12th European Conference on Machine Learning (ECML'01)
5th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'01)

Workshop

Semantic Web Mining

Gerd Stumme, Andreas Hotho, Bettina Berendt

September 3, 2001
Freiburg
Germany

Foreword

When we started the organization of the workshop, it was clear that the Semantic Web and Web Mining are two fast-developing research areas which have many points of contact. However, there was not yet a precise idea what the integration of the two areas might look like in detail. The workshop aims to advance the convergence between Semantic Web and Web Mining research by bringing together researchers and practitioners from these two areas. Our aim is to improve, on the one hand, the results of Web Mining by exploiting the new semantic structures in the web, and on the other hand to exploit Web Mining for building the Semantic Web.

The Semantic Web is based on a vision of Tim Berners-Lee. He suggests to enrich the web by machine processable information which is organized on different levels (see http://www.w3.org/DesignIssues/Semantic.html). For Web Mining, the levels from XML and RDF to ontologies and logics are of particular interest. Web Mining applies data mining techniques on the web. Three areas can be distinguished: Web usage mining analyzes user behavior, web structure mining utilizes the hyperlink structure, and web content mining exploits the contents of the documents on the web.

In the workshop, we want to discuss the use of XML, RDF, ontologies, and logics for the three web mining areas; and the support of web mining techniques for building XML and RDF schemes and ontologies. There are quite a number of people from different communities that approach the field of Semantic Web Mining from different, interesting angles. The goal of the workshop is to establish communication between these communities.

The contributions to this workshop represent different approaches to Semantic Web Mining:

A number of methods are proposed for learning domain ontologies. These range from learning from natural-language data to exploiting existing metadata representations. *Engels, Bremdal, and Jones* present a method that extracts information from natural-language data and produces a graph visualization of relations between concepts, as well as XML/RDF ontologies. *Kurematsu, Nakaya, and Yamaguchi* describe an approach for constructing domain ontologies from machine-readable dictionaries and text corpora. *Clerkin, Cunningham, and Hayes* cluster objects described by their attributes to generate class hierarchies expressible as RDF schemas. *Kavalec, Svátek, and Strossa* use web directories like Yahoo! as training data for automated meta data extraction.

Domain ontologies become more useful when they are related to the users and their individual interests and preferences. Learning from a user's prior interaction with the system can supply useful data for it. *Kiss and Quinqueton* deal with the learning of user preferences. Their multi-agent system is dedicated to

corporate memory management in an intranet. Semantic structure on the Web, together with the mining of user navigation histories, can directly lead to better adapted user interfaces. *Mobasher* describes an adaptive agent for information retrieval which uses a concept hierarchy of terms found in documents, together with clusters summarizing user search histories, to (semi-)automatically improve queries.

Lastly, it is important to develop integrating architectures that range from ontology learning to the display of results for the user. *Haustein* describes an agent-based blackboard architecture that connects mining components, ontologies, and applications, as for instance an interface for generating HTML. *Le Grand and Soto* cluster XML topic maps, online equivalents of printed indexes, by means of Formal Concept Analysis, to define profiles. Their aim is to support navigation and understanding of the set of documents.

With this collection of research papers, we aim to provide a starting point for the convergence of the Semantic Web and Web Mining. We wish to express our appreciation to all the authors of submitted papers, to the members of the program committee, and to the additional reviewers for making the workshop a valuable contribution to Semantic Web Mining.

July 2001

Bettina Berendt
Andreas Hotho
Gerd Stumme

Organization

The Semantic Web Mining Workshop was organized as a workshop within the 12th European Conference on Machine Learning (ECML'01) and the 5th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'01). It was held on September 3, 2001, in Freiburg, Germany.

Workshop Chairs

Gerd Stumme
Institut für Angewandte Informatik
und Formale Beschreibungsverfahren (AIFB)
Universität Karlsruhe
D–76128 Karlsruhe, Germany
http://www.aifb.uni-karlsruhe.de/WBS/gst
stumme@aifb.uni-karlsruhe.de

Andreas Hotho
Institut für Angewandte Informatik
und Formale Beschreibungsverfahren (AIFB)
Universität Karlsruhe
D–76128 Karlsruhe, Germany
http://www.aifb.uni-karlsruhe.de/WBS/aho
hotho@aifb.uni-karlsruhe.de

Bettina Berendt
Abteilung Pädagogik und Informatik
Humboldt-Universität zu Berlin
Geschwister-Scholl-Straße 7
D–10099 Berlin, Germany
http://www.educat.hu-berlin.de/~berendt
berendt@educat.hu-berlin.de

Program Committee

Soumen Chakrabarti
(Indian Inst. of Technology, Bombay)
Rosine Cicchetti
(Univ. de la Méditerranée, Marseille)
Stefan Decker
(Stanford University, Palo Alto)
Ronen Feldman
(Bar-Ilan University, Ramat Gan)
Klaus-Peter Huber
(SAS, Heidelberg)
Alexander Mädche
(Universität Karlsruhe)
Tom Mitchell
(WhizBang! Labs, Pittsburgh)
Bamshad Mobasher
(DePaul University, Chicago)
Katharina Morik
(Universität Dortmund)
Claire Nedellec
(Université Paris Sud)
Myra Spiliopoulou
(Handelshochschule Leipzig)
Rudi Studer
(Universität Karlsruhe)

Further Reviewers

Martin Lacher
(Stanford University, Palo Alto)
Lotfi Lakhal
(Univ. de la Méditerranée, Marseille)
Carsten Pohle
(Handelshochschule Leipzig)
Karsten Winkler
(Handelshochschule Leipzig)

Table of Contents

unedited draft: EXML/PKDD workshop (2001), Freiburg, Germany 2001

CORPORUM: a workbench for the Semantic Web

R. H. P. Engels, B. A. Bremdal and R. Jones*
CognIT a.s
P.O. Box 610, N-1754
Halden, Norway
{rob.engels, bernt.bremdal, richard.jones}@cognit.no

July 31, 2001

Abstract

'Web semantics' has for a long time been a term without much content. The web is organizing itself, and its pages are typically added in a random and *ad hoc* fashion by everybody who feels like contributing. Typically, there has not been much concern about how to present contents in the best way, other then pure lay-out issues. This fact, combined with the fact that the representation language used at the world wide web is mainly format oriented (i.e. not depending on a complex formal logic representation mechanism), makes publishing on the WWW easy, giving it its enormous expressibility. Although widely acknowledged for its general and universal advantages, the increasing popularity of the web also shows us some major draw-backs. The developments of the information contents on the web during the last year alone, clearly marks the need for some changes. Perhaps one of the most felt problems with the web as a distributed information system is the difficulty to find and compare information which is provided on it. Many people add private, educational or organizational content to the web which is of immense diverse nature. Content on the web is growing closer to a real universal *knowledge base*, where there is only one problem relatively 'undealt' with; the problem of the interpretation of such contents.

In this paper, the authors provide a discusson on a technical solution which is aimed at helping the web to become more *semantic*. The CORPORUM tool set that is developed for this task exists of a set of programs that can fulfill a variety of tasks, either as 'stand-alone', or augmenting each other. As the aim of the semantic web is to enhance the *precision* and *recall* of search, but also enable the use of *logical reasoning*on web contents in order to answer queries. Important tasks that are dealt with by CORPORUM are related to information retrieval (find relevant documents, or support the user finding them), but also information extraction (can we built a knowledge base from web documents to answer queries?), information dissemination (summarizing strategies and information visualisation), and automated document classification strategies performed by so-called intelligent agents which are present on the world wide web on a pertinent basis. The current article discusses the CORPORUM tool set and shows how it can support generation and utilisation of semantics on the web.

*CognIT a.s is a full partner in the EU-funded project "OnToKnowledge" IST-1999-10132. The authors wish to thank aidministrator Nederland BV for the usage of their visualisation tools used to visualise the structure of figure 4 within this article. Please refer to the workshop website for the final version.

1 Two scenarios to put more semantics on the web

Generally speaking, there are two fundamentally different scenarios in which the world wide web could evolve further. Either the currently existing mass of documents available on the web can be analysed in its current 'as is' form and contents can be extracted from it, or the representation format of the world wide web is changed up front so that documents are available in a format that expresses such 'semantics' more explicit.

Each of these approaches have their own drawbacks, a fact that might be the reason that there is still an ongoing debate on what the next generation Internet should look like. The disadvantage of 'flat', mainly format based representation languages (cf. HTML, LaTeX) is that they mix information on content (the text a writer wants to disseminate) and the format in which this is done (lay out issues). Such a disadvantage is to be opposed to a rather easy to learn language, so that virtually anybody with web access can easily publish information, knowledge and opinions.

Another reason for the need for more *web semantics* is that although the web is a media for publishing content, far from all its contents are created on or for it! In most cases documentation has to be reformatted and analysed before it can be published on the web, and extracting semantic contents from such un(web)structured documents might appear not to be easy at all.
Using an explicit representation language with clear semantics, where *content* is represented explicitly, usually sets a halt to the ease of use for most average users. Using 'higher' representation languages (cf. XML/RDF, or formal languages) in a similar manner as todays web publishing tools might therefore not be the best way to go, because it is expected to thwart publishing and sharing his or her knowledge and thoughts due to its higher complexity. Additionally, *backward compatibility* of a new *semantic web* representation language should be guaranteed.

Having had this debate for a few years now, in the meanwhile consensus seems to be that a combination of the two approaches could solve most of its drawbacks. As possible solution one can imagine tool support in order to either analyse pages that are not represented in a 'semantically rich' manner, or offering graphical interfaces (editors) to people that support creating such semantic representations (semi-)automatically.

These two scenarios are both seen as important, as long as they coexist on the web. A variety of projects with semantic representation languages have shown that representing all web contents in 'higher order languages' might not be feasible unless more automated approaches become available. However, an increasing acceptance of more semantic representation language on the web can be noticed and several initiatives aim at supporting them. Several project are initiated world-wide to support the *semantic web* ([FvHKA99], [Hen00], etc.), languages, extensions on languages and query languages are defined ([BG00], [FHH+00]) and tools for (semi-)automatic content extraction are implemented ([BJ99], [oMAG00], [aA00]).

Nevertheless, last years of research, be it conducted government supported or private, have shown emerging technologies that, although often predicted and already initiated for many years ago, only recently unleashed some of the power that lies in a combination of semantic analysis and distributed information systems. One of these tools is developed in the private sector during the past four years, and has grown mature enough to serve as bottom technology in a variety of products, as well as in co-operation projects on a European level (cf. the OnToKnowledge project [FvHKA99]).

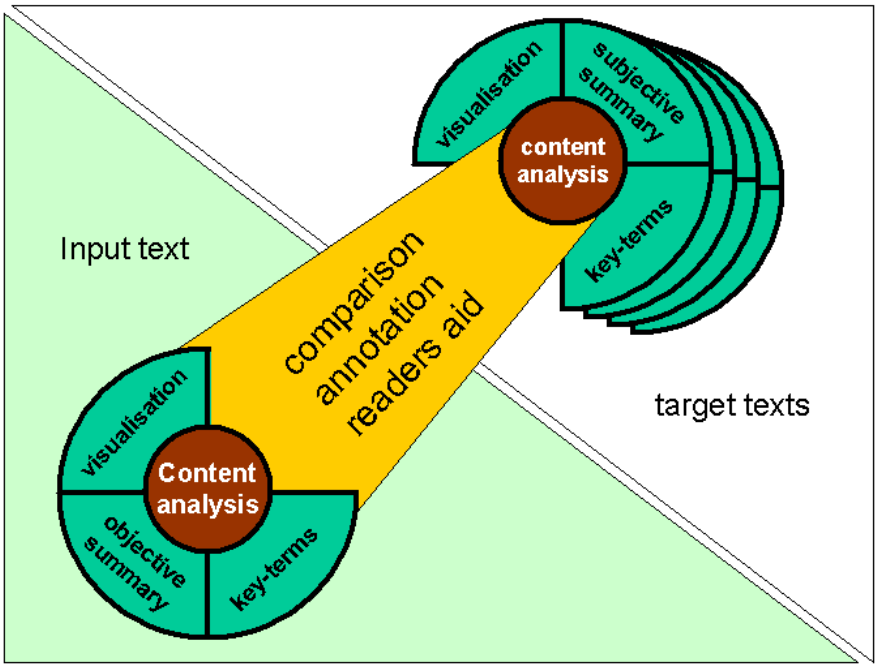


Figure 1: Core Analyser components' MiMir functionality.

2 Description of the CORPORUM system

For building up, utilising and maintaining the semantic web, there are a variety of tasks that are to be dealt with. All of these tasks find their *raison d'etre* in the fact that people need to get on top of the information overflow they get offered to them. This holds for individuals learning, organising and interacting on the web as much as for organisations that want their employees to mutually benefit of a better directed, better understandable and more clear information and knowledge sharing facility ([BJS+99]).

On the theoretical side the *semantic web* is defined as the means by which this could be reached. At the technological side the CORPORUM tool set is defined as the server for semantic analyses ([BJ99], [EB00]). These analyses are performed by CORPORUMs' core component, a semantic analyser component called MiMir. Whereas MiMir is the core analysis component in the CORPORUM tool set, this very component can be used in a variety of settings due to its ability to extract contents, generate a semantic representation of the concepts (implicit as well as explicitly present in texts), relationships and roles. MiMirs' functionality is based on more formal computational linguistics. The computational linguistic paradigm (cf. figure 2 and [EB00]) takes place on three main levels: the *phonological level*, *word level*, *sentence level* and the *supra-sentential level* (very similar to the *discourse level*).

On top of this basic functionality, the analyser component has the ability to compare such representations of meaning, in order to find out how similar they are. Based on the results of this similarity analysis, the MiMir component offers advice on which documents are most pertinent to a specific analysed text, and return those parts in targeted documents most similar to a particular input text (cf. figure 1).

As soon as embedded in the CORPORUM tool set, the MiMir component is able to unleash its real strengths and serves as the 'brains' of intelli-

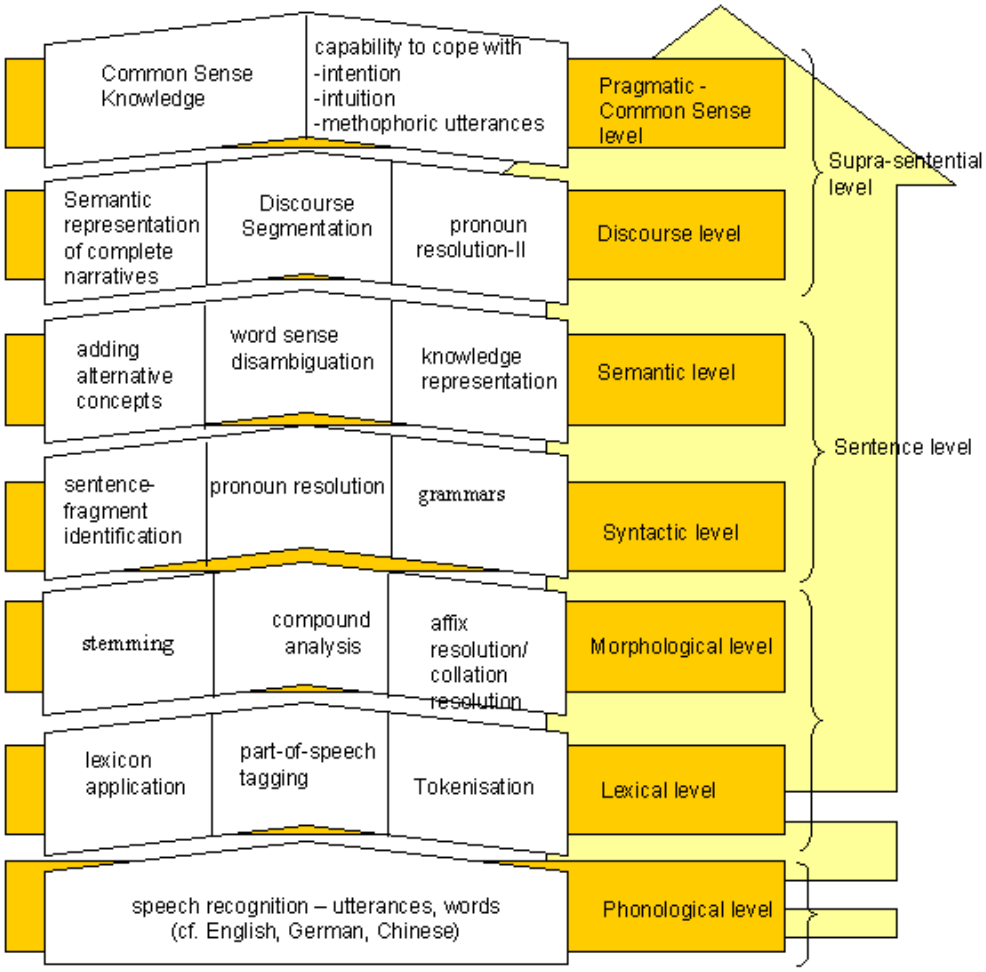


Figure 2: Grammatical analysis of texts.

gent agents gathering intelligence of all sorts on the web, be it medical knowledge for a specific new pro-leukine based medicine, a student wanting to collect material for a course or business intelligence about potential market opportunities or treats. For the intelligent agent scenario, a web server component, a database server, mission schedulers and a client server component are included. Another component in the CORPORUM tool set is the Summariser, which is capable of making summaries of texts based on a MiMir supported analysis of where the real information contents reside in the document. Alternatively such summaries can be made interest-driven, by using an interest profile (in the form of a natural language text) and generate summaries according to these.

Reflecting on the above, it can be said that three main scenario's for application of MiMir are most pertinent: a) *extraction* of information from texts for building knowledge bases, b) *retrieval* of information from other sources (search scenarios) and c) strategies to compact, visualise and disseminate information to people (dissemination and navigation). With the *semantic web* philosophy of an explicitly represented semantics as a given (RDF/OIL), the scenarios b) and c) become less important for the current discussion and we will therefore only pro-

```
<?xml version="1.0"?>
<!DOCTYPE CONCEPTGRAPH []>
<CONCEPTGRAPH>
  <CONCEPTLIST>
    <CONCEPT>
     <NAME>text interpretation program</NAME>
    </CONCEPT>
    <CONCEPT>
     <NAME>text analysis engine</NAME>
    </CONCEPT>
  </CONCEPTLIST>

  <INSTANCELIST>
     <INSTANCE>
      <NAME>corporum</NAME>
    </INSTANCE>
    <INSTANCE>
     <NAME>mimir</NAME>
    </INSTANCE>
  </INSTANCELIST>

  <RELATIONLIST>
      <RELATION TYPE="ISA">
         <CONCEPTNAME>corporum</CONCEPTNAME>
         <STRENGTH>0.4000</STRENGTH>
         <CONCEPTNAME>text interpretation program
              </CONCEPTNAME>
      </RELATION>
      <RELATION TYPE="ISA">
         <CONCEPTNAME>corporum</CONCEPTNAME>
         <STRENGTH>0.4000</STRENGTH>
         <CONCEPTNAME>text analysis engine</CONCEPTNAME>
      </RELATION>
      <RELATION TYPE="ISA">
         <CONCEPTNAME>mimir</CONCEPTNAME>
         <STRENGTH>0.7000</STRENGTH>
         <CONCEPTNAME>text analysis engine
              </CONCEPTNAME>
      </RELATION>

      <RELATION TYPE="UNIV">
         <CONCEPTNAME>corporum</CONCEPTNAME>
         <STRENGTH>0.4000</STRENGTH>
         <CONCEPTNAME>mimir</CONCEPTNAME>
      </RELATION>

              .....


      <RELATION TYPE="SUBCLASSOF">
         <CONCEPTNAME>text interpretation program
              </CONCEPTNAME>
         <STRENGTH>0.1000</STRENGTH>
         <CONCEPTNAME>program</CONCEPTNAME>
      </RELATION>

      <RELATION TYPE="SUBCLASSOF">
         <CONCEPTNAME>text analysis engine
              </CONCEPTNAME>
         <STRENGTH>0.1000</STRENGTH>
         <CONCEPTNAME>engine</CONCEPTNAME>
      </RELATION>

  </RELATIONLIST>
</CONCEPTGRAPH>
```

Figure 3: An XML export based on a SemStruc.

vide short examples of them. Focus of this text will be on the generation of explicit knowledge and information from a specific text, so that it can be used for building knowledge bases and question answering (cf. RDF query language and tools [KCPA00]). Eventually generated knowledge bases contain results of semantical analysis of (web) documents and techniques to "mine" the underlying concepts and relations.

2.1 Making content explicit

For the question answering scenarios, but even for visualisation of contents for easy graphical interpretation, the content of the texts found on f.e. the web should be made explicit. There are several ways of performing content analysis, all having their own definition of *meaning*. An often found approach to content analysis is the statistical approach. In such approaches, words are not regarded as representing real-world artifacts of specific sorts, but are merely seen as patterns with statistical properties (frequencies and co-occurance frequencies). Typically an advantage of such an approach is that information retrieval can be made relatively language independent (pattern matching is universal), and is implemented rather computationally efficient. Instead of using pure statistical methods, Vector Space Models possibly combined with neural net technology or genetic algorithms, are also used. A mayor drawback of such approaches is the fact that elements in word-vectors typically have to be exact matches, causing certain word forms not to be recognized as being similar, even if they principally are (cf. plural and singular forms of words, different tenses of verbs, etc.). Whereas the problem with different suffixes (as in plu-

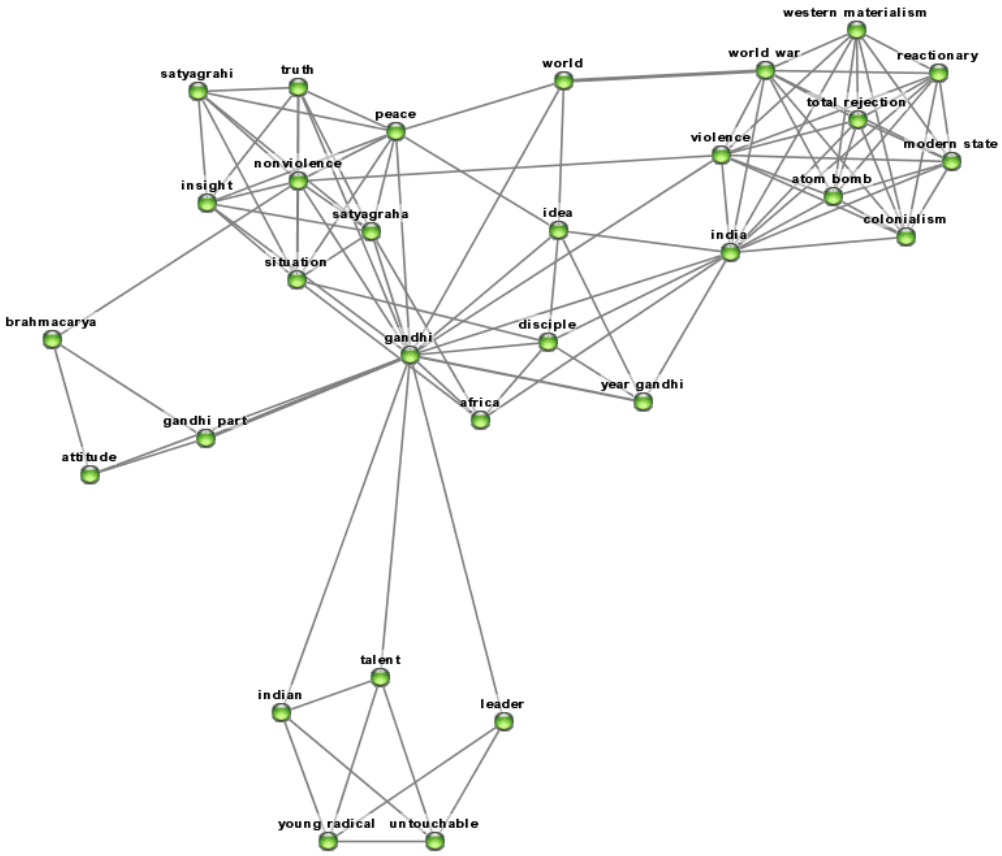


Figure 4: A visualised SemStruc generated from a CogLet (simplified version)

ral/singular and verb-tenses) has some solutions, there is less clarity on how to deal with synonyms, antonyms, etc.

On the other hand, there are pure formal grammar approaches, aiming to grasp meaning and semantics in a more formal sense. The definition of content as used in the *semantic web* defines meaning as an *explicit representation of the intention of a texts' author*. A natural way to represent such an explicit representation could be a (graphical) structure containing all concepts that play a role in a certain discourse, including intended concepts and relations between those (cf. three related concepts "Luther", "Martin" and "King" vs. a single 'intended' concept "Martin Luther King" bearing all semantic information in a single artifact. The concept of MLK could then be related to for example the concept of "civil rights leader" through grammatical sentence analysis). On top of this basic structure, concepts can be classified (f.e. 'instance', 'concepts', 'numbers' and 'names'). Currently MiMir is able to grasp the difference between specific types of relations that hold as well as a categorisation of the concepts it deals with. This capability makes MiMir not only suitable for typical Information Retrieval tasks, but also supports knowledge building for the semantic web and provides the information needed for *question answering*.

Linguistic Text Analysis

As discussed above, the basic analysis of a text as performed by MiMir is based on a tokeniser, a Part-Of-Speech tagger, stemming algorithms, Named Entity recognition facilities as well as a propri-

etary algorithm for generating concepts out of single words (cf. figure 2). From the information that is gathered during these analyses, a CogLet representation is generated which puts all information in relation and defines the context in which information should be interpreted.

The information residing in a single CogLet can now be used to export semantic structures (so-called *SemStrucs*). SemStrucs can be represented in XML format, which could be fed into visualisation algorithms. CogLets also contain the necessary information to analyse web pages and augment them with a Resource Description Framework (RDF) part describing document meta data (according to Dublin Core) and a lightweight ontology based on the analysed natural language text contained in the document.

Semantic Structures in XML

Information contained in a CogLet can also be exported into an XML format, so that it can be used as semantic annotation on a web site or in a knowledge base. XML has been chosen because it is regarded as the next step upward from standard HTML annotations. Only a subset of the information in the CogLet is used for the XML generation, while containing enough information for the generation of graphics.

Figure 3 provides an example of such an XML representation. Relations in such visualisations are not only typed, but also annotated with a calculated heuristic strength. The XML represented information in figure 3 could be used as input for the graph visualiser.

Visualising Semantic Structures

As mentioned before, one of the strengths of SemStructs is that they can be used for visualisation interests and contents. This capability allows for usage in visual browsers and navigators based on larger document sets, and to offer people an at-a-glance overview over the information they have access to.

Figure 4 shows a simplified[1] structure created from a SemStruc generated by a CogLet and visualised with CCAviewer[2]. The structure shows the semantic clusters around the person "Ghandi". There are three main clusters recognisable, one dealing with Ghandi's roles (`<young radical>`, `<leader>` and `<talent>`), one dealing with his philosophy (`<satyagraha>`, `<non-violence>` and `<insight>`) and one dealing with the violent world Ghandi fought against (`<colonialism>`, `<violence>`, `<total rejection>`, `<western materialism>`).

Pictures that are thus automatically generated from natural language texts provide an at-a-glance overview over a piece of information. Such pictures can then be used in order to augment executive summaries and readers aids, but they are also used as visual interfaces to databases (preferably in corporate settings). As such they augment knowledge management systems, where they provide a visual entrance to pieces of information pertinent to specific interest groups within an enterprise.

As an example of the expressive power of the SemStrucs, one might take some time to analyse figure 4 and try to imagine what the original text is about, and which 'discourses' the original document contained.

Augmenting web sites with RDF

Within the OnToKnowledge project, RDF with extensions are used as representation language for the *semantic web* (cf. OTK: [FvHKA99], OIL: [HFB+99]). The CORPORUM$_{OntoExtract}$ component is directed to the generation of a

[1] As SemStrucs represented in XML/RDF/OIL are formal representations, they will easily grow too large for inclusion in a paper. Hence the very short 'CORPORUM' text example. The visualisation of SemStrucs *condenses* texts, and could therefore be based on a larger-sized text (about Ghandi).

[2] The CCA viewer is a product by Aidministrator Nederland BV. It uses CogLet generated SemStrucs to generate pictures based on so-called augmented Spring Embedder technology (cf. figure 4). CCA stands for Central Concept Area, referring to the information created by the SemStrucs.

'light-weight ontology' based on linguistic analysis by CORPORUM in combination with the information that SemStrucs can provide. This means that formal taxonomic relationships that hold in the discourse at hand are disclosed and made explicit as a set of RDF tuples. Additionally, traditional web pages are augmented with Dublin Core meta data, also generated automatically by the CORPORUM$_{OntoExtract}$ component[3].

Figure 5 provides an example of such automatically generated DC and ontologic knowledge. The attentive reader will notice that there are two constructs declared that are not used in the example, i.e. `<isRelated>` and `<hasSomeProperty>`. These two constructs are defined in the OIL language. Whereas a typical ontology often represents a taxonomy (the ontology in the example is no exception on this), `<isRelated>` refers to cross-taxonomic links that can hold within a domain and, if represented, can make a difference in finding needed information based on context descriptions. As an example one can imagine two CCA concepts like `<oil-rig>` and `<ship>`. Such concepts are not typically 'close' in a traditional ontology, where they are not found as sub-classes of vehicles (`<oil-rigs>` are not typically means of transportation), neither as sub-class of a concept like `<building>`, `<floating device>`, etc. Nevertheless, people working in the oil industry typically regard the two concepts as highly related, not in the least due to their natural 'symbiosis' in everyday 'life on the rig'. CORPORUM is however able to capture such *cross-taxonomic* links and represent them using the `<isRelated>` structure.

The other construct (`<hasSomeProperty>`) is the most general, universal relation type reflecting *part-whole* relations within a taxonomy. It is currently used in CORPORUM$_{OntoExtract}$ to define not further specified *part-whole* relationships between a `<concept>` and a `<specifier>` of that concept. However, in some cases there is knowledge available from the Knowledge Base that allows us to further refine the type of properties are actually present. In such cases, CORPORUM$_{OntoExtract}$ will query the KB in order to find out how it can enhance its knowledge representation. At current this process extends Ontology generation in RDF from being single text based linguistic analysis into an augmentation process where previously generated ontologic knowledge (containing "background" knowledge about the domain) is taken into consideration as much as possible. After having augmenting the ontology generated by CORPORUM$_{OntoExtract}$ thus, the complete RDF(S) representation is send to the RDF repository maintained for ontology storage (OntoKnowledge - Sesame at the moment).

[3]DC meta data includes information about author, key concepts, summary of the content of a document, its URI, etc. Dublin Core meta data is described at: `http://purl.oclc.org/dc`.

Future Developments

While used in a variety of commercial products, ranging from *Intelligence Portals*, *Intelligent Crawler Systems* to *Summarising Tools* and *Visualising Components*, the CORPORUM tool set is subject to continuous improvement. Currently the system is able to deal with English, German and Norwegian texts, whereas more of the European languages (French, Spanish, Dutch) are expected to be added soon.

The MiMir component is also subject to continuous improvement, so that the CogLet generated SemStrucs get an even richer 'meaning' representation model. At the same time the functionality of the core MiMir component is enhanced in such a way that it can serve many more tasks (think of enhanced summarising, including smoothing, discourse recognition, as well as a more flexible Natural Language based readers aid.).

An issue that is currently dealt with is directed to scenarios where one wants to 'answer questions'. In such cases

Figure 5: An excerpt of CORPORUM$_{OntoExtract}$ generated RDF annotation including Dublin Core meta data.

more separated information should be available that f.e. can make the difference between a *concept* (i.e. `<car manufacturer>`) and an *instance* thereof (i.e. 'Renault'). The question what the difference between *instances* and *concepts* actually is is not always straightforwardly answered, as can be learned from ongoing discussions at the academic level on this topic. Therefore further development of CORPORUM$_{OntoExtract}$ within the OnToKnowledge project will be directed towards the (semi-?) automatic generation of RDF represented 'semantic' knowledge, which is to be used by reasoning and query engines developed within the very same project. More specifically, the algorithms defining the `<isRelated>` relationships will be refined in order to more precisely reflect the specifiers of concepts holding in specific domains (i.e. instead of currently stating that a specific instance`<car_01>` has a property `<isRelated>` with value `<red>`, it might be able to refine the `<isRelated>` property with a subrelation `<hasColour>` with the same value.

The CORPORUM tool set as such tends to grow with the functionality of its core component as well as with the imagination of and familiarity with Knowledge Management scenarios by key 'Knowledge Managers' in the large enterprises we cooperate with. It is our experience that in many situations there is a larger problem in making people understand the potential on a human and organisational level of semantic tools, as that there is showing the technical principles behind it. One can discuss why this is the case, the main reason possible being that larger enterprises tend to have capable people working with what we would call 'Knowledge Management', although the enterprise as a whole does not always seem to realise enough the benefits of actually integrating/implementing solutions developed by such KM departments at a company width scale. An acceptance of the industry of the possibilities of the semantic web should be boosted by the availability of tools to support it. Although currently only tested in smaller, controlled environments, the tool set discussed in this paper seems to address many of the issues raised in this paper.

References

[aA00] Knowledge Management Group at AIFB. Ontology Engineering Environment OntoEdit. Technical report, Angewandte Informatik und Formale Beschreibungsverfahren, University of Karlsruhe, D, http://ontoserver.aifb.uni-karlsruhe.de, 2000.

[BG00] D. Brickley and R.V. Guha. Resource Description Framework (RDF) Schema Specification 1.0. Technical report, W3C Consortium, http://www.w3.org/TR/rdf-schema/, 2000.

[BJ99] B. Bremdal and F. Johansen. CORPORUM; Technology and Applications. Technical report, CognIT a.s, Halden, Norway, http://www.cognit.no/, 1999.

[BJS+99] B. A. Bremdal, F. Johansen, Ch. Spaggiari, R. Engels, and R. Jones. Creating a Learning Organisation through Content Based Document Management. Technical report, CognIT a.s, Halden, Norway, http://www.cognit.no/, 1999.

[EB00] R.H.P. Engels and B.A. Bremdal. Information Extraction. Technical report, OnToKnowledge Consortium, http://www.ontoknowledge.org/del.shtml, 2000.

[FHH+00] D. Fensel, I. Horrocks, F. Van Harmelen, S. Decker, M. Erdmann, and M. Klein. OIL in a nutshell. In R. Dieng et al., editor, *Knowledge Acquisition, Modeling and Management: Proceedings of the European Knowledge Acquisition Conference (EKAW-2000)*. Springer, Berlin, Heidelberg, New York, 2000.

[FvHKA99] D. Fensel, F. van Harmelen, M. Klein, and H. Akkermans. On-To-Knoweldge: Ontology-based Tools for Knowledge Management. In *Proceedings of the Ebusiness and Ecommerce Conference*, Madrid, Spain, 1999.

[Hen00] J. Hendler. DARPA Agent Markup Language. Technical report, Defense Advanced Research Projects Agency (DARPA), http://www.daml.org/, 2000.

[HFB+99] I. Horrocks, D. Fensel, J. Broekstra, S. Decker, M. Erdmann, C. Goble, F. van Harmelen, M. Klein, S. Staab, and R. Studer. The Ontology Inference Layer OIL. Technical report, OnToKnowledge EU-IST-10132 Project Deliverable No. OTK-D1, http://www.ontoknowledge.org, 1999.

[KCPA00] G. Karvounarakis, V. Christophides, D. Plexousasikis, and S. Alexaki. Querying Community Web Portals. Technical report, ICS-FORTH, Heraklion, Greece, http://www.ics.forth.gr/proj/isst/RDF/RQL/rql.(html, pdf, ps, dvi), 2000.

[oMAG00] University of Manchester, Free University Amsterdam, and Interprice GmbH. OilEd. Technical report, University of Manchester, UK, http://img.cs.man.ac.uk/oil/, 2000.

Acquiring Conceptual Relationships from a MRD and Text Corpus

Masaki Kurematsu[1], and Naomi Nakaya[2] and Takahira Yamaguchi[2]

[1] Faculty of Software and Information Science, Iwate Prefectural University
152–52 Takizawasugo Takizawa Iwate 020-0193 JAPAN
kure@soft.iwate-pu.ac.jp
[2] Dept. Computer Science, Shizuoka University
3–5–1 Johoku Hamamatsu Shizuoka 432-8011 JAPAN
{cs7068, yamaguti}@cs.inf.shizuoka.ac.jp

Abstract. How to exploit a machine-readable dictionary (MRD) and text corpus in supporting the construction of domain ontologies that specify taxonomic and non-taxonomic relationships among given domain concepts are discussed here. a) In building taxonomic relationships (hierarchy structure) of domain concepts, some hierarchy structure can be extracted from MRD with marked sub-trees that may be modified by a domain expert, using both matching result analysis and trimmed result analysis. Domain-specific hierarchical structure can also be extracted from text corpus, using pairs of concepts that turn to be located near and have similar context by WordSpace. Thus two different kinds of hierarchical structure change into unified one with additional modification by a domain expert. b) In building non-taxonomic relationships (specification templates) of domain concepts, we construct concept specification templates that come from pairs of concepts that turn to be located near and have similar context by WordSpace. A domain expert does the task based on them later. The case study with some law called CISG shows us that the trade-off between precision and recall is so important in practically building domain ontologies.

1 Introduction

Although ontologies have been very popular in many application areas, we still face the problem of high cost associated with building up them manually. In particular, since domain ontologies have the meaning specific to application domains, human experts have to make huge efforts for constructing them entirely by hand.

In order to reduce the costs, automatic or semi-automatic methods have been proposed using knowledge engineering techniques and natural language

This paper is identical to the one published in IJCAI'01 Workshop on Ontology Learning

processing ones (cf. Ontosaurus [Swartout et. al. 1996]). The authors have also developed a domain ontology refinement support environment called LODE [Kurematsu and Yamaguchi 1997] and a domain ontology rapid development environment called DODDLE [Sekiuchi et. al. 1998], using machine readable dictionaries. However, these environments facilitate the construction of only a hierarchically structured set of domain concepts, in other words, taxonomic conceptual relationships.

As domain ontologies have been applied to widespread areas, such as knowledge sharing, knowledge reuse, software agents and information integration, we need software environments that support a human expert in constructing the domain ontologies with not only taxonomic conceptual relationships but also non-taxonomic ones. In order to develop the environments, it seems better to put together two or more techniques such as knowledge engineering, natural language processing, machine learning and data engineering, as seen in the workshop on ontology learning in ECAI2000 (e.g. [Maedche and Staab 2000]).

Here in this paper, we extend DODDLE into DODDLE II that constructs both taxonomic and non-taxonomic conceptual relationships, exploiting WordNet [Fellbaum 1998] and domain-specific texts with the automatic analysis of lexical co-occurrence statistics, based on WordSpace [Marti and Schutze] that has the idea that a pair of terms with high frequency of co-occurrence statistics can have non-taxonomic conceptual relationships. Furthermore, we evaluate how DODDLE II works in the field of law, the Contracts for the International Sale of Goods (CISG). The empirical results show us that DODDLE II can support a law expert in constructing domain ontologies.

2 DODDLE II: A Domain Ontology Rapid Development Environment

Figure 1 shows an overview of DODDLE II, "a Domain Ontology rapiD Development Environment" that has the following two components:

- Taxonomic relationship acquisition module using WordNet
- Non-taxonomic relationship learning module using domain-specific texts

A domain expert gives a set of domain terms to the system.
A) The taxonomic relationship acquisition module (TRA module) does "spell match" between the input domain terms and WordNet. The "spell match" links these terms to WordNet. Thus the initial model from the "spell match" results is a hierarchically structured set of all the nodes on the path from these terms to the root of WordNet. However the initial model has unnecessary internal terms (nodes). They do not contribute to keeping topological relationships among matched nodes, such as parent-child relationship and sibling relationship. So we can trim the unnecessary internal nodes from the initial model into a trimmed model, as shown in Figure 2. In order to refine the trimmed model, we have the following three strategies that we will describe later in the context of interaction with an user:

- Matched result analysis

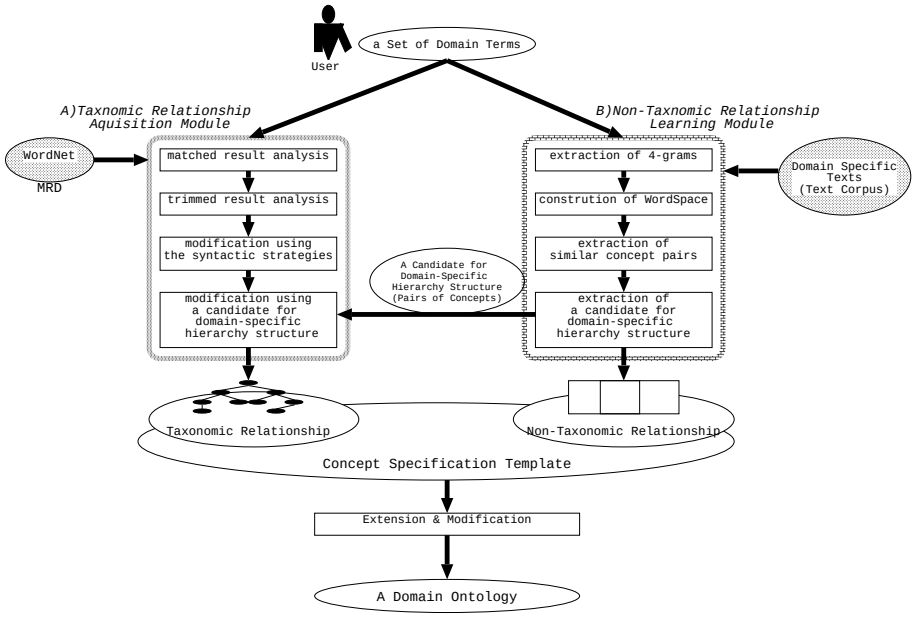


Fig. 1. DODDLE II overview

- Trimmed result analysis
- Using a candidate for domain-specific hierarchy structure extracted from text corpus.

B) The non-taxonomic relationship learning module (NTRL module) extracts the pairs of terms that should be related by some relationship from domain-specific texts, analyzing lexical co-occurrence statistics, based on WordSpace that is a multi-dimensional, real-valued vector space where the cosine of the angle between two vectors is a continuous measure of their semantic relatedness. Thus the pairs of terms extracted from domain-specific texts are the candidates for non-taxonomic relationships. We can built concept specification templates by putting together taxonomic and non-taxonomic relationships for the input domain terms. The relationships should be identified in the interaction with a human expert.

3 Taxonomic Relationship Acquisition

After getting the trimmed model, TRA module is refined by interaction with a domain expert, using the following three strategies: matched result analysis, trimmed result analysis and using domain-specific hierarchy structure extracted from text corpus.

Looking at the trimmed model, it turns out that it is divided into a PAB (a PAth including only Best spell-matched nodes) and a STM (a Sub-Tree that

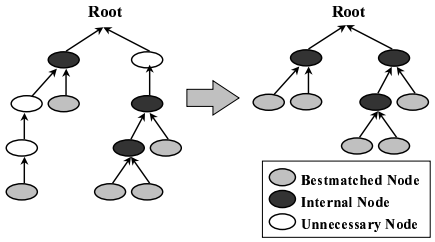


Fig. 2. Trimming Process

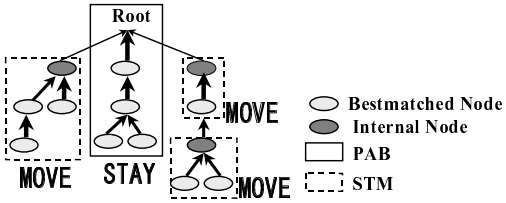


Fig. 3. Matched Result Analysis

includes best spell-matched nodes and other nodes and so can be moved) based on the distribution of best-matched nodes. On one hand, a PAB is a path that includes only best-matched nodes that have the senses good for given domain specificity. Because all nodes have already been adjusted to the domain in PABs, PABs can stay in the trimmed model. On the other hand, a STM is such a sub-tree that an internal node is a root and the subordinates are only best-matched nodes. Because internal nodes have not been confirmed to have the senses good for a given domain, a STM can be moved in the trimmed model. Thus DODDLE II identifies PABs and STMs in the trimmed model automatically and then supports a user in constructing a conceptual hierarchy by moving STMs. Figure 3 illustrates the above-mentioned matched result analysis.

In order to refine the trimmed model, DODDLE II can use trimmed result analysis as well as matched result analysis. Taking some sibling nodes with the same parent node, there may be many differences about the number of trimmed nodes between them and the parent node. When such a big difference comes up on a sub-tree in the trimmed model, it is be better to change the structure of the sub-tree. DODDLE II asks the user if the sub-tree should be reconstructed or not. Based on the empirical analysis, the sub-trees with two or more differences

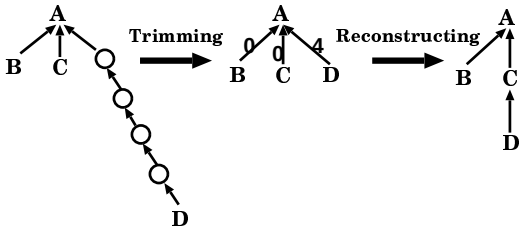


Fig. 4. Trimmed Result Analysis

may be reconstructed. Figure 4 illustrates the above-mentioned trimmed result analysis.

TRA module can not make suggestions about domain-specific hierarchy structure using above-mentioned strategies. Because these strategies don't know domain specific information. So, in addition to do that, TRA module makes suggestions using a candidate for domain-specific hierarchy structure extracted from text corpus by NTRL module. Domain-specific hierarchy structure is a set of pair of concepts. We will describe how to extract the structure later. When there are two different kinds of hierarchy structure between two concepts, DODDLE II asks the user if the hierarchy structure should be changed into unified one or not.

Finally DODDLE II completes taxonomic relationships of the input domain terms manually from the user.

4 Non-Taxonomic Relationship Learning

Non-taxonomic Relationship Learning almost comes from WordSpace, which derives lexical co-occurrence information from a large text corpus and is a multi-dimension vector space (a set of vectors). The inner product between two word vectors works as the measure of their semantic relatedness. When two words' inner product is beyond some upper bound, there are possibilities to have some non-taxonomic relationship between them.

4.1 Construction of WordSpace

WordSpace is constructed as shown in Figure 5.

1. extraction of high-frequency 4-grams Since letter-by-letter co-occurrence information becomes too much and so often irrelevant, we take term-by-term co-occurrence information in four words (4-gram) as the primitive to make up co-occurrence matrix useful to represent context of a text. We take high frequency 4-grams in order to make up WordSpace.

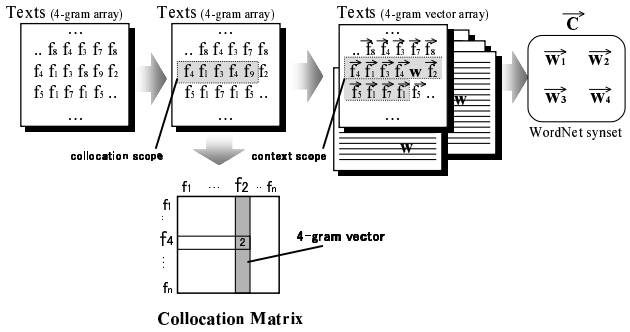


Fig. 5. Construction Flow of WordSpace

2. construction of collocation matrix A *collocation matrix* is constructed in order to compare the context of two 4-grams. Element $a_{i,j}$ in this matrix is the number of 4-gram f_i which comes up just before 4-gram f_j (called *collocation area*). The collocation matrix counts how many other 4-grams come up before the target 4-gram. Each column of this matrix is the *4-gram vector* of the 4-gram f.
3. construction of context vectors A *context vector* represents context of a word or phrase in a text. A sum of 4-gram vectors around appearance place of a word or phrase (called *context area*) is a context vector of a word or phrase in the place.
4. construction of word vectors A word vector is a sum of context vectors at all appearance places of a word or phrase within texts, and can be expressed with the following formula. Here, $\tau(w)$ is a vector representation of a word or phrase w, $C(w)$ is appearance places of a word or phrase w in a text, and $\varphi(f)$ is a 4-gram vector of a 4-gram f. A set of vector $\tau(w)$ is WordSpace.

$$\tau(w) = \sum_{i \in C(w)} \Big(\sum_{f \text{ close to } i} \varphi(f) \Big)$$

5. construction of vector representations of all concepts The best matched "synset" of each input terms in WordNet is already specified, and a sum of the word vector contained in these synsets is set to the vector representation of a concept corresponding to a input term. The concept label is the input term.

4.2 Constructing and Modifying Concept Specification Templates

Vector representations of all concepts are obtained by constructing WordSpace. Similarity between concepts is obtained from inner products in all the combination of these vectors. Then we define certain threshold for this similarity. A

concept pair with similarity beyond the threshold is extracted as a similar concept pair. A set of similar concept pairs becomes concept specification templates. Both of the concept pairs, whose meaning is similar (with taxonomic relation), and has something relevant to each other (with non-taxonomic relation), are extracted as concept pairs with context similarity in a mass. However, by using taxonomic information from TRA module with co-occurrence information, DODDLE II distinguishes the concept pairs which are hierarchically close to each other from the other pairs as TAXONOMY.

A user constructs a domain ontology by considering the relation with each concept pair in the concept specification templates, and deleting an unnecessary concept pair.

4.3 Extracting Domain-Specific Hierarchy Structure

In order to make suggestions about domain-specific hierarchy structure, NTRL module tries to extract pairs of concepts which form part of a candidate for domain-specific hierarchy structure. In order to do that, we pay attention to the distance between two concepts in a document. In this paper, the distance between two concepts means the number of words between them. If the distance between two concepts is small and the similarity between them is close, we suppose that one concept explains the other. If the distance is large and the similarity is close, we suppose that they form part of domain-specific hierarchy structure. According to above-mentioned idea, we calculate the proximally rate between two concepts within a certain scope. It is the number of times both concepts occur within the scope divided by the number of times only one concept occurs within it. We define certain threshold for this proximally rate. Pairs of concepts whose proximally rate is within this threshold and the similarity between them is beyond the threshold for similarity are extracted as part of a candidate for domain-specific hierarchy structure.

5 Case Studies for Taxonomic Relationship Acquisition

In order to evaluate how DODDLE is doing in practical fields, case studies have been done in a particular field of law called Contracts for the International Sale of Goods (CISG). Two lawyers joined the case studies. In the first case study, input terms are 46 legal terms from CISG Part-II. In the second case study, they are 103 terms including general terms in an example case and legal terms from CISG articles related with the case. One lawyer did the first case study and the other lawyer did the second.

Table 1 shows the result of the case studies . Figure 6 shows how much of the intermediate products is included in final domain ontology at each DODDLE activity.

Generally speaking, in constructing legal ontologies, 70 % or more support comes from DODDLE. About half portion of the final legal ontology results in the information extracted form WordNet. Because the two strategies just imply

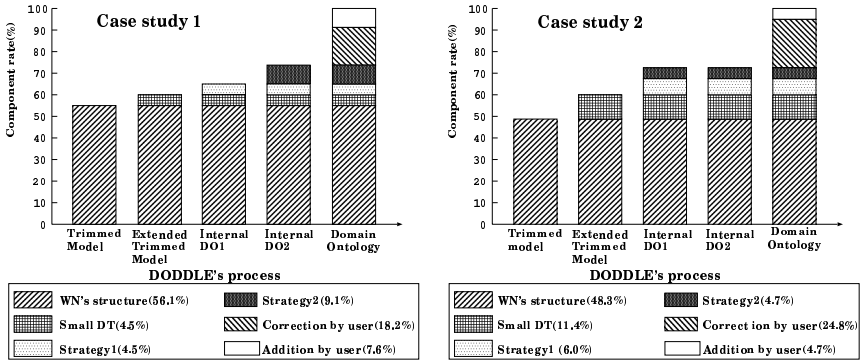


Fig. 6. The Component Rate of the Final Domain Ontology

Table 1. The Case Studies Results

The number of X	The first case study	The second case study
Input terms	46	103
Small DT(Component terms)	2(6)	6(25)
Nodes matched with WordNet(Unmatched)*	42(0)	71(4)
Salient Internal Nodes(Trimmed nodes)	13(58)	27(83)
Small DT integrated into a trimmed model(Unintegrated)	2(0)	5(1)
Modification by the user(Addition)	17(5)	44(7)
Evaluation of strategy1**	4/16(25.0%)	9/29(31.0%)
Evaluation of strategy2**	3/10(30.0%)	4/12(33.3%)

* "Nodes matched with WordNet" is the number of input terms which have be selected proper senses in WordNet and "Unmatched" is not the case.

** The number of suggestions accepted by a user/The number of suggestions generated by DODDLE

the part where concept drift may come up, the part generated by them has low component rates and about 30 % hit rates . So one out of three indications based on the two strategies work well in order to manage concept drift. Because the two strategies use such syntactical feature as matched and trimmed results, the hit rates are not so bad. In order to manage concept drift smartly, we may need to use more semantic information that is not easy to come up in advance in the stratagies.

6 A Case Study for Non-Taxonomic Relationship Learning

DODDLE II, domain ontology rapid development environment, which refer to MRD and domain-specific texts, is being implemented on Perl/Tk now. Figure 7 shows the ontology editor (left window) and the concept graph editor (right window).

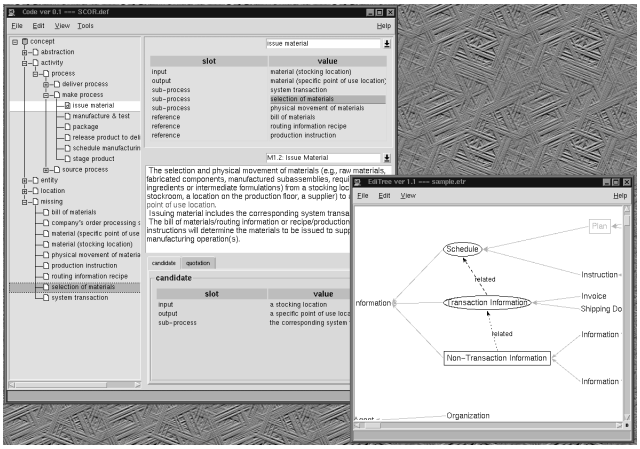


Fig. 7. The Ontology Editor

Table 2. significant 46 concepts in CISG part II

acceptance	delivery	offer	reply
act	discrepancy	offeree	residence
addition	dispatch	offerer	revocation
address	effect	party	silence
assent	envelope	payment	speech act
circumstance	goods	person	telephone
communication system	holiday	place of business	telex
conduct	indication	price	time
contract	intention	proposal	transmission
counteroffer	invitation	quality	withdrawal
day	letter	quantity	
delay	modification	rejection	

Subsequently, as a case study for non-taxonomic relationship acquisition, we constructed the concept definition for significant 46 concepts of having used on the first case study (Table 2) with editing the concept specification template using DODDLE II, and verified usefulness. The concept hierarchy, which the lawyer actually constructed using DODDLE in the first case study was used here (Figure 8).

6.1 Construction of WordSpace

High-frequency 4-grams were extracted from CISG (about 10,000 words) and 526 kinds of 4-grams were obtained. In order to keep density of a collocation matrix high, the extraction frequency of 4-grams must be adjusted according to the scale of text corpus. As CISG is the comparatively small-scale text, the extraction frequency was set as 8 times this case. Then, the collocation matrix was constructed by counting the number of each 526 kinds 4-gram just before a 4-gram for each kind. Since 526 kinds of 4-grams were extracted, the collocation

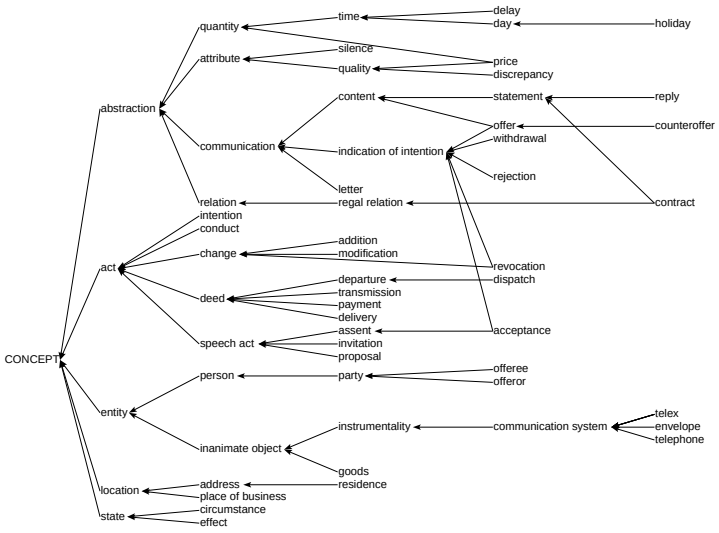


Fig. 8. domain concept hierarchy of CISG part II

matrix also became 526 dimensions. In order to construct a context vector, a sum of 4-gram vectors around appearance place circumference of each of 46 concepts was calculated. One article of CISG consists of about 140 4-grams. The number of 4-gram vectors in context area was set as 60 from experience. For each of 46 concepts, the sum of context vectors in all the appearance places of the concept in CISG was calculated, and the vector representations of the concepts were obtained. The set of these vectors is used as WordSpace to extract concept pairs with context similarity.

6.2 Constructing and Modifying Concept Specification Templates

Having calculated the similarity from the inner product for the 1035 concept pairs which is all the combination of 46 concepts, and having used threshold as 0.9993, 90 concept pairs were extracted, and concept specification templates were constructed. Table 3 is the list of the extracted similar concepts corresponding to each concept. A concept in bold letters is either an ancestor, descendant or a sibling to the left concept in the concept hierarchy constructed using DODDLE in the first case study. In concept specification templates, such a concept is distinguished as TAXONOMY relation. As taxonomic relationships and non-taxonomic relationships may be mixed in the list based on only context similarity, the concept pairs which may be concerned with non-taxonomic relationships are obtained by removing the concept pairs with taxonomic relationships. Figure 9 shows concept specification templates extracted about the concept "assent". The concepts underlined are in taxonomic relation with each other.

Table 3. the concept pairs extracted according to context similarity (threshold 0.9993)

CONCEPT	CONCEPT LIST IN SIMILAR CONTEXT
acceptance	**communication**, **offer**, **indication**, telex
act	offeror, **assent**, effect, payment, person, quantity, time, goods, **delivery**, **dispatch**, price, contract, delay, withdrawal, offeree, place, quality
assent	offeror, **act**, effect, offer, person, offeree, withdrawal, time, **proposal**
communication	**acceptance**, **offer**, **telex**, conduct, **indication**
conduct	party, telex, communication
contract	effect, act, person, delivery, payment, quantity
delay	delivery, offer, act, payment
delivery	**payment**, quantity, goods, place, **act**, delay, time, contract, person, effect, quality
dispatch	goods, price, **act**, person, quantity, offeror
effect	person, assent, act, offeror, contract, proposal, payment, time, withdrawal, party, delivery
goods	dispatch, quantity, delivery, payment, act, person, price, quality
indication	intention, **acceptance**, **communication**
intention	indication
offer	**acceptance**, assent, **communication**, delay
offeree	withdrawal, **offeror**, assent, act, price
offeror	act, assent, withdrawal, **offeree**, **person**, effect, time, price, dispatch
party	conduct, effect, place, **person**
payment	quantity, **delivery**, place, act, goods, quality, delay, effect, person, contract, time
person	effect, **offeror**, act, proposal, goods, assent, withdrawal, contract, dispatch, payment, delivery, **party**, place, price
place	payment, delivery, time, quantity, party, act, person
price	dispatch, act, offeror, goods, withdrawal, offeree, person
proposal	person, effect, withdrawal, **assent**
quality	quantity, payment, goods, act, delivery
quantity	payment, delivery, goods, act, quality, dispatch, place, contract, **time**
telex	conduct, **communication**, acceptance
time	act, offeror, delivery, place, effect, payment, **quantity**, assent
withdrawal	offeree, offeror, person, price, act, assent, effect, proposal

The final concept definition is constructed from consideration of concept pairs in the templates. Figure 10 shows the definition of the concept "assent" constructed from the templates. Although relation AGENT exists also in assent-offeree and assent-offeror, it is represented by definition inheritance and not described.

6.3 Extracting Domain-Specific Hierarchy Structure

We have defined the threshold for the proximally rate as 0.78, the certain scope as the same sentence and tried to extract domain-specific hierarchy structure in the first case study. As a result, DODDLE II extracted 128 pairs of concepts regarded as part of domain-specific hierarchy structure from text corpus. 8 pairs out of them have occurred in the concept hierarchy constructed by the user and have not occurred in the trimmed model. That is, they and modifications by the user were same. It shows that DODDLE II can make useful suggestions about domain-specific hierarchy structure using candidate for them extracted from text corpus. But the rate of same suggestions as modification by the user is about 6%(8/128) and is not good. So, we have to improve extraction of candidate.

assent	*non-TAXONOMY?* :	**offeror**
	TAXONOMY :	**act**
	non-TAXONOMY? :	**effect**
	non-TAXONOMY? :	**offer**
	non-TAXONOMY? :	**person**
	non-TAXONOMY? :	**offeree**
	non-TAXONOMY? :	**withdrawal**
	non-TAXONOMY? :	**time**
	TAXONOMY :	**proposal**

Fig. 9. The concept specification templates for "assent"

assent	AGENT	: **person**
	LEGAL-SEQUENCE	: **offer**
	ANTONYM	: **withdrawal**

Fig. 10. The concept definition for "assent" with editing the templates

6.4 Results and Evaluation

The user with legal knowledge did evaluation about extraction of concept pairs. Note that the concept definition constructed in this case study is only for the 46 concepts as input terms, and is not the whole concept definition which should be constructed from CISG. The detail of the extracted concept pairs in this case study are shown in Table 4.

Taxonomic or non-taxonomic relationships existed in 59% from the top of the list of concept pairs with high context similarity between the concepts. Since a concept pair with high context similarity has a high possibility that it has some kind of relation, concept definitions can be led by considering these pairs.

The problems obtained from this case study are the follows.

Determination of a Threshold Threshold of the context similarity changes in effective value with each domain. It is hard to set up the most effective value in advance. Figure 11 is the relation between the numbers of the extracted concept pairs and recall and precision in this case study.

Specification of a Concept Relation Concept specification templates have only concept pairs based on the context similarity, it requires still high cost to specify relationships between them. It is needed to support specification of concept relationships on this system in the future work.

Ambiguity of Multiple Terminology For example, the term "transmission" is used in two meanings, "transmission (of goods)" and "transmission (of communication)", in the article, but DODDLE II considers these terms as the same and creates WordSpace as it is. Therefore constructed vector expression may not be exact. In order to extract more useful concept pairs, semantic specialization of a multisense word is necessary, and it should be considered that the 4-grams with same appearance and different meaning are different 4-grams.

Table 4. The detail of the extracted concept pairs

Threshold	Extracted concept pair	Advisable	Unknown	Improper
0.9993	90	53	14	23

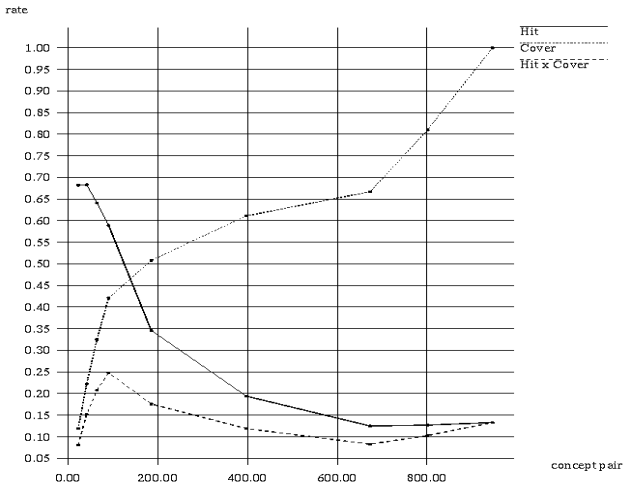


Fig. 11. recall and precision

7 Related Work

In the research using verb-oriented method, the relation of a verb and nouns modified with it is described, and the concept definition is constructed from these information (e.g. [Hahn 1998]). In [Faure and Nédellec 1999], taxonomic relationships and Subcategorization Frame of verbs (SF) are extracted from technical texts using a machine learning method. The nouns in two or more kinds of different SF with a same frame-name and slot-name is gathered as one concept, base class. And ontology with only taxonomic relationships is built by carrying out clustering of the base class further. Moreover, in parallel, Restriction of Selection (RS) which is slot-value in SF is also replaced with the concept with which it is satisfied instantiated SF. However, proper evaluation is not yet done. Since SF represents the syntactic relationships between verb and noun, the step for the conversion to non-taxonomic relationships is necessary.

On the other hand, in ontology learning using data-mining method, discovering non-taxonomic relationships using a association rule algorithm is proposed by [Maedche and Staab 2000]. They extract concept pairs based on the modification information between terms selected with parsing, and made the concept pairs a transaction. By using heuristics with shallow text processing, the generation of a transaction more reflects the syntax of texts. Moreover, RLA, which is their original learning accuracy of non-taxonomic relationships using the existing taxonomic relations, is proposed. The concept pair extraction method in our

paper does not need parsing, and it can also run off context similarity between the terms appeared apart each other in texts or not mediated by the same verb.

8 Conclusion

In this paper, we discussed how to construct a domain ontology using existing MRD and domain-specific texts. In order to acquire taxonomic relationship, two strategies have been proposed: matched result analysis and trimmed result analysis. Furthermore, in order to learn non-taxonomic relationships, concept pairs may be related to concept definition, extracted on the basis of the co-occurrence information in domain-specific texts, and a domain ontology is developed by the modification and specification of concept relations with concept specification templates. It serves as the guideline for narrowing down huge space of concept pairs to construct a domain ontology.

It is almost craft-work to construct a domain ontology, and it is still difficult to obtain the high support rate on system. The DODDLE II mainly supports for construction of a concept hierarchy with taxonomic relationships and extraction of concept pairs with non-taxonomic relationships now. However a support for specification concept relationship is indispensable. The future work follows: improvement in the scalability of the definition support by learning of heuristics, introduction of the useful data-mining method instead of WordSpace, and system integration of taxonomic relationship acquisition module and non-taxonomic relationship learning module (now implementing).

Acknowledgments

We would like to express our thanks to Mr. Takamasa Iwade (a graduate student of shizuoka university) and the members in the Yamaguchi-Lab.

References

[Swartout et. al. 1996] Bill Swartout, Ramesh Patil, Kevin Knight and Tom Russ: "Toward Distributed Use of Large-Scale Ontologies", Proc. of the 10th Knowledge Acquisition Workshop (KAW'96), (1996)

[Kurematsu and Yamaguchi 1997] Masaki Kurematsu and Takahira Yamaguchi: "A Legal Ontology Refinement Support Environment Using a Machine-Readable Dictionary", "*Artificial Intelligence and Law 5*", 119-137, (1997)

[Sekiuchi et. al. 1998] Rieko Sekiuchi, Chizuru Aoki, Masaki Kurematsu and Takahira Yamaguchi: "DODDLE : A Domain Ontology Rapid Development Environment", PRICAI98, (1998)

[Maedche and Staab 2000] Alexander Maedche, Steffen Staab: "Discovering Conceptual Relations from Text", ECAI2000, pp.321-325 (2000)

[Fellbaum 1998] C.Fellbaum ed: "Wordnet", The MIT Press, 1998. see also URL: http://www.cogsci.princeton.edu/~wn/

[Marti and Schutze] Marti A. Hearst, Hinrich Schutze: "Customizing a Lexicon to Better Suit a Computational Task", in Corpus Processing for Lexical Acquisition edited by Branimir Boguraev & James Pustejovsky, pp.77-96

[Hahn 1998] Udo Hahn, Klemens Schnattinger: "*Toward Text Knowledge Engineering*", AAAI98, IAAAI-98 proceedings, pp.524-531 (1998)

[Faure and Nédellec 1999] David Faure, Claire Nédellec, "Knowledge Acquisition of Predicate Argument Structures from Technical Texts Using Machine Learning: The System ASIUM", EKAW'99

[Sono and Yamate 1993] Kazuaki Sono, Masasi Yamate: United Nations convention on Contracts for the International Sale of Goods, Seirin-Shoin(1993)

Ontology Discovery for the Semantic Web Using Hierarchical Clustering

Patrick Clerkin, Pádraig Cunningham, Conor Hayes

Department of Computer Science
Trinity College Dublin
Patrick.Clerkin@cs.tcd.ie
Padraig.Cunningham@cs.tcd.ie
Conor.Hayes@cs.tcd.ie

Abstract. According to a proposal by Tim Berners-Lee, the World Wide Web should be extended to make a Semantic Web where human understandable content is structured in such a way as to make it machine processable. Central to this conception is the establishment of shared ontologies, which specify the fundamental objects and relations important to particular online communities. Normally, such ontologies are hand crafted by domain experts. In this paper we propose that certain techniques employed in data mining tasks can be adopted to automatically discover and generate ontologies. In particular, we focus on the conceptual clustering algorithm, COBWEB, and show that it can be used to generate class hierarchies expressible in RDF Schema. We consider applications of this approach to online communities where recommendation of assets on the basis of user behaviour is the goal, illustrating our arguments with reference to the Smart Radio online song recommendation application.

1 Introduction

Tim Berners-Lee has proposed and extension to the existing World Wide Web known as the Semantic Web (Berners-Lee, 1998, Berners-Lee et al, 2001). Most of the Web's existing content is designed to be read and understood by humans, and cannot readily be parsed and processed by software agents. The central idea behind the Semantic Web is to develop and use machine-understandable languages for the expression of the semantic content of Web pages. This promises to enhance the ability of software agents to navigate the Web's information space and carry out tasks for humans, without the need for sophisticated artificial intelligence.

Central to the Semantic Web project is the concept of an ontology. Web pages are conceived of as being composed of statements relating objects. The denotations of the terms making up the statements need to be fixed relative to a particular universe of discourse, which is represented in an ontology. The ontology codifies a shared and common understanding of some domain. An ontology is usually constructed by domain experts. In this paper, we examine the possibility of generating ontologies automatically using hierarchical conceptual clustering, and consider certain online

communities where such methods are highly appropriate, since there is no existing conceptualisation of the site resources. It is important to emphasis at this point that we are concerned with generating ontologies from behavioural and usage data relating to resources of interest, rather than from the free text data that might be found on web pages.

Since we aim to demonstrate how this technique may be practically implemented, we first present an overview of some of the technologies being used to build the Semantic Web, focusing in particular on how basic ontologies can be represented using RDF Schema (Brickley and Guha, 2000). In subsequent sections we discuss the concept formation system, COBWEB (Fisher, 1987), and demonstrate how the concept hierarchies discovered by this algorithm can be represented as ontologies with RDF Schema. We conclude with a discussion of the application of this approach to online communities - dealing in particular with the Smart Radio system (Hayes and Cunningham, 2000), developed as a test bed for our ideas - and point to some further research directions.

2 Implementing the Semantic Web

The Uniform Resource Identifier (URI)[1] provides the foundation for the Web, since it allows us to give any object or concept a uniquely identifying name. URIs are decentralized, in the sense that no one person or organisation controls their defintion and use. Since anyone can create a URI, we inevitably end up with multiple URIs representing the same thing, so it is important for the Semantic Web to provide a means for resolving names correctly.

This is provided for in the use of an ontology, which usually takes the form of a taxonomy defining classes and relations among them. The meaning of terms can now be resolved if they point to particular ontologies, and if equivalence relationships are defined between ontologies.

The Resource Description Framework (RDF)[2] provides a means for software agents to exchange information on the Web. It defines a simple model for describing relationships between Web resources in terms of properties and their values. However, RDF itself provides no means of declaring such properties. This task is left to RDF Schema, which can be used to represent simple ontologies.

RDF can be written using XML tags, but it is important to note that XML is not sufficient for building the Semantic Web. XML facilitates the arbitrary creation of tags that can be used to annotate Web pages. If a programmer knows in advance what these tags signify, then it is possible for her to write software to process these Web pages automatically. However, in the absence of such knowledge, it is not possible to write such programs, since XML builds no semantics into its structures. RDF, on the other hand, encodes machine-processable structures into its statements. An RDF

[1] See http://www.w3.org/Addressing/ for an overview of naming and addressing schemes used on the World Wide Web.

[2] See http://www.w3.org/RDF/.

statement consists of a triplet, which asserts that a particular thing has a certain property with a certain value. For example, the sentence

```
  Ora Lassila is the creator of the resource
  http://www.w3.org/Home/Lassila.
```

can be represented in RDF by

```
<?xml version="1.0"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:s="http://description.org/schema/">
  <rdf:Description
about="http://www.w3.org/Home/Lassila">
    <s:Creator>Ora Lassila</s:Creator>
  </rdf:Description>
</rdf:RDF>
```

The RDF Schema defines a collection of RDF resources that can be used to describe properties of other RDF resources that define application-specific RDF vocabularies. The RDF Schema type system is similar to the type systems of object-oriented programming languages such as Java. For the purposes of this paper, it is sufficient to present this system in the context of an example. Consider the following class hierarchy. We first define a class MotorVehicle. We then define three subclasses of MotorVehicle, namely PassengerVehicle, Truck and Van. We then define a class Minivan which is a subclass of both Van and PassengerVehicle. In representing this hierarchy we must make use of some core classes and properties defined by RDF Schema. In particular: all things being described by RDF expressions are called *resources*, and are considered to be instances of the class rdfs:Resource; when a resource has an rdf:type property whose value is some specific class, we say that the resource is an *instance of* the specified class; when a schema defines a new class, the resource representing that class must have an rdf:type property whose value is the resource rdfs:Class; the rdfs:subClassOf property specifies a subset/superset relation between classes. Our example class hierarchy is therefore represented by the following diagram:

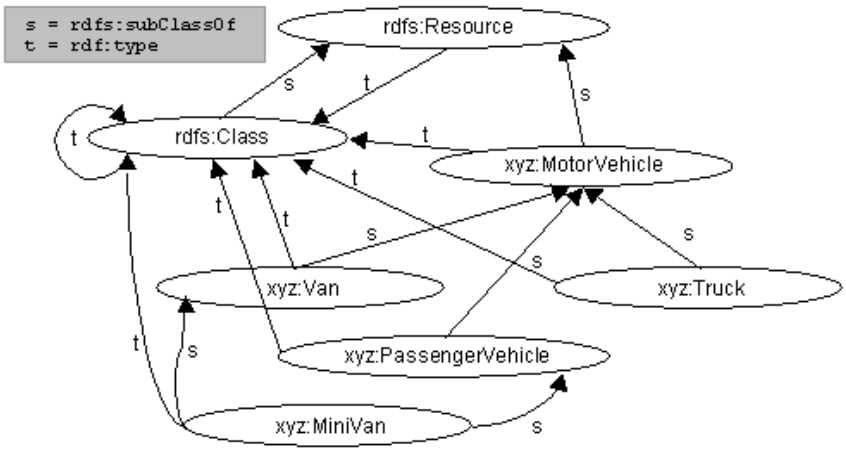


Fig. 1 Class Hierarchy for MotorVehicle class and its subsets (Brickley and Guha, 2000)

This model can be rendered in XML as follows:

```
<rdf:RDF xml:lang="en"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-
ns#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

<rdf:Description ID="MotorVehicle">
 <rdf:type resource="http://www.w3.org/2000/01/rdf-
schema#Class"/>
 <rdfs:subClassOf
  rdf:resource="http://www.w3.org/2000/01/rdf-
schema#Resource"/>
</rdf:Description>

<rdf:Description ID="PassengerVehicle">
 <rdf:type resource="http://www.w3.org/2000/01/rdf-
schema#Class"/>
 <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
</rdf:Description>

<rdf:Description ID="Truck">
 <rdf:type resource="http://www.w3.org/2000/01/rdf-
schema#Class"/>
 <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
</rdf:Description>
```

```
<rdf:Description ID="Van">
 <rdf:type resource="http://www.w3.org/2000/01/rdf-
schema#Class"/>
 <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
</rdf:Description>

<rdf:Description ID="MiniVan">
 <rdf:type resource="http://www.w3.org/2000/01/rdf-
schema#Class"/>
 <rdfs:subClassOf rdf:resource="#Van"/>
 <rdfs:subClassOf rdf:resource="#PassengerVehicle"/>
</rdf:Description>

</rdf:RDF>
```

3. Hierarchical Conceptual Clustering

Data mining is the extraction of implicit, previously unknown, and potentially useful information from data. The objective is to build computer programs that automatically detect regularities or patterns in databases. Useful patterns, if found, should generalise to make accurate predictions on future data. Thus, the final objective of data mining activity is knowledge discovery.

Machine learning provides the technical basis of data mining. It is used to extract information from the raw data in databases. The process is one of abstraction in order to find patterns. Usually, we also require that the system should provide us with an explicit structural description, so as to provide the observer with an explanation of what has been learned and an explanation of the basis for new predictions.

Clustering is a data-mining task that has at its goal the unsupervised classification of a set of objects. Classification is unsupervised in the sense that there are no *a priori* target classes used during training. Clustering techniques rely on the existence of some suitable similarity metric for objects. Clustering algorithms may be classified according to a number of criteria. Some are distance-based and describe clusters purely by enumerating their members, while others represent the clusters by means of a description. This description may take the form of a set of necessary and sufficient conditions for membership of a given cluster, or it may be a probabilistic description where no such set of conditions is tenable. Furthermore, a set of clusters may be "flat" in the sense that no cluster is "contained" in any other cluster, or it may be hierarchical, providing a taxonomy of clusters with definite relationships between them.

COBWEB is an incremental conceptual clustering algorithm with represents concepts probabilistically. It was initially inspired by research on *basic level* effects. For example, humans can typically verify that an item is a bird more quickly than they can verify the same item is an animal, vertebrate, or robin. Thus, the concept of birds is said to reside at the basic level. COBWEB's design assumes that principles which dictate basic concepts in humans are good heuristics for machine concept formation as well.

COBWEB is designed to produce a hierarchical classification scheme. It carries out a hill-climbing search - which consists of taking the current state of the search, expanding it, evaluating the children, selecting the best child for further expansion, etc, and halting when no child is better than its parent – through a space of schemes, and this search is guided by an heuristic measure called *category utility*.
The category utility metric was originally developed by Gluck and Corter (1985) to predict the basic level in human classification categories. In adopting it as a criterion for evaluating concept quality in AI systems, Fisher notes that it can be viewed as a function that rewards traditional virtues held in clustering generally – similarity of objects within the same class, and dissimilarity of objects in different classes.

$$CU(\{C_1, C_2, ..., C_n\}) = \frac{\sum_k \left((P(C_k) \left[P(A_i = V_{ij} \mid C_k)^2 - \sum_i \sum_j P(A_i = V_j)^2 \right] \right)}{n} \quad \textbf{(1)}$$

COBWEB performs its hill-climbing search of the space of possible taxonomies and uses category utility to evaluate and select possible categorisations. It initialises the taxonomy to a single category whose features are those of the first instance. For each subsequent instance, the algorithm begins with the root category and moves through the tree. At each level it uses CU to evaluate the taxonomies resulting from:

1. Classifying the object with respect to an existing class.
2. Creating a new class.
3. Merging: combining two classes into a single class.
4. Splitting: dividing a class into several classes.

4. Ontology generation using COBWEB

We propose that COBWEB may be used to automatically generate ontologies. Let us consider the following artificial and simple example. We have a domain consisting of only four resources, namely, the following four cells:

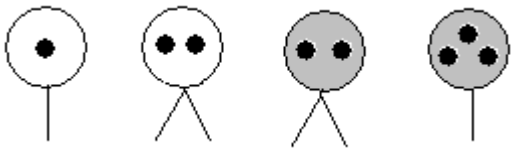


Fig. 2 The four cells to be clustered by COBWEB. (Gennari et al, 1989, Luger and Stubblefield, 1998)

COBWEB generates the following hierarchy of concepts:

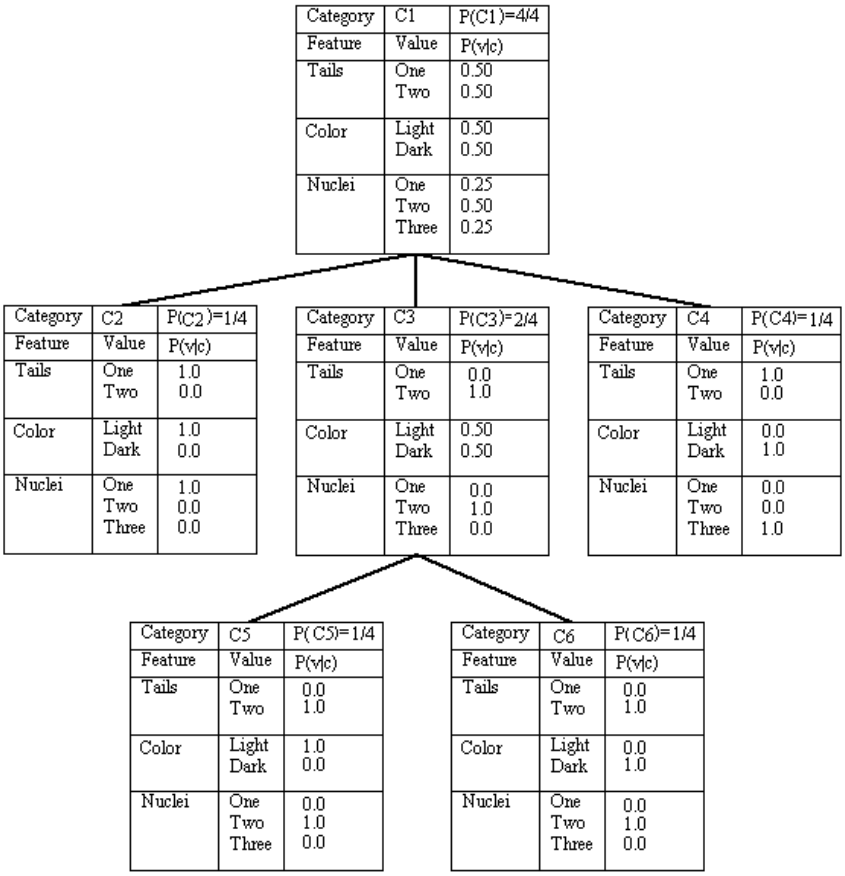


Fig. 3 The concept taxonomy produced by COBWEB. (Gennari et al, 1989, Luger and Stubblefield, 1998)

Just as in our previous example, we can represent this hierarchy in RDF Schema with a diagram:

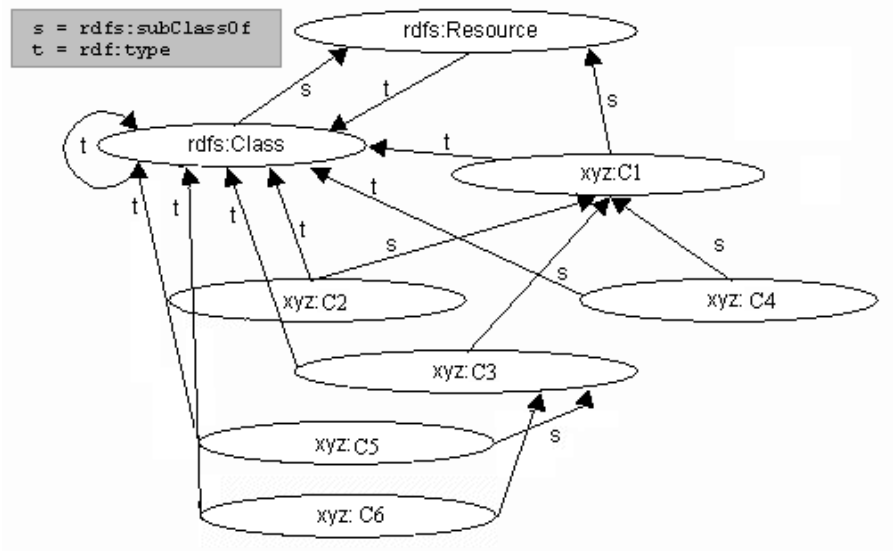


Fig. 4 The Class Hierarchy corresponding to the COBWEB concept taxonomy

The actual XML looks like this:

```
<rdf:RDF xml:lang="en"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-
ns#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
<rdf:Description ID="C1">
 <rdf:type resource="http://www.w3.org/2000/01/rdf-
schema#Class"/>
 <rdfs:subClassOf
  rdf:resource="http://www.w3.org/2000/01/rdf-
schema#Resource"/>
</rdf:Description>
<rdf:Description ID="C2">
 <rdf:type resource="http://www.w3.org/2000/01/rdf-
schema#Class"/>
 <rdfs:subClassOf rdf:resource="#C1"/>
</rdf:Description>
<rdf:Description ID="C3">
 <rdf:type resource="http://www.w3.org/2000/01/rdf-
schema#Class"/>
 <rdfs:subClassOf rdf:resource="#C1"/>
</rdf:Description>
<rdf:Description ID="C4">
```

```
 <rdf:type resource="http://www.w3.org/2000/01/rdf-
schema#Class"/>
 <rdfs:subClassOf rdf:resource="#C1"/>
</rdf:Description>
<rdf:Description ID="C5">
 <rdf:type resource="http://www.w3.org/2000/01/rdf-
schema#Class"/>
 <rdfs:subClassOf rdf:resource="#C3"/>
</rdf:Description>
<rdf:Description ID="C6">
 <rdf:type resource="http://www.w3.org/2000/01/rdf-
schema#Class"/>
 <rdfs:subClassOf rdf:resource="#C3"/>
</rdf:Description>
</rdf:RDF>
```

5. Applications

Now we turn to the question of an application for this approach to ontology generation. We will discuss this in the context of Smart Radio, which is a web-based song recommendation system that relies on users' ratings of songs. The user builds playlists consisting of ten songs that they choose from the database. They may then listen to the songs and rate them on a scale of one to five, where one indicates a strong dislike for song, and five indicates a strong liking.

Track	Artist	Your Rating (1=bad,5=fab)
Wednesday Night Prayer Meeting	Mingus, Charles	1 2 3 4 X
My Jelly Roll Soul	Mingus, Charles	1 2 3 4 X
Everytime We Say Goodbye	Coltrane, John	1 2 3 X 5
My Funny Valentine	Baker, Chet	1 X 3 4 5
Someone To Watch Over Me	Baker, Chet	X 2 3 4 5
It Ain't Necessarily So	Hancock, Herbie	1 2 3 4 5
St. Thomas	Rollins, Sonny	1 2 3 4 5
Satin Doll	Duke Ellington	1 2 3 X 5
Take the "A" Train	Fitzgerald, Ella	1 2 3 4 5
Freddie Freeloader	Davis, Miles	1 2 3 4 X

Fig. 5 An example Smart Radio playlist showing the songs rated by the user.

This results in something like the following matrix of values:

Table 1. **Smart Radio data showing how users have rated songs.**

	Song 1	Song 2	Song 3	Song 4	Song 5	Song 6
User 1	5	2		4	2	
User 2	2	5	1		2	5
User 3	1	1	5			3
User 4					4	2
User 5	3	1	5	5		
User 6		4	1	1	5	3
User 7	1			1		
User 8		3	1	4	1	5

The Smart Radio system currently relies solely on Automated Collaborative Filtering (ACF) to make recommendations. We are experimenting with using knowledge discovery techniques to enhance the quality of these recommendations. In particular, we have employed the COBWEB algorithm to generate a hierarchy of concepts. To do this, we define a song to be *good* for a user if and only if she gives that song a rating of 4/5 or more; below this, the song is *bad* for the user. Then, we characterise each song in the database as an object with the same number of attributes as there are users in the database. Each attribute can take on the value *good* or *bad* according to how the user rated the song. The following is an example song object, where question marks symbolise missing values:

```
good bad ? ? good ? good ? ? ? ? ? ? good ? ? ? ? ? ?
? ? good ? ? ? ? ? good bad ? ? ? ? good good ? ? ? bad
? ? bad ? bad good ? ? ? bad ? bad ?
```

When COBWEB is run on the complete set of these objects, we acquire a hierarchy of clusters and associated probabilistic descriptions. An example cluster, arbitrarily named ‘C112’, is presented below:

```
['Break On Through (To The Other Side)', 'Mr Tambourine
man', 'Say hello wave goodbye', 'Hallelujah', 'Unfin-
ished Sympathy', 'Where The Wild Roses Grow', 'Please
Forgive Me', 'The Girl From Ipanema', 'Gin Soaked Boy',
'Street Spirit (Fade Out)', "La Femme D'Argent", 'Right
Here, Right Now', 'Redemption Song']
```

We can then go on to translate the output of COBWEB into a class hierarchy that can be rendered in RDF Schema as outlined above.

The advantages of this approach to this sort of domain is best discussed in light of the fundamental goal of the Semantic Web project, which is to create ‘an environment where software agents roaming from page to page can readily carry out sophisticated tasks for users’ (Berners-Lee et al, 2001). In the context of an online music community, one of these tasks is going to be the recommendation of songs to users. A user might, for example, request that an agent finds her new songs similar to those she has liked in the past. Or she might request that new songs be provided which are some-

what dissimilar to those that she has previously encountered, but which she may nevertheless like. An ontology facilitates the fulfilment of these requirements, because similar songs will fall under the same concept, and degrees of similarity/dissimilarity will hopefully be captured in the relationships between concepts.

The usual way of creating an ontology is for domain experts to establish the fundamental concepts, objects, relations, etc, which exist for a given community. This presupposes that these ontological elements can be uncovered *a priori*. However, in domains such as that of Smart Radio, it is not at all clear that any *a priori* analysis by a team of experts could yield the sort of concepts important to recommendation tasks. While songs may be categorised according to artist, and while to a much lesser extent, genres and sub-genres may be employed, this approach is inadequate, since it does not account for the fact that many people like songs that are widely divergent according to the artist and genre criteria. By using such algorithms as COBWEB to cluster songs based on user ratings, we hope to discover structures more truly reflective of the similarities and dissimilarities between songs. We need only evaluate the resulting conceptual structures in terms of their impact on recommendations, and we need not worry that users may be unable to articulate the hypothesised perceived similarities and dissimilarities between songs. Furthermore, we do not expect that the discovered conceptual hierarchy will map onto any existing and already familiar network of human concepts. Rather, we expect to discover structures that it was never feasible for human experts to detect.

A further advantage conferred by the automatic generation of ontologies using COBWEB and related systems is that such concept formation algorithms are *incremental*, in the sense that observations are not processed *en masse*. There is a stream of objects, which is processed over time. In the case of Smart Radio, this means that the conceptual hierarchy can automatically evolve over time as new songs are added to the database, and as new users join the system. This is something that would be very costly if human experts were involved, and yet such a capacity to evolve over time is essential to a constantly expanding online community resource.

Finally, we may wonder at how such automatically generated ontologies, which do not map onto any existing human understandable ontologies, can fulfil the requirement for interoperability across web sites. So far, we have considered how one online community can be held together and enhanced by such ontologies, but we now turn to a consideration of how other online communities with similar assets (in this case, songs) could exchange information, via agents, with our site. In traditional ontology engineering, collaboration is required between the people who run Web sites and online communities. It is no different in the case where we are employing automatic ontology generation techniques. The only difference is that it is not human beings who collaborate, but, rather, machines. If the Smart Radio database is accessible to a COBWEB-based agent, and another, different, database of songs and user ratings from a hypothetical Smart Radio II, is also accessible to the same agent, then there would be no problem in that agent constructing, maintaining, and evolving a shared ontology for both sites. The only limitation is that the agent must be able to understand the structure of all such databases. Collaboration requires a set of standards and conventions for the construction or description of such databases, and such collaboration is entirely within the spirit of the Semantic Web project.

Summary and conclusion

We have discussed how hierarchical clustering algorithms may be employed to automatically construct basic ontologies, and illustrated this in the context of COBWEB and RDF Schema. We have argued that such an approach is highly appropriate to domains where no expert knowledge exists, or where it proves inadequate, and have gone on to propose how we might employ software agents to collaborate, in place of human beings, on the construction of shared ontologies. This benefits recommendation tasks, in particular, by allowing for the evolution of concept hierarchies which do not match any articulated human conceptual structures, but which are, hopefully, closely reflective of the criteria that people employ in rating online assets. If the task of Semantic Web project is to render human understandable resources processable by machines, we might say that the task envisaged here is to extend the resources processable by machines beyond the domain of human understanding – but always with a view to helping humans carry out their online tasks.

References

1. Berners-Lee, T. (1998). Semantic Web Road map. Available online at http://www.w3.org/DesignIssues/Semantic.html.
2. Berners-Lee, T., Hendler, J., Lassila, O. (2001). The Semantic Web. *Scientific American* feature article, May 2001. Available online at http://www.scientificamerican.com/2001/0501issue/0501berners-lee.html.
3. Brickley, D., Guha, R.V., (Eds.) (2000). Resource Description Framework (RDF) Schema Specification 1.0. W3C Candidate Recommendation 27 March 2000. Available online at http://www.w3.org/TR/rdf-schema/.
4. Fisher, D. H. (1987). Knowledge acquisition via incremental conceptual clustering. Machine Learning, 2, 139-172.
5. Gennari, J.H., Langley, P., Fisher, D. (1998). Models of incremental concept formation. Artificial Intelligence, 40(1-3):11-62.
6. Gluck, M.A., Corter, J.E. (1985). Information, uncertainty, and the utility of categories. *Proceedings of the Seventh Annual Conference on Artificial Intelligence*, pp. 831-836, Detroit, MI: Morgan Kaufmann.
7. Hayes, C., Cunningham, P. (2000) *Smart Radio: Building Music Radio on the Fly*. Expert Systems 2000, Cambridge, UK, December 2000.
8. Luger, G.F., Stubblefield, W.A., (1998). *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*. Third Edition. Addison-Wesley, 1998.

Web Directories as Training Data for Automated Metadata Extraction

Martin Kavalec, Vojtěch Svátek and Petr Strossa

Department of Information and Knowledge Engineering
University of Economics, Prague, 13067 Praha 3, Czech Republic
e-mail: kavalec@vse.cz, svatek@vse.cz, kizips@vse.cz

1 Introduction

Although *man-made* annotations are considered as the main 'knowledge fuel' for the Semantic Web, the majority of existing commercial pages are still poorly equipped with any kind of metadata, never mind the forthcoming standards such as the RDF syntax or the Dublin Core semantics. *Information Extraction*, relying on characteristic patterns in text, can be applied even on such 'legacy' pages, in order to obtain metadata containing, for example, the names, types, and domains of activity of the WWW subjects (companies).

The two decades of development of Information Extraction techniques have shown that extraction patterns applicable on real-world, unstructured text data cannot be satisfactorily prepared by hand; instead, Machine Learning (ML) became the enabling technology. The common assumption in ML-based Information Extraction is that the training cases are chunks of text pre-labelled by a human indexer. This is acceptable for domain-specific resources with limited vocabulary and more-or-less conventional structure, such as computer science department pages [2] or housing advertisement pages [7]. However, if we proceed to broad categories such as 'pages of companies offering products or services', the number of training cases needed will explode, the acquisition of text fragments becomes difficult, and their manual labelling simply infeasible. Yet, there is a promising resource of web data that has already undergone a process of human indexing, of a sort: web directories such as Open Directory or Yahoo!

In this paper, we analyse the possibility of reusing the knowledge embedded in the structure of the directories in order to obtain *labelled* training data for Web Information Extraction with limited human effort. In section 2 we show the results of preliminary experiments consisting in mining the fragments of web pages, obtained with the help of web directory information, for indicator terms usable for subsequent extraction of semantic information from other pages. In section 3 we outline an ontology of web directories, and suggest the way it can be used to refine the above process. In section 4, our approach is compared to some other projects. Finally, in section 5, we summarise our plans for the future.

2 Mining Indicator Terms through Directory Headings

Our assumption is that the *directory headings* (such as `... /Manufacturing /Materials/Metals/Steel/...`) coincide with the generic names of products and services—let us nickname them *informative terms* in this paper—offered by the owners of the pages referenced by the respective directory page. By matching the headings with the page fulltexts, we obtain sentences that contain the informative terms. The terms situated near the informative terms in the structure of the sentence are candidates for *indicator terms*, provided they occur frequently on pages from various domains. The resulting collection of indicator terms can, conversely, play the role of 'extraction patterns' for discovering informative terms in previously unseen pages.

The knowledge asset embedded in web directories is the judgement of human indexers who have assigned the pages under the particular heading(s). Naturally, informative terms on the page need not always correspond to the existing directory headings, e.g. due to synonymy. As consequence, our method will extract (without the help of a thesaurus) only a fraction of the sentences with informative terms. This however does not disqualify the method, since, in this training phase, we aim at discovering indicator terms rather than at identifying the informative terms themselves. The small degree of completeness of the method is actually compensated by the hugeness of the material available[1] in the directories. Namely, the 'Business' subhierarchy of Open Directory (`www.dmoz.org`), which we have exploited in our experiments, points to approx. 150,000 pages overall, each of these containing the 'heading' terms (from the referencing node or one of its ancestors) in two sentences, on the average.

We have tested the training phase of our method on a sample of 14,500 sentences[2] containing the 'heading' terms. The syntactical analysis has been carried out using the *Link Grammar Parser* [6]. The *verbs* which occurred the closest (in the parse tree) to informative terms have been counted, and arranged into a frequency table. In Table 1, the essence of the table is shown, mostly featuring verbs that are likely to be associated with the informative terms (e.g. 'our assortment *includes...*', 'we *manufacture...*', 'in our shop you can *buy...*'). The table contains only the verbs that occurred in at least 50 sentences[3]. We hope to build a more comprehensive collection using a larger sample of pages. Furthermore, the plain verbs (in particular 'to be', which has no significance of its own) can be extended to more complex *phrases*, again via selecting the neighbouring terms with frequent occurrence.

[1] As we dispense with manual labelling, processing a larger sample of data is merely the matter of computer time/storage.

[2] I.e. about 5% of the total of such sentences.

[3] In this display, they are however not arranged according to the relative frequency of occurrence in the neighbourhood of the informative term (P_n), but according to the ratio of this frequency to the relative frequency of occurrence in the whole of the extracted, possibly compound, sentence (P_s). This visibly pushes down the universal verbs such as 'to be'.

P_n/P_s	Verb	P_n	P_s
2.23	includes	0.0048	0.0021
2.20	manufacture	0.0038	0.0017
2.17	buy	0.0038	0.0017
2.09	including	0.0057	0.0027
2.08	supply	0.0036	0.0175
1.96	offers	0.0119	0.0060
1.92	provides	0.0135	0.0070
1.92	offer	0.0200	0.0104
1.89	include	0.0062	0.0032
1.85	specializing	0.0051	0.0027
1.78	providing	0.0091	0.0051
1.70	specializes	0.0045	0.0026
1.66	specialize	0.0053	0.0032
1.56	using	0.0037	0.0024

P_n/P_s	Verb	P_n	P_s
1.55	provide	0.0191	0.0122
1.40	use	0.0046	0.0033
1.39	sell	0.0039	0.0028
1.26	see	0.0046	0.0036
1.25	are	0.0740	0.0589
1.24	were	0.0042	0.0034
1.23	made	0.0040	0.0032
1.22	make	0.0066	0.0054
1.15	need	0.0053	0.0046
1.12	is	0.0988	0.0880
1.05	get	0.0043	0.0041
1.03	find	0.0060	0.0058
1.03	meet	0.0043	0.0041
1.00	related	0.0035	0.0035

Table 1. Frequent verbs in sentences containing the headings

3 Ontology of Web Directory Headings

As we have shown in the previous section, interesting pieces of information can be extracted from web directories even without specific assumptions about the nature of the heading terms. Nevertheless, we believe that only deeper *ontological analysis* of the headings can bring the automated discovery of indicators to its full potential, in particular for complex terms spanning across multiple levels of headings. We will thus now outline an ontology of web directory headings.

The semantic information associated with the particular page, in the context of a web directory, is defined by the sequence (or, several sequences, in the case of a non-tree hierarchy) of headings preceding the node pointing to that page. The headings essentially belong to one of the following classes:

1. 'Entity' terms (most often nouns), which correspond to real-world entities. Note that their meaning may depend on the preceding terms: for example, pages referenced by the node preceded by the subpath `Cranes/Accessories` are likely to offer accessories for cranes but not clothing accessories. Nevertheless, the word 'accessories' can possibly be found on the page even without the attribute 'for cranes', since the latter can be assumed by context.
2. 'Property' terms (most often adjectives), which correspond to properties of entities. They are *restrictive* rather than descriptive, since they (usually) restrict the scope of the immediately preceding 'entity' term to denote a narrower class of entities. For example, the pages referenced by the node preceded by the (sub)path `Telecommunications/Wireless` can be viewed as 'indexed' by the compound term 'wireless telecommunications'. The 'property' term is completely dependent on the given 'entity' term, seeking it independently on a page would be spurious.

The 'entity' terms can be further refined to:

1. *Subjects* (active entities) such as `Manufacturers`, `Publishers`, or `Associations`.
2. *Objects* (passive entities) such as `Materials`, `Aircraft` or `Textiles`.
3. *Domains* of activity such as `Telecommunications` or `Publishing`.

In addition, we can identify a *common* subclass of both *object* and *domain*, which can be denoted as *activity*. An activity is a domain, since it fits into the generality hierarchy of domains, but it is also an object, since it can be viewed as a 'commodity' offered by a certain subject, e.g. `Manufacturing` or `Construction`. Furthermore, a distinct feature of an activity is the aptitude of being *applied* on an (other) object.

The diagram at Fig. 1 depicts the essence of the ontology. Boxes correspond to classes, full edges to named relations, and dashed edges to the class-subclass relationship. Reflexive binary relations are listed inside the respective boxes.

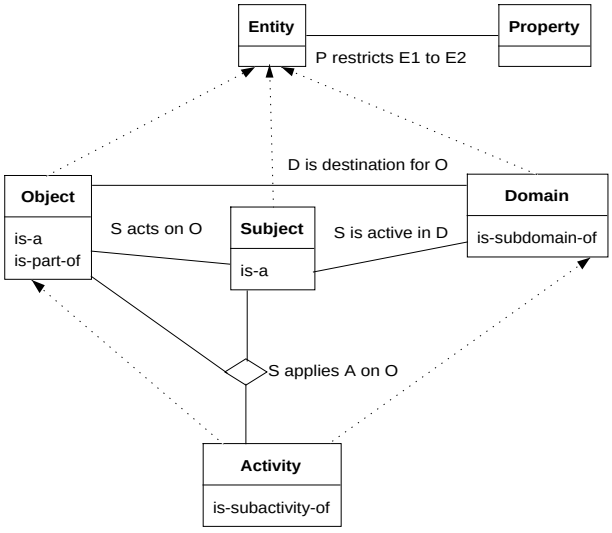


Fig. 1. The ontology of web directory headings

Given the ontology, the method described in section 2 can be enhanced in the following way:

1. *Select* the most promising paths and nodes in the directory structure.
2. Assign *class labels* to the headings from the selected paths, and arrange them into a semantic network of *relations*.
3. Generate full-text *queries* based on the headings (and their classes), and apply them on the pages to extract sentences.

4. The mining of indicator terms could then be done *separately* for each class of 'entity' terms, thus obtaining collections of 'extraction patterns' specific for different type of information to be extracted.

The structure over the headings should enable to generate[4] finer queries, and thus to obtain a more comprehensive sample of training data for indicator learning. As an example, given the path segment `Cranes/Accessories`, in which both `Cranes` and `Accessories` were pre-classified as 'objects', and `Cranes` identified as 'destination' for `Accessories`, the training cases containing e.g. the expression 'accessories *for* cranes' might me the most desirable ones.

The importance of step 1 (selection) follows from the fact that step 2 (labelling) has to be done manually, but, in distinction to 'classical' labelling, a single labelling action can lead to class (or, relation) assignment to several training cases. The efficiency of manual labelling is thus closely related to the number of pages referenced by the node being labelled, as well as to the number of sentences (from these pages) containing the headings. In order to obtain a high 'assignments-to-actions' ratio, we have to trade off the high number of pages (for general headings close to the root, pertaining to huge subhierarchies) with the higher number of sentences per page (for the headings close to the leaves, which are better tuned to the page content, and even subsume several 'ancestor' terms). The parameters of the respective utility function could be determined in the future.

4 Related Work

The common approach to overcome the lack of classified training examples in text categorisation is to apply *statistical techniques* consisting in iterative automated labelling of unclassified examples based on a few classified ones (bootstrapping, see [1], [4], [5]). So far, we have not considered such techniques, and instead rely on the prior work of a human indexer of the web directory. While directories have already been used for learning to classify *whole documents* [3], their use for *information extraction* seems to be rather innovative.

Our work is actually rather similar to Brin [1], which targets on automated discovery of extraction patterns using *search engines*. The patterns can be used to find relations, such as books, i.e. pairs (author, title). The patterns are based simply on characters surrounding the occurence of investigated relation. In comparison, we aim at finding less structured information, for which such simple patterns wouldn't be sufficient; we therefore search for linguistic indicators, which are based on syntax analysis. (The indicators themselves can be thought of as 'syntactic patterns'.)

[4] We are currently working on a rewriting grammar that will automatically convert the set of relational expressions on headings into a layered set of query terms.

5 Conclusions and Future Work

We have suggested a novel method for learning *indicative terms*, which can be, in turn, used to extract *important terms* (in fact, meta-data) from web pages. The source of learning cases is a *web directory*: thanks to the prior work of human indexers of the directory, the burden of manual case labelling is either completely removed, or significantly reduced. Preliminary results in a rather restricted setting suggest that the method may be viable.

As we have mentioned in the end of section 2, we will soon extend the non-interactive method of mining the indicators by *searching forth in the parse trees*, beyond the neighbouring verb (in particular for the 'unclear' verbs). The *accuracy on unseen pages* also has to be thoroughly tested. Furthermore, the prospective use of the web directory *ontology* has been described in section 3.

Finally, we anticipate that best results could be obtained by combining our reuse of human effort (with rather precise but incomplete results) with bootstrapping techniques mentioned in section 4 (more complete but possibly imprecise), in a more distant future.

The research has been partially supported by the grant no. 201/00/D045 (Knowledge model construction in connection with text documents) of the Grant Agency of the Czech Republic.

References

1. Sergey Brin. Extracting Patterns and Relations from the World Wide Web. In: WebDB Workshop at EDBT'98.
2. Dayne Freitag. Information Extraction From HTML: Application of a General Learning Approach. In *Proc. 15th National Conference on Artificial Intelligence (AAAI-98)*.
3. Dunja Mladenic. Turning Yahoo into an Automatic Web-Page Classifier. In *Proceedings of the 13th European Conference on Aritficial Intelligence, ECAI'98*, pp. 473-474.
4. Andrew McCallum and Kamal Nigam. Text Classification by Bootstrapping with Keywords, EM and Shrinkage. In *ACL'99 Workshop for Unsupervised Learning in NLP*, 1999.
5. Ellen Riloff and Rosie Jones. Learning Dictionaries of Information Extraction by Multi-Level Bootstrapping. In *Proc. 16th Nat. Conf. Artificial Intelligence (AAAI-99)*.
6. Daniel Sleator and Davy Temperley: Parsing English with a Link Grammar. In *Third International Workshop on Parsing Technologies*, August 1993.
7. Stephen Soderland. Learning Information Extraction Rules for Semi-Structured and Free Text. *Machine Learning* Vol. 34, 1999, pp.233-272.

Multiagent Cooperative Learning of User Preferences

Adorjan Kiss and Joël Quinqueton

LIRMM, Multiagent Team, 161 rue Ada, F-34392 Montpellier Cedex, France
{kiss,jq}@lirmm.fr,
WWW home page: http://www.lirmm.fr/~kiss

Abstract. We present in this paper a Machine Learning method designed to predict preference knowledge in a multi-agent context. In the first part we show some theoretical properties of our learning scheme and then present an application of it to a corporate knowledge management system.

1 Introduction

In this paper we will present some attempts to design a Machine Learning method to predict preference knowledge in a multi-agent context. Here we define preference knowledge as knowledge about a preference between elements of a set.

For instance, the documents found by a search engine on the web are ordered according to a preference function computed from the user request. Thus, they can be considered as ordered according to a preference relation.

The framework that gave birth to this work is a joint research project, CoMMA[1], dedicated to corporate memory management in an intranet. The main objective of the project is to implement and test a Corporate Memory management framework integrating several emerging technologies in order to optimize its maintenance and ease the search inside it and the use of its content by members of the organization.

The main challenge is to create a coherent system that relies upon several promising new technologies which are in the middle of their struggle to become standards:

- Multi-agent architecture: it is well suited to the heterogeneity of the Corporate Memory; its flexibility eases the system maintenance and keeps the rhythm with the dynamics and evolution of the Corporate Memory; cooperating and adaptive agents assure a better working together with the user in his pursuit to more effectively achieve his goals. The FIPA standard, supported by the CoMMA project, offers the specifications for interoperable intelligent multi-agent systems.

[1] This work was supported by the CoMMA (Corporate Memory Management through Agents) project [Con00] funded by the European Commission under Grant IST-1999-12217, which started beginning of February 2000.

– XML: is a standard recommended by the World Wide Web Consortium intended to offer a human and machine understandable description language: a good choice if it is important to ensure an easy maintenance, and seamless flow through various information processing systems that are expected to evolve in time.
– RDF/RDFS another W3C recommendation, that creates a semantic level on the top of XML formal description. RDF annotations allow having an integrated, global view of the Corporate Memory keeping untouched (in terms of storage, maintenance) the heterogeneous and distributed nature of the actual info sources. RDF also allows us to create a common ontology to represent the enterprise model. The ontological commitment, a fundamental choice in our approach to design the Corporate Memory, is motivated by our belief that the community of corporate stakeholders is sharing some common global views of the world that needs to be unified and formalized (RDFS) to form the basis of the entire information system.
– Machine Learning Techniques make the system adaptive to the user, and come even more naturally due to the previous choices, as presented in the following section.

2 The use of Preference knowledge

Here we define preference knowledge as knowledge about a preference between elements of a set. Such knowledge can be stated in various forms: a numerical value assigned for each item, a total ordering relation, a partial ordering relation or even a preordering of the set.

Logical models have been proposed to deal with such knowledge, some dealing directly with the comparison abilities [Sch96] and others inspired from discrete linear-time temporal logics [SR99].

It is generally admitted that a preference stands for an order that maybe partial, even a preorder, but that it is often convenient to represent it by a linear extension (which is a total order) or a numeric value compatible with the known orderings.

Then, in terms of Machine Learning, different strategies may be used, depending on the form of the preference knowledge.

2.1 Numeric labelling

A numeric labeling, i.e. a mapping of our set of examples into a set of real numbers, is a convenient way to summarize a preference relation. Some Machine Learning methods are available to learn numerical variables [Gas89,Bre96b,Bre96a].

Generally, the methods for learning to predict a numerical variable v measure the quality of a predictive rule R by the standard deviation $\sigma^2(v)$ of the value of the variable among the set of objects verifying the concept R to be tested.

$$Q(R,v) = \frac{1}{|R(x)true|} \sum_{x}^{R(x)true} (v(x) - \bar{v})^2$$

The lower $Q(R, v)$ is, the better R is to predict v, because the mean value of v can be used with less error. With such criteria, any learning method will lead to grouping several neighbour values around their mean. Then, the learnt rules will not be very different from rules learnt from examples roughly rated with a finite set of values.

2.2 The order relation

By definition, a binary relation, which we note $\lhd$, is an order if it has the following properties:

- reflexive $x \lhd x$,
- transitive: if $x \lhd y$ and $y \lhd z$, then $x \lhd z$,
- antisymmetric: if $x \lhd y$ and $y \lhd x$, then $x = y$.

Then, we can imagine to learn the binary relation by learning each of its elements, that is, learn on each couple of objects (a, b) such that $a \lhd b$. Then, let us summarize the suitable properties of such a learning set for this approach to work correctly.

First, if (a, b) with $a \lhd b$ is an example, then (b, a) is a counter-example. Then, what happens to (a, a)? We can see that they would be both examples and counter-examples, then it is better to consider the strict order relation, and eliminate diagonal elements.

With these hypotheses, the description of an example is made of 2 parts: the attributes which are modified between a and b, and those which keep the same value. We can notice here that these attributes are the same as those involved in the sorting tree of our examples.

Then, our method appears to be "half lazy", in comparison with lazy learning methods, like kNN or LWR [Aha92]. Our learned knowledge is partly explicit, but in the classification step, we need to compare a new instance with several elements of the learning set (maybe in a dichotomic way) to put it in the right place.

2.3 Statistical evaluation criteria

Usually in Machine Learning, particularly for building of decision trees, the learned classification rules are evaluated by their similarity to the desired classification.

We can use the same principle here, and we have two possible families of criteria. If we can compute a rank for each element, the similarity is computed by measuring the rank correlation to the expected ranking. Otherwise, each pair must be given: then we use a pairwise comparison between the expected order and the learnt order.

Several measures of similatity between 2 different orderings of the same data have been proposed. In each case, one has to deal with tied elements.

The Spearman rank order correlation The Spearman rank order correlation r_s is a correlational measure that is used when both variables are ordinal. The traditional formula for calculating the Spearman rank-order correlation is

$$Corr(r, r') = 1 - \frac{6 \sum_{i=1}^{i=n} (r_i - r'_i)^2}{n(n^2 - 1)}$$

where r and r' are the ranks to compare of paired ranks. When there are tied cases they should be assigned the mean of their ranks. The mean of the ranks from $p+1$ to $p+n$ is $\frac{1}{n}(\frac{(n+p)(n+p+1)}{2} - \frac{p(p+1)}{2})$, which become after simplification $\frac{n+2p+1}{2}$.

The Kendall pairwise τ criterion When we have to compare each pair of data, they can be classified as either tied (T), concordant (P), or discordant (Q).

The best measure for this case is Kendall's τ_b which takes into account a correction for tied pairs. Its formula is

$$\tau_b = \frac{P - Q}{\sqrt{(P + Q + T_x)(P + Q + T_y)}}$$

where T_x is the number of pairs tied on X but not Y, and T_y is the number of pairs tied on Y but not X.

2.4 Verifying the consistency of the method

In the case we perform a pairwise learning of the order relation, we can notice that a fundamental property, the transitivity, can be guaranteed by the learning process itself, as we show below for a version space method [Mit97].

We can check that, if, for 3 examples the transitivity holds, then it is not necessary to add the 3rd pair as example to learn the relation :

- let $(a, b) = (a_1 \dots a_n, b_1 \dots b_n)$ and $(b, c) = (b_1 \dots b_n, c_1 \dots c_n)$. Then, S is of the form $L \wedge R$, with the left part L as a generalisation of both a and b, and the right part R of both b and c. Then, as L is a generalisation of a and R of c, S is a generalisation of (a, c).
- with the same conventions, G has a disjunctive form whose elements reject all the examples, then, if we represent any of its elements as $L \wedge R$. If (b,a) and (c,b) are rejected, it means that L rejects b or R rejects a, and L rejects c or R rejects b. But G must also be a generalisation of S.

Of course, this is only a scheme of the proof, and is, strictly speaking, only available for version-space-like learning. In a more general case, like decision tree learning, we can only make the hypothesis that it is true. We concluded that we could learn directly a sorting rule (in a greedy way, like decision trees) and evaluate the obtained rule with the τ criteria defined in section2.3.

Let us now describe more widely our application.

3 Using preference in Knowledge Management

This section aims to present and discuss how our work on preference learning fits into CoMMA project's Knowledge Management System [Con00].

3.1 Getting the user profile

One of the advantages of an enterprise that should be exploited by such a corporate information management system is that the users (i.e. the employees) can be known (their domains of interest/competence, their current activities/tasks). This can be especially useful in some cases where users are likely to be overwhelmed by the quantity of information to process and navigate themselves through (new employees during accommodation, technology monitoring scientists) who would appreciate personalized automated help in their process of information retrieval. Nevertheless, using Machine Learning to reach this goal can present some challenges. Some generic solutions are presented in [WPB01].

3.2 Using semantic annotations

Secondly, we have "human and machine understandable" semantic information upon the corporate knowledge offered by the RDF formalization, based upon an "enterprise ontology" (RDF schema).

The combination of these two sources of information can provide a rich ground to infer knowledge about the users' probable/possible preferences. This combination is made possible due to the fact that we use the same RDF standard for formalizing the user profile; the same base ontology for the enterprise and user models.

It can be imagined that the info combined from these sources form sets of attributes that will be used as input for an ML mechanism.

In order to set up such a ML mechanism, there are two main tasks to complete:

1. Getting and formalizing the information to be decomposed as attributes to feed the ML mechanism.
2. Defining the ML methodology to process this info

3.3 Collecting the information to create a set of most meaningful attributes

We will need to answer the following question: *Why does a user prefer a document?*

In our attempt to give an example of some possible answers, we are gradually going deeper and deeper into details in case of complex answers: *The document is interesting.*

– Because it has been stated so:

.

  - By the user himself (the user has already seen the document, and "told" the system, that he is interested in)
  - By someone else (someone, maybe "close" to the user, wanted to share a favorable opinion about a document)
- Because it concerns a topic close to the user's *interest fields*:
  - by the relation with the user:
    * Personal interest fields
    * Professional interest fields (known by his role in the enterprise)
  - by the way they are obtained:
    * Declared interest fields (the user has stated his interest documents concerning a topic)
    * Implied interest fields (the user is included in a community of interest which is close to a topic)

The second question, that introduces the notion of temporality into the preference: Why does a user prefer a document at a given moment?

In other words, to make the difference from the first question: *Why does a user prefer a document at a given moment, and does not prefer it at another moment?*

- The document is interesting only if seen the first time (or the first few times)
- It is interesting during a certain period (when the user performs a certain activity, etc.)

These answers are just some samples, one can think of many other possible reasons. Though, we realize that it is a very important to find the right questions and answers, that include the majority of possible situations. Indeed, getting the right questions and answers and translating them into quantifiable attributes, and making sure that the highest number of possible situations are observed is a key to the success of such a learning mechanism, that may even outclass in importance the chosen learning technique.

Nevertheless, we will present our approach in the Comma project to choose some typical answers and attributes, but we keep more focused on the second issue: the preference learning methodology.

3.4 Learning in the CoMMA system

After a short presentation of the design of the CoMMA system, this section presents the multiagent interaction in which Machine Learning is performed in CoMMA.

The chosen MAS consists of a society of coarse-grained agents, that fulfill in general multiple roles, and are organized in a small number of functional subsocieties. The MAS architecture was designed in order to optimize task-division, flexibility and robustness of the system, and network layout (extensibility, scalability, traffic optimization).

For the implementation of the prototype system, the Jade agent platform was chosen, which is an Open Source Project developed by project partners,

University of Parma and CSELT. Jade is a FIPA compliant agent platform, implemented in Java, and has also the advantages of a wide opening towards Internet and the Web, interoperability with other MAS-s, and future systems.

In the current status of implementation, the CoMMA system will help the user in three main tasks:

- insertion and RDF annotation of documents,
- search of existing documents, and
- autonomous document delivery in a push fashion to provide her/him with information about new interesting documents.

We have already experimented such an architecture in Network Supervision [EDQ96], with Machine Learning abilities [QEN97]. Here, for the Machine Learning part, we choose to use the Weka open source Java library [WF99], which enables us to experiment various learning methods and frameworks.

3.5 The learning agent

The first context to assess preference learning was chosen to be the document retrieval scenario, via semantic annotations. The search engine used for document retrieval in the CoMMA system is an inference engine called CORESE [CDH00] developed by INRIA, one of the partners of the project. CORESE uses Conceptual Graphs and combines the advantages of using the RDF language for expressing and exchanging metadata, and the query and inference mechanisms available in CG formalism. In order to produce inferences, CORESE exploits the common aspects between CG and RDF: it defined a mapping from annotation statements (RDF triples) to Conceptual Graphs and vice-versa.

One of the shortcomings of such a query retrieval engine is that there is no standard method to sort the information returned, such as keyword frequency in keyword-based search engines. The returned data set must be post-processed, filtered and sorted to present the user with the relevant information. Here comes the aid offered by our ML mechanism.

In the CoMMA system, information that feeds the ML comes from several sources: The document sub-society (the annotations accompanying a query response), the user sub-society (user monitoring and explicit user feedback), and ontology sub-society (to help getting the "meaning" of the results). And of course the user profile. Therefore, the learning behavior was "awarded" to the User Profile Manager agent, which belongs to the connection dedicated sub-society, and performs notably a role of middleman between agents. This decision was justified also by network traffic optimization reasons, especially because in reaction to a user action (query), several interactions can be triggered between agents of different roles.

For example, during a query retrieval, the main interactions are as described in the following diagram.

In this scenario, the role of the ML component starts when the query answers are collected and transmitted to the profile magager agent. First, the

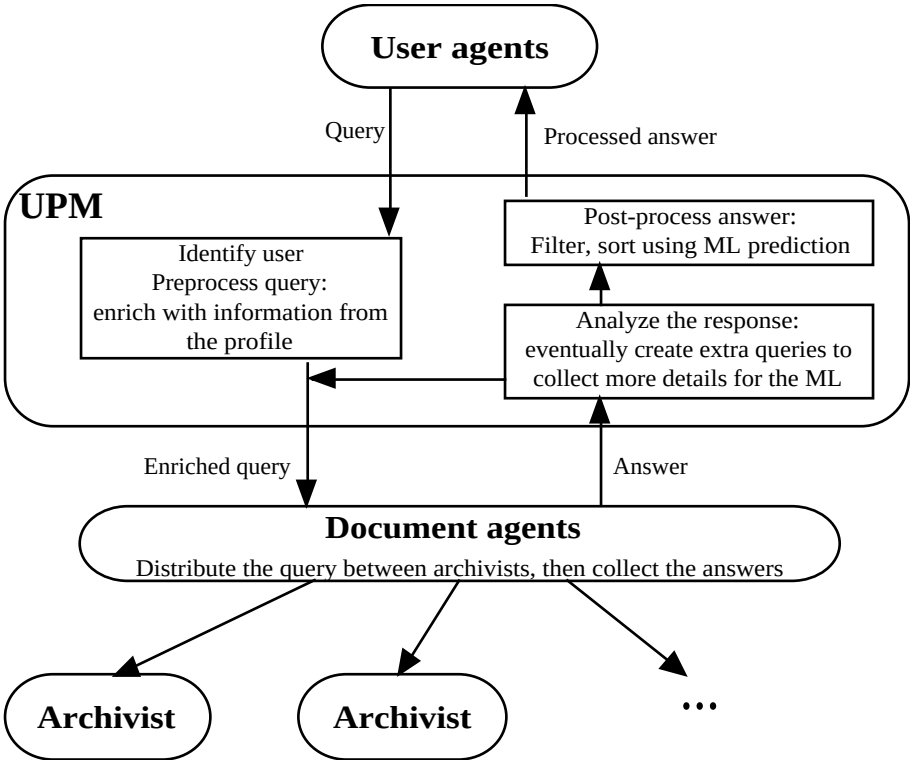


Fig. 1. The main interactions in CoMMA during a query

semantic annotations are extracted and analysed. They are combined with the relevant fields from the user profile in the attempt to compute the values of the input attributes (such as the ones listed in the section about attributes). In case the system finds that there could be more semantic annotations related to the documents retrieved that could help in computing the input attributes, a supplementary query can be formulated to retrieve them before going on with processing and forwarding the answer.

The second step is to use the learnt preference measures to rank the documents and sort the answers list to be returned by these predictions.

3.6 The learning cycle

The goal of the ML component is to produce a set of rules that will be used to produce predictions about user preferences. It can be supposed that the system starts with a set of predefined rules, that will be gradually improved during the process of adaptation to users (method known as theory refinement). Otherwise, the system may start with an empty knowledge base, and will be undergo a training period, to accumulate sufficient knowledge to allow its deployment.

In our approach, user adaptability comes from both implicit observation of the user (user monitoring subsystem), and explicit user feedback.

We use a "predict-or-learn" type protocol, that is, when the input is coming from query answers, the system tries to use its knowledge to predict, otherwise, when the input comes from the user in the form of negative feedback, the system tries to update its rules set.

3.7 A sample set of attributes

A sample set of attributes we used to create instances from the answers we gave as examples to the question of preference is listed in the followings. These attributes may not be the most relevant ones, or adapted to any case, we only tried to make it diverse and for most of them restricted the scope as much as possible for the sake of simplicity of our prototype.

Is the document related to the user's role We suppose that the users, depending on their roles, will be assigned certain interest fields, that will be recorded in their profiles. Then, the documents can have a generic $Concerns$ property, that associates them with such topics. In this scenario, the notions like $Topic$, $Concerns$, $Interestedby$, etc are defined as concepts in the ontology that constitutes the basis of the enterprise model.

As an observation, the notion *topic* or $interestfield$ is used in a general sense to categorise documents.

Then we can extract this information from the user profile and document annotation and combine them to create an instance attribute. We can define this attribute as taking a binary value.

In a complete implementation it is likely that this relationship can be seen as more complex and eventually be split into several attributes, and/or may take a wider range of values.

Is the document related to the user's COINs (communities of interests) This attribute reflects a further differenciation we have made in interest topics when answering the preference question: topics can be assigned or chosen by the user; inherited or inferred by the system, etc.

In the same conditions as for the attribute above, the attribute will take a binary value, and in case we define relationships, we can use the same procedure as above.

User experience at the company Since in our project addressing the New Employee scenario is particularly focused upon, we considered important making the system behave differently towards new versus experienced employees. We have segmented the range of values so that this may take values such as: $lessthan1week$, $lessthan1month$, $lessthan2months$, ... all these regarding the novice user, and $morethan3months$ (or simply emphexpert, or whatever the opposite for *NE* is). A tip for implementation, if such a segmentation is desired, is

to define these intervals in the enterprise ontology, so it can adapt to its specific needs.

The following subset of attributes are related to *user monitoring*.

In our example we have imagined a simple user monitoring scenario, that supposes tracking each consultation of a document by the user, and building a navigation history (or consultation trace).

Document last seen Usually it is important if a document was seen before or not, and if it was then how long before. After extraction from the user's navigation history it should also be discretised into several intervals: ¡1hour, ¡1day, ¡1week, ... , never.

Average return frequency In certain cases the user may return more often to a document, in other cases an information can only present interest when first seen. This attribute may also tell something about the situation for the current case. The value will also result from processing the navigation history, and it would probably be enough to use some rough intervals (like once, small, large, etc.)

Document category touch frequency It might present an interest to extend the above attribute for categories the document belongs to. In this case the value will be processed the same way as for the previous attribute. In case a specific strategy for creating implicit communities of interest is envisaged, it should be checked if there are possible conflicts with the use of this attribute.

The next attributes contain specific information about the particular document being analysed:

The user's rating for the document For some specific documents, the user may explicitly wish to manifest his interest or non-interest. In this case the user profile should allow storing this information. It is a general vote for the document, and does not have the temporal aspect implied by the output of this classifier.

Public ratings Sometimes a user would like to share his opinion about a document with others. In this case it must be foreseen in the enterprise model so that it may be stored in the form of an annotation, that can be used also by our classifier. A method should be formalised to allow storing information about people possibly interested about this opinion.

This was a list we used in our first trials. Once again we make the remark that the list of attributes should be open, and checking the completeness and exhaustiveness of it has to be an important and ongoing task. In other words,

watching that the factors that contribute to a document being seen as more or less relevant in a situation have been well captured, and that there are no other decisive factors that were omitted, or can not be represented. Because in either case, the system might make major mistakes in certain situations, whatever learning algorithm was used.

3.8 Document ranking and sorting

In our case, the first goal is to sort documents in a query response by the order of predicted user preference. The two principal strategies that can be used to achieve that, are: grouping documents using numeric labeling or learning the order relation (or the sorting rule).

For the first trial of our system, we choose to learn a rough labelling in a finite set of classes (namely 5), to classify the documents retrieved by the system after a user query. Then, we clearly fall in the first kind of strategy, whose goal is not to obtain a fine grained ranking, but only a coarse grained rating.

This is equivalent to the use of numeric labeling, associating numeric values from a finite set to categories of documents representing their importance. But the drawback is that the system will not be aware of the semantic of the order relation. And there is also a risk, that in case of large number of items to classify, many of them will fall into the same class, and there will be no further means to differentiate them.

On the other hand, if an order relation is used (either by pairwise learning or by learning the sorting rules), we will have no distance measure to separate the values. That is, to give an idea about for instance "how much a document is more important then another".

Then, a perspective for document retrieval systems of this kind can be to use both methods in a complementary way:

1. in serial-coupling (one method used to pre-process, the second to post-process)
2. in parallel (eventually in distinct agents) then putting together the results and solving conflicts.

In order to evaluate the contribution of learning to the efficiency of the retrieval system, we have designed an experimental protocol. It consists on evaluating the learning step by comparing the use trace with and without learning, in the various scenarii taken in account in the CoMMA system.

4 Conclusion

In the CoMMA project, the Machine Learning and user adaptability component is one of the main features. In this paper we presented the advances that we have made in this domain, especially focussing on the learning of preference data for document retrieval.

We proposed a specific method to learn preference data, and a framework to experiment and evaluate it. The main choice we focus on does not only present the usefulness of Machine Learning, but also tries to overcome some of the limitations of semantic information retrieval systems.

In our opinion, the choice we made here will give interesting results during the first trial we planned in the project, then allows an experimental evaluation through feedback from the user.

The implementation is currently well advanced, and we begin to have some experimental results.

References

[Aha92] D. Aha. Tolerating noisy, irrelevant, and novel attributes in instance-based learning algorithms. *International Journal of Man Machine Studies*, 36(2):267–216, 1992.

[Bre96a] L. Breiman. Bagging predictors. *Machine Learning*, 24:123, 1996.

[Bre96b] L. Breiman. Stacked regression. *Machine Learning*, 24:49, 1996.

[CDH00] Olivier Corby, Rose Dieng, and C. Hébert. A Conceptual Graph Model for W3C Resource Description Framework. In *the 8th International Conference on Conceptual Structures (ICCS'00)*, number LNCS 1867 in Lecture Notes in Artificial Intelligence, Darmstadt, Germany, 2000. Springer Verlag, Springer Verlag.

[Con00] CoMMA Consortium. Corporate Memory Management through Agents. In *E-Work and E-Business conference, Madrid*, October 2000.

[EDQ96] Babak Esfandiari, Gilles Deflandres, and Joël Quinqueton. An interface agent for network supervision. In *Intelligent Agents for Telecommunication Applications*, Budapest, Hungary, 1996. ECAI'96 Workshop IATA, IOS Press.

[Gas89] O. Gascuel. A conceptual regression method. In Katharina Morik, editor, *EWSL-89, 4th European Working Session on Learning*, pages 81–90, Montpellier, France, Décembre 1989. Jean Sallantin and Joel Quinqueton, CRIM, Pitman, Morgan Kaufman.

[Mit97] Tom M. Mitchell. *Machine Learning*. Mac Graw Hill, 1997.

[QEN97] Joël Quinqueton, Babak Esfandiari, and Richard Nock. Chronicle learning and agent oriented techniques for network management and supervision. In Dominique Gaïti, editor, *Intelligent Networks and Intelligence in Networks*. Chapman & Hall, September 1997.

[Sch96] P-Y. Schobbens. A comparative logic for preferences. In Pierre-Yves Schobbens, editor, *Working Notes of 3rd ModelAge Workshop: Formal Models of Agents*, Sesimbra, Portugal, January 1996.

[SR99] P.-Y. Schobbens and J.-F. Raskin. The logic of "initially" and "next": Complete axiomatization and complexity. *Information Processing Letters*, 69(5):221–225, March 1999.

[WF99] Ian H. Witten and Eibe Frank. *Data Mining: practical machine learning tools and techniques with Java implementations*. Morgan Kaufmann, 1999.

[WPB01] G. I. Webb, M. J. Pazzani, and D. Billsus. Machine learning for user modeling. *User Modeling and User-Adapted Interaction (UMUAI)*, 11, 2001.

ARCH: An Adaptive Agent for Retrieval Based on Concept Hierarchies

B. Mobasher

We present a client-side agent, named ARCH, for assisting users in one of the most difficult information retrieval tasks, i.e., that of formulating an effective search query. The agent utilizes a hierarchically-organized semantic knowledge base in aggregate form, as well as an automatically learned user profile, to enhance user queries. In contrast to traditional methods based on relevance feedback, ARCH assists users in query modification prior to the search task. The initial user query is (semi-)automatically modified based on the user's interaction with an embedded, but modular, concept hierarchy. The modular design of the agent allows users to switch among the representations of different domain-specific hierarchies depending on the goals of the search. ARCH passively learns a user profile by observing the user's past browsing behavior. The profiles are used to provide additional context to the user's information need represented by the initial query. The full system also incorporates mechanisms for categorizing and filtering the search results, and using these categories for performing refined searches in the background. Preliminary experiments have shown that the agent can substantially improve the effectiveness of information retrieval both in the general context of the Web, as well as for search against domain-specific document indexes.

Utilising an Ontology Based Repository to Connect Web Miners and Application Agents

Stefan Haustein

University of Dortmund, Computer Science VIII,
Baroper Str. 301, D-44221 Dortmund, Germany,
stefan.haustein@udo.edu,
http://www-ai.cs.uni-dortmund.de

Abstract. Ontologies are important for providing a shared understanding of a domain for web mining agents and other agents accessing the gathered information. When the information access is decoupled from the mining process – for example when building a semantic web server – an additional storage compliant to the application ontology is needed. The COMRIS information layer was built to serve that purpose for a system supporting conference participants. It is able to provide permanent access to gathered or aggregated information suitable for both, humans and software agents by providing FIPA ACL and HTML interfaces.

1 Introduction

The goal of the COMRIS project was to design an agent based conference support system. Conference participants were equipped with a wearable electronic device that was able to recognise other participants wearing a similar device. The purpose of the device was to introduce participants to each other, to filter requests and to provide background information depending on the current context [1]. In order to perform its task, the agent controlling the device needed background information during the short period of time a certain context was valid: When a person has passed by, it is too late to introduce that person.

The background information should be provided by a web mining process. Since starting mining on demand seemed too slow for the given application, web mining was already performed beforehand. The approach is similar to using web spiders for search engines: If they were launched just when somebody enters a keyword, web searching would not be really practical.

2 The Mining Task

The gathering agents enrich the conference information by gathering information from different sources in the WWW. In our case, the agents just collect all information available about the registered conference participants, and new persons discovered in the gathering process were not investigated further. The information was used to enrich the knowledge about a person and its relations to other persons (e.g. co-author, project partner).

In the conference scenario, we were using three different types of gathering agents: The CORDIS collector is able to query the CORDIS project database of the European Union, the KA (Knowledge Acquisition) and ILP (Inductive Logic Programming) agents are able to query two different bibliography databases for the corresponding community. Each agent takes into account the special structure of its source, but they all stem from one generic agent. Together, the gathering agents are able to find European projects the conference participants were involved in and most of their publications.

Before the actual start of the conference, a learning step was applied to information gathered for a set of training persons. The Rule Discovery Tool (RDT [2]) of the MOBAL machine learning system [3, 4] was used to learn indicators for a "may-want-to-meet" relation between participants. While the learning step itself was performed off-line, the rules learned were applied to the information gathered in the runtime system, in order to create default instructions for the personal representation agents of the participants.

3 Complex Mining Tasks require Ontologies

When operating on a highly structured information space, it is no longer sufficient to just store words in a huge database. This is where the application ontology comes into play. Both, gathering and application agents need a common language. Also, in order to be able to perform the gathering beforehand, some kind of repository for the gathered information is needed. For the COMRIS conference scenario, the amount of concepts to be modelled, like participants, speakers, talks, sessions, rooms, agents, booths, schedules etc., became quite large. Moreover, all concepts had a lot of complex relations to other concepts.

Using relational tables for this purpose seemed inadequate because of the complicated mapping that is required to transform the ontology to a high number of tables. Also, the table solution seemed inflexible because ontological changes would cause a lot of changes in database tables and additional "agentification" wrappers.

Description Logic [5] systems like KL-ONE [6] provide additional features like automated classification that are computational expensive but not required in the system. All reasoning was intended to be performed by the specialised agents. Like for the relational tables, additional wrappers for an Agent Communication Language (ACL) would be required. Also, using Description Logics would require globally unique slot names, leading to additional negotiation efforts between the project partners designing "their" part of the application ontology.

OntoBroker, a system extracting ontologies from the web, is able to automatically unfold the stored knowledge and provides persistence for the ontology itself [7]. While its centralised structure would be a good starting point for learning mechanisms, the system interface is not agent but human oriented.

4 Information Layer System Architecture

For the given reasons, we decided to build a new kind of information system that

– provides ACL access in the first place,
– is agent based itself,
– and is built on an ontology that is not hard-wired to the system.

The main purpose of the system was to act as blackboard [8, 9] for decoupled communication between the conference organisers entering the initial participant information, the gathering agents annotating this information with web content, and the application agents utilising the gathered information for their tasks helping the conference participants.

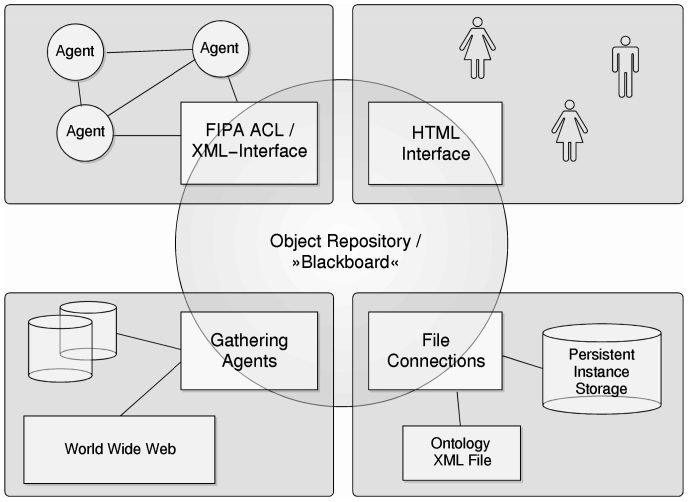


Fig. 1. Information layer architecture overview

The kernel of this system, the COMRIS information layer, provides only a memory representation of information structured corresponding to a given ontology. All other features were delegated to additional modules or agents, performing specialised tasks like:

– Handling communication with other agents
– Applying the learned rules to transform gathered data to default agent instructions
– Synchronisation with the underlying persistent data storage
– Building a generic HTML presentation from the ontology and the actual information layer content

The HTML presentation was not an initial part of the system, but once the system was built, it seemed a waste of resources to set up a separate conference web site built on traditional techniques. Instead, a wrapper agent transformed HTTP requests to ACL messages and forwarded them to the corresponding agents. Figure 1 shows an overview of the system architecture.

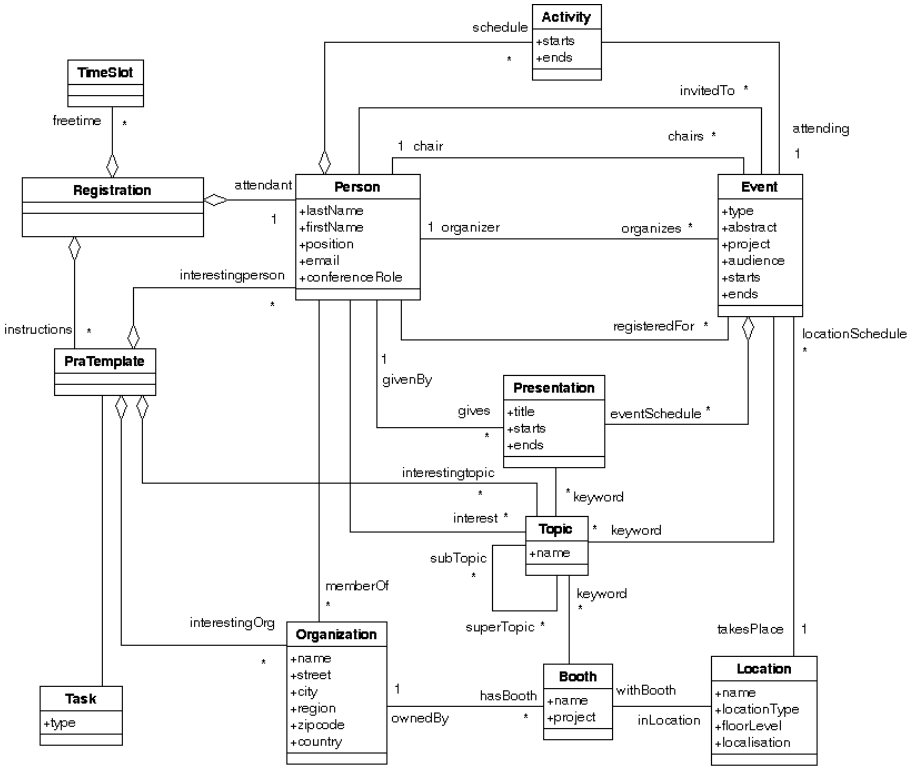


Fig. 2. Sample UML ontology diagram

5 Ontology and Data Model

The information layer uses an object-oriented model for data representation. Objects consist of atomic attributes and relations to other objects. The consistency of relations in both directions is ensured automatically, avoiding inconsistencies inside the system. The concepts and relations are defined application-dependent in an external ontology definition file. All files used by the information layer are stored as XML documents.

The ontology used in the COMRIS information layer is defined using an Unified Modelling Language (UML) model [10, 11] encoded in a simple XML format. Compared to other languages suitable for ontology modelling, UML currently still lacks clearly defined semantics. However, there are significant efforts to solve this problems [12, 13].

Figure 2 shows the UML diagram of the shared parts of the COMRIS ontology. Gathered information about publications and projects was transformed to templates for the Personal Representative Agents (`PraTemplate`) by applying the learned rules. The raw data gathered was also stored in the information system, but was not shared among all agents.

6 Communication and Content Languages

The communication and content languages for software agents and system components are based on XML, too. An XMLified version of FIPA ACL [14] is used as communication language, whereas the actual content language format is derived from the ontology automatically. Figure 6 shows the content language encoding of Tanja Katschenko and Carlos Gomez working at IBM corresponding to the ontology example in the previous section.

Linked Structure

```
<Organization id="555777">
   <name>IBM</name>
</Organization>

<Person id="888543">
   <name>Katschenko</name>
   <firstName>Tanja</firstName>
   <memberOf idref="555777"/>
</Person>

<Person id="878653">
   <name>Gomez</name>
   <firstName>Carlos</firstName>
   <memberOf idref="555777"/>
</Person>
```

Nested Structure

```
<Organization id="555777">
  <name>IBM</name>
  <members>
    <Person id="888543">
      <name>Katschenko</name>
      <firstName>Tanja</firstName>
    </Person>
    <Person id="878653">
      <name>Gomez</name>
      <firstName>Carlos</firstName>
    </Person>
  </members>
</Organization>
```

Fig. 3. Content language examples

Relations between instances can be described using the `idref` attribute, or by embedding related instances in the relation element. The encoding used for sending instances to software agents or other entities can be controlled by the corresponding entity to fit its particular needs best.

Readers familiar with the Resource Description Format (RDF) will have noticed a strong similarity of the formats. While it would be possible to migrate to RDF, there would be no improvement concerning human readability, which turned out crucial for system integration and maintenance. Moreover, RDF uses a property-centric data model, causing compatibility issues with traditional object oriented systems. The high number of RDF syntax variants leads to integration problems with other XML building blocks like XML Schema and XSLT [15] [16]. For those reasons, we will replace the current XML representation by the serialisation format of the Simple Object Access Protocol (SOAP) [17], improving the compactness and readability of the format as well as compatibility to SOAP based third party systems. However, migration to SOAP does not exclude building an additional RDF based interface if required.

7 Query Interface

The information layer supports a subset of OQL [18] as query language for agents. Additional languages may be plugged in by adding corresponding agents. By subscribing to the information layer, it is possible to keep an agent up to date without polling [19].

8 HTML Generation

The information layer contains a module that provides built-in web-server functionality. Since XML is not fully supported by web browsers yet, the server is able to generate HTML dynamically: For any object, the attributes are simply displayed, and the relations are converted to sets of hyperlinks to the related objects (figure 4). The HTML interface can also be used to edit the content of the system using forms generated dynamically based on the ontology.

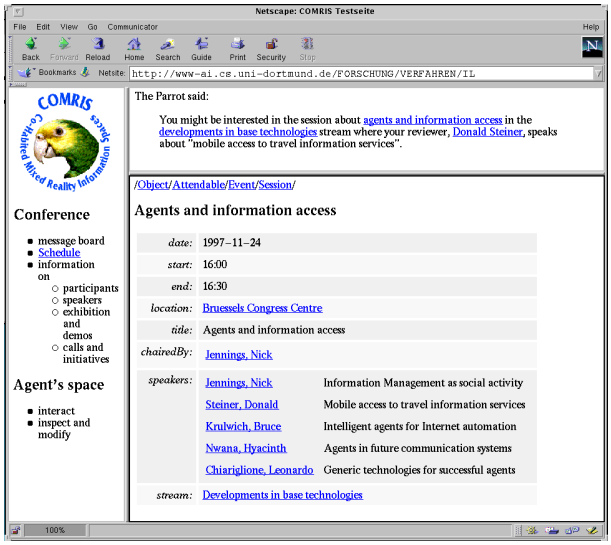


Fig. 4. Access to the Information Layer using a Web Browser.

In the COMRIS project, the HTML interface was used for interaction with the end user as well for as debugging and inspection purposes.

In addition to generic HTML generation, templates can be used in order to generate HTML pages conforming to a given look and feel. In the COMRIS project, we have also used the template mechanism to generate the input structure required by the text generation system TG/2 ([20]) which was used to generate natural language output for the wearable device.

The template mechanism was also used to generate questionnaires for evaluating the mining and learning results of the gatherers [21, 22].

9 Conclusion and Outlook

The main purpose of the implemented system was to provide an ontology based persistent blackboard communication mechanisms for connecting mining and application agents.

Using ontologies and agent technologies enabled a simple extension of the system beyond the original purpose. The system can now also be used to publish structured and massively linked data to the traditional "human readable" web using template based (X)HTML generation. The system proved useful not only for modelling some aspects of a conference but also for other applications with many sets of small and massively linked objects.

Currently, the COMRIS information layer is used for two internal projects and as the training server of MLnet[1]. In the future, it is planned to use the information layer in the MiningMart project for storing and editing data mining meta information.

The most important future developments are to make the information layer compliant to SOAP serialisation [17] and XMI in order to use a standardised XML formats for the message content language and for the ontology definition. It is also planned to include structure translation mechanisms for connecting systems using different but related application ontologies.

Acknowledgements

The research reported in this paper was supported by the ESPRIT LTR 25500 COMRIS project.

References

1. Plaza, E., Arcos, J.L., Noriega, P., Sierra, C.: Competing agents in agent-mediated institutions. Personal Technologies Journal **2** (1998) 1–9
2. Kietz, J.U., Wrobel, S.: Controlling the complexity of learning in logic through syntactic and task–oriented models. In Muggleton, S., ed.: Inductive Logic Programming. Number 38 in The A.P.I.C. Series. Academic Press, London [u.a.] (1992) 335–360
3. Sommer, E., Emde, W., Kietz, J.U., Wrobel, S.: Mobal 4.1b9 User Guide. GMD – German National Research Center for Information Technology, AI Research Division (I3.KI), St. Augustin, Germany (1996)
4. Morik, K., Wrobel, S., Kietz, J.U., Emde, W.: Knowledge Acquisition and Machine Learning - Theory, Methods, and Applications. Academic Press, London (1993)
5. Nebel, B.: Reasoning and Revision in Hybrid Representation Systems. Number 422 in Lecture Notes in Artificial Intelligence. Springer-Verlag (1990)
6. Brachman, R.J., Schmolze, J.G.: An overview of the KL-ONE knowledge representation system. Cognitive Science **9** (1985) 171–216
7. Decker, S., Erdmann, M., Fensel, D., Studer, R.: Ontobroker: Ontology based access to distributed and semi-structured information. In Meersman, R., other, eds.: Semantic Issues in Multimedia Systems, Kluwer Academic Publisher, Boston, 1999. Kluwer Academic Publisher, Boston (1999)
8. Hayes-Roth, B.: A blackboard architecture for control. Artificial Intelligence **26** (1985)
9. Kollingbaum, M., Heikkilae, T., McFarlane, D.: Persistent agents for manufacturing systems. In: AOIS 1999 Workshop at the Third International Conference on Autonomous Agents. (1999)

[1] http://www.mlnet.org/training

10. Object Management Group: OMG Unified Modeling Language Specification, version 1.3. http://www.omg.org/technology/documents/formal/unified_modeling_language.htm (2000)
11. Cranefield, S., Purvis, M.: Uml as an ontology modelling language. In: Proceedings of the Workshop on Intelligent Information Integration, 16th International Joint Conference on Artificial Intelligence (IJCAI-99. (1999)
12. Precise UML Group: The Precise UML Group home page. http://www.puml.org (2001)
13. Clark, T., Evans, A., Kent, S., Brodsky, S., Cook, S.: A feasibility study in rearchitecting UML as a family of languages using a precise OO meta-modeling approach. Report, Precise UML Group (2000) http://www.cs.york.ac.uk/puml/mml/mmf.pdf.
14. Foundation for Intelligent Physical Agents: FIPA web site (2001) http://www.fipa.org/specs/fipa00023/XC00023F.pdf.
15. World Wide Web Consortium: XSL Transformations (XSLT) version 1.0. http://www.w3.org/TR/xslt (1999)
16. Haustein, S.: Semantic Web languages: RDF vs. SOAP serialization. In: Proceedings of the Second International Workshop on the Semantic Web at WWW10. (2001) http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-40/haustein.pdf.
17. Box, D., Ehnebuske, D., Kakivaya, G., Layman, A., Mendelsohn, N., Nielsen, H.F., Thatte, S., Winer, D.: Simple Object Access Protocol (SOAP) 1.1. Note, World Wide Web Consortium (2000) http://www.w3.org/TR/2000/NOTE-SOAP-20000508.
18. Cattell, R.G.G., ed.: The Object Database Standard: ODMG 2.0. Morgan Kaufmann (1997)
19. Haustein, S., Lüdecke, S.: Towards Information Agent Interoperability. In Klusch, M., Kerschberg, L., eds.: Cooperative Information Agents IV – The Future of Information Agents in Cyberspace. Volume 1860 of LNCS., Boston, USA, Springer (2000) 208 – 219
20. Busemann, S.: A shallow formalism for defining personalized text. In: Workshop Professionelle Erstellung von Papier- und Online-Dokumenten at the 22nd Annual German Conference on Artificial Intelligence (KI-98), Bremen (1998)
21. Haustein, S.: Information environments for software agents. In Burgard, W., Christaller, T., Cremers, A.B., eds.: KI-99: Advances in Artificial Intelligence. Volume 1701 of LNAI., Bonn, Germany, Springer Verlag (1999) 295 – 298
22. Haustein, S., Lüdecke, S., Schwering, C.: The Knowledge Agency. In Sierra, C., Gini, M., Rosenschein, J.S., eds.: Proceedings of the Forth International Conference on Autonomous Agents, Barcelona, Spain, ACM SIGART, ACM Press, New York (2000) 205 – 206

XML Topic Maps and Semantic Web Mining

Benedicte Le Grand, Michel Soto

Laboratoire d'Informatique de Paris 6
8, rue du Capitaine Scott 75015 Paris, France
Benedicte.Le-Grand@lip6.fr, Michel.Soto@lip6.fr

Abstract. Navigation and information retrieval on the Web are not easy tasks; the challenge is to extract information from the large amount of data available. Most of this data is unstructured, which makes the application of existing data mining techniques to the Web very difficult. However, new semantic structures which improve the results of Web Mining are currently being developed in the Web. This paper presents how one of these semantic structures - XML topic maps – can be exploited to help users find relevant information in the Web. This paper is organised as follows: first, we introduce XML topic maps in the context of Tim Berners-Lee's Semantic Web vision. Then, we show how topic maps allow to characterise and "clean" Web data through the definition of a profile; this is achieved by the analysis of a lattice generated by a classification algorithm - called Galois algorithm. This profile may be used to evaluate the relevance of a web site with regard to a specific request on a traditional search engine. We finally explain how data on the Web can be clustered, organised and visualised in different ways so as to enhance users' navigation and understanding of these documents.

1 Introduction

Navigation and information retrieval on the Web are not easy tasks; the challenge is to extract information from the large amount of data available. Most of this data is unstructured, which makes the application of existing data mining techniques to the Web very difficult. However, new semantic structures which improve the results of Web Mining are currently being developed in the Web. This paper presents how one of these semantic structures - XML topic maps – can be exploited to help users find relevant information in the Web. This paper is organised as follows: first, we introduce XML topic maps in the context of Tim Berners-Lee's Semantic Web vision [2]. Then we show how topic maps allow to characterise Web sites through the definition of a profile. This profile may be used to evaluate the relevance of a web site with regard to a specific request on a traditional search engine. We finally explain how data on the Web can be clustered, organised and visualised in different ways so as to enhance users' navigation and understanding of these documents.

2 XML Topic Maps and the Semantic Web

Finding information on the Web is very difficult. Search engines may return hundreds or more links to users' queries – provided that the right keywords are used. Choosing the most relevant Web sites to explore is not trivial, because no semantics help evaluate the relevance of each hit. The next step is not easier: once a link is chosen, navigation is not always intuitive. Users can get lost easily: they may not find the information they are looking for even though it does exist. Sometimes they do not manage to go back to a page they have already visited. This is due to the lack of structure of the Web. Therefore it is necessary to add structure and semantics as well as to provide a mechanism which allows a more precise description of data on the Web. According to Tim Berners-Lee from W3C [2]:

"The Semantic Web is a vision: the idea of having data on the web defined and linked in a way that it can be used by machines – not just for display purposes, but for using it in various applications."

This Semantic Web can be achieved by adding semantic structures to the current Web. Many candidate techniques were proposed, such as semantic networks [16], conceptual graphs [5], the W3C Resource Description Framework (RDF) [14] and XML Topic Maps [11]. Semantic networks are basically directed graphs (networks) consisting of vertices linked by edges. Edges express semantic relationships between the vertices.

The conceptual graphs theory developed by Sowa [10] is a language for knowledge representation based on linguistics, psychology and philosophy.

RDF data consists of nodes and attached attribute/value pairs. Nodes can be any web resource (pages, servers, basically anything for which you can give a URI), or other instances of metadata. Attributes are named properties of the nodes, and their values are either atomic (character strings, numbers, etc.), metadata instances or other resource. This mechanism allows us to build labelled directed graphs.

Topic maps, as defined in ISO/IEC 13250 [8], are used to organise information in a way that can be optimised for navigation. Topic maps were designed to solve the problem of large quantities of unorganised information. Information is not useful if it cannot be found or linked. In the paper publishing world, there are several mechanisms to organise and index the information contained within a book or document. Indexes allow readers to go directly to the portion of the document that is relevant to their information needs. Topic maps can be thought of as the online equivalent of printed indexes. Topic maps are also a powerful way to manage link information, much as glossaries, cross-references, thesauri and catalogs do in the paper world. Topic Maps allow users to create a large quantity of metadata and tightly interconnected data. They constitute a kind of semantic network above the data themselves.

A new specification which aims at applying the topic map paradigm to the Web is currently being written; this initiative is called XTM (XML Topic Maps) [11]. XML Topic Maps allow to structure data on the Web and therefore make Web mining more efficient.

It was recently proven that the RDF and Topic Map models could inter-operate at a fundamental level [9]. Both standards are concerned with defining relationships between entities with identity. Each language can be used to model the other.

All the techniques described previously have the same goals and many of them are compatible. We decided to further investigate XML Topic Maps and study how they could enhance Semantic Web Mining.

We aim at helping users find relevant information and we contribute at three levels:

1. by evaluating Web sites relevance to users needs based on semantic criteria,
2. by filtering the topic map; the topic map profile constitutes a reference that can be used to select the most semantically significant objects (called *regular* objects). This allows to identify the major subjects which the topic map deals with and to discard less relevant topics.
3. by enhancing navigation on the Web through the aggregation of conceptually related topics and through the visualisation with different scales – or levels of details.

The different steps of topic maps – or Web sites[1] - analysis are represented in figure 1:

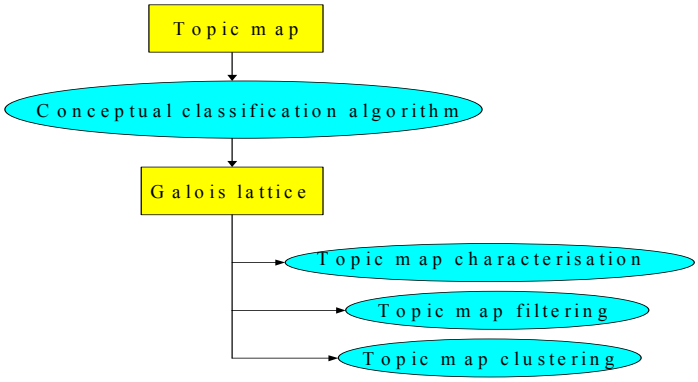


Fig. 1. Web sites analysis algorithm

We propose to achieve the first goal by defining profiles of topic maps – and consequently Web sites. These profiles characterise topic maps – or web sites - and help evaluate their relevance to users' information needs. The computation of this sort of topic map "DNA" and its interpretation are described in section 4.

The topic map may contain topics which are not semantically significant or not much related to others. We call them *singular* topics. They may be eliminated from the topic map so as to clean it, as explained in section 4.2.

[1] In the following, we will use the term "topic map" which is more general than "Web site". Topic maps may apply to any kind of data.

Our third contribution consists in enhancing navigation and information retrieval in a Web site. Information retrieval varies according to the needs of the user. If he looks for an answer to a specific question, query languages (like "tolog" [6]) are adapted. Their strength is to exploit the relationships between objects, which allows to answer questions better. For example, one may seek the Beatles' songs which were not written by John Lennon. This kind of information would be difficult to find with a traditional search engine.

If the subject of interest is clearly identified, it is easy to explore the corresponding topic in the topic map. This topic can be reached through a list of topics, for example an alphabetical list. Tools to navigate in topic maps have been designed so that any topic can be reached in 7 mouse clicks at most.

If the user has no precise question nor any clear subject of interest, none of the search modes described above can apply. This is the case of a beginner user who wishes to have a global understanding of the topic map so as to decide where to start his navigation. Therefore he first needs a simplified view of the topic map, with no detail, then he can decide to see more precise information as his subject of interest gets clearer. Let us compare this to geographical maps: there is no point in displaying very specific data on a map of the world. However, more and more details may be added as the user focuses on some part of the map. We propose to use clustering algorithms to group semantically related topic together at different abstraction levels. Clusters computation and visualisation are described in section 5.

The figure 1 shows that topic maps – or Web sites – characterization, filtering and clustering are deduced from the results of a conceptual classification algorithm based on Formal Concept Analysis and Galois connections. This algorithm is presented in section 3.

3 Conceptual classification algorithm

The starting point of our Web analysis is a conceptual classification algorithm based on Formal Concept Analysis and Galois connections. FCA is a mathematical approach to data analysis which provides information with structure. FCA may be used for conceptual clustering as shown in [4] and [12]. Let us first define a few terms:

- an object is a topic or an association of the topic map,
- the objects have characteristics called properties. We describe how these properties are determined in 3.1.

A profile allows to characterise a topic map in a structural way. With this footprint, one can tell if the topic map is specific or general. We can also tell if the objects of a topic map are similar or very different. In order to characterise objects, we use a Galois algorithm to classify the objects conceptually. This algorithm groups objects in concepts according to the properties they have in common. It is very powerful because it performs a semantic classification without having to express semantics explicitly. We will first describe how the objects and their

properties are generated from a topic map. Then we will describe Galois lattices and detail the statistical computations made on the objects. We will finally explain how the profile is determined.

3.1 Objects and properties generation

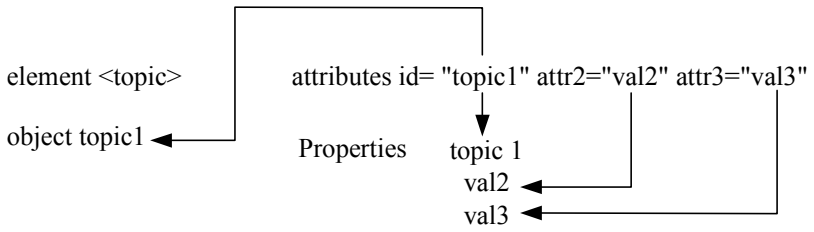
The generation of objects and properties is a 2-steps process.

- First step:

Every time there is an element with an identifier (that is an *id* attribute), a new object is created. The name of the object is the value of the identifier. As stated in the DTD (Document Type Definition), all topics and associations of the topic map have an identifier, so there will be the same number of objects as the number of topics and associations.

An object's properties correspond to the values of this object's attributes (including the value of the *id* attribute), as well as the values of his children' attributes. These properties are weighted (for instance, the weight of the values of *instanceOf* attributes may be greater than the weight of the values of *href* attributes).

Generation of object and properties (first step):



Example: consider the following extract of a topic map about music, written by Kal Ahmed[2]:

```
<topic id="t-the-clash">
    <instanceOf>
      <topicRef xlink:href="tt-band"/>
    </instanceOf>
    <baseName>
      <baseNameString>The Clash</baseNameString>
      <variant>
        <parameters>
          <topicRef xlink:href="http://www.topicmaps.org/xtm/1.0/psi-sort"/>
        </parameters>
        <variantName>
          <resourceData>clash the</resourceData>
        </variantName>
```

[2] Kal Ahmed works for Ontopia, http://www.ontopia.net

```
        </variant>
        <variant>
          <parameters>
            <topicRef xlink:href="http://www.topicmaps.org/xtm/1.0/psi-sort"/>
          </parameters>
          <variantName>
            <resourceData>Clash, The</resourceData>
          </variantName>
        </variant>
      </baseName>
    </topic>
```

An XML document is made of elements limited by tags and is hierarchically structured. In the example we studied, *topic*, *instanceOf* and *baseName* are elements. An element may have characteristics called attributes. The attributes of an element are declared inside the opening tag of the element. The element *topic* has an attribute *id* with a value *tt-clash*. The element *instanceOf* has no attribute.

When parsing the topic map, we find a topic which has an identifier with the value *t-the-clash*. An object *t-the-clash* is thus created.

In order to determine the properties of these objects, we look for all the attributes of this element. In this case, the only one is the identifier.

Then, we have a look at the children of this element (that is all the XML elements included in the element) to find their attributes. We repeat this for all the children.

In this example, the analysis of this abstract of the topic map creates an object *t-the-clash* with the properties *t-the-clash (*weight e.g. 0.5*)*, *tt-band* (weight e.g. 2) and *http://www.topicmaps.org/xtm/1.0/psi-sort (weight e.g. 0.2)*. The weights shown here correspond to one possible scenario - in which the type of a topic (weight 2) is more important than its name (weight 0.5), its occurrences (weight 0.2) or the associations it is involved in (weight 1).

In the same way, the analysis of the following abstract:

```
<topic id="tt-band">
      <instanceOf>
        <topicRef xlink:href="tt-music"/>
      </instanceOf>
      <baseName>
        <baseNameString>Band</baseNameString>
      </baseName>
</topic>
```

leads to the creation of an object *tt-band* with the properties *tt-band* (weight e.g. 0.5) and *tt-music* (weight e.g. 2).

The last example concerns an association:

```
<association id="assoc6">
      <instanceOf>
        <topicRef xlink:href="at-recorded"/>
```

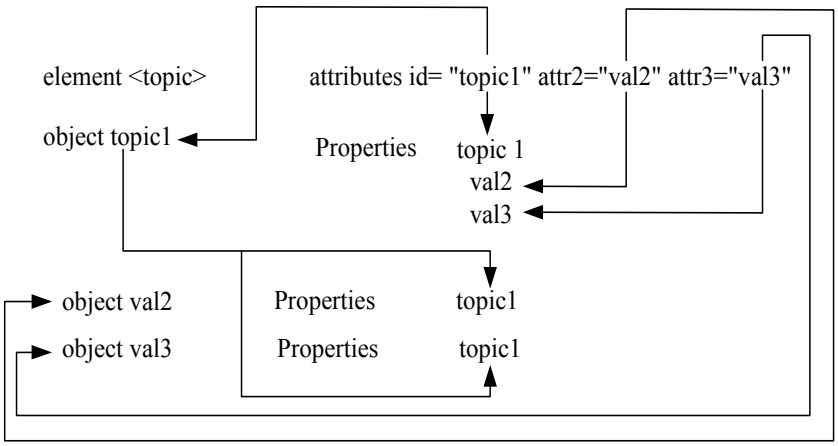
```
        </instanceOf>
        <member>
          <instanceOf>
            <topicRef xlink:href="tt-band"/>
          </instanceOf>
          <topicRef xlink:href="t-the-clash"/>
        </member>
        <member>
          <instanceOf>
            <topicRef xlink:href="tt-track"/>
          </instanceOf>
          <topicRef xlink:href="t-i-fought-the-law"/>
        </member>
    </association>
```

The object *assoc6* is created and has the properties *assoc6* (weight e.g. 0.5*)*, *at-recorded* (weight e.g. 2), *tt-band*, *t-the-clash*, *tt-track* and *t-i-fought-the-law*.

So far, the properties of an object are only intrinsic properties. Indeed, the object *t-the-clash* takes a part in the association *assoc6*, but this does not appear in its properties yet, since the association is not declared inside the topic which has the identifier *t-the-clash*. The second step takes these characteristics into account.

- Second step:

Generation principle of the objects and properties (second step):



The second steps adds non intrinsic properties to the objects by “crossing” the data. In fact, for an object O with a set of properties P, each property P becomes an object with O (amongst others) as a property. The properties of an object are its intrinsic properties and all the properties that were added recursively.

In the previous examples, the object *assoc6* has the properties *assoc6*, *tt-band* and *t-the-clash*. The property *assoc6* is added to the objects *tt-band* and *t-the-clash*. So all the objects know the associations they appear in.

Moreover, the object *t-the-clash* has the property *tt-band*. The data is crossed by adding *t-the clash* to the object *tt-band*. This example illustrates a new type of information, which was not present in the first step: the object *tt-band* knows it has an instance of *t-the-clash*. In the preceeding scenario, *t-the-clash* was the only one to know its superclass.

In the end, *tt-band* has the properties *tt-band* (weight e.g. 0.5), *tt-music*(weight e.g. 2) , *t-the-clash* (weight e.g. 1), *assoc1* (weight e.g. 1), *assoc2* (weight e.g. 1) and *assoc6* (weight e.g. 1). The object *t-the-clash* has the characteristics http://www.topicmaps.org/xtm/1.0/psi-sort (weight e.g. 0.2), *tt-band, t-the-clash* (weight e.g. 0.5), *assoc1* (weight e.g. 1), *assoc2* (weight e.g. 1) and *assoc6* (weight e.g. 1).

Note that the properties *assoc1* and *assoc2* correspond to other associations in which *tt-band* and *t-the-clash* appear. These associations are present in the topic map but not in the extracts we presented.

3.2 Introduction to Galois lattices

The notion of Galois lattice for a relationship between two sets is the basis of a set of conceptual classification methods. This notion was introduced by Birkhoff in [3] and by Barbut and Monjardet in [1]. Galois lattices consist in grouping objects into classes that materialise concepts of the domain under study. Individual objects are discriminated according to the properties they have in common. This algorithm is very powerful as it performs semantic classification. Topic maps are semantic structures themselves, but they may be very large and complex, so this algorithm is interesting to extract more semantics from them. The algorithm we implemented is based on the one that was proposed in [7].

Let us first introduce Galois lattices basic concepts.

Let two finite sets E and E' (E consists of a set of objects and E' is the set of these objects' properties), and a binary relation $R \subseteq E \times E'$ between these two sets. Figure 2 shows an example of binary relation between two sets. According to Wille's terminology [13], the triple (E, E', R) is a formal context which corresponds to a unique Galois lattice. It represents natural regroupings of E and E' elements.

Let P(E) be the powerset of E and P(E') the powerset of E'. Each element of the lattice is a couple, also called concept, noted (X, X'). A concept is composed of two sets $X \in P(E)$ and $X' \in P(E')$ which satisfy the two following properties :

$$X' = f(X) \text{ where } f(X) = \{ x' \in E' \mid \forall x \in X, xRx' \}$$
$$X = f'(X') \text{ where } f'(X') = \{ x \in E \mid \forall x' \in X', xRx' \} \quad (1)$$

A partial order on concepts is defined as follows :
Let C1=(X1, X'1) and C2=(X2, X'2),

$$C1 < C2 \Leftrightarrow X'1 \subseteq X'2 \Leftrightarrow X2 \subseteq X1 \quad (2)$$

This partial order is used to draw a graph called a Hasse diagram, as shown on figure 2. There is an edge between two concepts C1 and C2 if C1<C2 and there is no other element C3 in the lattice such as C1<C3<C2. In a Hasse diagram, the edge direction is upwards. This graph can be interpreted as a representation of the generalisation / specialisation relationship between couples, where C1<C2 means that C1 is more general than C2 (and C1 is above C2 in the diagram).

R	a	b	c	d	e	f	g	h
1	1	0	1	0	0	1	0	1
2	1	0	1	0	0	0	1	0
3	1	0	0	1	0	0	1	0
4	0	1	1	0	0	1	0	1
5	0	1	0	0	1	0	1	0

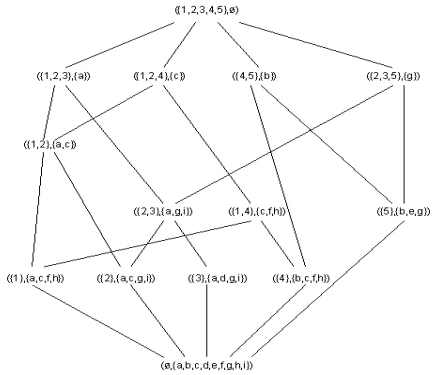


Fig. 2. Binary relationship and associated Galois lattice representation (Hasse diagram)

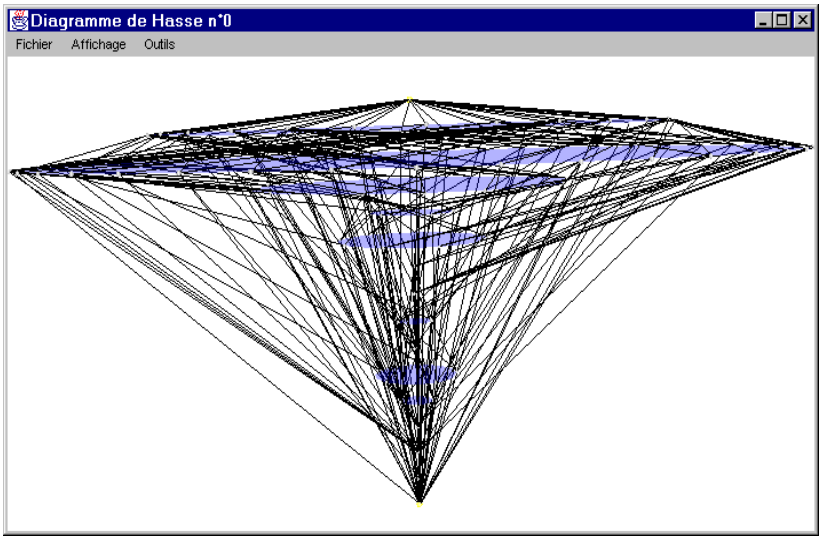


Fig. 3. Concept lattice of a topic map about music

The concept lattice shows the commonalities between the concepts of the context. The first part of a concept is the set of objects. It is called "extension". The second set – the intention - reveals the common properties of the extension's objects. The figure 3 shows the concept lattice generated from our example topic map about music.

4 Topic maps characterisation: conceptual profile

4.1 Calculating the statistics for every object.

We calculate statistics for every object of the topic map. We compute a weighted mean of these statistics. Each object has a weight which is assigned according to its importance in the topic map (the number of occurrences of the object in the XML source file).

Consider an object O. It is characterised by a vector with 6 components:

- The first component (A1) is the percentage of concepts of the sub-lattice where the object is present in the list of extensions. This value tells if O is present in many concepts of the lattice. A low value for A1 may indicate that O has few common characteristics with other objects. However, the other components allow to increase our knowledge.
- The second component is the maximum number of objects with which O is grouped, divided by the total number of objects. We have to select the concept containing O and with the largest number of objects. We add a constraint on this concept: it must contain at least one property. Indeed, we wish to group objects with common properties. The component A2 shows if O is grouped with many other objects. However, this value is a maximal value. The validity of A2 must be checked using A3.
- A3 is the mean number of objects with which O is grouped divided by the number of objects. This time we can tell if O is linked to a large number of objects and determine the significance of A2. If A3 is high, then there is a concept with O and many other concepts. On the other hand, if A3 is low, O is grouped with very few objects. The selected concept is thus an exception and we should not base our analysis on it.
- Let S be the set of objects which are grouped with O in one –or more- concepts of the lattice; these objects have at least one of O's properties. A4 is the maximum number of properties O shares with the objects contained in S, divided by the total number of objects. This component is deduced from the concept containing the object O and which has the greatest number of properties. Again, we add a constraint on this concept: it must contain at least two objects, that is at least one object different from O. We want to evaluate the number of shared properties, thus we need at least one object with which O shares them. A4 tells if the objects which are close to O share many common properties with O or not. Objects are more similar when they share an increasing number of properties. This similarity

can either be structural or conceptual. However, this value is a maximum number which must be validated with A5.

- A5 is the mean number of properties O shares with other objects, divided by the total number of properties. This tells the degree of significance of A4.
- Finally, A6 is about the topic map itself, and not about the lattice. It is the number of occurrences of the object in the topic map divided by the number of occurrences of objects of the same type (topic or association). A6 is used to compute the topic map's profile. This profile represents the characteristics of a mean object. Each component of this vector is the mean of the components of each object in the topic map, with a weight A6 given to each of these objects. Thus, objects with a high number of occurrences in the topic map will influence the profile much more than objects with few occurrences.

Note that the five first components are deduced from an analysis of the lattice whereas the last component only depends on the XML document.

4.2 Topic map – Web site - profile and selection of objects

When the statistics have been computed for every topic and association, the profile can be deduced. It is a vector for which each component is a mean of the components of all the objects with the weight A6 of each object. For N objects O_1, O_2, … O_N, each component A_i of the profile vector P is computed as follows:

$$P.A_i = \sum_{j=1}^{N} O_j.A_i * O_j.A_6 \quad \textbf{(3)}$$

where $O_j.A_i$ is the component A_i of the j-th object.

We wish to keep the most relevant objects, that is the ones which share "many" common properties with "many" other objects. These objects are called *regular* objects, they are semantically more significant than others. The significance of the words "many" (properties) and "many" (objects) is given by the topic map profile. A regular object is associated to at least as many objects and shares as many properties as the profile.

Among the statistics presented in section 4.1, the values A3 and A5 are more relevant than A2 and A4: maximum values may not give a reliable information because they may correspond to an exception. The comparison between the objects and the profile is thus done using the components A3 and A5.

A regular object O must verify the following conditions:

$$\begin{aligned} O.A_1 &\geq profile.A_1 \\ O.A_2 &\geq profile.A_2 \end{aligned} \quad \textbf{(4)}$$

This should be refined using the standard deviation. The standard deviation for A3 is the mean distance between an object's value of A3 and the profile's value of A3.

$$std.dev.A_3 = \frac{\sum_{j=1}^{N} |O_j.A_3 - P.A_3|}{N} \quad (5)$$

For A5, the standard deviation is computed in the same way:

$$std.dev.A_5 = \frac{\sum_{j=1}^{N} |O_j.A_5 - P.A_5|}{N} \quad (6)$$

Thus, a regular object is defined as follows:

$$O.A_1 + std.dev.A_1 \geq P.A_1$$
$$O.A_2 + std.dev.A_2 \geq P.A_2 \quad (7)$$

The regularity conditions can be changed (to be more or less restrictive) with a coefficient (C). Thus, a regular object meets the two following requirements:

$$O.A_1 + C \times std.dev.A_1 \geq P.A_1$$
$$O.A_2 + C \times std.dev.A_2 \geq P.A_2 \quad (8)$$

A non regular object is called a *singular* object –it conveys little semantics. When the objects of the topic map are submitted to these conditions, singular objects are eliminated. When C increases, more objects are suppressed since the conditions are harsher.

After this selection, we have a new list of objects which are used as an input for the Galois classification algorithm. A new lattice is generated and the statistics computed on this new panel of objects provide a new profile. We can thus select once again the regular objects for this new footprint of the topic map. The new regular objects are used again as an input for the Galois algorithm, etc. until all the objects become regular. This happens when no object is eliminated. The algorithm stalls and we get a stable list of regular objects which we must group together.

4.3 Results

Several topic maps – of different sizes and subjects - were analysed. The figure 4 displays the distribution of objets in three topic maps. The coordinates of the center of a disk correspond to the values of A3 and A5 attributes. The diameter of a circle is proportional to the number of objects which have these values for A3 and A5. All the objets of the *simple* topic map are very close. This topic map is qualified of "homogeneous", which means that all topics have the same semantic significance. *Music* and *icc* are "heterogeneous" structures. The objects in the lower left corner have low values for A3 and A5: they are "singular" - not much related to other

objects in the topic map. These topic maps can be filtered easily by eliminating these singular objects.

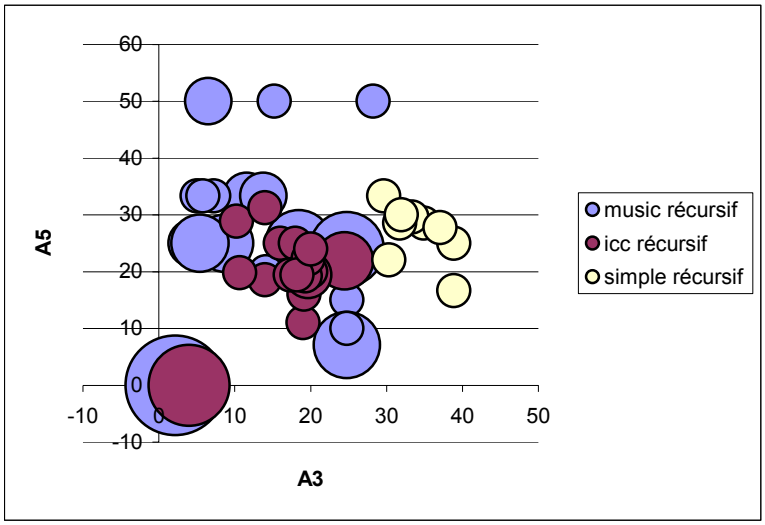


Fig. 4. Topics conceptual distribution

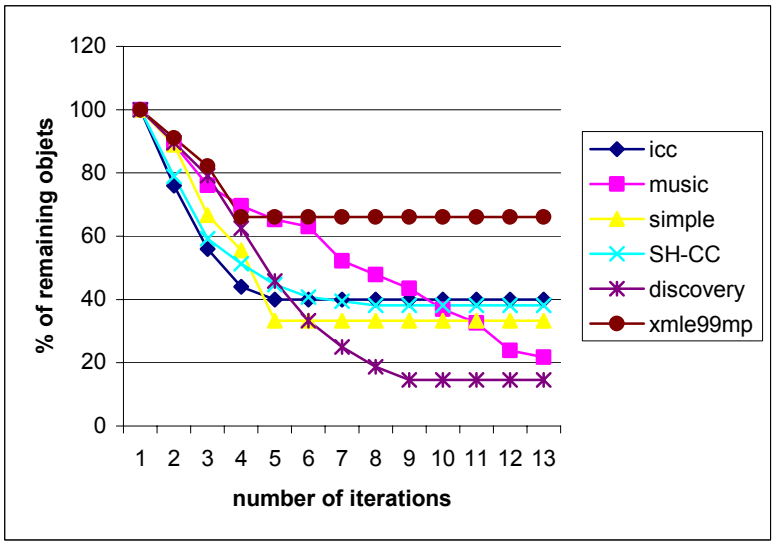


Fig. 5. Topic map filtering

The figure 5 illustrates the filtering of six topic maps. Some topic maps can be simplified a lot; this is the case of *discovery*. On the other end, after the last iteration, *xmle99mp* still contains almost 70% of its topics. This means that it is more difficult to filter this topic map: all topics have the same semantic value.

5 Topic maps clustering and visualisation

5.1 Clustering algorithm

The Galois lattice which is generated from a topic map contains some concepts which are made up of a set of topics which share common properties. The lattice gives an exhaustive description of the input data and the number of concepts generated may be very high. The concept lattice shown in the figure 3 is quite complex although it was generated from a small topic map (which contains 46 objects). We wish to group topics together into clusters in order to provide different level of detail (or scales) of the topic map. We propose to extract a tree from the Galois lattice. The concepts contained in this tree are the clusters. Thus, we have a hierarchy of clusters. The root of the tree contains all the topics; it is a gross cluster which provides no additional information. The next level groups some topics together, the next level executes a finer grouping of topics, etc. The number of levels of detail is given by the depth of the tree.

Many clustering algorithms exist; we chose to implement a clustering algorithm based on Galois conceptual classification. The clusters we generate are thus conceptually and semantically relevant. This algorithm also allows us to use the generalisation/specialisation relationship inherent to the Galois lattice.

To build the tree of clusters, we start from the representation which provides the greatest level of detail. Every cluster corresponds to an object: the objects are not grouped together. We begin to construct the leaves of the tree: these clusters correspond to the fathers of the upper bound of the lattice (which is represented at the bottom of the lattice on the Hasse diagram). This is the most specific level.

For each leaf, we select one unique father which is a generalisation of the concept. This selection is done according to a hierarchy of criteria which will be developed in the following. One father is selected for each selected node, and so on until the lower bound of the lattice is reached. At the end of this process, a tree is created. Each level of the tree contains clusters which correspond to a level of detail.

We defined a hierarchy of selection criteria when a concept has several fathers in the lattice.

- first, we consider the distance between each father and the lower bound of the lattice (this distance corresponds to the minimum number of edges between them).

- if one of the fathers' distance to the lower bound is smaller than the others, this node is selected. Being at lower distance from the lower bound means that this concept is semantically richer.
- if several nodes are at a minimum distance from the lower bound, we compare the sum of the weights of the properties contained in their intention. The node with the highest value is selected.
  - if several fathers meet this requirement, the algorithms chooses the one which minimises the total number of branches in the tree. If this condition is not unique, different scenarios are considered, one for each possible father.

5.2 Clusters analysis

Once the tree of clusters is generated, different measures may be computed, e.g. the proportion of concepts of the initial lattice which were not selected to be clusters.

The depth of the tree is interesting because it indicates the number of navigation levels that may be provided to the user. We also study the distribution of clusters at each abstraction level. If a cluster has no father, it means that it cannot be generalised. On the other hand, a cluster with no children corresponds to the most specific level.

We may also compute the distances between clusters. The distance between two clusters may be the average – or minimum, or maximum – distance between two objects (one in each cluster). Let O1 and O2 be two objects. Let P1 be the set of properties of O1 and P2 the set of properties of O2. Let INTER be the intersection of P1 and P2, and UNION the union of P1 and P2. The similarity between O1 and O2 is defined as:

$$S(O1,O2)=\frac{\sum_{i=1}^{card(INTER)} wi}{\sum_{j=1}^{card(UNION)} w'j} \quad (9)$$

The distance between O1 and O2 is given by (2):

$$D(O1,O2)=100-\frac{1}{S(O1,O2)} \quad (10)$$

5.3 Clusters representation

The levels of detail are symbolised by different colours. At each abstraction level, clusters are represented by portions of a disk, as shown in figure 6. Each cluster's size is proportional to the number of children this concept has. When the pointer of the mouse is over a cluster, its extension – the set of topics contained in this cluster – or intention – the set of these topics' properties – is displayed. When the user clicks on a part of the disk, this cluster becomes the current context – i.e. the whole disk -

and its content is displayed in greater detail. The disk in the upper left corner represents a global view of the topic map before focusing on a specific cluster.

The figure 6 shows the results of this clustering algorithm on our example topic map about music. These representations are SVG (Scalable Vector Graphics) graphics [15]. SVG is a language for describing two-dimensional graphics in XML. SVG drawings can be interactive and dynamic. SVG leverages and integrates with other W3C specifications and standards efforts. By leveraging and conforming to other standards, SVG becomes more powerful and makes it easier for users to learn how to incorporate SVG into their Web sites.

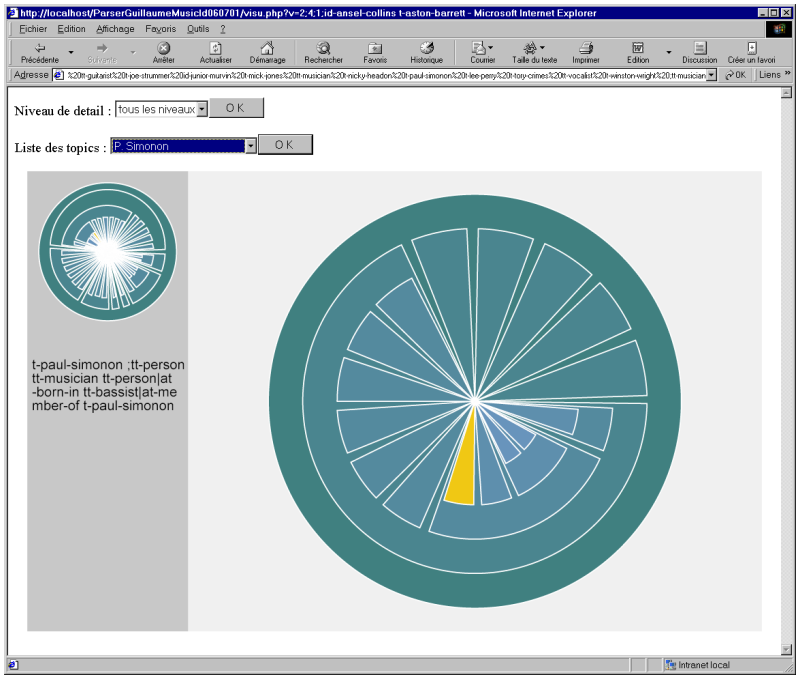


Fig. 6. Clusters visualisation

6 Conclusion and further work

This article presented how XML topic maps could be exploited to help users find relevant information on the Web. This contribution is at several levels: first, we characterise Web sites by defining their profile. This may be used to evaluate Web sites relevance with regard to a specific query. Second, our analysis identifies topics that have no interest – semantically speaking – which allows to "clean" the topic map. Finally we showed how we could enhance navigation by clustering Web pages and displaying them with different levels of details.

These results were deduced from the analysis of Galois lattices generated from Web sites with a conceptual classification algorithm. This algorithm is very powerful as it groups topics semantically.

In the future, we will study Web sites clusters in more details. For example, we noticed that some of the clusters are less relevant than others; it may thus be possible to further filter the Web site if it is really too large.

We will also investigate how ontologies may be used to characterise our clusters. Galois algorithm generates clusters which have a semantic value without expressing this semantics explicitly. Ontologies may help us make this information explicit.

7 Références

1. Barbut, M., Monjardet, B., *Ordre et classification*, Algebre et combinatoire, Tome 2, Hachette, 1970.
2. Berners-Lee, T., *A roadmap to the Semantic Web*, **http://www.w3.org/DesignIssues/Semantic.html**, Sept 1998.
3. Birkhoff, G., *Lattice Theory*, First Edition, Amer. Math. Soc. Pub. 25, Providence, R. I., 1940.
4. Carpineto, C., Romano, G., *Galois: An order-theoretic approach to conceptual clustering*, Proc. Of the 10th Conference on Machine Learning, Amherst, MA, Kaufmann, pp. 33-40, 1993.
5. Chein, M., Mugnier M.-L., *Conceptual Graphs : Fundamental Notions*, Revue d'intelligence artificielle, Volume 6 - n°4/1992, pp365-406, 1992.
6. Garshol, L. M., *"tolog" – A Topic Map Query Language*, XML Europe 2001, Berlin, Germany, 21-25 May 2001.
7. Godin, R, Chau, T.-T., *Incremental concept formation algorithms based on Galois Lattices*, Computational intelligence, 11, n° 2, p246 –267, 1998.
8. International Organization for Standardization, ISO/IEC 13250, *Information Technology-SGML Applications-Topic Maps*, Geneva: ISO, 1998.
9. Moore, G., *RDF and Topic Maps, An Exercise in Convergence*, XML Europe 2001, Berlin, Germany, 21-25 May 2001.
10. Sowa, J. F., *Conceptual Information Processing in Mind and Machine*, Reading, Massachusetts, Addison-Wesley, 1984.
11. TopicMaps.Org XTM Authoring Group, *XTM: XML Topic Maps (XTM) 1.0: TopicMaps.Org Specification*, 3 March 2001.
12. Wille, R., *Line diagrams of hierarchical concept systems*, Int. Classif. 11, pp. 77-86, 1984.
13. Wille, R., Concept lattices and conceptual knowledge systems, Computers & Mathematics Applications, 23, n° 6-9, pp. 493-515, 1992.
14. World Wide Web Consortium, Resource Description Framework (RDF) Model and Syntax Specification, W3C Recommendation, 22 February 1999.
15. World Wide Web Consortium, Scalable Vector Graphics (SVG) 1.0 Specification, W3C Candidate Recommendation, 2 November 2000.
16. Woods, W.A., What's in a link: foundations for semantic networks, In D.G. Bobrow and A.M. Collins, (Eds.), Representation and Understanding: Studies in Cognitive Science,. New York: Academic Press. p. 35-82, 1975.