science, education and open knowledge society”.

References

1. Kaczmarek, P., Downar, M.: Interoperability analysis of sensor interface in ubiquitous environments. *Information Systems, Architecture, and Technology*, 99–109 (2010)
2. Downar, M.: Problemy bezpieczeństwa informacji w środowisku przetwarzania wszechobecnego. In: *KASKBOOK 2009, Gdańsk–Bytów* (2009)
3. Downar, M.: Języki opisu scenariuszy realizacji przedsięwzięć. In: *KASKBOOK 2010, Gdansk* (2010)
4. Dziubich, K.: Metody i języki opisu scenariuszy zachowań aplikacji przetwarzania wszechobecnego. In: *KASKBOOK 2009, Gdańsk–Bytów* (2009)
5. Manning C.D., Raghavan, P., Schütze, H.: *An Introduction to Information Retrieval*. Cambridge University Press (2009)
6. Czarnul, P., Kuryłowicz, J.: Automatic conversion of legacy applications into services in Beesy. In: *2010 2nd International Conference on Cluster Information Technology (ICIT), June 28-30*, pp. 21–24 (2010)
7. Iskandar, D.N.F.A.: Visual ontology query language Networked Digital Technologies. In: *First International Conference on NDT 2009, July 28-31*, pp. 65–70 (2009)
8. Kapetanios, E., Baer, D., Glaus, B., Groenewoud, P.: MDDQL-Stat: data querying and analysis through integration of intentional and extensional semantics. In: *Proceedings. 16th International Conference on Scientific and Statistical Database Management, June 21-23*, pp. 353–356 (2004)
9. Tun, M.T., Thein, N.L.: Informative Query Answering by using RDQL for E-Commerce Information and Telecommunication Technologies, 2005. In: *6th Asia-Pacific Symposium on APSITT 2005 Proceedings, November 10-10*, pp. 142–147 (2005)
10. Lv, L., Jiang, H., Ju, L.: Research and Implementation of the SPARQL-TO-SQL Query Translation Based on Restrict RDF View. In: *2010 International Conference on Web Information Systems and Mining, October 2010, vol. 1*, pp. 309–313 (2010)

11. Malik, S., Goel, A., Maniktala, S.: A comparative study of various variants of SPARQL in semantic web. In: 2010 International Conference on Computer Information Systems and Industrial Management Applications (CISIM), October 8-10, pp. 471–474 (2010)
12. Wu, G.T.J.: SERQL: an ER query language supporting temporal data retrieval. In: Conference Proceedings. Tenth Annual International Phoenix Conference on Computers and Communications, March 27-30, pp. 272–279 (1991)

Author Index

- Bazan, Jan G. 1
Bembenik, Robert 121
Betliński, Paweł 97
Brocki, Łukasz 243
Cyganek, Bogusław 217
Czyżewski, Andrzej 227
Downar, Marek 251
Frączek, Rafał 217
Gawrysiak, Piotr 37
Goczyła, Krzysztof 179
Gora, Paweł 97
Grzegorowski, Marek 49
Herba, Kamil 61, 97
Jamro, Ernest 203
Janusz, Andrzej 61
Jaśkiewicz, Grzegorz 77
Kijowski, Michał 49
Korżinek, Danijel 243
Kostek, Bożena 227
Kowalski, Marcin 49
Kozłowski, Marek 37
Krawczyk, Henryk 251
Kupryjanow, Adam 227
Marasek, Krzysztof 243
Mazurek, Cezary 153
Nguyen, Hung Son 1, 9
Nguyen, Linh Anh 9
Nguyen, S. Hoa 77
Nguyen, Trung Tuan 97
Pardel, Przemysław 49
Podsiadły-Marczykowska, Teresa 121
Protaziuk, Grzegorz 121
Rybiński, Henryk 121
Ryżko, Dominik 37
Sielski, Krzysztof 153
Skowron, Andrzej 1
Ślęzak, Dominik 1, 49
Stawicki, Sebastian 97
Stencel, Krzysztof 49
Stroiński, Maciej 153
Świeboda, Wojciech 77
Szczuka, Marcin 61
Walkowska, Justyna 153
Waloszek, Aleksander 179
Waloszek, Wojciech 179
Wasilewski, Piotr 107
Węglarz, Jan 153
Werła, Marcin 153
Wiatr, Kazimierz 203
Więch, Przemysław 37
Wielgosz, Maciej 203
Wróblewska, Anna 121
Zawadzka, Teresa 179
Żurek, Dominik 203