Proceedings of the 2007 workshop on

Contexts and Ontologies Representation and Reasoning

(C&O:RR-2007)

Paolo Bouquet
Jérôme Euzenat
Chiara Ghidini
Deborah L. McGuinness
Valeria de Paiva
Luciano Serafini
Pavel Shvaiko
Holger Wache
(Editors)

AUGUST 2007 ROSKILDE UNIVERSITY COMPUTER SCIENCE RESEARCH REPORT #115

Paolo Bouquet, Jérôme Euzenat, Chiara Ghidini, Deborah L. McGuinness, Valeria de Paiva, Luciano Serafini, Pavel Shvaiko, and Holger Wache

Computer Science
Roskilde University
P. O. Box 260
DK–4000 Roskilde
Denmark

Telephone: +45 4674 3839
Telefax: +45 4674 3072
Internet: http://www.ruc.dk/dat/
E-mail: datalogi@ruc.dk

ISSN 0109–9779

This research report constitutes the proceedings of the *2007 workshop on Contexts and Ontologies Representation and Reasoning (C&O:RR-2007)* which is held in conjunction with the 6th International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT 2007), Roskilde University, Denmark, August 2007.

Research reports are available electronically from:

http://www.ruc.dk/dat/

P. Bouquet, J. Euzenat, C. Ghidini, D. L. McGuinness,
V. de Paiva, L. Serafini, P. Shvaiko, H. Wache (Eds.)

Contexts and Ontologies Representation and Reasoning C&O:RR-2007

Working notes of the workshop
collocated with CONTEXT 07
Roskilde University, Denmark,
21 August 2007
http://www.ruc.dk/dat_en/research/reports/115/

Foreword

Contexts and ontologies play a crucial role in knowledge representation and reasoning. Computer systems which act intelligently need the ability to represent, utilize and reason about contexts and ontologies. Recently we have seen a number of projects devoted to the definition and usage of contexts as well as ontologies in intelligent KR systems. With the advent of the web and the ubiquitous connectivity it brings, contexts and ontologies have become a relevant notion also in other, more recent, applications. Many such applications, including for example, information integration, distributed knowledge management, semantic web, multi-agent systems, distributed reasoning, data grid and grid computing, pervasive computing and ambient intelligence, and peer-to-peer information management systems, have acknowledged the need for methods to represent and reason about knowledge which is scattered in a large collection of contexts and ontologies.

The workshop Contexts and Ontologies: Representation and Reasoning (abbreviated as C&O:RR) is the result of a merge of two successful lines of workshops (Contexts and Ontologies: Theory, Practice and Applications (C&O) and Context Representation and Reasoning (CRR)) investigating these issues. The previous workshops have focused on the themes of combining ontologies, contexts, and contextual reasoning. The new workshop keeps the focus on the combination of contexts and ontologies, but also emphasizes the representation and reasoning aspects of this research. The Context and Ontologies Representation and Reasoning workshop welcomes papers on contexts and ontologies, with a focus on approaches to semantic heterogeneity and on the analysis and understanding of the combination of contexts and ontologies from the knowledge representation and reasoning perspectives.

The organizers would like to thank the members of our programme committee for their careful work and our invited speakers, Professors Frank Wolter and David Robertson for their inspiring contributions.

Organizing Committee

- Paolo Bouquet, University of Trento, Italy
- Jérôme Euzenat, INRIA Rhône-Alpes & LIG, France
- Chiara Ghidini, Fondazione Bruno Kessler (ITC-IRST), Italy
- Deborah L. McGuinness, Stanford University, USA
- Valeria de Paiva, Palo Alto Research Center (PARC), USA
- Luciano Serafini, Fondazione Bruno Kessler (ITC-IRST), Italy
- Pavel Shvaiko, University of Trento, Italy
- Holger Wache, University of Applied Sciences Northwestern Switzerland, Switzerland

Program Committee

- Horacio Arlo-Costa, Carnegie Mellon University, USA
- Alex Borgida, Rutgers University, USA
- Fausto Giunchiglia, University of Trento, Italy
- Vasant Honavar, Iowa State University, USA
- Yannis Kalfoglou, University of Southampton, UK
- David Leake, Indiana University, USA
- Maurizio Marchese, University of Trento, Italy
- Leo Obrst, MITRE, USA
- Fano Ramparany, France Telecom R&D, France
- Chantal Reynaud, Universit Paris-Sud, France
- David Robertson, University of Edinburgh, UK
- Riccardo Rosati, University of Rome "La Sapienza", Italy
- Thomas Roth-Berghofer, DFKI, Germany
- Aviv Segev, Technion, Israel
- Heiner Stuckenschmidt, University of Mannheim, Germany
- York Sure, AIFB, University of Karlsruhe, Germany
- Andrei Tamilin, Fondazione Bruno Kessler (ITC-IRST), Italy
- Sergio Tessaris, Free University of Bolzano/Bozen, Italy
- Rich Thomason, University of Michigan, USA
- Roy Turner, University of Maine, USA
- Ludger van Elst, DFKI, Germany
- Frank Wolter, University of Liverpool, UK

Acknowledgements

We appreciate the support from the KnowledgeWeb European Network of Excellence (IST-2004-507482).

Invited Talk: Modularity in logical theories and ontologies
Frank Wolter, University of Liverpool, UK

Modularity of logical theories is a classical subject in mathematical logic and philosophy of science. Due to the ever increasing size and complexity of logical theories that are used to represent ontologies and software specifications, the problem of defining appropriate notions of modularity and of providing reasoning support for dealing with modularity has recently become an important research topic also in these areas. In this talk, we introduce and survey recent progress in the field. We start with a general introduction to modularity in the context of propositional and first-order logic, highlighting the connection to classical notions such as conservative extensions and interpolation. We then switch to ontologies and description logics, introduce different notions of modularity, analyze their interrelation and possible use, and consider a variety of reasoning tasks such as the extraction of a module from an ontology.

Biographical Note:

Frank Wolter is Professor for Logic and Computation at the Department of Computer Science, University of Liverpool. He works in Knowledge Representation and Reasoning and Logic in Computer Science. His main interests are in Modal Logic (theory and applications), Description Logic and their application as ontology languages, Spatial and Temporal Reasoning, and Combining Logics. Frank Wolter is co-editor of the Handbook of Modal Logic (Elsevier, 2007) and co-author of the research monograph Many-dimensional Modal Logic: Theory and Applications (Elsevier, 2003). He is member of the Steering Committees 'Advances in Modal Logic (AiML)', 'Description Logic Workshop (DL),' and 'Frontiers of Combining Systems (FroCoS)'.

Invited Talk: Interaction as context: the OpenKnowledge experience
David Robertson, University of Edinburgh, UK

Context often is viewed by traditional knowledge engineers as a problem: our beautifully crafted ontologies tend to break as we shift them from one context to another and then we become frustrated in our attempts to prevent such breakages by attempting to standardise across ontologies. The OpenKnowledge project (www.openk.org) has taken a different view; it accepts that context will radically influence the semantics of the knowledge conveyed during interaction between systems (both automated and human) and requires all knowledge sharing to be situated with respect to a (standardised) model of the interaction for which that knowledge is being shared. We have provided a lightweight infrastructure in which this sort of context can be shared at very low cost, providing a form of web service choreography in the process. With this infrastructure we then have a different starting point for addressing problems such as ontology mapping, service matchmaking and assessment of reputation of services.

Biographical Note:

Dave Robertson is the Director of the Centre for Intelligent Systems and their Applications, part of the School of Informatics at the University of Edinburgh. His current research is on formal methods for coordination and knowledge sharing in distributed, open systems - the long term goal being to develop theories, languages and tools that out-perform conventional software engineering approaches in these arenas. He is coordinator of the OpenKnowledge project (www.openk.org) and was a principal investigator on the Advanced Knowledge Technologies research consortium (www.aktors.org), which are major EU and UK projects in this area. His earlier work was primarily on program synthesis and on the high level specification of programs, where he built some of the earliest systems for automating the construction of large programs from domain-specific requirements. He has contributed to the methodology of the field by developing the use of "lightweight" formal methods - traditional formal methods made much simpler to use in an engineering context by tailoring them to a specific type of task. As an undergraduate he trained as a biologist and continues to prefer biology-related applications of his research, although methods from his group have been applied to other areas such as astronomy, simulation of consumer behaviour and emergency response.

Table of Contents

Full papers

Short papers

Ontology-Driven Association Rule Extraction: A Case Study

Andrea Bellandi[1], Barbara Furletti[1], Valerio Grossi[2], and Andrea Romei[2]

[1] IMT - Lucca Institute for Advanced Studies
Piazza S. Ponziano, 6 - 55100 Lucca, ITALY
{a.bellandi, b.furletti}@imtlucca.it

[2] Department of Computer Science - University of Pisa
Largo B. Pontecorvo, 3 - 56127 Pisa, ITALY
{romei, vgrossi}@di.unipi.it

Abstract. This paper proposes an integrated framework for extracting *Constraint-based Multi-level Association Rules* with an ontology support. The system permits the definition of a set of domain-specific constraints on a specific domain ontology, and to query the ontology for filtering the instances used in the association rule mining process. This method can improve the quality of the extracted associations rules in terms of relevance and understandability.

1 Introduction

The *Data Mining* (DM) results, i.e. the models, represent relations in the data and are usually employed for classifying new data or for describing correlations hidden in the data. In this paper, we focus on the *Association Rule Mining* as originally introduced by Agrawal et al. in [2] and on a way for improving the process results. There are several ways to reduce the computational complexity of Association Rule Mining and to increase the quality of the extracted rules: (i) reducing the search space; (ii) exploiting efficient data structures; (iii) adopting domain-specific constraints. The first two classes of optimizations are used for reducing the number of steps of the algorithm, for re-organizing the itemsets, for encoding the items, and for organizing the transactions in order to minimize the algorithm time complexity. The third class tries to overcome the lack of user data-exploration by handling domain-specific constraints. This paper focuses on these optimizations by representing a specific domain by means of an ontology and driving the extraction of association rules by expressing constraints. The aim of this work is to reduce the "search space" of the algorithm and to improve the significance of the association rules.

Paper Organization. Section 2 provides some notions of OWL ontologies, data mining and association rules. Section 3 introduces the syntax of the constraints and describes the process. Section 4 presents a case study based on a real dataset. Section 5 discusses the related works and section 6 proposes some ideas for further improvements.

2 Background knowledge

OWL Overview

OWL is a family of three ontology languages: $OWL - Lite$, $OWL - DL$, and $OWL - Full$. The first two languages can be considered syntactic variants of the $\mathcal{SHIF}(\mathcal{D})$ and $\mathcal{SHOIN}(\mathcal{D})$ description logics (DL), respectively, whereas the third language was designed to provide full compatibility with RDF(S). We focus mainly on the first two variants of OWL because OWL-Full has a nonstandard semantics that makes the language undecidable and therefore difficult to implement. OWL comes with several syntaxes, all of which are rather verbose. Hence, in this paper we use the standard DL syntax [3]. The main building blocks of DL knowledge bases are concepts (or classes), representing sets of objects, roles (or properties), representing relationships between objects, and individuals representing specific objects. OWL ontologies consist of two parts: intensional and extensional. The former part consists of a *TBox* and an *RBox*, and contains knowledge about concepts (i.e. classes) and the complex relations between them (i.e. roles). The latter part consists of an *ABox*, and contains knowledge about entities and how they relate to the classes and roles from the intensional part. In our scenario, TBox and RBox shall provide supermarket domain knowledge, while all the supermarket items constitute ABoxes which are interlinked with intensional knowledge.

The semantics for OWL DL is fairly standard. An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is a tuple where $\Delta^{\mathcal{I}}$, the domain of discourse, is the union of two disjoint sets $\Delta^{\mathcal{I}}_{O}$ (the object domain) and $\Delta^{\mathcal{I}}_{D}$ (the data domain) and $\mathcal{I}$ is the interpretation function that gives meaning to the entities defined in the ontology. $\mathcal{I}$ maps each OWL class C to a subset $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}_{O}$, each object property P_{Obj} to a binary relation $P^{\mathcal{I}}_{Obj} \subseteq \Delta^{\mathcal{I}}_{O} \times \Delta^{\mathcal{I}}_{O}$, and each datatype property P_{Data} to a binary relation $P^{\mathcal{I}}_{Data} \subseteq \Delta^{\mathcal{I}}_{O} \times \Delta^{\mathcal{I}}_{D}$. The whole definition is in the OWL W3C Recommendation (http://www.w3.org/TR/owl-semantics/).

Data Mining and Association Rules

Data mining is the analysis of (often large) observational data sets to find unsuspected relationships and to summarize the data in novel ways that are both understandable and useful to the data owner. The relationships and summaries derived through a data mining exercise are often referred to as models or patterns. The main tasks of Data mining are generally divided in two categories: *Predictive* and *Descriptive*. The objective of the predictive tasks is to predict the value of a particular attribute based on the values of other attributes, while for the descriptive ones, is to derive patterns (correlations, trends, clusters, ...) that summarize the relationships in the data.
The Association rule mining is one of the major techniques of data mining and it is perhaps the most common form of local-pattern discovery in unsupervised learning systems. These methodologies retrieve all possible interesting patterns in the database. Given a database D of transactions, where each transaction

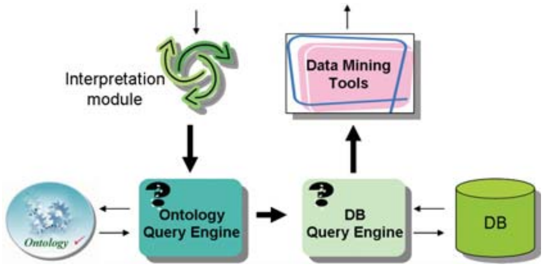

Fig. 1. The system architecture.

$T \in D$ is a set of items, an association rule is a (statistical) implication of the form $X \rightarrow Y$, where $X, Y \in D$ and $X \cap Y = \emptyset$. A rule $X \rightarrow Y$ is said to have a *support* (or frequency) factor s if and only if, at least $s\%$ of the transations in T satisfy $X \cup Y$. A rule $X \rightarrow Y$ is satisfied in the set of transactions T with a $confidence$ factor c if and only if, at least $c\%$ of the transactions in T that satisfy X also satisfy Y. The support is a measure of statistical significance, whereas the confidence is a measure of the strength of the rule. A rule is said to be "interesting" if its support and confidence are greater than user-defined thresholds sup_{min} and con_{min}, respectively, and the objective of the mining process is to find all such interesting rules [13].

3 Description of the approach

In this section, we describe our approach for guiding the extraction process of *Multi-level Constraint-based Association Rules* with an ontology support. Our scenario consists of the set of components shown in figure 1. The ontology ($O_{\mathcal{D}}$) describes the domain of interest ($\mathcal{D}$) and it is used as a means of meta-data representation. The interpretation module translates the requests of an user (*user constraints*) into a set of $formal$ *constrains* ($Q_{\mathcal{D}}$ defined on $O_{\mathcal{D}}$) so that they can be supplied to the Ontology Query Engine by means of a suitable query language. The aim of these constraints is to exclude some items from the output association rules, or to characterize interesting items according to an abstraction level. The *user constraints* syntax is formalized in table 1. It includes both *pruning constraints*, used for filtering a set of non-interesting items, and *abstraction constraints*, which permit a generalization of an item to a concept of the ontology. By using pruning constraints, one can specify the exclusion of a set of items from the input transactions set, and, as a consequence, from the extracted rules. This kind of constraints refers either to a single item, or to an ontology concept, and they can include a condition expressed on a set of ontology properties. Abstraction constraints permit exploring different levels of the ontology concepts. The generalization to a predefined level of the hierarchy

$\mathcal{I}$ is the set of items ($i_1, i_2, ...i_n \in \mathcal{I}$).
$\mathcal{C}$ is the set of the concepts of the ontology ($c_1, c_2, ...c_n \in \mathcal{C}$).
$\mathcal{P}_c$ is the set of the properties of the concept $c \in \mathcal{C}$ ($p_1, p_2, ...p_n \in \mathcal{P}_c$).
$cond_c$ is a Description Logic expression.
ALL represents all the instances defined in the ontology.

A constraint is defined on $\mathcal{I}$, $\mathcal{C}$ and $\mathcal{P}_\mathcal{C}$ in the following form:

1. **Pruning Constraints.** A pruning constraint is of one of the following forms:
   (a) $prune(e)$, where $e \in \mathcal{I} \cup \mathcal{C} \cup \{ALL\}$.
   (b) $prune_{cond_c}(c)$, where $c \in \mathcal{C} \cup \{ALL\}$.
2. **Abstraction Constraints.** An abstraction constraint is of one of the following forms:
   (a) $abstract(e, c)$, where $e \in \mathcal{I} \cup \mathcal{C}$, $c \in \mathcal{C}$ and c is a super-concept of e.
   (b) $abstract_{cond_{c_1}}(c_1, c_2)$ where $c_1 \in \mathcal{C} \cup \{ALL\}$, $c_2 \in \mathcal{C}$ and c_2 is a super-concept of c_1.
   (c) $abstract^l_{cond_e}(e)$, where $e \in \mathcal{I} \cup \mathcal{C} \cup \{ALL\}$, and l is a non-negative integer indicating the level of the hierarchy; *cond* can be unspecified.

Table 1. User constraints syntax.

improves the support of association rules, and consequently avoids the discovery of a massive quantity of useless rules, especially in case of sparse data.

The ontology query engine interacts with the ontology by performing the set $Q_\mathcal{D}$ of queries. The resulting $R_\mathcal{D}$ instances set, is used by the DB query engine for retrieving the instances that contain the filtered/abstracted/pruned items (i.e., the items specified in $R_\mathcal{D}$). The data base is the repository of the data to pass in input to the data mining tools. The box "Data Mining Tools" contains the tool for analyzing and processing the data. In our context we refer to a specific algorithm for extracting association rules, but we would like to point out that the system can operate with other kinds of DM tools. The support and the confidence measures are initially provided by the user.

4 Case study: a Market Basket Analysis application

In this section we show the results of a case study by using data taken from a national supermarket, and stored in a relational database (DB). The aim of this study is to construct and test the framework described in the previous section with real data and w.r.t. specific market analysis. In this case, the data consist of a set of purchase transactions $T = [transID, item]$, where *transID* is the cash voucher identification and *item* is the purchased item. The DB contains 775,000 transactions. According to the approach proposed in sec. 3, meta-data (description of the items) and data to analyze have been organized respectively in separate structures:

The ontology - contains the description of the items and their hierarchical organization. Starting from the DB structure (tables and fields)[3], we derived the OWL ontology schema mapping the fields of the DB tables in classes and properties of the ontology. Also, we automatically filled up the ontology with about 30,000 items, and their attributes (approximately 100).

Let us consider the item *Vodka Keglevich Melon*. The correspondent hierarchical structure and the list of the item attributes are shown in the table below.

Hierarchical Structure	Attributes of Vodka
⊙ *owl:Things*	*hasColour*: transparent;
∇ ⊙ *XXX Supermarket*	*hasAlcoholicContent*: high;
∇ ⊙ *L0 Foodstuffs and Drinks Department*	*hasFlavour*: Melon;
∇ ⊙ *L1 Drinks*	*hasBrand*: Keglevich;
∇ ⊙ *L2 Vodka*	*isFizzy*: No;
∇ ⊙ *L3 Spicy*	*hasPrice*: EUR 7.56;
⊙ *Vodka Keglevich Melon*	*hasSize*: 70 cl;

The DB - contains the transactions T.

The experimentation has been conducted using *SeRQL* ("*Sesame* RDF Query Language") [4] language for querying the ontology and the Apriori algorithm [1] for mining association rules. *SeRQL* is an RDF/RDFS query language that is currently being developed by Aduna as part of *Sesame* [5]. It combines the best features of other (query) languages (RQL, RDQL, N-Triples, N3) and adds some of its own. Sesame is a RDF database which can be employed to manage RDF triples.

In the first two tests we abstract all items to two upper levels (level L2 and level L1) for verifying what categories of items are bought together. In this way we abstract all items to only 14 high level concepts in the first case and to only 4 high level concepts in the second one. These abstraction constraints can be expressed respectively as:

$$Query\ 1 \equiv abstract^2(ALL)$$
$$Query\ 2 \equiv abstract^1(ALL)$$

The third test concerns an investigation for organizing a future promotional campaign during the holidays (Christmas and Easter). The focus is on typical sweets and cakes (with well-known brands) of the two holidays, and the alcoholic drinks. The objective is to verify how those articles are related. All kinds of sweets/cakes are abstracted to *Foodstuffs* (associated with the item brand) and all kinds of alcoholic drinks to *Drinks*. These constraints can be expressed as:

$$\begin{aligned} Query\ 3 \equiv\ & prune_{(\exists hasBrand.=_{null})}(ALL) \wedge abstract_{(\exists hasBrand.<>_{null})}(Alcoholic,\ Drinks) \\ & \wedge abstract_{(\exists hasBrand.<>_{null})}(Sweets,\ FoodStuffs) \\ & \wedge abstract_{((\exists hasRecurrence.=_{Easter}) \sqcup (\exists hasRecurrence.=_{Christmas}))}(Sweets,\ FoodStuffs) \end{aligned}$$

[3] We considered the DB table named *Marketing* that, for each article, specifies a hierarchical structure w.r.t. the department organization in the supermarket.

The part of the ontology schema (i.e. the part of the DL knowledge base) related to the *Query* 3 can be expressed by the following TBox fragment:

$$\begin{aligned}\mathcal{T}_3 = {} & EatableThing \sqsubseteq (\exists hasBrand.string) \sqcap (= 1 hasBrand) \\ & \sqcap\ EatableThing \sqsubseteq (\exists hasRecurrence.string) \sqcap (\geq 0 hasRecurrence) \\ & \sqcap\ (Drinks \sqsubseteq EatableThing) \sqcap (FoodStuffs \sqsubseteq EatableThing) \\ & \sqcap\ (Alcoholic \sqsubseteq Drinks) \sqcap (Sweets \sqsubseteq FoodStuffs)\end{aligned}$$

According to the interpretation function $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ defined in section 2, the semantic interpretation of the conditions expressed by the *abstract* clauses is:

$$\begin{aligned}& \Bigg(Alcoholic \sqsubseteq (\exists hasBrand. <>_{null}) \\ & \sqcup\ Sweets \sqsubseteq (\exists hasBrand. <>_{null}) \\ & \sqcap \Big((\exists hasRecurrence. =_{Easter}) \sqcup (\exists hasRecurrence. =_{Christmas}) \Big) \Bigg)^{\mathcal{I}} \\ & = Alcoholic^{\mathcal{I}} \cap \{x_a \mid \exists y_a.(x_a, y_a) \in hasBrand^{\mathcal{I}} \wedge y_a \neq null^{\mathcal{I}}\} \\ & \quad \cup\ Sweets^{\mathcal{I}} \cap \{x_s \mid \exists y_s.(x_s, y_s) \in hasBrand^{\mathcal{I}} \wedge y_s \neq null^{\mathcal{I}}\} \\ & \qquad \cap \Big(\{z \mid \exists w.(z, w) \in hasRecurrence^{\mathcal{I}} \wedge w = Easter^{\mathcal{I}}\} \\ & \qquad \cup\ \{h \mid \exists k.(h, k) \in hasRecurrence^{\mathcal{I}} \wedge k = Christmas^{\mathcal{I}}\} \Big) \\ & = \{\mathcal{A}\} \cap \{x_a \mid \exists y_a.(x_a, y_a) \in \{(a, b_a)\} \wedge y_a \neq null\} \\ & \quad \cup\ \{\mathcal{S}\} \cap \{x_s \mid \exists y_s.(x_s, y_s) \in \{(s, b_s)\} \wedge y_s \neq null\} \\ & \quad \cap \Big(\{z_s \mid \exists w_s.(z_s, w_s) \in \{(p, r_p)\} \wedge w_s = Easter\} \\ & \quad \cup\ \{h \mid \exists k_s.(h_s, k_s) \in \{(q, r_q)\} \wedge k_s = Christmas\} \Big) \\ & = \{\mathcal{A}\} \cap \{(a, brand)\} \\ & \quad \cup\ \{\mathcal{S}\} \cap \{(s, brand)\} \\ & \quad \cap \Big(\{(p, Easter)\} \cup \{(q, Christmas)\} \Big) \\ & = \{(alcoholic, b_a)\} \cup \{(sweets_{Easter}, b_s)\} \cup \{(sweets_{Christmas}, b_s)\}\end{aligned}$$

where $\{\mathcal{A}\}$ and $\{\mathcal{S}\}$ are the instances sets of the classes *Alcoholic* and *Sweets* respectively, with $a \in \{\mathcal{A}\}$ and $s, p, q \in \{\mathcal{S}\}$; b_a, b_s are any well-known brands of *Alcoholic* and *Sweets* respectively. The semantic expressed by *prune* clause is very similar to *abstract* so we omit it for lack of space.

In the last test, we consider the case in which the supermarket augments its services by introducing a new department (*Assisted Service*). This event introduces an innovation in the supermarket domain, so we have to modify the ontology[4] i.e. we have to introduce a new data property, for some category (*typeOfService* (ToS) with enumerated type *Assisted Service*, *Take Away*, *Free Service*). We abstract to level $L2$ all the items with *typeOfService* equals to *Assisted Service* or *Take Away*, ignoring the others. This constraint can be expressed as:

[4] Notice that, the introduction of a new property does not imply the re-engineering of the structure, but only the introduction of the property in the higher classes so that the property is inherited by each subclasses.

$$Query\ 4 \equiv abstract^2_{(\exists hasToS.=_{AssistedService})}(ALL)$$
$$\wedge\ abstract^2_{(\exists hasToS.=_{TakeAway})}(ALL)$$
$$\wedge\ prune_{((\exists hasTos.<>_{AssistedService})\sqcap(\exists hasTos.<>_{TakeAway}))}(ALL)$$

For the lack of space we omit the semantic interpretation of the *Query*4.

For evaluating our framework we submitted to the system the queries introduced above. Our framework automatically translates these constraints into *SeRQL* language for querying the ontology. In all tests we applied the Apriori implementation of the KDDML System [10], setting the support threshold to 1%[5], and confidence to 50%. In Table 2, the five rows represent the results of the tests. The first query labeled $\text{no}_{constraints}$ represents the request without any constraints. *#Trans* reports the number of transactions that satisfy the constraints, *#Items* reports the total number of different articles that compose the transactions, *#Itemsets* and *#Rules* report the number of itemsets and the rules computed by the Apriori, respectively. Furthermore *LI* and *AI* contain statistical information about the number of items contained in the largest transaction, and the average number of items contained in a transaction. In figure 2

Query ID	**#Trans**	**#Items**	**#Itemsets**	**#Rules**	**LI**	**AI**
no $\text{constraints}_{Query0}$	91563	123	176	50	76	7.68
Test 1_{Query1}	80765	11	524	1248	12	3.86
Test 2_{Query2}	76323	4	15	31	4	2.83
Test 3_{Query3}	352	33	60	9	6	2.17
Test 4_{Query4}	69534	10	200	258	10	4.03

Table 2. Queries summary results

we report the supports graph of the queries. In the abscissa there are the top 50 frequent itemsets, while in the ordinate there is the support related to the i^{th} frequent item. As you can notice, in the picture the result of Test 2 has not been reported because it contains only 15 frequent itemsets. The use of real data typically brings issues related to the quality of the extracted model. Items at the lower levels of the taxonomy may not have enough support to appear in any frequent itemsets. This aspect is underlined in figure 2 in which we can notice that the *Query* 0 retrieves only itemsets with a very low support. This is mainly due to the large number of articles. Moreover, rules extracted at the lower levels of a concept, are too much specific, and may not be interesting. Consider for example the following rule extracted at low level:

$$\{bread,\ red\ wine,\ ham,\ chocolate\ cake\} \Rightarrow \{roasted\ chicken,\ cooked\ lasagne\}$$
$$[supp = 0.02, conf = 0.57].$$

The rule is not relevant due to the low support. Consider instead the following rule, that corresponds to the previous, but at an higher level of abstraction, and

[5] This low support threshold is dued to the large number of items.

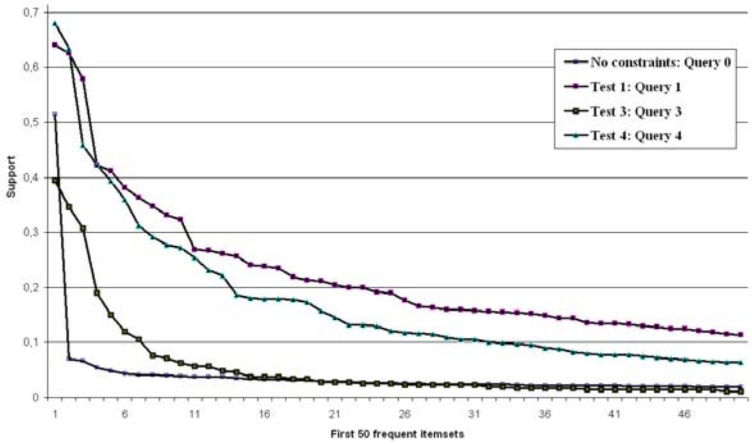

Fig. 2. Compared Supports.

satisfying $Query$ 4:

$\{FoodStuffs\ AssistitedService\} \Rightarrow \{FoodStuffs\ TakeAway\}$

$[supp = 0.26, conf = 0.68]$.

This rule abstracts all the items to level L2 of the ontology and each of them is selected by the *typeOfService* property. The information extracted from this association rule can suggest that the assisted service department has to provide to the customers also (take away) cooked meals ($roastedchicken, cookedlasagne$). In general, items abstracted at the higher levels, tend to have higher support counts. This fact increases the quality of the extracted rules, and as consequence, helps the analyst in the decision support. Association rules related to $Query$ 3, for example, emphasize the concept of multi-level rule correlating concepts at different abstraction level. For example the concept *FoodStuffs* (level L2) with $BAULI$ and $MOTTA$ as brands, and *Drinks* (level L2) with $ASTI$[6], are related to $Red\ Meats$ (level L7) slaughtered and packed by the supermarket. It can suggest to the analyst some marketing decisions on these products during Easter or Christmas period.

The study of multi-level association rules is well-known in literature, and in this context, our work may not seem innovative. The focus of our approach is the introduction of the expressive power of ontologies for constraint-based multi-level association rule mining. The main advantages can be summarized in terms of *extensibility* and *flexibility*. Our framework is extensible because data properties and concepts can be introduced in the ontology without either changing the relational database containing the transaction, or the implementation of our framework. The flexibility is guaranteed from the separation of the data to

[6] $MOTTA$, $BAULI$ and $ASTI$ are Italian food and drink brands.

analyze (the transactions) from the meta-data (description of the data). Furthermore it interesting to point out that our approach is general, and can be adapted to further data mining analysis.

5 Related Works

Methods to define and integrate item constraints are originally introduced by Srinkant and Agrawal in [11] and by Han and Fu in [7]. Recently, in [12] and [9], we can find the attempt to integrate the item-constraints evaluation directly in the rule extraction algorithm. In [12], the authors concentrate on improving the Apriori algorithm, while in [9] the authors focus on the definition of a two-phase approach: specification of the constraint association queries, and submission of the constraints in the mining process.

Our approach follows the research line proposed by the cited works, nevertheless it introduces three main differences: (i) we employ an ontology to represent an item taxonomy; (ii) constraints can be defined on the basis of specific properties of the items; (iii) by using an ontology instead of a taxonomy, a new item property or a concept can be added without re-engineering the (meta-data) representation model or the relational database.

Other studies concern the merging of the association rules mining with a domain ontology. In [6], the authors use an ontology to improve the counting support during the association rule mining phase by using a taxonomy. Another interesting approach is presented in [8], where an ontology-based algorithm is employed for discovering rules of product fault causes, in an attempt to discover high-level clearer rules. In this case, the system enables the user only to specify an ideal level of generality of the extracted rules. In addition, our framework also enables the users to specify different levels of abstraction for different items, depending on the specific properties of such items. A concise syntax has been defined to this aim. In our view, the use of an ontology enforces constraints definition, enabling us to use data properties in domain-specific constraints.

6 Conclusions and future works

We proposed an integrated framework for the extraction of constraint-based multi-level association rules with the aid of an ontology. Our system permits the definition of domain-specific constraints by using the ontology for filtering the instances used in the association rule mining process. The main advantages of the proposed framework can be summarized in terms of *extensibility* and *flexibility*.

In our case study, the supermarket domain is modeled only by classes and data properties and it would be very interesting to study: (i) how object properties (and more complex logical relationships) can be employed in our framework; (ii) what aspects they can improve. Other important future works are the possibility of modeling the antecedent and the consequent of an association rule as ontology concepts in order to express constraints on the association rules structure. Furthermore we could improve the system by integrating the constraints evaluation directly in the mining algorithm.

Acknowledgement. This work is supported by *MUSING* project (www.musing.eu/).

References

[1] Agrawal, R., Methta, M., Shafer, J., and Srikant, R.: Fast algorithms for mining association rules in large databases. In Proceedings of the 20th International Conference on Very Large Databases (VLDB '94), Santiago de Chile, Chile, pp. 478-499.

[2] Agrawal, R., Srikant, R., and Swami, A.: Mining association rules between sets of items in large databases. In Proceedings of ACM SIGMOD Conference on Management of Data (SIGMOD '93), San Diego, CA, pp. 207-216.

[3] Baader, F., Calvanese, D.,McGuinness, D., Nardi, D., Patel-Schneider, P., eds.: The Description Logic Handbook. Cambridge University Press (2003).

[4] Broekstra, J., Kampman, A.: SeRQL: An RDF Query and Transformation Language, http://www.cs.vu.nl/ jbroeks/papers/SeRQL.pdf, 2004.

[5] Broekstra, J., Kampman, A., van Harmelen, F.: Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In Ian Horrocks and James Hendler, editors, Proceedings of the first International Semantic Web Conference (ISWC 2002), number 2342 in Lecture Notes in Computer Science, pages 5468, Sardinia, Italy, June 9 12, 2002. Springer Verlag, Heidelberg Germany.

[6] Chen, X., Zhou, X., Scher, R., and Geller, J.: Using an interest Ontology for Improved Support in Rule Mining. In Proceedings of the 5th International Conference of Data Warehousing and Knowledge Discovery (DaWaK 2003), Prague, Czech Republic, pp. 320–329

[7] Han, J., and Fu, Y.: Discovery of multiple-level association rules from large databases. In Proceedings of the 21st International Conference on Very Large Data Bases (VLDB '95) San Francisco, CA, pp. 420–431

[8] Hou, X., Gu, J., Shen, X., and Yan, W.: Application of Data Mining in Fault Diagnosis Based on Ontology. In Proceedings of the 3rd Conference on Information Technology and Applications (ICITA '05), Sydney, Australia, pp. 260–263.

[9] Ng, R., T., Lakshmanan, L., V., S., Han, J., and Pang A.: Exploratory mining and pruning optimizations of constrained associations rules. In Proceedings of the 1998 ACM SIGMOD international conference on Management of data (SIGMOD Seattle, WA, pp. 13–24.

[10] Romei, A., Ruggieri, S., and Turini, F.: KDDML: a middleware language and system for knowledge discovery in databases, Data and Knowledge Engineering, 57(2):179–220, 2006.

[11] Srikant, R., and Agrawal, R.: Mining Generalized Association Rules, In Proceedings of the 21st International Conference on Very Large Data Bases (VLDB '95), San Francisco, CA, pp. 407–419.

[12] Srikant, R., Vu, Q., and Agrawal, R: Mining Association Rules with Item Constraints In Proceedings of the 3rd International Conference of Knowledge Discovery and Data Mining (KDD '97), Newport Beach, CA, pp. 67–73.

[13] Tan, P.N., Steinbach, M., and Kumar, V.: Introduction to Data Mining, Pearson International Edition - Addison Wesley, 2006.

Contexts and ontologies in schema matching

Paolo Bouquet*

Department of Information and Communication Technology, University of Trento
Via Sommarive, 14 – 38050 Trento (Italy)

Abstract. In this paper, we propose a general model of schema matching based on the following ideas: on the one hand, a schema is viewed as a *context* (namely as a partial and approximate representation of the world from an agent's perspective); on the other hand, a schema cannot be assigned any arbitrary interpretation, as the meaning of the expressions used to label nodes (and possibly arcs) may be constrained by shared social conventions or agreements expressed in some *lexical* or *domain ontologies*. Accordingly, the proposed schema matching method can be viewed as an attempt of coordinating intrinsically context-dependent representations by exploiting socially negotiated constraints on the acceptable intepretations of the labels as codified in shared artifacts like lexicons or ontologies.

1 Introduction

In the literature, we find many different approaches to the problem of schema matching, and each of them reflects a theoretical view on what a schema is (a graph, a linguistic structure, a data model, ...). In this paper, we propose a general model based on the following ideas: on the one hand, a schema can be viewed as a context in the sense defined in [**?**] (a partial and approximate representation of the world from an agent's perspective); on the other hand, a schema cannot be assigned any arbitrary interpretation, as the meaning of the expressions used to label nodes (and possibly arcs) may be constrained by shared social conventions or agreements expressed in some lexical or domain ontologies. Accordingly, a schema matching method can be viewed as an attempt of coordinating intrinsically context-dependent representations by exploiting socially negotiated constraints on the acceptable intepretations of the labels as codified in shared artifacts like lexicons or ontologies.

Our claim is that this type of approach may also provide a general view on the relation between contexts and ontologies. The idea is the following: contexts are representations which encode an agent's point of view; to be shared or communicated, these representations need to be linguistically expressed; however, this linguistic representation cannot be arbitrary, otherwise agents would never succeed in cokmmunication; lexical and domain ontologies are the reification

* Part of the material for this paper was developed in collaboration with Stefano Zanobini as part of his PhD work.

of partial and evolving agreements achieved in a linguistic or in other types of communities on the use of terms used for communication (perhaps in limited domains).

The paper is structured as follows. First, we present the intuitions underlying our approach. Then we show how these intuitions are captured in a formal model. Finally we argue that this model can also be used to explan what other approaches to schema matching do.

2 The building blocks

Let us start with a few general definitions.

Definition 1 (Schema). *Let L_{ext} be a set of labels. A* schema $\mathcal{S}$ *is a 4-tuple* $\langle N, E, \text{lab}_N, \text{lab}_E \rangle$*, where* $\langle N, E \rangle$ *is a graph,* $\text{lab}_N : N \rightarrow L_{ext}$ *is a function that associates each node with a label in* L_{ext}*, and* $\text{lab}_E : E \rightarrow \mathcal{L}_{NL}$ *is a function that associates each edge to a label in* $L_{ext} \cup \emptyset$*.*

In this definition, L_{ext} is the language used to externalize a schema outside an agent's mind (e.g. for publishing the schema and sharing it with other agents). In many real situations, it may be a subset of some natural language. For example, in most web directories, the communication language is basically the portion of English which is needed to label the directory categories, and the sign `>` to denote the sub-category relation (e.g. `Music > Baroque > Europe` is a subcategory of `Music > Baroque`)

Now we need to capture the intuition that an agent a may associate a set of objects to a schema element e based on her understanding of the meaning of e, and that different agents may have a different understanding of e. Let a_1 and a_2 be two agents and L_{ext} a suitable communication language. We now introduce the notion of a representation language for an agent, namely the internal (mental?) language which represents agents know about their environment. Let L^i be the representation language of the agent a_i (where $i = 1, 2$), W a set of worlds, and C a set of contexts of use. Intuitively, W is the set of all possible interpretations of L^j, and C represents a collection of distinct contexts of use of expressions belonging to L_{ext} (C is necessary to model the fact that many communication languages, including natural languages, are polysemous, namely the same word may have a different meaning in different contexts of use). We do not make any special assumption on L_{ext} and L^j; the only important requirement is that they are distinct – and possibly different – languages. For the sake of this paper, we will assume that the representation languages are some sort of Description Logic (DL) language (see [1] for an introduction to DL languages).

The connection between schema elements and data happens in two steps: in the first, we take the specification of a schema element e in L_{ext} and build the representation of its meaning in L^j (given a context of use c); in the second, we provide an interpretation function from the resulting expression of L^j into a set of objects in the domain of W.

The first step is formalized by the following *translation function*.

Definition 2 (Translation function). *Let* L^j *be a representation language,* L_{ext} *a communication language and* c *a context in* C*. The* translation function $\mathcal{T}_c^j : L_{ext} \to L^j$ *is the function which associates an expression of* L^j *to an expression of* L_{ext} *when used in* c*.*

We notice that the translation function is indexed with an agent's name, as it reflects the way such an agent assign a (subjective) meaning to the (public) expressions of a communication language. In the following, we will use the notation $\mathcal{T}^j$ to denote the family of functions $\{\mathcal{T}_c^j \mid c \in C\}$.

The second step is formalized by the following *projection function.*

Definition 3 (Projection function). *Let* L^j *be a representation language and* $w \in W$*. The* projection function $\mathcal{P}_w^j : L^j \to 2^w$ *is a function which, for any possible world* w*, associates an extension to any term of* L^j*.*

In the following, we will use the notation $\mathcal{P}^j$ to denote the family of functions $\{\mathcal{P}_w^j \mid w \in W\}$.

To model the fact that agents may have domain knowledge about the concepts which they associates to schema elements, we introduce the notion of an agent's ontology, expressed as a set of axioms in the agent's representation language: $\mathcal{O}^j = \{t_i \sqsubseteq t_k \mid t_i, t_k \in L^j\}$[1].

Finally, an *agent* a_j which uses L_{ext} as a communication language is defined as follows:

Definition 4 (Agent). *An* agent $a_j^{L_{ext}} = \left\langle \mathcal{T}^j, \mathcal{P}^j, L^j, \mathcal{O}^j \right\rangle$ *is a quadruple, where* $\mathcal{T}^j$ *is a family of translation functions,* $\mathcal{P}^j$ *is a family of projection functions,* L^j *is the agent's representation language,* $\mathcal{O}^j$ *is the agent's knowledge, and the following holds:*

$$\forall w \in W \;\; t_i \sqsubseteq t_k \in \mathcal{O}^j \Rightarrow \mathcal{P}_w^j(t_i) \subseteq \mathcal{P}_w^j(t_k)$$

Now we have all the necessary building blocks for defining the formal object of schema matching in this model.

3 Semantic Coordination

The main idea of our model is that in no real world sitation one can guarantee that two agents share meanings just because they share a communication language. Indeed, the notion of shared meaning is not available (meaning is always mediated through concepts, and therefore partially private). Therefore, to model schema matching, we need to introduce a notion of agreement which does not

[1] For the sake of simplicity, we assume that the agent knowledge can be represented as a set of entailment axioms between concepts. Notice that this formalization of the knowledge basis is not a novelty, but it is the standard one used in Description Logics [1]. Following this approach, for expressing the *Is-A* relation between the concepts, we use the DL symbol '$\sqsubseteq$'.

presuppose that shared meanings are available and which we call *semantic coordination*: two agents are semantically coordinated on the use of two expressions t_1 and t_2 of a communication language L_{ext}, with respect to a relation R and in some context of use c, when the interpretation they give to the two linguistic expressions is *compatible*[2]:

Definition 5 (Semantic Coordination). *Let* $a_1^{L_{ext}} = \left\langle \mathcal{T}^1, \mathcal{P}^1, L^1, \mathcal{O}^1 \right\rangle$ *and* $a_2^{L_{ext}} = \left\langle \mathcal{T}^2, \mathcal{P}^2, L^2, \mathcal{O}^2 \right\rangle$ *be two agents,* t_1 *and* t_2 *two expressions of a communication language* L_{ext}*, and* R *be any set–theoretical relation. We say that* a_1 *and* a_2 *are* semantically coordinated *on* t_1 *and* t_2*, with respect to* R *in some context of use* c*, if the following holds:*

$$\forall w \in W \quad \mathcal{P}^1_w(\mathcal{T}^1_c(t_1)) \; R \; \mathcal{P}^2_w(\mathcal{T}^2_c(t_2))$$

Imagine, for example, that R is an equivalence relation and $t_1 = t_2 =$ 'cat'; then the definition above says that two agents are coordinated with respect to the use of the word 'cat' (in English) in a context of use c if, for any world $w \in W$, they associate to it the set of objects belonging to the domain of w (via translation and projection).

In what follows, we will use the notation $coord(a_1, a_2, t_1, t_2, R, c)$ to denote that the agents a_1 and a_2 are semantically coordinated on t_1, t_2 with respect to R in the context c.

We now introduce a syntactic notion of mapping across schemas:

Definition 6 (Mapping). *Let* $\mathcal{S}_1 = \left\langle N_1, E_1, \mathrm{lab}^1_N, \mathrm{lab}^1_E \right\rangle$ *and* $\mathcal{S}_2 = \left\langle N_2, E_2, \mathrm{lab}^2_N, \mathrm{lab}^2_E \right\rangle$ *be two schemas and* $\Re = \{r_1, \ldots, r_n\}$ *be a set of binary relations which may hold between elements of the two schemas. A* mapping $\mathcal{M}_{\mathcal{S}_1^P \rightarrow \mathcal{S}_2^P}$ *is a set of* mapping elements $\langle n_1, n_2, r_i \rangle$*, where* $n_1 \in N_1$*,* $n_2 \in N_2$*, and* $r \in \Re$*.*

We say that a mapping element $m = \langle n_1, n_2, R, q \rangle$ is correct if the two agents a_1 and a_2 are semantically coordinated with respect to n_1, n_2 and R, in a context c:

Definition 7 (Correct Mapping). *Let* $\mathcal{M}_{\mathcal{S}_1 \rightarrow \mathcal{S}_2}$ *be a mapping between the schemas* $\mathcal{S}_1$ *and* $\mathcal{S}_2$*.* $\mathcal{M}_{\mathcal{S}_1 \rightarrow \mathcal{S}_2}$ *is correct if and only if, for any mapping element* $m = \langle n_1, n_2, r \rangle \in \mathcal{M}_{\mathcal{S}_1 \rightarrow \mathcal{S}_2}$*, it holds that:*

$$coord(a_1, a_2, n_1, n_2, r, c)$$

To illustrate the generality of the model, consider the two following cases. In the first, we imagine that a_1 and a_2 are the same agent; in this case, semantic

[2] Compatibility here refers to a precise formal notion which was defined in [15] as part of a logic of contextual reasoning. For lack of space, we will not try event to summarize this notion in any detail. We only stress that compatibility captures the idea of logical constraints holding between two distinct logical languages, and therefore seems especially suitable in this paper, where we imagine that agents have distinct representation languages.

coordination boils down to the translation of elements of different schemas into the same representation language, and to checking whether the relation r holds between the two expressions of the representation language itself. In the second, imagine that a_1 and a_2 are different, but $\mathcal{S}_1$ and $\mathcal{S}_2$ are the same schema; then, a_1 and a_2 might be not coordinated even on the same schema element n, as they might assign a different meaning to n and therefore r might not hold between them.

4 Default rules for semantic coordination

Many theoretical results can be used to prove that semantic coordination (and therefore the correctness of a mapping) can't be directly checked, as it would require knowledge on what any agent really means by a word in a context, and this would be equivalent to look into an agent's mind. However, in this section we show that the condition of semantic coordination can be (and actually is) approximated by three types of *default rules* which are used by agents to "jump to the conclusion" that semantic coordination holds.

4.1 Syntactic default rules

The first type of default rule has to do with the used of a communication language L_{ext} in a community of agents. The idea is the following. Take any two expressions t_1 and t_2 of L_{ext}, and a family R^+ of relations which connect some syntactic features of any two expressions of L_{ext} (e.g. string identity, substring, permutations of strings, and so on). Now suppose that we have a "table" associating the elements of R^+ with a family of set–theoretic relations R. The syntactic default rule (SDR) says that, whenever $r^+ \in R^+$ holds between t_1 and t_2, then $r \in R$ holds between the meaning of t_1 and t_2.

Definition 8 (Syntactic Default Rule). *Let let a_1 and a_2 be two agents, t_1 and t_2 be two expressions of the language L_{ext}, r^+ a relation between expressions of L_{ext}, and r the set-theoretical relation which corresponds to the syntactic relation r^+. Then:*

$$\begin{array}{rl} \textit{if} & t_1\ r^+\ t_2 \quad \textit{in a context } c \\ \textit{then} & \textit{coord}(a_1, a_2, t_1, t_2, r, c) \end{array}$$

As an example, let t_1 be the phrase 'black and white images', t_2 be the word 'images' and r^+ a relation holding two strings when one *contains* the other. Imagine that this relation is associated with set inclusion ($\subseteq$). Then one would be allowed to conjecture that an agent a_1 using the expressions 'black and white images' is semantically coordinated with an agent a_2 using the expression 'images' with respect to set inclusion.

We should recognize that this default rule is extremely powerful and widely used even in human communication, in particular in the special case of a single term (we typically assume that, unless there is any evidence to the contrary, what other people mean by a word t is what we mean by the same word in a given context). The intuitive correctness and the completeness of this default rule essentially depends on the fact that a syntactical relation r^+ be an appropriate representation of a set–theoretical relation r, and vice–versa, that a set–theoretical relation r is appropriately represented by some syntactical relation r^+. But, in general, polysemy does not allow to guarantee the correctned and completeness of this rule even in the trivial case when $t_1 = t_2$; and the existence of synonyms makes it quite hard to ensure completeness.

4.2 Pragmatic default rule

The second type of default rules says that agents tend to induce that they are semantically coordinated with other agents from a very small number of cases in which the agreed with another agent upon the use of a word. For example, from the fact that they agreed upon a few examples of objects called "laptops", they tend to induce a much stronger form of coordination on the meaning of the word "laptop". Formally, this pragmatic default rule can be expressed as follows:

Definition 9 (Pragmatic Default Rule). *Let t_1 and t_2 be two expressions of the language L_{ext}, W a set of worlds, $c \in C$ a context of use and r a set–theoretical relation. Furthermore, let $w_A \in W$ be a finite world. Then:*

$$\begin{array}{rl} \textit{if} & \mathcal{P}^1_{w_A}(\mathcal{T}^1_c(t_1)) \; R \; \mathcal{P}^2_{w_A}(\mathcal{T}^2_c(t_2)) \\ \textit{then} & \textit{coord}(a_1, a_2, t_1, t_2, r, c) \end{array}$$

For exmaple, if $t_1 = t_2 = t$ and R is the equality symbol ('='), then the pragmatic default rule says that the restricted notion of semantic coordination can be inferred when two agents associate the same subset of the current world to t.

Pragmatic default rules are a very strong form of induction from the particular to the universal, and it is well-known that this not a valid pattern of reasoning. Indeed, if the positive examples are taken from a very small domain, then the two agents may happen to induce their coordination on equivalence simply because they never hit a negative example (lack of correctness); or vice versa they may fail to recognize their coordination simply because they could not find any positive example (lack of completeness).

4.3 Conceptual default rule

Finally, we discuss a third type of default rule, which can be stated as follows:

$$\text{if } \mathcal{T}^i_c(t_1) \; r^* \; \mathcal{T}^j_c(t_2), \text{ then } \forall w \in W \;\; \mathcal{P}^i_w(\mathcal{T}^i_c(t_1)) \; r \; \mathcal{P}^j_w(\mathcal{T}^j_c(t_2))$$

where r^* would be any relation between concepts.

However, there are two major problems with this definition:

– first of all, the premise of the rule does not make sense, as we do not know how to check in practice whether $\mathcal{T}_c^i(t_1)\ R^*\ \mathcal{T}_c^j(t_2)$ holds or not, as the two concepts belong to different (and semantically autonomous) representation languages;
– second, it not clear how to determine a relation between concepts. Indeed, the syntactic and pragmatic default rule are based on conditions (relation between strings, and relations between sets of objects) which can be externally verified. But how do we check whether the concept of "cat" is subsumed by the concept of "mammal"?

A possible way out for the second issue may be that agents infer that such a relation holds (or does not hold) from what they know about the two concepts. This, of course, introduces an essential directionality in this rule, as it may be that two agents have different knowledge about the two concepts to be compared. So we need two distinct checks:

$$\begin{array}{rl} \text{if} & \mathcal{O}^i \models \mathcal{T}_c^i(t_1)\ R^*\ \mathcal{T}_c^j(t_2) \\ \text{then} & \forall w \in W\ \ \mathcal{P}_w^i(\mathcal{T}_c^i(t_1))\ R\ \mathcal{P}_w^j(\mathcal{T}_c^j(t_2)) \end{array}$$

and

$$\begin{array}{rl} \text{if} & \mathcal{O}^j \models \mathcal{T}_c^i(t_1)\ R^*\ \mathcal{T}_c^j(t_2) \\ \text{then} & \forall w \in W\ \ \mathcal{P}_w^i(\mathcal{T}_c^i(t_1))\ R\ \mathcal{P}_w^j(\mathcal{T}_c^j(t_2)) \end{array}$$

Suppose we accept this asymmetry. How can we address the first problem?

Senses and Dictionaries Here is where we introduce the notion of socially negotiated meanings. Indeed, most communication languages (for example, natural languages) provide dictionaries which list all accepted *senses* of a word (WordNet [13] is a well-known example of an electronic dictionary). A sense can be viewed as a tentative bridge between syntax and semantics: its goal is to list possible meanings (semantics) of a word (syntax), but this is done by providing definitions which are given in the same language which the dictionary is supposed to define. So, dictionaries have two interesting properties:

– on the one hand, they provide a *publicly accessible* and *socially negotiated* list of acceptable senses for a word;
– however, senses cannot *ipso facto* be equated with a list of shared meanings for the speakers of that language, as senses are (circularly) defined through other words, and do not contain the concept itself.

However, we believe that dictionaries are crucial tools for communication languages, and indeed a linguistic community can be defined as a group of speakers which agree on a common dictionary. Let us show how this idea can be used to define a surrogate of a conceptual default rule.

Definition 10 (Lexical Default Rule). *Let t be an expression of the language L_{ext}, W a set of worlds, and $c \in C$ a context of use. Furthermore, let $a_1^{L_{ext}}$ and $a_1^{L_{ext}}$ be to agents of the same linguistic community. Then:*

$$\begin{array}{rl} \textit{if} & SR_c^1(t) = SR_c^2(t) \\ \textit{then} & coord(a_1, a_2, t, t, \equiv, c) \end{array}$$

where $SR_c^i(t)$ is a function that, given a context c and a term $t \in L_{ext}$, returns a suitable sense for t from a dictionary (notice that this function is again parametric on agents).

Such default rules overcomes the first issue of an the ideal conceptual default rule, i.e. comparing terms from different representation languages. Indeed, it is a verifiable condition whether two agents refer to the same dictionary sense of a word in a given context of use. However, as it is, it can only be used to infer the restricted form of semantic coordination, as it applies only to a single term t of L_{ext}. The general default rule should say that two agents a_1 and a_2 are semantically coordinated with respect to two expressions t_1 and t_2 of the communication language L_{ext}, and with respect to a relation r, when the dictionary senses individuated by the sense retrieval function are r^*–related, where r^* is a relation between senses corresponding to the relation r between concepts. As we said at the beginning of this section, the relation r^* can be determined only with respect to an agent's knowledge about the relation between concepts corresponding to senses. To capture this aspect, we need to make a further assumption, namely that there is a mapping from the concepts is an agent's ontology $\mathcal{O}^j$ and dictionary senses. For example, if a_1's ontology $\mathcal{O}^1$ contains the axiom '$\mathsf{cat} \sqsubseteq \mathsf{animal}$' (where cat and animal are expressions of L^1), then $[\mathcal{O}^1]$ contains the axiom '$s_g \sqsubseteq s_h$', where s_g is the dictionary sense 'feline mammal' associated to the word 'cat' and s_h is the dictionary sense 'a living organism' associated to the word 'animal'[3]. Now, we can introduce an extended lexical default rule.

Definition 11 (Lexical Default Rule Extended). *Let t_1 and t_2 be two expressions of the language L_{ext}, W a set of worlds, $c \in C$ a context of use and r a set–theoretical relation. Furthermore, let a_1 and a_1 be two agents belonging to the same linguistic community. Then:*

$$\begin{array}{rl} \textit{if} & [\mathcal{O}^1] \models SR_c^1(t_1)\ r^*\ SR_c^2(t_2) \\ \textit{then} & coord(a_1, a_2, t_1, t_2, r, c) \quad \textit{w.r.t.}\ [\mathcal{O}^1] \end{array}$$

and

$$\begin{array}{rl} \textit{if} & [\mathcal{O}^2] \models SR_c^1(t_1)\ r^*\ SR_c^2(t_2) \\ \textit{then} & coord(a_2, a_1, t_1, t_2, r, c) \quad \textit{w.r.t.}\ [\mathcal{O}^2] \end{array}$$

[3] The problem of lexicalizing the ontologies with respect to some dictionary is not completely new. In computer science area, a lot of studies are dedicated to this problem. Among them, in our opinion the most relevant approaches are described in [3, 14, 30].

where r^* is the relation between senses corresponding to r with respect to the conceptual level of meaning.

Essentially, this rule says that, if the sets of senses associated to two expressions t_1 and t_2 by the agents a_1 and a_2 are in some relation r^* with respect to the (lexicalized) knowledge of the agent a_i (for $i = 1, 2$), then they are semantically coordinated with respect to t_1, t_2 and r. An example will better clarify the situation. Suppose t_1 is 'images of cats', and t_2 is 'images of animals'. Furthermore, imagine that the senses that a_1 associates to t_1 are $\langle s_q, s_w, s_e\rangle$ ($SR^1_c(t_1) = \langle s_q, s_w, s_e\rangle$), and that the senses that a_2 associates to t_2 are $\langle s_q, s_w, s_r\rangle$ ($SR^2_c(t_2) = \langle s_q, s_w, s_r\rangle$), where s_q = 'a visual representation', s_w = 'concerning', s_e = 'feline mammal' and s_r = 'a living organism'. Imagine now that the (lexicalized) ontology $[\mathcal{O}^1]$ contains the following two axioms: (i) 'for each pair of concepts c, d, if $c \sqsubseteq d$, then 'a visual representation' 'concerning' c is less general than 'a visual representation' 'concerning' d; (ii) 'feline mammal' $\sqsubseteq$ 'a living organism'. In this case, we can deduce that $\langle s_q, s_w, s_e\rangle$ 'less general than' $\langle s_q, s_w, s_e\rangle$, and, by applying the default rule, that the two agents are coordinated with respect to the relation 'less general than' ($\subseteq$). The same considerations can be done if we take into account the agent knowledge $[\mathcal{O}^2]$ of the other agent.

As we announced at the beginning of the section, the extended lexical default rule introduces a form of *directionality* in the notion of semantic coordination, as the computation of the relation r^* between (lexicalized) concepts relies on an agent's knowledge about them, and such knowledge may lack, or be different in two different agents. However, as it was proved in [2], this directionality effect can be weakened. Indeed, the relation computed by a_1 is guarantee to be correct also for a_2, if we can prove that $\mathcal{O}^1 \sqsubseteq \mathcal{O}^2$.

The correctness and completeness of the lexical default essentially depends on the condition that two agents use the same function to associate dictionary senses to concepts in their internal representation and vice versa. Clearly, this condition cannot be guaranteed, as we can always conceive a situation where two agents point to the same dictionary sense s_k for a term t, but then their internal representation of s_k is different. However, this type of rule makes an essential use of socially negotiated tools, which provide a powerful extension to purely syntactic or pragmatic methods.

5 Conclusions

The model we propose leds to two general results. The first is negative, as it says that the condition required to prove that a mapping is correct (even in the weak sense of semantic coordination) can never be formally proved. The second, however, is that all the proposed matching methods can be classified into three broad families, one for each default rule[4]:

[4] Wever, see [25, 27] for a classification of matching methods based on different principles.

Syntactic methods: methods which try to determine a mapping by a purely syntactic analysis of the linguistic expressions occurring in different schemas, namely by comparing the properties of the strings that are used to label nodes, and reasoning on their arrangement into a schema. Examples can be found in [35, 32, 24, 6, 12, 34, 7, 19];

Pragmatic methods: methods which assume that the relation between schema elements can be inferred from the relation between the data associated to them. Examples can be found in [33, 8, 31, 7, 9, 29, 11, 5, 10, 20, 28];

Conceptual methods: methods which try to compute a mapping by comparing the (lexical representation of the) concepts associated by the schema creators to the schema elements. Examples can be found in [26, 4, 16].

The three types of methods have their own pros and cons. For example, *syntactic methods* are highly effective, as they exploit and reuse very efficient techniques from graph matching; however, their meaningfulness is quite low, as they take into account a very superficial level of meaning which disregards potential ambiguities and cannot capture semantic relations between concepts which are not reflected in the pure syntax (for example, the relation between "cats" and "mammals"). *Pragmatic methods* are extremely meaningful (if two schemas were used to classify the same data set, and if the data set was representative of the domain of discourse, then the outcome of pragmatic methods would be always correct – independently from the analysis of labels and from the arrangement of nodes); however, the necessary preconditions (same data set and appropriate coverage) are extremely hard to match, and in practice we can never know when they are matched (we can only rely on statistical methods). Finally, *conceptual methods* – like pragmatic methods – have the advantage of being independent from the syntactical structure of the schemas, as in principle they can find the correct relation between two schema elements even if labels are (syntactically) very different and nodes are arranged in different orders (when such an order is inessential from a semantic point of view). In addition, like syntactic methods, they have the advantage of being independent from the data contained into the schema elements. However, conceptual methods may fail in two crucial steps. First of all, the function which returns a sense for a word in a context of use is quite complex, and inherits most well-known issues related to word sense disambiguation in NLP. Second, an agent might lack part of the relevant knowledge to compute a mapping between two concepts. However, we should add that these methods are the only ones which are semantically incremental: one can always know why a mapping was not found (or why a wrong match was computed) and fix the problem is a general way (and not, for example, by tuning some parameters, which may have bad effects on the performance on different schemas).

Probaby to overcome some of these limitations, most actual methods are indeed hybrid, as they use techniques which are based on more than one default rule (for example, syntactic methods use lexical information from thesauri, and some conceptual methods use string matching techniques for improving the quality of their results).

References

1. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors. *The Description Logic Handbook. Theory, Implementation and Applications.* Cambridge University Press, January 2003.
2. M. Benerecetti, P. Bouquet, and S. Zanobini. Soundness of schema matching methods. In L. Torenvliet A. Gòmez-Pèrez, J. Euzenat, editor, *second European Semantic Web Conference (ESWC 2005)*, volume 3532 of *LNCS*, Heraklion, Crete, Greece, May 29–June 1, 2005 2005. Springer.
3. S. Bergamaschi, S. Castano, and M. Vincini. Semantic integration of semistructured and structured data sources. *SIGMOD Record*, 28(1):54–59, 1999.
4. P. Bouquet, L. Serafini, and S. Zanobini. Semantic coordination: a new approach and an application. In D. Fensel, K. P. Sycara, and J. Mylopoulos, editors, *The Semantic Web – 2nd international semantic web conference (ISWC 2003)*, volume 2870 of *Lecture Notes in Computer Science (LNCS)*, pages 130–145, Sanibel Island, Fla., USA, 20-23 October 2003. Springer Verlag.
5. J. Broekstra, M. Ehrig, P. Haase, F. van Harmelen, M. Menken, P. Mika, B. Schnizler, and R. Siebes. Bibster - a semantics-based bibliographic peer-to-peer system. In *In Proceedings of the SemPGrid 04 Workshop*, New York, USA, May 2004.
6. J. Carroll and HP. Matching rdf graphs. In *Proc. in the first International Semantic Web Conference - ISWC 2002*, pages 5–15, 2002.
7. A. Doan, P. Domingos, and A. Y. Halevy. Reconciling schemas of disparate data sources: A machine-learning approach. In *SIGMOD Conference*, 2001.
8. A. Doan, P. Domingos, and A. Y. Levy. Learning source description for data integration. In *WebDB (Informal Proceedings)*, pages 81–86, 2000.
9. A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Learning to map between ontologies on the semantic web. In *Proceedings of WWW-2002, 11th International WWW Conference, Hawaii*, 2002.
10. M. Ehrig and S. Staab. Qom - quick ontology mapping. In *In Proceedings of the 3rd ISWC*, Hiroshima (JP), November 2004.
11. J. Euzenat. Brief overview of t-tree: the tropes taxonomy building tool. In *In Proceedings of the 4th ASIS SIG/CR workshop on classification research*, pages 69–87, Columbus (OH), USA, 1994.
12. J. Euzenat and P. Valtchev. An integrativive proximity measure for ontology alignment. Proceedings of the ISWC2003 workshop on *Semantic Integration*, Sanibel Island (Florida, USA), October 2003.
13. Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database.* The MIT Press, Cambridge, US, 1998.
14. A. Gangemi. *Comparative and Functional Genomics*, volume 4, Some tools and methodologies for domain ontology building, pages 104–110. John Wiley & Sons, Ltd., 2003.
15. C. Ghidini and F. Giunchiglia. Local Models Semantics, or Contextual Reasoning = Locality + Compatibility. *Artificial Intelligence*, 127(2):221–259, April 2001.
16. F. Giunchiglia and P. Shvaiko. Semantic matching. *The Knowledge Engineering Review Journal*, 18(3):265–280, 2003.
17. P. Hitzler, J. Euzenat, M. Krotzsch, L. Serafini, H. Stuckenschmidt, H. Wache, and A. Zimmermann. D2.2.5 – Integrated view and comparison of alignment semantics. Deliverable of the EU funded network of excellence *KnowlegdeWeb*, 2005.
18. P. Bouquet, L. Serafini, and S. Zanobini. Bootstraping semantics on the web: meaning elicitation from schemas. Proceedings of *WWW2006*, Edinburgh, Scotland, 22nd – 26th May 2006.

19. W. Hu, N. Jian, Y. Qu, and Y. Wang. Gmo: A graph matching for ontologies. In *Integrating ontologies Workshop Proceedings, K-CAP 2005*, pages 41–48, 2005.
20. Ryutaro Ichisem, Hiedeaki Takeda, and Shinichi Honiden. Integrating multiple internet directories by instance–base learning. In *AI AND DATA INTEGRATION*, pages 22–28, 2003.
21. N. Jian, W. Hu, G. Chen, and Y. Qu. Falcon-ao: Aligning ontologies with Falcon. In *Integrating ontologies Workshop Proceedings, K-CAP 2005*, pages 85–91, 2005.
22. J. Madhavan, P. A. Bernstein, and E. Rahm. Generic schema matching with CUPID. In *The VLDB Journal*, pages 49–58, 2001.
23. T. Milo and S. Zohar. Using schema matching to simplify heterogeneous data translation. In *Proc. 24th Int. Conf. Very Large Data Bases, VLDB*, pages 122–133, 24–27 1998.
24. M. Pelillo, K. Siddiqi, and S. W. Zucker. Matching hierarchical structures using association graphs. *Lecture Notes in Computer Science*, 1407, 1998.
25. E. Rahm and P.A. Bernstein. A survey of approaches to automatic schema matching. *VLDB Journal*, 10(4), 2001.
26. L. Serafini, S. Zanobini, S. Sceffer, and P. Bouquet. Matching hierarchical classifications with attributes. In *European Semantic Web Conference (ESWC2006)*, 2006.
27. P. Shvaiko. A classification of schema-based matching approaches. Technical Report DIT-04-093, Dipartimento di Informatica e Telecomunicazioni, University of Trento, 2004.
28. U. Straccia and R. Troncy. OMAP: Results of the ontology alignment context. In *Integrating ontologies Workshop Proceedings, K-CAP 2005*, pages 92–96, 2005.
29. Gerd Stumme and Alexander Maedche. FCA-merge: bottom-up merging of ontologies. In *In Proceedings of 17th IJCAI*, pages 225–230, Seattle (WA), USA, 2001.
30. W. R. van Hage, S. Katrenko, and G. Schreiber. A method to combine linguistic ontology-mapping techniques. In Y. Gil, E. Motta, V. R. Benjamins, and M. A. Musen, editors, *The Semantic Wec – ISWC 2005*, volume 3729 of *LNCS*. Springer, 2005.
31. C. Clifton W. Li. Semint: A tool for identifying attribute correspondences in heterogeneous databases using neural networks. In *Data & Knowledge Engineering*, volume 33(1), pages 49–84, 2000.
32. J. T. Wang, K. Zhang, K. Jeong, and D. Shasha. A system for approximate tree matching. *Knowledge and Data Engineering*, 6(4):559–571, 1994.
33. Q. Y. Wang, J. X. Yu, and K. Wong. Approximate graph schema extraction for semi-structured data. In *Proceedings of the 7th International Conference on Extending Database Technology*, pages 302–316. Springer-Verlag, 2000.
34. L. Xu and D. W. Embley. Using domain ontologies to discover direct and indirect matches for schema elements. In *Proceedings of the workshop on Semantic Integration*, Sanibel Island (Florida, USA), October 2003.
35. K. Zhang, J. T. L. Wang, and D. Shasha. On the editing distance between undirected acyclic graphs and related problems. In Z. Galil and E. Ukkonen, editors, *Proceedings of the 6th Annual Symposium on Combinatorial Pattern Matching*, volume 937, pages 395–407, Espoo, Finland, 1995. Springer Verlag.

Textual Inference Logic: Take Two

V. de Paiva, D. G. Bobrow, C. Condoravdi, R. Crouch, L. Karttunen,
T. H. King, R. Nairn, and A. Zaenen

Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto CA 94304*

Abstract. This note describes a logical system based on concepts and contexts, whose aim is to serve as a representation language for meanings of natural language sentences. The logic is a theoretical description of the output of an evolving implemented system, the system Bridge, which we are developing at PARC, as part of the AQUAINT program. The note concentrates on the results of an experiment which changed the underlying ontology of the representation language from CYC to a version of WordNet/VerbNet.

1 Introduction

This note describes a second version of a logical system based on concepts and contexts, whose aim is to serve as a representation language for meanings of natural language sentences. This representation language is constrained in two different directions: on the one hand we want the *mapping* from English to this language to be as easy as possible, hence we want a very expressive logical language. On the other hand we want to do *reasoning* with this language, so we want to constrain its complexity as much as possible.

The first version of this logic of contexts and concepts was called TIL (for Textual Inference Logic) and was described in [1]. The logical system in this paper, which we call TIL2 (for Textual Inference Logic Two), the formalization of the implemented system Bridge that we are developing at PARC as part of the AQUAINT framework, shares with TIL its main characteristics: it is a logical system of concepts and contexts, where declaration of instantiability of an instance of a concept in a context specifies the truth of assertions concerning that concept in that context. Uninstantiability is the negation of instantiability. Some higher level discussion of the rationale behind the systems can be found in [6, 2, 4].

The main difference between TIL and TIL2 is the change of the underlying ontology from the CYC one to an in-house version of a merge of WordNet/VerbNet. Here we concentrate on the results of this experiment in the change of ontology. We also discuss briefly the problem of evaluating the quality of the representations produced by the system Bridge.

* This work was sponsored in part by DTO. Approved for Public Release; distribution unlimited.

For those interested, the system Bridge parses sentences in English using our industrial-strength parser XLE and a hand-crafted lexical functional (LFG) grammar[11]. Parsed sentences are mapped to f-structures and f-structures are then mapped to linguistic semantic structures. These are mapped to AKR (Abstract Knowledge Representation) structures using a robust rewriting system [6]. This layered approach to producing logic from text is useful and natural. We have discussed elsewhere the (perhaps less natural) main characteristics of our approach: the 'packing' of all these structures. By 'packing' we mean that instead of disambiguating structures (grammatical ones, semantic ones and knowledge representation ones) and pruning the less likely ones at each stage of the pipeline, our algorithms allow us to keep a condensed representation of all possibilities, effectively avoiding premature pruning of the correct choices.

Given that our logical AKR representations are intimately connected to the underlying ontology, one might expect that the change of ontology from CYC to WordNet/VerbNet would necessitate a total reworking of the system Bridge. This turned out not to be case, the re-architecture of the system was surprisingly easy and almost trouble-free. It is true that new trade-offs were made and these are some of the issues that we discuss here. But before discussing trade-offs we should explain *why* this change of ontology and *what* are the representations obtained in TIL2.

Our aim is to map free text to logical formulae based on a conceptual hierarchy that one can reason with. Our initial intuition was to try to use the concepts provided by the biggest knowledge base available CYC, and to take advantage of its reasoning component, which was familiar to some of us. Although we used CYC concepts for our first logic, we found it useful to map the text to an *abstract* form of knowledge representation (AKR), that could be realized as CYC or KM or any other knowledge representation formalism. The design of this AKR aimed for a sweet spot between ease of mapping from text to a formalism, and mapping from that formalism to standard logical representations. A happy surprise was our realization that the AKR representations were already good enough for some important classes of textual inferences that we wanted to concentrate on. In general, the inferences we wanted to concentrate on were immediate, almost simple-minded, but necessary for the understanding of the text. For example, if the text says that "John managed to close the door" then we can safely infer that "John closed the door" and this kind of immediate inference is absolutely necessary to answer questions, based on snippets of text, as is the case in our primary application. Furthermore, these inferences did not seem to depend crucially on the particular ontology; they were much more dependent on the articulation of inference patterns surrounding the use of particular classes of words which appear quite often in open texts.

At the same time, we were having serious difficulties completing mappings from open texts when using the CYC system. CYC's mappings from word to CYC concepts are very sparse, as might be expected from a knowledge base not built to model language. We realized that having good information, very deep, about some concepts and nothing at all about others was worse than having superficial information about most words. Thus we decided to move to a WordNet/VerbNet ontology, or more precisely, to the projection of WordNet/VerbNet obtained from our own Unified Lexicon[5].We considered trying to extend the mappings from WordNet to CYC; however, we found that concepts implicit in WordNet covered a broader range than those in CYC, and we found no automatic way of extending the mappings from WordNet senses to CYC concepts.

As in the previous version of the system the logic is based on the notion of *events* expressed in a neo-Davidsonian style [10]. We use the neo-Davidsonian notation because it supports easy handling of optional/missing arguments. We couple this with use of *contexts* based on McCarthy's ideas [8]. McCarthy's contexts have two properties that we cash out in our system. The first is that within a context, reasoning can be done locally. So for example, if *Ed leaves Berlin.* then whether this is in a hypothetical/counterfactual, or real-world context, in that context one can conclude that Ed was in Berlin. The second property of McCarthy's contexts is the existence of context-lifting rules that relate statements in one context to ones in nested contexts. We show how linguistic structures provide a framework for different classes of such context-lifting rules.

We first describe the logical system, using several examples. Rather than listing all kinds of relations between contexts and concepts that the implemented system produces, we aim to give a feel for how our representations look like. Then we discuss the changes, gains and losses, caused by the change of ontology from CYC to WordNet/VerbNet. Finally we discuss methods and criteria for evaluating the coverage of the logical system obtained. We close with some ideas for further work.

2 TIL2 via examples

It is traditional for logics of Knowledge Representation to be fragments of first-order logic (FOL). It is traditional for logics for natural language semantics to be higher-order intensional logics. Our logic has concepts, which make it look like "description logics", that is, fragments of FOL, but it also has contexts, a possible-worlds-like construct that, we hope, is expressive enough for the needs of natural language.

Concepts, the way we conceive them, come from both neo-Davidsonian event semantics and, somewhat independently, from description logics. Some of our reasons for using a concept denoting analysis instead of an individual denoting analysis when mapping noun phrases to logic are discussed in [3]. The main reasons are being able to deal with non-existent entities (for example when mapping *"Negotiations prevented a strike"* we do not want to say that there exists negotiations N and there exists a strike S and $prevented(N, S)$, as the prevented strike does not really exist in the actual world).

One of the main differences between TIL and TIL2 consists in the type of concepts that are used. While in the previous logic TIL the basic ontology was the CYC ontology, for TIL2 the basic ontology is WordNet/VerbNet. But whatever the basic ontology, concepts in our logic are of two very different kinds: the first kind of concepts are given a priori, sitting in a established hierarchy, based on the hierarchy underlying either CYC or the synsets of WordNet, considered as a taxonomy. The second kind of concepts are dynamic, created by the implemented system Bridge when we feed it an English sentence. The dynamic concepts are created and placed in the hierarchy in use, as best as we can, at run time.

For example, when using the CYC ontology, for the sentence *A zebra slept*, we use two CYC concepts `Zebra` and `Sleeping` and two dynamic concepts $zebra : 1$, a subconcept of the CYC concept `Zebra` and $sleep : 11$, a subconcept of the CYC concept `Sleeping`. Now when the same sentence is analyzed in the WordNet/VerbNet version of the system, the dynamic concept $zebra : 1$ will be mapped to a subconcept of the WordNet synset corresponding to the zebra animal, but the dynamic concept corresponding to the zebra's sleeping, $sleep : 11$ will be mapped to **two** different static concepts in WordNet, one corresponding to the WordNet meaning of animal sleep, the other corresponding to "accommodate", as in the sentence *The tent sleeps six*.

The concepts in WordNet are treated by Bridge, following WordNet convention, using the synset numbers. These are not very easy to read, hence the system pretty-prints it as a head word of the synset, followed by a number. The dynamic concepts are written as the word colon a number, showing that this is simply a Skolem constant. For example a clause like $subconcept(sleep : 11, [sleep - 1, sleep - 2])$ means that the dynamic subconcept of the zebra sleep ($sleep : 11$) is either a subconcept of $sleep - 1$ or of $sleep - 2$.

The most underspecified concept in the WordNet hierarchy is **entity**, which corresponds to the concept `Thing` in CYC. All our concepts are subconcepts of the most underspecified concept. We assume that that there are no circularities nor inconsistencies[1] in the given initial hierarchy, be that CYC or WordNet.

The second main difference between TIL and TIL2 has to do with how the concepts are related, when expressing propositions. In both systems concepts are related via "role" assertions, but the kinds of roles available are different. Thus continuing on with the same example *"The zebra slept"* when using the CYC ontology, we were able to use the CYC role `bodilyDoer` to connect the sleeping event concept to the zebra concept, so the representation ends up with the two subconcept clauses plus a clause for $role(bodilyDoer, sleep : 11, zebra : 1)$, while the representation using the WordNet/VerbNet ontology is very similar, but has instead $role(Agent, sleep : 11, zebra : 1)$ The CYC role `bodilyDoer` is much more specific than the role `Agent` from the much more limited collection of VerbNet roles. Our unified lexicon ([5]) provides a mapping from the grammatical relations produced by our XLE/LFG parser to the concept and role structure based on the information in VerbNet. While many of our roles resemble linguistic "thematic roles", the view here is more general and we have many roles that do not correspond to thematic roles, see below. Roles are written as ternary relations, in a prefix notation, i.e. $role(t_1, t_2, t_3)$ where t_1 is the name of the role and t_2 and t_3 are the concepts in the binary relation named by t_1. Thus the intuitive meaning of $role(Agent, sleep : 11, zebra : 1)$ is that in a particular

[1] This is a big assumption, but we hope others are working on the problem.

context there is an sleeping event (a generic sub-concept of the sleeping concept) and there is a zebra (a generic sub-concept of the concept of zebra) such that the relation *Agent* relates this zebra and this sleeping event.

The logical system described so far looks like a description logic. We have concepts Concept with their own partial order (written as $subconcept(t_1, t_2)$) and roles Role, which are binary relations on the set of concepts Concept. We write *clauses* that either relate concepts via subconcept relations or relate roles to pairs of concepts, like $role(Agent, sleep : 11, zebra : 1)$. And we write collections of clauses that correspond to representations of natural language sentences and hence correspond to propositions.

But our simple logic has contexts Context, as well as concepts. There is a first initial context (written as t) that corresponds roughly to what we take the world to be like, as far the author of the sentence is committed to. But since this circumlocution is awkward, we will usually talk about this top level context as the 'true context'.

Contexts in our logic were conceived as syntactic ways of dealing with intensional phenomena, including negation and non-existent entities. They support making existential statements about the existence and non-existence in specified possible worlds of entities that satisfy the intensional descriptions specified by our concepts. The possible worlds reflect the worlds implicitly (partially) described by the author of a text. Authors statements of propositional attitudes clearly require use of intensional terms, since no existence in the real world can be implied by such descriptions. It is clear that intensional notions are required when dealing with the representation in logic of *propositional attitudes*. We use propositional attitudes as an example of our use of contexts.

Propositional attitudes predicates relate contexts and concepts in our logic. Thus a concept like 'knowing' or 'believing' or 'denying' introduces a context that represents the proposition that is known, believed or denied. For example, if we want to represent the sentence *Ed denied that the diplomat arrived*, we will need concepts for the arriving event, for the denying event, for the diplomat and for Ed. And we will need roles that describe how these concepts relate to each other. Thus we need to say who did the 'denying' and 'what was denied' and who did the arriving. The content of what was denied in the denying event is the proposition corresponding to *The diplomat arrived*. The role corresponding to 'what was denied' relates a dynamic concept, the concept of the denying event (written as $deny : 4$), to (the contents of) a new context. To name this new context we use its 'context head'. The context head is the arriving event, so the new context is called $context(ctx(arrive : 4))$ ('contex-head' is one of the many roles in the system that is not a thematic role).

Contexts allow us to localize reasoning: the existence of the denying event and of Ed are supposed to happen in the true world, but the existence of the arrival of the diplomat is only supposed to happen in the world of the things denied by Ed. In particular the arrival event could be considered as not happening, if Ed is known as a reliable source. (The system takes no position as to the instantiability or not of the arrival event in the top context: the instantiability of the arriving is only stated in the context of the things denied by Ed.) In some cases (for example if the sentence was *Ed knew that the diplomat arrived*) we can percolate up the truth of assertions in inner contexts up to the outside context. In many cases we cannot. The happening or not of events is dealt with by the *instantiability/uninstantiability* predicate that relates concepts and contexts.

While we may be prepared to make the simplifying assumption that if 'X is known' than 'X is true', we certainly do not want to make the assumption that if 'X is said' than 'X is true'. We say that the context introduced by a knowing event is *veridical* with respect to the initial context t, while the context introduced by a saying event is *averidical* with respect to the initial context. Negation introduces a context that is *anti-veridical* with respect to the original context. Thus we have a fairly general mechanism of contexts (these can clearly be iterated), which can represent some positive and some negative information. Similarly to McCarthy's logic we also have 'context lifting rules' that allow us to transfer veridicality statements between contexts, in a recursive way.

Our representations also have a (preliminary) layer of temporal representation on them. The idea is to order events according to their times of happening and with respect to some generic time 'Now'.

A few words on related work: Clearly our goals and motivations are very similar to the SNePS project[12]. We share the use of intensional notions and of contexts, with a logic approach that strives for the right amount of expressivity. But the differences are overt: we do not feel the need for belief revision. We deal with snapshots of the author's world, not with systems of beliefs. Our basic logic system is constructive, not relevant and paraconsistent. While both logics have been (and are being) designed to support natural language processing and commonsense reasoning, they are implemented very differently.

3 Changing the Ontology

Changing the ontology allowed us to talk about ambiguity-enabled or packed representations. While we could, in principle, do the same with the CYC ontology and we did so, to a limited extent, in practice we simply didn't have the different concepts for each word. For many words we did not have a single concept associated to it, for very few we did have more than one. So we were restricted to what the ontologists in CYC thought the meaning of a given word was. (Of course we are now constrained to the meanings that the lexicographers at WordNet think one should have, but the pool is much bigger. So we do not have the problem of "missing concept for skolem", by and large). Thus a sentence like "Ed arrived at the bank" will not be assigned simply one of possible meanings of "bank" (river bank or financial institution). Actually it will map to any of the ten possible meanings of bank in WordNet. Also "arrive" will be mapped to two different meanings, the physical reaching of a destination and the somewhat metaphoric, succeed in a big way. But instead of having twenty different representations for the meaning of the sentence, sharing the concepts 'Ed', 'arrive', 'bank' and the

VerbNet roles for 'arrive', we have a single representation packing all of this as

$$subconcept(arrive : 4, [arrive - 1, arrive - 2])$$
$$role(Experiencer, arrive : 4, Ed : 1)$$
$$role(Cause, arrive : 4, bank : 15)$$
$$subconcept(Ed : 1, [male - 2]])$$
$$subconcept(bank : 15, [bank - 1, \ldots bank - 10])$$

One bad side of this is that we are forced (to begin with, at least) to use the very uninformative VerbNet roles. Thus in the example above we end up with one sensibly named role $role(Experiencer, arrive : 4, Ed : 1)$ and one not so sensibly named $role(Cause, arrive : 4, bank : 15)$. We have discussed ways of augmenting the number of roles of VerbNet (from less than twenty) to a reasonable number, presumably much less than the 400 that CYC has, but have found that a daunting task, so we are still exploring possibilities. Roles in our system are supposed to support inference and at the same time are supposed to make the mapping from language feasible. For the latter purpose (mapping from language feasibly) VerbNet roles are well-suited, but they are too underspecified to help with inference. The quest is on to find a collection of roles that keeps feasibility of the mapping, but improves the inferential capabilities.

While the mechanism that implements the packing of representations could be used with the CYC ontology too, the actual details of the previous implementation, which looked at noun concepts before verb concepts (given CYC's more extensive coverage of nouns) made packed representations the exception rather than than the norm. In any case packing makes more sense when using WordNet/VerbNet where we do have many concepts for each word.

Another feature of our use of the new ontology is that it does not enforce "sortal restrictions". Using CYC we could make sure that in the sentence *Ed fired the boy* the verb 'fire' was used with the meaning of what CYC calls `DischargeWithPrejudice`, while in *Ed fired the cannon* it was used with a `ShootingAGun` meaning. With the new ontology we do not weed out even the worst clashes of meanings. But a single representation covers a multitude of meanings. We take this as a shortcoming that we plan to address in future work.

4 Inferences and Design Decisions

The reason for introducing event concepts was the fact that they make some inferences that can be complicated in other semantical traditions very easy. For example it is obvious how to obtain *Ed arrived in the city* from the sentence *Ed arrived in the city by bus.* This inference corresponds simply to conjunction dropping in our logic. But of course there is much more to textual inference than simply dropping conjuncts.

To test textual inference our system provides a method for detecting entailment/contradictions, called "qa" for the application in question answering. When given two passages "qa" tells us whether the second passage is entailed by the first one or not. Simple subconcept/superconcept reasoning is handled.

In addition we support some pre and post condition reasoning. So *Ed arrived in the city* does entail that *A person arrived in the city*, since Ed is a person. Similarly *Ed arrived in Rome* should entail that *Ed arrived in a city*, as 'Rome' is a city, but given that the proper names in WordNet are somewhat sketchy, we do not use this facility.

Note that the clauses we construct satisfy the usual monotonicity patterns, both in positive and in negative form. Thus *Ed arrived in the city by bus* entails that *Ed arrived in the city.* But *Ed did not arrive in the city* entails that *Ed did not arrive in the city by bus*, while *Ed did not arrive in the city by bus* does **not** entail that *Ed did not arrive in the city.*

We have also implemented the transformation of nominal deverbals with their respective arguments into verb-argument structures. The work is described in [7]. It allows us to conclude from a sentence like *Alexander's destruction of the city happened in 332 B.C.* that the sentence *Alexander destroyed the city in 332 B.C.* follows.

We have done significant work on exploring how certain linguistic expressions support classes of context-lifting rules. Using the context structure of our logic we support inferences associated with kinds of verbs with implicative behavior. In our unified lexicon, the classes of such behavior are marked. This work is discussed in [9]. Here we simply give an example of each one of the classes of "implication signatures" or implicative behavior described by Nairn, Condoravdi and Karttunen. There are nine such classes, depending on whether positive environments are taken to positive or negative ones. Thus for example the verb "manage" takes positive predicates (e.g "Ed managed to close the door" → "Ed closed the door") to positive predicates and negative ones ("Ed didn't manage to close the door" → "Ed didn't close the door"). By contrast the verb "forget (to)" inverts the polarities: "Ed forgot to close the door" → "Ed didn't close the door" and "Ed didn't forget to close the door" → "Ed closed the door".

More complicated are the verbs that only show their implicative behavior either in positive or negative situations. For example we have positive implicatives like the verb "force (to)" takes positive polarities and produces positive polarities (e.g "Ed forced Mary to paint" → "Mary painted"), but if "Ed didn't force Mary to paint" we cannot tell whether Mary painted or not. While "refuse (to)" only works to produce negative polarity (e.g. 'Mary refused to sing" → "Mary did not sing"). There are also negative implicatives like "attempt (to)" and "hesitate (to)" which again only work for a negative polarity, but produce a positive one ("Ed didn't hesitate to leave" → "Ed left", but if "Ed hesitated to leave" we cannot tell whether he left or not).

Finally we have factives and counterfactives, examples being "forget (that)" ("Ed forgot that Mary left" → "Mary left" and "Ed didn't forget that Mary left" → "Mary left" and "pretend that" ("Ed pretended that Mary left" → "Mary didn't leave" and "Ed didn't pretend that Mary left" → "Mary left"). And the neutral class, where we cannot say anything about the veridicity of the complement ("Ed said/expected that Mary left"). Further work is in progress to mark implicative behavior of verbs that do not take sentential complements.

One of our difficult design decisions was over the treatment of copula. It was clear that one needed to have trivial inferences like "Ed is a clown" contradicts "Ed is not a clown". But the mechanism used to infer that should be also able to cope with answering yes to "Ed, the clown, slept" implies that "A clown slept" and several other similar and not so similar inferences.

5 Towards Evaluation

From the beginning we faced the problem of measuring the 'quality' of our representations. One can try to measure that by manually inspecting the representations themselves and checking that the arguments provided by the system correspond to our intuitions. But this is not very efficient nor objective. We can also try to measure the faithfulness of the representations by checking whether the system can answer correctly questions, using these representations. We have tried this indirect method in the AQUAINT pilot KB Evaluation and the measuring is quite difficult because question/answer pairs usually have to deal with several logic-linguistic issues at once. We devised pairs of question/answers that try to focus on a particular specific problem at a time. Thus we have test-suites checking mostly deverbal nouns, or anaphora resolution or coordination of sentences, etc. But besides being time consuming and laborious, it is not clear that this would measure adequacy or faithfulness of the representations in a fair way. At the moment, it seems to us that the best that can be done is to try to look at textual entailment, as originally proposed by the PASCAL RTE but modify it to deal with the issues that we consider important, like the implicative behavior of lexical items and especially the need to distinguish between entailment of a negation from not being able to draw a conclusion [13].

6 Conclusion

This note only starts the discussion of the kinds of inferences that we expect to be able to make using our logic of concepts and contexts. On the positive side we have an implemented system Bridge that it is easy to modify as it relies on a heavy duty rewriting system (the transfer system[6]) capable of packing efficiently large amounts of representations, be they f-structures or AKR-structures. This system proved to be robust enough to cope with a very radical change of ontologies. Moreover, the abstract description of the system needed almost no modification.

On the negative side, much work remains to be done to get the system working as well as we want it to. First we still have a long way to go as far as improving the representations is concerned. Amongst the issues we have not discussed here are how to deal with noun-noun compounds, how to deal with contexts introduced by adjectives and adverbs and how to deal with temporal modifiers and temporal interpretation in general. We have done some work on these problems and hope to describe that work elsewhere.

We have said nothing about how to deal with lexical entailments such as *Ed snored* implies that *Ed slept.* We are not sure whether this problem should be addressed by creating enriched representations (maybe the concept of 'snoring' must include a "concurrent" necessary condition of 'sleeping') , or whether such inferences should be handled in the entailment/contradiction algorithm. So we are back to the previous trade-off between easiness of mapping and easiness of reasoning.

References

1. D. Bobrow, C. Condoravdi, R. Crouch, R. Kaplan, L. Karttunen, T. H. King, V. de Paiva, and A. Zaenen. A basic logic for textual inference. In *Proceedings of the AAAI Workshop on Inference for Textual Question Answering, Pittsburg, PA*, 2005.
2. C. Condoravdi, D. Crouch, R. Stolle, V. de Paiva, and D. Bobrow. Entailment, intensionality and text understanding. In *Proceedings Human Language Technology Conference, Workshop on Text Meaning, Edmonton, Canada*, 2003.
3. C. Condoravdi, R. Crouch, M. van der Berg, , J. Everett, R. Stolle, V. de Paiva, and D. Bobrow. Preventing existence. In *Proceedings of the Conference on Formal Ontologies in Information Systems (FOIS). Ogunquit, Maine*, 2001.
4. D. Crouch, C. Condoravdi, R. Stolle, T. King, V. de Paiva, J. O.Everett, and D. Bobrow. Scalability of redundancy detection in focused document collections. In *Proceedings First International Workshop on Scalable Natural Language Understanding (ScaNaLU-2002), Heidelberg, Germany*, 2002.
5. R. Crouch and T. H. King. Unifying lexical resources. In *Proceedings of the Interdisciplinary Workshop on the Identification and Representation of Verb Features and Verb Classes, Saarbruecken, Germany.*, 2005.
6. Richard Crouch. Packed rewriting for mapping semantics to KR. In *Proceedings Sixth International Workshop on Computational Semantics, Tilburg, The Netherlands*, 2005.
7. O. Gurevich, R. Crouch, T. H. King, and V. de Paiva. Deverbals in knowledge representation. In *Proceedings of FLAIRS'06, Melbourne Beach, FA*, 2005.
8. J. McCarthy. Notes on formalizing context. In *Proceedings of the 13th Joint Conference on Artificial Intelligence (IJCAI-93)*, pages 555–560, 1993.
9. R. Nairn, C. Condoravdi, and L. Karttunen. Computing relative polarity for textual inference. In *Proceedings of ICoS-5 (Inference in Computational Semantics). April 20-21, 2006. Buxton, UK*, 2006.
10. T. Parsons. Underlying eventualities and narrative progression. In *Linguistics and Philosophy, 681-99.*, 2002.
11. S. Riezler, T. H. King, R. Kaplan, R. Crouch, J. T. Maxwell, and M. Johnson. Parsing the wall street journal using a lexical-functional grammar and discriminative estimation techniques. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02), Philadelphia, PA*, 2002.
12. S. C. Shapiro. SNePS: A logic for natural language understanding and commonsense reasoning. In *Natural Language Processing and Knowledge Representation: Language for Knowledge and Knowledge for Language, L. Iwanska, S. Shapiro (editors), AAAI Press/MIT Press*, 2000.
13. A. Zaenen, L. Karttunen, and R. Crouch. Local textual inference: can it be defined or cicumscribed? In *Proceedings of the AC 2005 Workshop on Empirical Modelling of Semantic Equivalence and Entailment, Ann Arbor, MI*, 2005.

The User Model and Context Ontology GUMO revisited for future Web 2.0 Extensions

Dominik Heckmann, Eric Schwarzkopf, Junichiro Mori, Dietmar Dengler, Alexander Kröner

German Research Center for Artificial Intelligence
D-66041 Saarbrücken, Germany
heckmann@dfki.de

Abstract. We revisit the top-level ontology `Gumo` for the uniform management of user and context models in a semantic web environment. We discuss design decisions, while putting the focus on ontological issues. The structural integration into user model servers, especially into the U2M-UserModel&ContextService, is also presented. We show ubiquitous applications using the user model ontology `Gumo` together with the user model markup language `UserML`. Finally, we ask how data from Web 2.0 and especially from a social tagging application like del.icio.us as a basis for user adaptation and context-awareness could influence the ontology.

Keywords ubiquitous user modeling, semantic web, ontological engineering, web 2.0, user model markup language

1 Motivation and Introduction

A commonly accepted top level ontology for user and context models is of great importance for the user modeling and context research community. This ontology should be represented in a modern semantic web language like OWL and thus be available for all user-adaptive systems at the same time via internet. The major advantage would be the simplification for exchanging user model and context data between different user-adaptive systems.

However, the current trends of web 2.0 and social computing tell us that the users like to create their own tag spaces, naming conventions and taxonomies. The masses of tagging, rating and even blogging define a kind of "wisdom of the crowds". Now the question arises how this new bottom-up approach can be combined with the more top-down approach of ontology engineering. Does a revisiting of a domain ontology like the user model and context ontology GUMO make sense? There are two directions of mutual influence possible. An existing ontology could be used in taxonomy learning of tag spaces in a way of seeding, or the other way round, the taxonomies that are dynamically generated by the tagging behavior of communities can be used to correct or update existing ontologies. Approaches for tag-space mining are presented in [Schmitz et al., 2006], [Heymann and Garcia-Molina, 2006] and [Golder and Huberman, 2006]. And in [Mika, 2005] a first attempt is shown how to learn ontologies from tag-space mining. Please notice that we present in this paper only initial thoughts in the direction of the duality of ontology engineering and tag-space mining. Back to the ontological approach. The problem of syntactical and structural differences between existing user modeling and context systems could be overcome with a commonly accepted taxonomy, specialized for user modeling tasks. Note, that we are talking about a user model ontology rather than a user model*ing* ontology, which would include, the inference techniques, or knowledge about the research area in general. We are analyzing the user's dimensions that are modeled within user-adaptive systems like the *user's heart beat*, the *user's age*, the *user's current position* or the *user's birthplace*.

Ontologies provide a shared and common understanding of a domain that can be communicated between people and heterogeneous and widely spread application systems, as pointed out in [Fensel, 2001]. Since ontologies have been developed and investigated in artificial intelligence to facilitate knowledge sharing and reuse, they should form the central point of interest for the task of exchanging user models. The design choices in our approach are described in the following. The main conceptual idea for the construction of the specialized user model ontology `Gumo` was to divide the descriptions of user model dimensions into three parts: `auxiliary` - `predicate` - `range`. For example if one wants to say *something about the user's interest in football*, one could divide this into the auxiliary part: "interest", the category part "football" and the range part: "low-medium-high". If a system wants to express something like *the user's knowledge about Beethoven's Symphonies*, one could divide this into the triple: "knowledge" - "Beethoven's Symphonies" - "poor-average-good-excellent". As a third example, *the user's hair-color* would lead to: "property" - "hair-color" - "black-red-brown-blonde-white". First of off all, important groups of auxiliaries have to be identified. A list of identified important user model auxiliaries could be { has Property, has Interest, has Believe, has Knowledge, has Preference, has Regularity, has Plan, has Goal, has Location }. This listing is not intended to be complete, but it is a start with which, most of the important user facts can be realized. Then the user model predicates have to be classified and analyzed. But it turned out that actually everything can be a category for the auxiliary "interest" or "knowledge", thus a whole world-ontology would be needed, what leads to a real problem if one does not work modularized. The crucial idea is to leave this part open for existing other ontologies like the general CYC ontology (see [Lenat, 1995] for example), the UbisWorld ontology (see [Stahl and Heckmann, 2004]), or any other. This insight leads to a modular approach which forms a key feature rather than a disadvantage. Nevertheless the problem of finding a commonly accepted, specialized top level ontology for the user modeling research group is moved into the user's property section: Which classes of user dimensions can be identified? In [Jameson, 2001] and in [Kobsa, 2001] rough classifications for such categories can be found. However, no top level user model ontology has been proposed so far.

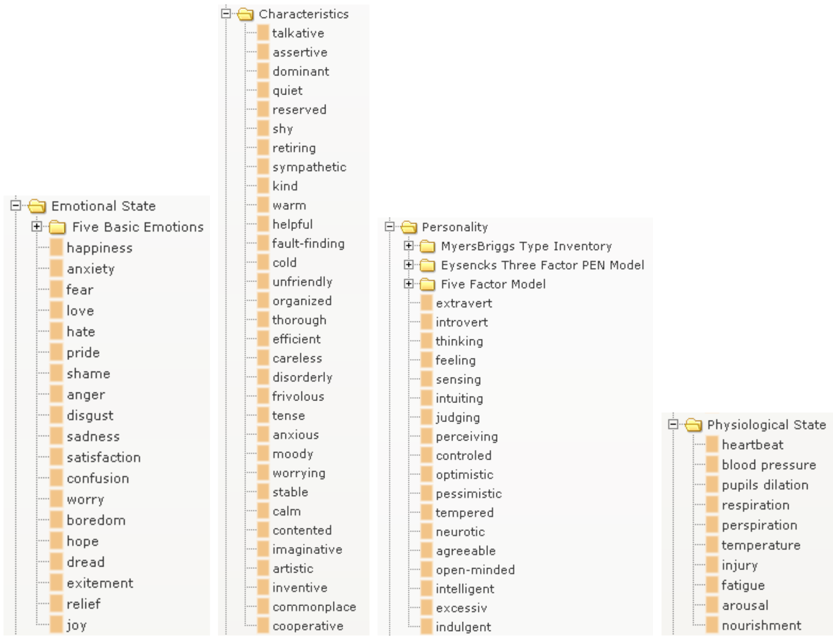

Fig. 1. Several User Model Property Dimensions: Emotional States, Characteristics and Personality with included sub models like the "Five Factor Model"

2 Representation of `Gumo` in OWL

In this section we discuss, why we have chosen the web ontology language OWL. We present three concept definitions, namely the class "Physiological State", the user model dimension "Happiness" and the auxiliary "has Knowledge".

2.1 Three example concept definitions from `Gumo`

Figure 2 presents as a first example the concept of the user model dimension class *Physiological State* which is realized as a `owl:Class`. A class defines a group of individuals that belong together because they share some properties. Classes can be organized in a specialization hierarchy using `subClassOf`. There is a built-in most general class named Thing that is the class of all individuals and a superclass of all OWL classes. The *Physiological State* is defined as subclass of *Basic User Dimensions*.

Every new concept has a unique `rdf:ID`, that can be resolved into a complete URI. Since the handling of these URIs could become very unhandy, a short identification number was introduced, the so called `u2m:identifier`. The identification number in this case is 700016, it has been chosen arbitrarily but seen

```
<owl:Class rdf:ID="PhysiologicalState.700016">
    <rdfs:label> Physiological State </rdfs:label>
    <u2m:identifier> 700016 </u2m:identifier>
    <u2m:lexicon>the state of the body or bodily functions</u2m:lexicon>
    <u2m:website rdf:resource="&UserOL;concept=700016" />
    <rdfs:subClassOf rdf:resource="#BasicUserDimensions.700002" />
</owl:Class>
```

Fig. 2. The OWL class definition of "Physiological State"

under its namespace, it is unique. It has the advantage of freeing the textual part in the `rdf:ID` from the need of being semantically unique. The term *mouse* for example, could be read as the animal mouse, or as the computing device mouse. Apart from solving the problem of conceptual ambiguity, this number facilitates the work within relational databases, which is important from the implementation point of view.

Figure 2 also defines the lexical entry `u2m:lexicon` of the concept of *Physiological State* as "the state of the body or bodily functions", while this textual definition could also be realized through a link to an external lexicon. The attribute `u2m:website` points towards a web site, that has its purpose in presenting this ontology concept, to a human reader. The abbreviation `&UserOL;` is a shortcut for the complete URL to the `Gumo` ontology.

```
<rdf:Description rdf:ID="Happiness.800616">
    <rdfs:label> Happiness </rdfs:label>
    <u2m:identifier> 800616 </u2m:identifier>
    <u2m:durability> Hour.520060 </u2m:durability>
    <u2m:image rdf:resource="http://u2m.org/UbisWorld/img/happiness.gif" />
    <u2m:website rdf:resource="&UserOL;concept=800616" />
    <rdf:type rdf:resource="#EmotionalState.700014" />
    <rdf:type rdf:resource="#FiveBasicEmotions.700015" />
</rdf:Description>
```

Fig. 3. GUMO definition of "Happiness"

Figure 3 defines the user model dimension *Happiness* as an `rdf:Description`. It contains a `rdfs:label`, a `u2m:identifier` and a `u2m:website` attribute. Additionally it provides a default value of the average durability `u2m:durability`. It carries the qualitative time span of how long the statement is expected to be valid (like minutes, hours, days, years). In most cases when user model dimensions or context dimensions are measured, one has a rough idea about the expected durability, for instance, emotional states change normally within hours, however personality traits won't change within months. Since this qualitative time span is dependent from every user model dimension, a definition mechanism is prepared within the `Gumo`. Some examples of rough durability-classifications, without any attempt of proven correctness, are:

- physiologicalState.heartbeat - can change within seconds
- mentalState.timePressure - can change within minutes
- emotionalState.happiness - can change within hours
- characteristics.inventive - can change within months
- personality.introvert - can change within years
- demographics.birthplace - can't normally change at all

Another important point that is shown in the definition of *happiness* in figure 3 is the ability in OWL of multiple-inheritance. In detail, happiness is defined as `rdf:type` of the class *Emotional State* as well as `rdf:type` of the class *Five Basic Emotions*. Thus OWL allows to construct complex, graph like hierarchies of user model concepts, which is especially important for ontology integration. Figure 4 defines the auxiliary *has Knowledge* as `rdfs:subPropertyOf` of the

```
<rdf:Property rdf:about="hasKnowledge.600120">
    <rdfs:label> has Knowledge </rdfs:label>
    <u2m:identifier> 600120 </u2m:identifier>
    <u2m:website rdf:resource="&UserOL;concept=600120" />
    <rdfs:domain rdf:resource="#Person.110003" />
    <rdfs:subPropertyOf rdf:resource="#UserModelAuxiliary.600020" />
</rdf:Property>
```

Fig. 4. GUMO Property hasKnowledge as example for general auxiliaries

resource *user model auxiliary* with the `rdf:domain` *#Person*, which is not part of the user model ontology itself, but which is part of the general UbisWorld Ontology, see [Stahl and Heckmann, 2004]. The acronym `u2m` stands for *ubiquitous user modeling* and forms a collection of standards, that are available online at `http://www.u2m.org/`. The new vocabulary for the user model ontology language consists of `u2m:identifier, u2m:durability, u2m:image, u2m:website u2m:lexicon` . The main User Model Dimension that we identified so far are *MentalState PhysicalState, Demographics, ContactInformation, Role, EmotionalState, Personality, Characteristics, Ability, Proficience* and *Motion*.

To support the distributed construction and refinement of the top level user model ontology, we developed a specialized online editor, that helps with introducing new concepts, adding their definitions and transform the information automatically into the required semantic web ontology language. Currently supported are RDF and OWL.

3 The U2M-UserModelServer

A user model server manages information about users or individuals in general. The U2M-UserModel&ContextService, see [Heckmann, 2003a] is an application-independent server with a distributed approach for accessing and storing user information, while the focus lies on the possibility to exchange and understand the data between different applications, as well as adding privacy and transparency to the statements about the user itself. The semantics for all concepts is mapped to the `Gumo` ontology.

Applications can retrieve or add information to the server by simple HTTP requests, alternatively, by the "UserML" WebService. UserML, see for example [Heckmann and Krüger, 2003], is an XML application which is based on the concept of "situational statements", as introduced in [Heckmann, 2003b]. A request could look like:

```
http://www.u2m.org/UbisWorld/UserModelServer.php?
subject=Joerg.210006&auxiliary=hasProperty&predicate=Age.800302
```

Mentionable is the optional naming convention for disambiguation, like "Joerg.210006" or "Age.800302". These names are unique identifiers for the particular, intended concepts. A general problem when one wants to talk about objects, individuals or concepts is the non-uniqueness of names, as seen before, especially in an open web-based system. In the Semantic Web approach, each resource is mapped to a (hopefully) unique URI. But the URIs have the disadvantage that they are rather long and uneasy to read. The used naming-format "Name.Id" can be seen as a shortcut for such a unique URI. Those unique resource identifiers, for the area of user modeling, are established in the `Gumo`.

The user model server "u2m.org" can be used by every user adaptive system to manage user related data, but also by the modeled user himself. A specialized UserModelEditor is provided which displays the information in a web-browser form that allows the change and privacy control, see `http://www.u2m.org`. The access, the purpose and the retention of every situational statement can be controlled in the "editor view modus". Each statement can contain meta information like creator, method, evidence or confidence. Figure 5 shows the overall architecture of the UserModelServer with its input and output information flows *Query*, *Answer* and *Add* that are represented as arrows. The main block of the illustration contains four piled, dotted rectangles. The lowest one indicates the distributed storage of the so called SituationalStatements, which are explained in detail in [Heckmann and Krüger, 2003]. The second rectangle shows the filter, ranking and conflict resolution strategies that are applied to the set of Situational Statements. The User Model Server itself, which is responsible for communication, handling requests and responses, is based on both introduced rectangles as well as the rectangle on the top for distributed knowledge bases in form of semantic web ontologies. A query or request, that is received in the so called `UserQL` query language will be handled by the user model server in the following way: first all matching situational statements are retrieved, then the filter and resolution strategies are applied and finally the semantics is given by referencing to web ontologies.

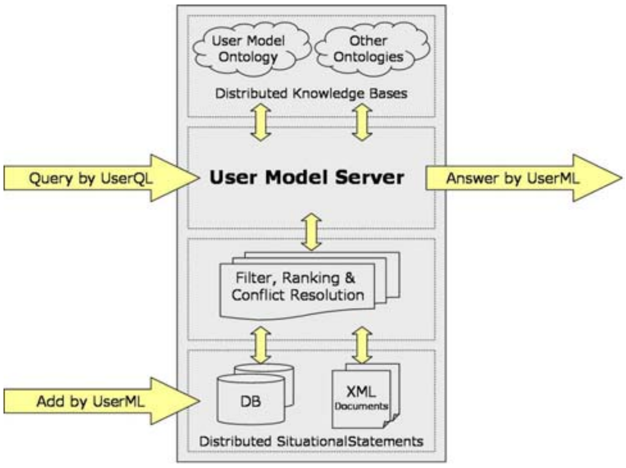

Fig. 5. Architecture of the UserModelServer

4 How to further develop GUMO in the era of Web2.0?

The Semantic Web is based on the content-oriented description of digital documents with standardized vocabularies that provide machine understandable semantics. The result is the transformation from a Web of Links into a Web of Meaning / Semantic Web, (see arrow A in Fig. 6). On the other hand, the traditional Web 1.0 has recently been orthogonally shifted into a Web of People / Web 2.0 where the focus is set on folksonomies, collective intelligence and the wisdom of crowds (see arrow B in Fig. 6). Only the combined muscle of semantic web technologies and broad user participation will ultimately lead to a Web 3.0 with completely new business opportunities in all segments of the ITC market. Without Web 2.0 technologies and without activating the power of community-based semantic tagging, the emerging semantic web cannot be scaled and broadened to the level, that is needed for a complete transformation of the current syntactic web. On the other hand, current Web 2.0 technologies cannot be used for automatic service composition and open domain query answering without adding machine-understandable content descriptions based on semantic web technologies. The ultimate world-wide knowledge infrastructure cannot be produced fully automatically, but needs massive user participation based on open semantic platforms and standards.

The interesting and urging question that arises is: what happens when the emerging Semantic Web and Web 2.0 meet with their full potential power?

There are no new technologies introduced by Web 2.0, but the role and value of the user has been changed significantly. We focus in this paper on *tagging*.

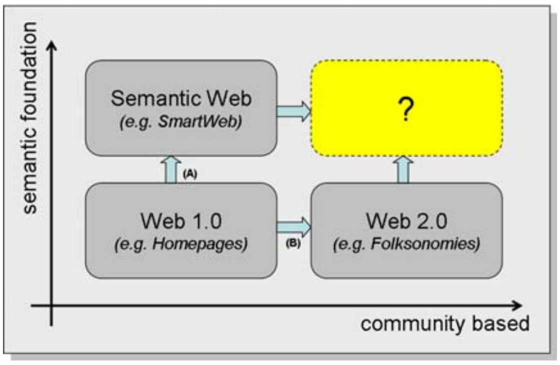

Fig. 6. Joining Semantic Web and Web 2.0

However, a social *rating* system could also be of interest in order to improve the ontologies.

Tag spaces are an obvious source of data for user modeling. The user of a social tagging tool could provide access to his personal tag space to an e-commerce site which could use the data to tailor its structure and presentation to the user. For example, a music store could attempt to assess where a user lives given data from a social bookmarking site. Then, if the user is interested in an album by an artist who will give a concert in the vicinity of the user's home town, the store could offer him tickets for the event. How can we use a tag space and a user's tagging data to create a user model and adapt a system? Furthermore, how can we use the already developed general user model and context Ontology `Gumo` to improve the tagging taxonomy and the generated user model and context rules? Figure 7 shows the possible connection of `Gumo` and Web 2.0.

The approach we are proposing starts with automatically learning a structure of the tag space, then manually defining adaptation rules based on that structure, and finally automatically mapping a user's data into the structure in order to decide what adaptation rules to apply. This implies that the set of possible adaptation rules depends on the learned structure. For instance, creating a rule with a precondition on the home town of a user is sensible only if this information is part of the structure. Not all tag spaces are suitable for this type of user modeling. Because we want to learn something about the user's interests, we require tagging data used by the user for himself (as in del.ico.us) and not for others (as in flickr).

We are aiming for a taxonomy of tags, where subtags of a tag tag (for example, pop-music should be a subtag of music). For the designer of an adaptive system, identifying the semantics of a tag (by using its predecessors and successors its generality (the higher it is in the taxonomy, the more users will Hence, we think a taxonomy is a good underlying structure for the a taxonomy from a

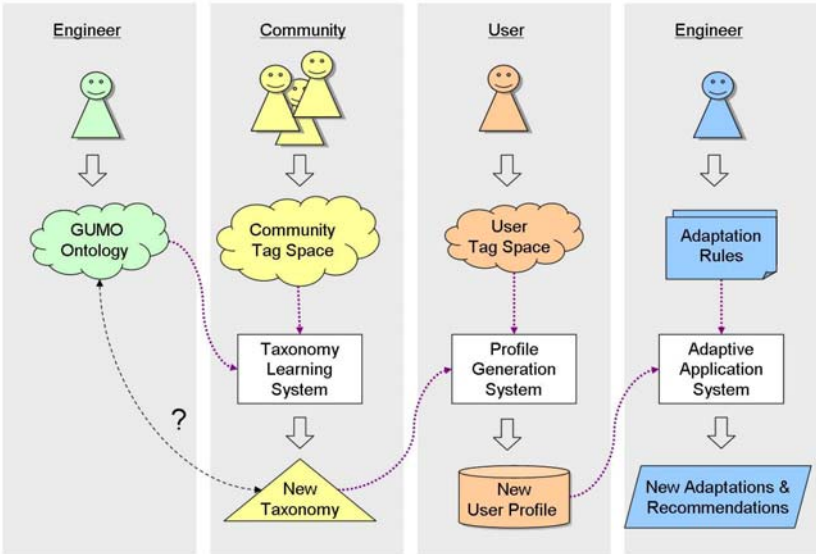

Fig. 7. A possible connection of GUMO and Web 2.0

tag space is the main subject of this paper. See [Schwarzkopf et al., 2007] for a detailed description of this approach.

Summary We have revisited the user model and context ontology `Gumo` in the semantic web ontology language OWL together with the exchange language `UserML` and the U2M UserModel&ContextServer. This work is highly under progress and the future goal is to find out the influence of social computing in Web 2.0 to the so far only semantic web approach in order to determine the possible advantages of combining tag-space mining and ontology engineering.

References

[Fensel, 2001] Fensel, D. (2001). *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*. Springer-Verlag Berlin Heidelberg.

[Golder and Huberman, 2006] Golder, S. A. and Huberman, B. A. (2006). Usage patterns of collaborative tagging systems. *J. Inf. Sci.*, 32(2):198–208.

[Heckmann, 2003a] Heckmann, D. (2003a). Integrating Privacy Aspects into Ubiquitous Computing: A Basic User Interface for Personalization. In Krüger, A. and Malaka, R., editors, *Artificial Intelligence in Mobile Systems (AIMS 2003)*, pages 106–110, Seattle, USA. in conjunction with the Fifth International Conference on Ubiquitous Computing.

[Heckmann, 2003b] Heckmann, D. (2003b). Introducing situational statements as an integrating data structure for user modeling, context-awareness and resource-adaptive computing. In *ABIS2003*, pages 283–286, Karlsruhe, Germany.

[Heckmann and Krüger, 2003] Heckmann, D. and Krüger, A. (2003). A user modeling markup language (UserML) for ubiquitous computing. *Lecture Notes in Artificial Intelligence*, 2702:393–397.

[Heymann and Garcia-Molina, 2006] Heymann, P. and Garcia-Molina, H. (2006). Collaborative creation of communal hierarchical taxonomies in social tagging systems. Technical Report 2006-10, Stanford University.

[Jameson, 2001] Jameson, A. (2001). *Systems That Adapt to Their Users: An Integrative Perspective*. Habil, Saarbrücken, Germany.

[Kobsa, 2001] Kobsa, A. (2001). Generic user modeling systems. *User Modelling and User-Adapted Interaction Journal*, 11:49–63.

[Lenat, 1995] Lenat, D. B. (1995). CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38.

[Mika, 2005] Mika, P. (2005). Ontologies are us: A unified model of social networks and semantics. *In International Semantic Web Conference, volume 3729 of Lecture Notes in Computer Science*, pages 522–536.

[Schmitz et al., 2006] Schmitz, C., Hotho, A., Jaschke, R., and Stumme, G. (2006). Mining association rules in folksonomies. *In V. Batagelj, H.-H. Bock, A. Ferligoj, and A. iberna, editors, Data Science and Classification, Studies in Classification, Data Analysis, and Knowledge Organization, Berlin, Heidelberg, Springer*, pages 261–270.

[Schwarzkopf et al., 2007] Schwarzkopf, E., Heckmann, D., Dengler, D., and Krner, A. (2007). Mining the structure of tag spaces for user modeling. In *Complete On-Line Proceedings of the Workshop on Data Mining for User Modeling at the 11th International Conference on User Modeling*, pages 63–75, Corfu, Greece.

[Stahl and Heckmann, 2004] Stahl, C. and Heckmann, D. (2004). Using semantic web technology for ubiquitous hybrid location modeling. In *UbiGis*.

Context-Sensitive Referencing for Ontology Mapping Disambiguation

Heiko Paulheim, Michael Rebstock, and Janina Fengel*

Hochschule Darmstadt, University of Applied Sciences
{lastname}@fbw.h-da.de

Abstract. Ontologies can be used for e-business integration, for example by describing existing e-business standards as ontologies. If cooperating parties use different ontologies, ontology mappings are needed, which can be ambiguous, thus making ontology mapping disambiguation necessary. Different disambiguation strategies exist, such as community-driven or context-sensitive referencing of ontologies, where the latter is what we developed in our project. In this paper, we show that community-driven referencing can be realized using a context-sensitive referencing service in a way that the user administration is transparent to the referencing system.

Keywords: Semantic Synchronization, Ontology Mapping, Ontological Engineering, Context-Sensitivity, Communities

1 E-Business Integration with Ontologies

Standards play an important role in electronic business. Unfortunately, there are different and competing standards for describing products, processes, documents, and the like. To allow interoperability, mechanisms that allow parallel usage of elements from different e-business standards in the same process are needed.

Nowadays, such mechanisms mainly either exist in the users' minds, or in fixed translation tables that require a major project effort and do not allow dynamic change. Furthermore, semantic synchronizations carried out manually are not persistent. With the framework presented in this paper, we provide a general architecture for the implementation of an evolutionary semantic synchronization service that can be integrated into different e-business systems to support users with semantic knowledge.

Following [1], we look at e-business standards as ontologies, thus, the elements to be synchronized are the ontologies' concepts and properties. This enables us to use methods and tools from the field of ontological engineering. Some existing e-business standards, like UN/SPSC [2] and eCl@ass [3], have already been transferred into ontology languages. Furthermore, a lot of research has been conducted in the past years on technologies for processing ontologies, so there are a couple of components ready to use, including ontology representation, visualization, mapping, and reasoning. We have implemented a framework on

* The authors are supported by the German Federal Ministry of Education and Research under grant 1716X04, project ORBI.

top of JENA2 [4] and Java that allows connecting such components to form a coherent semantic referencing service [5], as well as reusing techniques from information retrieval (IR).

The service allows users to find references between ontologies. References may either be created manually or established automatically by a mapping tool. However, as stated in [6], more than one reference can exist for the same element, caused by different modelling approaches and granularities of the individual standards, even more so if proprietary or in-house standards are used. Therefore, reference disambiguation strategies are needed, which filter appropriate results and/or sort results by relevance. The framework developed in our project evaluates context information to provide reference disambiguation.

The rest of this paper is structured as follows. Section 2 describes the basics on ontologies, references, and context. Section 3 explains two approaches for reference disambiguation: community-based and context-sensitive referencing. Section 4 shows how community-based referencing can be realized using context-sensitive referencing. Section 5 provides an overview on related work, and section 6 closes with a discussion of our results.

2 Ontologies, Semantic References, and Context

Ontologies are structured, machine-readable representations of knowledge. There are many different definitions of what an ontology actually is (for a comprehensive overview see [1]), however, we will look at ontologies as a collection of definitions of elements and their relations. Ontologies can be represented in different languages, the most dominant are RDF Schema [7], and the various dialects of OWL [8]. Ontologies are considered as a means for e-business integration [9], however, if two or more cooperating parties use different ontologies, further steps have to be taken to allow seamless interoperability.

Therefore, ontology matching solutions are needed, which produce mappings from elements in one ontology to elements in another. There are two main categories of ontology matching algorithms [10]. One are element-based approaches, which try to match single elements of an ontology, either using only the information given in the ontology itself (e.g., by measuring string distance using the edit distance), or by using external information, e.g. upper-level ontologies, such as WordNet [11]. The second are structure-level approaches, which do not only analyze elements isolated from each other, but also their relations and patterns they form in graphs. An overview and more detailed analysis of matching approaches can be found in [10] and [12]. Some approaches, like [13], combine the weighted results of several matching solutions in order to obtain mappings of higher quality.

Ontology matching tools provide references. In extension of [14], references can be described as a five-dimensional vector of the form

$$reference := \langle entity1, entity2, type, confidence, acceptance \rangle . \qquad (1)$$

The first two entries *entity*1 and *entity*2 are URIs of the elements from both ontologies to be referenced, *type* describes the kind of relation (like "equal", "subclass of", etc.), $confidence$ describes the degree of probability of the relation, and *acceptance* expresses the users' rating of that reference. For example, the reference

$$r_1 = \langle StandardA\#X, StandardB\#Y, equal, 0.87, 0.95\rangle \tag{2}$$

is read as "Element X in StandardA and element Y in StandardB are equal with a probability of 87%, and 95% of all users agreed on that statement". The acceptance value is calculated from the users' ratings.

In order to disambiguate such semantic references, we have developed an approach which uses context information. There has been a lot of research on context in the fields of machine translation and IR, yielding several ways of describing context. In machine translation, shallow and deep approaches [15], bag of words and relational approaches [16] are distinguished to solve the problem of word sense disambiguation. In IR, context data can be represented in different forms, from simple binary vectors to highly complex graphs, as proposed by [17]. An introduction to context queries in IR can be found in [18].

3 Approaches for Mapping Disambiguation

3.1 Community-Based Referencing

The idea of context mapping disambiguation by using communities has first been developed by Anna V. Zhdanova and Pavel Shvaiko in [19]. The general problem of community-based referencing can be formally defined as follows:

Definition 1. *Given a user being member in a non-empty set of communities S_U, find those references for an element x from a set of ontologies O_1 to a set of ontologies O_2 that have been created by a user being member in a non-empty set of communities S_C under the condition that $S_U \cap S_C$ is not empty.*

That means that a user issuing a query for semantic references on an element is presented all references for that element created by users with whom he has at least one community in common (note that we are considering the creators of *ontology references*, not of the ontologies themselves). The user's login and community data are directly processed by the referencing system.

Although the authors of [19] primarily focused on mapping reuse, this community-driven approach can also be seen as an ontology mapping disambiguation strategy: different semantic references caused by ambiguous use of elements in different communities are filtered and thereby disambiguated. We will call a semantic referencing service that allows disambiguation by using context information a *community-based semantic referencing service*.

Figure 1 demonstrates the idea of community-driven mapping disambiguation. There are two references for the element "switch" from a rather coarse-grained proprietary standard P to the more fine-grained standard eCl@ss [20], each having its right to exist in a given context. User 1 is a network administrator using standard P for ordering an ethernet LAN switch. Since the supplier

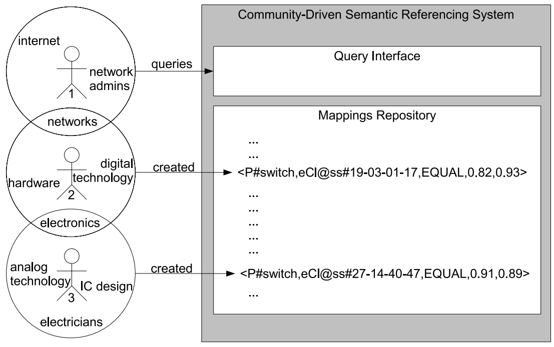

Fig. 1. Community-driven mapping disambiguation

uses eCl@ss, user 1 queries the semantic referencing system for references for the element "switch". The system returns the reference to "19-03-01-17" (which is the eCl@ss code for "network switch") created by user 2, since both users are in the "networks"-community, but does not return the reference to "27-14-40-47" (which is the eCl@ss code for "toggle switch") created by user 3, since users 1 and 3 do not share any communities. The list of references that exist for the element "switch" is thus filtered and thereby disambiguated.

3.2 Context-Sensitive Referencing

A different approach for disambiguating semantic references is the evaluation of the context of the term to be referenced. The general problem of context-sensitive referencing can be defined as follows:

Definition 2. *Given some context information $C(x)$, find the references for an element x from a set of ontologies O_1 to a set of ontologies O_2, with an acceptance value $acc_{C(x)}$ (which is the higher the more appropriate the reference is in this context), calculated dynamically for that context information and exceeding a minimum acceptance threshold acc_{min}.*

Such an acceptance value $acc_{C(x)}$ can be obtained in different ways. Since one of the design aims of our system was to minimize the need for manual preparatory work, we decided to calculate $acc_{C(x)}$ based on user ratings. Each user can rate (in the easiest case: accept or deny) a reference in his or her context, and the ratings are stored in the system. Each time a user requests a reference for an element in a context, the acceptance value is calculated using the distance-weighted k-nearest-neighbor rule [21], with the difference between the similarity of the request's context $C_Q(x)$ and the rating's context $C_R(x)$ as distances, given any similarity function sim. In other words, $acc_{C(X)}(Ref)$ is calculated as

$$acc_{C_Q(X)}(Ref) = \begin{cases} \sum_{R \in Ratings(Ref)} \frac{sim(C_Q(x), C_R(x))}{sum_{sim}} \cdot acc(R) & sum_{sim} > 0 \\ acc_{def} & sum_{sim} = 0 \end{cases} \quad (3)$$

where sum_{sim} is calculated as

$$sum_{sim} := \sum_{R \in Ratings} sim(C_R(x), C_Q(x)). \quad (4)$$

and acc_{def} is a configurable parameter which serves as a default acceptance if no ratings exist or if none of the ratings is at least minimally similar to the query's context. In the latter case, it is also possible to use the unweighted median of all ratings.

We will call a semantic referencing service which uses context-sensitive reference disambiguation a *context-sensitive semantic referencing service.*

As already stated in section 2, there are different ways to describe context. Since different client applications can have different strategies of gathering context information, using more specific context information (as in deep and relational approaches) narrows the variety of possible client applications. Therefore, we decided for a relational approach which uses a weighting factor for each context term, where the context terms are simple strings. Therefore, the context of an element x is defined as a set of context terms $C(x)$, and a normalized weighting function ω, defined as

$$\omega_{C(X)} : C(X) \to [0, 1] \text{ with } \max_{y \in C(X)} \omega_{C(X)} = 1. \quad (5)$$

That function can also be interpreted as a reverse of a distance function: the higher a context term's weight, the closer it is to the term in question.

Since many context similarity measures are defined for vectors, with the context terms used as dimensions and the weights as values, the weighting function can also be regarded as a weighting vector $w_{C(X)}$ with

$$w_{i,C(X)} := \omega_{C(X)}(t_i), t_i \in C(X), 1 \le i \le |C(X)|. \quad (6)$$

With those definitions, an acceptance value can be calculated for each reference, determining that reference's appropriateness in the query's context. Thereby, semantic references can be disambiguated. Details on context-sensitive reference disambiguation can be found in [22].

4 Community information as a special kind of context

4.1 Using communities as context information

A query for references in a community-driven scenario, as stated in definition 1, can be identified by a query term X and by a set S_U of community identifiers, where $S_U \subseteq S$, and S represents the set of all communities. A query in a context-sensitive scenario, as stated in definition 1, is identified by a query term X, a context set $C(X)$ (containing context terms), and a weighting function $\omega_{C(X)}$ as defined in (5).

Since, according to definition 1, the result set would be empty if the user was not a member of any community, we assume that each user issuing a query is a member of at least one community.

In order to transform a community-driven query to a context-sensitive one, we treat the community identifiers as simple strings and define:

$$C(X) := S \text{ and } \omega_{C(X)}(t) := \begin{cases} 1 & \forall\, t \in S_U \\ 0 & \forall\, t \in S - S_U \end{cases} \tag{7}$$

We are now going to show that our context-sensitive reference disambiguation approach answers context-based queries as defined above such that the following requirements are fulfilled:

Requirement 1: All references created by users that share at least one community with the user issuing the query are returned.

Requirement 2: No references created by users that do not share any community with the user issuing the query are returned.

To this end, we use the cosine similarity [18] as a similarity measure, and a default acceptance $acc_{def} = 0$. Furthermore, we assume that for each reference that one and only one rating exists, whose context is the community information of the reference's creator as defined above and whose acceptance value is 1. We will elaborate on how to assure this assumption in the next section.

Let $w_{C_Q(X)}$ be the query's weighting vector and $w_{C_Q(X)}$ be the rating's vector (containing the community information of the reference's creator), according to (6).

The cosine similarity is defined as

$$sim_{cos}\left(w_{C_Q(X)}, w_{C_R(X)}\right) := \frac{w_{C_Q(X)} \bullet w_{C_R(X)}}{\left\|w_{C_Q(X)}\right\| \left\|w_{C_R(X)}\right\|}. \tag{8}$$

Since each user is a member of at least one community, at least one element in both w_Q and w_R has a value of 1, thus, the denominator never equals 0. Furthermore, $w_{C_Q(X)} \bullet w_{C_R(X)}$ is greater than zero if and only if both vectors contain a non-zero element in the same position, e.g. if both users have at least one community in common, and zero otherwise. Thus, (3) reduces to

$$acc_{C_Q(X)}(Ref) = \begin{cases} > 0 & \text{if } sim\left(w_{C_Q(X)}, w_{C_R(X)}\right) > 0 \\ 0 & \text{if } sim\left(w_{C_Q(X)}, w_{C_R(X)}\right) = 0 \end{cases} \tag{9}$$

Thus, if all semantic references are filtered with a threshold of $acc_{min} = 0$, and only references with an acceptance value $acc_{C_Q(X)}(Ref) > 0$ are returned, the two requirements stated above are fulfilled. That shows that our system can provide community-driven reference disambiguation, put down to context-sensitive referencing.

4.2 Providing community-based reference disambiguation by a context-sensitive referencing service

Our original context-sensitive referencing service provides three main functions:

- Create a new reference,
- get a list of references in a given context,
- and rate a reference in a given context.

In order to assure that only one rating exists for each reference, as proposed in the section above, those functions are encapsulated to form a community-based referencing service as follows:

- Each time a user creates a reference using the community-driven referencing service, the reference is automatically rated with an acceptance value of 1 in the context derived from the user's community information.
- The request for a list of references remains the same.

With this approach, we have created a community-driven semantic referencing service by encapsulating our context-sensitive semantic referencing service, where the latter remains unchanged. The referencing system only processes context data, thus abstracting away from user and community administration. In principal, the algorithm is generic enough to solve other context-based disambiguation tasks as well.

5 Related Work

In the area of ontological engineering, much research work has already been conducted on ontology matching and ontology reasoning. Ontology matching deals with finding similarities between ontologies, often in order to merge them [10]. Ontology reasoning tries to derive new knowledge from knowledge already present in an ontology. There are also approaches trying to improve ontology mappings by means of ontology reasoning [23], while others propose an ontology mapping language capable of mapping heterogeneous information, like concepts to relations [24].

Some research projects deal with providing semantic references between e-business standards to allow semantic integration. Besides the already mentioned community-based approach developed by Zhdanova and Shvaiko [19], some other projects exist. [25] combine agents and ontology mapping to allow automatic e-business transactions. Some approaches try to collect references under the umbrella of one global ontology, like WordNet [26]. [27] propose a hierarchy of ontologies connected by mappings. Zimmermann and Euzenat haven shown in [28] that a context-sensitive approach is not possible for ontology alignment. However, it is a feasible approach for disambiguating semantic references. Other works, like [29], use ontologies, for example, to disambiguate items like person names in unstructured text by searching context terms in ontologies, unlike our approach, where context terms can be arbitrary strings that need not exist in any ontology.

The problem of context-sensitive referencing can be regarded as a special information retrieval problem. Extensive research has been conducted in this area. The present approaches stretch from using simple context term vectors [18] to describe context in rich semantic structures like RDF graphs [17]. There are also community-based information retrieval approaches, like [30], which uses the visualization of different perspectives in distinct communities for sharing information across community borders.

While our system is based on creating a collection of references, other approaches try on-the-fly mapping of ontologies [31], which is a reasonable approach when, like in the case of very large ontologies, the collection of mappings tends to become rather extensive. There are also works on matching blocks of partioned ontologies [32], which could be a possible approach to deal with the problem of large ontologies.

6 Discussion

In this paper, we have shown that a context-sensitive semantic referencing service, combined with user's ratings, can also be used for providing community-based semantic referencing. Both are feasible approaches for ontology mapping disambiguation, each having their advantages and drawbacks:

- Both approaches provide mechanisms to create a growing knowledge base of semantic references.
- Community-based referencing needs the additional implementation of user and community administration, while context-sensitive referencing also works from scratch (our implementation of the service also works with empty context information).
- On the other hand, community-based referencing is an appropriate approach to ensure that references remain private in a community and users from other communities will never come to see those references.
- The rating mechanism underlying our context-sensitive approach can also be made transparent to the user by observing the user's behavior: if a user works with a reference, it receives a positive rating, if s/he decides not to work with a proposed reference, it receives a negative rating.
- Both approaches have to cope with erroneous user's entries. Community-based referencing only has to deal with wrong references. Context-sensitive referencing also has to handle wrong ratings, which can mislead the system to calculate a wrong acceptance value and thus present a reference not appropriate in a context as being highly appropriate, and vice versa. However, the ratio of correct ratings to incorrect ones is high enough, the weight of wrong ratings decreases, and it is likely that many negative ratings will make a wrong reference fall below the lower acceptance threshold and thus make it "disappear" from the list of results displayed for the user.
- Since the usage context of a term in general can be expected to be similar within a community and different between distinct communities, context information can be looked at as implicit community information, and vice versa.

The approach presented in this paper does not yet allow using context-sensitive and community-driven semantic referencing in parallel (e.g. to further disambiguate different references used in a community). However, if this can be achieved by allowing two sets of context (the community information and the actual context information), calculating an acceptance value for each context and applying filters to each of the calculated acceptance values. Such an approach would also make the use of further types of context information possible, like documents, bookmarks, the user's role in a company, or previous projects the user has worked on, as proposed by [17].

References

1. Gómez-Pérez, A., Fernández-López, M., Corcho, O.: Ontological Engineering. Springer, London et al. (2004)
2. Klein, M.: DAML+OIL and RDF Schema Representation of UNSPSC. http://www.cs.vu.nl/mcaklein/unspsc, October 31, 2006
3. Hepp, M.: Ontologizing eCl@ss. http://www.heppnetz.de/eclassowl/, October 31, 2006
4. Jena: Jena - A Semantic Web Framework for Java. http://jena.sourcefourge.net/, October 25, 2006
5. Paulheim, H., Rebstock, M., Fengel, J.: Dynamic Semantic Synchronization in E-Business. In: Proceedings of the 1st European Semantic Technology Conference, May 31st - June 1st, Vienna, Austria (CD-ROM). (2007)
6. Schulten, E., Akkermans, H., Guarino, N., Botquin, G., Lopes, N., Dörr, M., Sadeh, N.: The E-Commerce Product Classification Challenge. IEEE Intelligent Systems **16**(4) (2001) 86–c3
7. RDF Core Working Group: RDF Vocabulary Description Language 1.0: RDF Schema (2004) http://www.w3.org/TR/rdf-schema/.
8. Web Ontology Working Group: Web Ontology Language (OWL) (2004) http://www.w3.org/2004/OWL/.
9. Uschold, M., Grüninger, M.: Ontologies: principles, methods, and applications. Knowledge Engineering Review **11**(2) (1996) 93–155
10. Shvaiko, P., Euzenat, J.: A Survey of Schema-based Matching Approaches. In: Journal on Data Semantics. (2005)
11. Fellbaum, C., ed.: WordNet: An Electronic Lexical Database. The MIT Press, Cambridge, MA (1998)
12. Kalfoglou, Y., Schorlemmer, M.: Ontology mapping: the state of the art. The Knowledge Engineering Review **18**(1) (2003) 1–31
13. Kalfoglou, Y., Hu, B., Reynolds, D., Shadbolt, N.: CROSI project, final report. Technical Report 11717, School of Electronics and Computer Science, University of Southampton, Southampton, UK (2005)
14. Euzenat, J.: Evaluating ontology alginment methods, extended abstract. In: Dagstuhl Seminar Proceedings 04391, Semantic Interoperability and Integration, http://drops.dagstuhl.de/opus/volltexte/2005/36, June 06, 2005. (2005)
15. Uszkoreit, H.: Shallow Language Processing, Deep Language Processing and Domain Ontologies. In: Natural Language Processing and Knowledge Engineering. (2005)
16. Ide, N., Véronis, J.: Introduction to the Special Issue on Word Sense Disambiguation: The State of the Art. In: Computational Linguistics 24 (1). (1998) 1–40

17. Henrich, A., Morgenroth, K.: Kontextbasiertes Information Retrieval zur Unterstützung von Entwicklungsprozessen. In: eOrganiztion: Service-, Prozess-, Market-Engineering. Volume 2., Universitätsverlag Karlsruhe (2007) 575–592
18. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. ACM Press, New York (1999)
19. Zhdanova, A.V., Shvaiko, P.: Community-Driven Ontology Matching. In Sure, Y., Domingue, J., eds.: The Semantic Web: Research and Applications ESWC 2006, LNCS 4011, Berlin Heidelberg, Springer (2006) 34–49
20. ECl@ss: Internationaler Standard zur Klassifizierung u. Beschreibung von Produkten u. Dienstleistungen. http://www.eclass.de/, December 12, 2006
21. Dudani, S.A.: The distance-weighted k-nearest-neighbor rule. In: IEEE Transactions on Systems, Man and Cybernetics Vol. 6. (1976) 325–327
22. Rebstock, M., Fengel, J., Paulheim, H.: Context-Sensitive Semantic Synchronization in Electronic Negotiations. In: Proceedings of the Group Decision and Negotiation Meeting, Mt. Tremblant (Montreal), May 14-17, 2007. Volume 2. (2007) 253–264
23. Huang, J., Gutiérrez, R.L.Z., García, B.M., Huhns, M.N.: A Schema-Based Approach Combined with Inter-Ontology Reasoning to Construct Consensus Ontologies. In: Proc. AAAI Workshop on Contexts and Ontologies: Theory, Practice and Applications. (2005) 80–87
24. Ghidini, C., Serafini, L.: Reconciling Concepts and Relations in Heterogeneous Ontologies. In Sure, Y., Domingue, J., eds.: The Semantic Web: Research and Applications ESWC 2006, LNCS 4011, Berlin Heidelberg, Springer (2006) 50–64
25. Malucelli, A., Palzer, D., Oliveira, E.: Combining Ontologies and Agents to help in Solving the Heterogeneity Problem in E-Commerce-Negotiations. In: Proceedings of the 2005 International Workshop on Data Engineernig Issues in E-Commerce. (2005)
26. Kim, W., Choi, D.W., Park, S.: Product Information Meta-search Framework for Electronic Commerce. In Gómez-Pérez, A., Euzenat, J., eds.: ESWC 2005, LNCS 3532, Berlin Heidelberg, Springer (2005) 408–422
27. Corcho, O., Gómez-Pérez, A.: Solving Integration Problems of E-commerce Standards and Initiatives through Ontological Mappings. In: Proceedings of the Workshop on E-Business and intelligent Web at the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI2001), Seattle, USA (2001)
28. Zimmermann, A., Euzenat, J.: Three Semantics for Distributed Systems and Their Relations with Alignment Compositions. In: Proc. of 5th International Semantic Web Conference, Athens, GA, USA, November 5-9, 2006, LNCS 4273. 16–29
29. Hassell, J., Aleman-Meza, B., Arpinar, I.B.: Ontology-Driven Automatic Entity Disambiguation in Unstructured Text. In: Proc. of 5th International Semantic Web Conference, Athens, GA, USA, November 5-9, 2006, LNCS 4273. 44–57
30. Novak, J.: Multiperspektivische Wissensvisualisierung für Wissensaustausch in heterogenen Netzen. In: eOrganiztion: Service-, Prozess-, Market-Engineering. Volume 2., Universitätsverlag Karlsruhe (2007) 611–628
31. Lopez, V., Sabou, M., Motta, E.: PowerMap: Mapping the Real Semantic Web on the Fly. In: Proc. of 5th International Semantic Web Conference, Athens, GA, USA, November 5-9, 2006, LNCS 4273. 414–427
32. Hu, W., Qu, Y.: Block Matching for Ontologies. In: Proc. of 5th International Semantic Web Conference, Athens, GA, USA, November 5-9, 2006, LNCS 4273. 300–313

The Difference a Day Makes –
Recognizing Important Events in Daily Context Logs

Michael Wessel[1], Marko Luther[2], and Matthias Wagner[1]

[1] Racer Systems GmbH & Co. KG, Blumenau 50, 22089 Hamburg, Germany
[2] DoCoMo Euro-Labs, Landsbergerstr. 312, 80687 Munich, Germany

Abstract. We study the extension of context ontologies towards enhanced qualitative spatio-temporal representations and reasoning. Our goal is to model and extract events that are important to the user from her context log, i.e. the history of context data collected over a longer period. We present a case study based on actual context ontologies and context data from the ContextWatcher mobile application. The presented work has been fully implemented in the DL-based reasoning engine RACERPRO.

1 Introduction

In this paper we present the representation and recognition of significant events within the context data that a mobile user collects over a longer period such as a couple of days. As contextual data sources we assume data collected by ContextWatcher [1]. ContextWatcher[3] is a mobile application that facilitates easy gathering and sharing of personal context from an underlying network of context providers. These context providers include the user's location in terms of the present location, location traces as well as frequently visited places, all kinds of user-tagged objects and activities, and location-specific information extracted from public sources, such as local weather information. In ContextWatcher, the value of personal context information is multiplied by sharing context with others through networks of defined social relationships.

ContextWatcher is implemented as a self-contained mobile client but can also connect to third party applications. A currently very popular application is the automatic compilation of gathered context into personal daily Web logs, for instance, to show pictures taken on the phone in a certain context, display visits to selected places or to disclose social encounters. Such contextual blogs have been a strong motivation for the work presented in this paper: to enhance the blog readability, to make sharing of posts easier and to simply make blogs more attractive, enhanced concepts to model and recognize important events are needed.

As most data delivered by our context providers is of quantitative nature in the first place, abstraction methods and context ontologies haven been introduced to deal with context at a higher level [2]. At the level of these context ontologies, complex conceptual dependencies between context elements are introduced to enrich contextual descriptions and to implement classification-based reasoning about the user's situation. Qualitative context descriptions were firstly introduced in ContextWatcher to describe user places as conceptual abstractions from exact locations. Examples include place descriptions like "Office", "Home" ore "Business Place". As the supporting context

[3] `http://www.contextwatcher.com`

ontologies evolved, more qualitative concepts that connect to these place descriptions were added. In the current version of ContextWatcher, the linkage between exact physical user locations and qualitative places is implemented through clustering methods which are applied to user traces.

In this paper we exploit extensions of the existing ContextWatcher ontologies towards enhanced qualitative spatial representation and reasoning. We study the implementation of a complex event recognition and management system with RACERPRO [4] as our DL-based reasoning component of choice. The rest of this paper is organized as follows. We first define required terminology and sketch the overall architecture of the proposed framework. Then we describe our RACERPRO event model. We illustrate the potentials of the modeling with an example scenario. Finally we conclude. In the following we assume some basic knowledge of Description Logics (DLs) [3] and related semantic technologies (e.g., W3C standards such as OWL[5] and semantic query languages such as nRQL [4]).

2 Context Awareness and Event Recognition

The most prominent definition of *context* was coined by A. K. Dey et al.: *Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.*

Such a piece of relevant "'information"' is also called a *context element*. As mentioned in the introduction, context elements are provided by *context providers.* In this case study, we are primarily considering the context elements *location* and *time*. The collection of all context elements is called the current context or current *situation*.

Situational reasoning [2] uses background knowledge specified in OWL ontologies to infer additional context elements from the asserted ones. One can claim that context providers merely provide *raw context data*, and that this context data can only be tranformed into context elements (being characterized as "information") by means of *interpretation.* This interpretation is performed with the help of logical reasoning.

Currently, the approach taken by ContextWatcher is to map the context data into *context assertions* in a *context or situation ABox* in RACERPRO. The *ABox realization service* (which is a standard DL inference service) is then used by ContextWatcher to derive the entailed, logically implied ABox context assertions. Each agent is represented as an individual in the ABox, describing the agent's current context. The ABox also includes social as well as spatial relationships.

The mapping function from *context data* to *context assertions* is currently defined procedurally. For example, the location of an agent is provided by a GPS device. So-called *location clusters* are acquired from GPS agent traces which are analyzed offline by statistical learning / clustering methods to find so-called location clusters. An acquired location cluster can then be annotated by the user with an OWL class or DL concept, e.g. *home* or *office*. Membership in these clusters is from now on recognized automatically by ContextWatcher, and appropriate *qualitative* location assertions are put into the context ABox. This mapping function (which not only takes care of location) is called the *Situation Description Generator* in the following. In many cases,

[5] `http://www.w3.org/TR/owl-ref/`

quantitative context data is mapped to *qualitative* context assertions such that OWL or DL reasoning can be exploited, which primarily works on a qualitative, symbolic level. However, by exploiting the *expressive concrete domain reasoning facilities* of RACERPRO we will also show how reasoning on quantitative (time) context data can be performed and exploited.

Context ABoxes in ContextWatcher so far can be described as static descriptions of "snapshots" in space-time. We claim that the recognition of *dynamic space-time histories*, so called events, can provide valuable additional context elements for ContextWatcher – in fact, certain situations can only be recognized if the *situational changes* are considered rather than the (static) situations themselves. For example, the situation *leaving home* is characterized by a *certain change in the agent's situation:* First the agent is *inside* its home cluster, then, in the next situation, he no longer is. In case the event takes place in the early morning hours of a working day, it is reasonable to assume that the agent is leaving his home for work. In case he should be too late, an SMS could be send automatically to his boss, apologizing in advance for being late. Moreover, as mentioned in the introduction, event structures which have been recognized in daily context logs can be used for the automated generation of diary-like blogs. Thus, a *dynamic* context ABox and DL-based event model is needed in which *notions of time and change play a major role.*

3 DL-based Event Recognition – A Case Study with RACERPRO

Our RACERPRO event recognition model includes *three basic building blocks:* a model of time, a model for situations, and an event model. In the following, a situation is called a *state* to make the resemblance with temporal modal logics or AI planning formalisms [5,6] more explicit.

Time Points and Intervals The basic temporal building blocks are *time points* and *intervals*. Let us start with time points. A time point is any ABox individual which has a real valued filler of the `time` attribute in the concrete domain of $\mathcal{ALCQHI}_{\mathcal{R}^+}(\mathcal{D}^-)$, which is the DL implemented by RACERPRO:

```
(define-concrete-domain-attribute time :type real)
(define-concept point-in-time (a time))
```

Two individual time points `p1` and `p2` can be modeled in the ABox as follows:

```
(instance p1 (= time 6.5))
(instance p2 (= time 8.0))
```

Certain day times can be modeled as defined concepts:

```
(define-concept early-morning-time (and (<= 6.0 time) (< time 7.0)))
```

Note that `p1` is an instance of `early-morning-time` then. We also want to be able to reason about the *relative locations* of time points to one another, e.g., we want to know whether `p1` is before of after `p2`. A mapping to qualitative relationships such as `before-point-in-time` and `after-point-in-time` is thus needed. In RACERPRO we can use *defined* NRQL *queries* or NRQL *ABox rules* to establish such a mapping:

```
(defquery before-point-in-time (?s1 ?s2)
          (and (?s1 point-in-time) (?s2 point-in-time)
               (?s1 ?s2 (constraint time time <))))
```

A `defquery` form can be understood as a simple macro which can be used in nRQL queries such as `(retrieve (?x ?y) (?x ?y before-point-in-time))`, which then returns `(((?x p1) (?y p2)))`. But in order to make ABox reasoning aware of the qualitative relationship holding between `p1` and `p2`, we must add an `before-point-in-time` *role assertion* to the ABox. This can be done with an ABox rule:

```
(firerule (?x ?y before-point-in-time)
    ((related ?x ?x before-point-in-time-role)))
```

This rule fires and adds a `(related p1 p2 before-point-in-time-role)` assertion to the ABox. Due to the added role assertion, `p1` can now for example be recognized as an instance of the concept

```
(define-concept has-successor-point
        (and point-in-time (some before-point-in-time-role point-in-time)))
```

Having modeled time points, we can continue defining *intervals* which have a start and end time point; moreover, the intervals duration shall be greater than zero:

```
(define-concrete-domain-attribute start-time :type real)
(define-concrete-domain-attribute   end-time :type real)
(define-concept interval (and (a start-time) (a end-time)
                              (< start-time end-time)))
```

Given this definition of interval, it is even possible to classify / recognize events as *short or long intervals*, again by means of the expressive concrete domain reasoning offered by RacerPro:

```
(define-concept short-interval
                (and interval (< end-time (+ start-time 1.0))))
```

A short interval is thus an interval that lasts at most one hour; note that `(< end-time (+ start-time 1.0))` is satisfied iff $end\_time - start\_time < 1$; this equation cannot be expressed in a more direct way in RacerPro.

The *point in interval* relationship is an important qualitative relationship. It can be modeled as a defined query as follows:

```
(defquery point-in-time-inside-interval (?s ?e)
          (and (?s point-in-time) (?e interval)
               (?e ?s (constraint start-time time <=))
               (?s ?e (constraint time end-time <=))))
```

It is now reasonable to define *certain special day times* as *interval individuals*, e.g., like `morning-hours`. The rationale is that these intervals can be used in queries such as *What happened during the morning hours?*:

```
(instance early-morning-interval
          (and interval (= start-time 6.0) (= end-time 7.0)))
```

Moreover, the famous *Allen temporal relationships* [7] provide well-known qualitative temporal relational vocabulary for intervals (meets, overlaps, during, …). Like the point in interval relation, the Allen relations can be defined as queries. If required, corresponding ABox rules can again add *Allen role assertions* to the ABox so that further reasoning processes are aware of the qualitative temporal relationships holding between the intervals. The `meets` relationship between intervals looks as follows:

```
(defquery meets (?e1 ?e2)
                (and (?e1 interval) (?e2 interval)
                     (?e1 ?e2 (constraint end-time start-time =))))
```

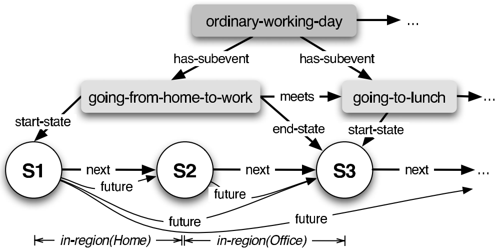

Fig. 1. States (Circles), Simple Events (Light Gray), and Complex Events (Gray)

From Time Points to States and Histories A *state* is a user/agent-specific description of the user's current situations as well as of its relevant (spatial, social, . . .) environment at a given time point. States will be generated by the Situation Description Generator. Every time point that *has some agent* associated with it is called *a state of that agent*:

```
(define-concept state (and point-in-time (some has-agent agent)))
```

An arbitrary amount of additional context information can be "attached" to a state individual; for example, information regarding the current location for which we are using the `in-region` role. Regions can be cluster regions annotated with location concepts, but also annotated map regions, points of interest etc:

```
(define-primitive-role in-region :domain state :range region)
```

A *sequence of states of an agent* is called a *history*. Like in a modal temporal logic based on a discrete linear model of time, we are are introducing a functional role `next` to reference the successor state. The inverse of next is called `previous`; `next` has a transitive super-role called `future` which can thus be used to access all future states from the current state. Obviously, `past` is the inverse of `future`.

Since the state individuals of the agents are generated by the Situation Description Generator, the generator can as well create the required `(related s1 s2 next)` role assertions to produce the time thread. However, since the quantitative time information is available, the required *next role assertions* can also be created with an ABox rule:

```
(firerule (and (?s1 state) (?s1 ?a has-agent)
               (?s2 state) (?s2 ?a has-agent)
               (?s1 ?s2 before-point-in-time)
               (neg (project-to (?s1 ?s2 ?a)
                      (and (?s1 ?s before-point-in-time)
                           (?s ?s2 before-point-in-time)
                           (?s ?a has-agent)))))
    ((related ?s1 ?s2 next)))
```

The variables `?s1` and `?s2` will be bound to states of the same agent `?a`. Moreover, `?s1` precedes `?s2` in time. We also have to verify that `?s2` is the *direct* successor of `?s1`. This means that there is no state `?s` *in between* `?s1` and `?s2` of that same agent. This is verified with the expression `(neg (project-to ...))`. If satisfying `?s1, ?s2` bindings are found, the rule adds a `(related ?s1 ?s2 next)` assertion to the ABox.

From Histories to Events Now we have an ABox containing all the histories of the agents. Events shall now be recognized on agent histories. An example history of an agent on which events have been recognized is shown in Fig. 1.

An event *is a time interval* having a start state s_1 and an end state s_2. An event either describes a *constancy holding between* s_1 *and* s_2, e.g., like *staying at home*, or a *certain change that happened between* s_1 *and* s_2, e.g. *going from home to work.* The former events are called *homogeneous events*. Such an event has the property that the described constancy does not only hold for the whole event, but also for all its subevents. Moreover, such events shall often be of *maximum length*, i.e., there shall be no proper subintervals which also satisfy the event property. In contrast, the events describing changes are often called *Gestalt events*, if they may *not* have subevents for which the property also holds. Thus, such events shall be of *minimum length*. We will show how these requirements (whose modeling would require an *until* modality in temporal modal logics) can be formalized in NRQL.

We distinguish *generic or non-thematic events* and *thematic events*. A thematic event requires background knowledge (e.g., social reasoning) in order to be recognized. For example, the event *staying in a region* is a (homogeneous) generic event, whereas the *staying at home* event is a thematic event. An additional discriminator is given by the distinction of *simple vs. complex events*, as illustrated in Fig. 2. Simple events have no subevents, whereas complex events have. The required relationships of the subevents to one another are specified with the help of Allen relations.

Events describing constancies can only be recognized if states are also automatically generated even if the situation description has *not changed*, but simply time has gone by. Thus, if a significant change of the value of the time attribute is considered as a relevant change in the situation description, then a new state will be generated automatically, so events describing constancies can be recognized from the similarity (non change) of attributes between situations. However, this also reveals the question of *how frequent* new states shall be constructed. We do not answer this question here.

Given the structure of the history ABoxes, the next important question to ask is: How to recognize the events? As a first idea, we can try to identify events with their *start states* and then exploit the temporal structure spawned by the *next* and *future* relationships. Thus, a *leaving home event* could be recognized with the following concept definition:

```
(define-concept leaving-home-event
     (and state (some in-region home) (some next (all in-region (not home)))))
```

However, due to the Open World Semantics [3, pp. 68] employed by DLs, we see that `(all in-region (not home))` can only be proven if appropriate closure assertions are added on the `next` successor's `in-region` role. Moreover, it does not seem to be adequate to identify events which have a certain duration and are thus conceptually intervals with their start states which are conceptually time points. Also, there is no way to access or refer to the duration of such an event, since role quantification on `next` and `future` can only *see* the require future states, but cannot *fix* them. Thus, *variables* are needed. Moreover, while / until operators known from temporal modal logics would be needed in order to express that an event has maximum or minimum length. Also, a concept such as `home` depends on the agent and thus cannot be used if more than one agent individual is present. Thus, we have to verify that the region is indeed the home *of the agent*.

We are thus defining events with the help of rules again. Events are instances of an event concept and reference their start and end states with the roles `start-state` and end-state. In case of a complex event, the subevents are aggregated using the has-subevent role. These events thus satisfy

```
(define-concept event
                (and interval (some has-agent agent)
                     (some start-state state) (some end-state state)))
```

Event rules have to construct *new individuals*. So-called *DL-safe rules* are rules whose variables only range over ABox individuals, i.e., all variables are *distinguished*. This is always the case in nRQL. However, since nRQL allows the creation of new individuals with rules we need to be careful, since rules may be applied to freshly created individuals as well. In order to avoid termination problems, nRQL does not offer an automatic rule application strategy; instead, API functions function are supplied to first identify the applicable rules, and then to fire (all or some of) them. This is called a single *rule application cycle*. In principle it is unclear how many cycles will be needed. Thus, the application runs a loop. To ensure termination, we make the antecedences of the rules *non-monotonic* such that a rule can only be fired once for a certain set of input individuals. The *general pattern / idiom for simple event rules* thus looks as follows:

```
(prepare-abox-rule
     (and (?s1 state) (?s2 state)
          (?s1 ?a has-agent) (?s2 ?a has-agent)
          (?s1 ?s2 next) // or future or past or previous
          ... // some more conditions on the states
          // ensure that rule can only be fired "once":
          (neg (project-to (?s1 ?s2)
                 (and (?e some-simple-event)
                      (?e ?s1 start-state) (?e ?s2 end-state)))))
  ((instance (new-ind new-simple-event ?s1 ?s2) some-simple-event)
   (related  (new-ind new-simple-event ?s1 ?s2) ?s1 start-state)
   (related  (new-ind new-simple-event ?s1 ?s2) ?s2 end-state)))
```

If the antecedence of the rule identified appropriate start and end states ?s1 and ?s2 in the same history (belonging to the same agent ?a), and such an event has not already been constructed, then a new event instance referencing ?s1, ?s2 is created. The new-ind operator is used to construct a new ABox individual; if ?s1 is bound to s1 and ?s2 to s2, then the expression (new-ind new-simple-event ?s1 ?s2) creates a new individual new-simple-event-s1-s2.

Using this pattern, we can define homogeneous and gestalt *generic simple events.* For example, we have the following spatial events: *leaving a region, entering a region, staying in a region,* and the *in no region event.* By means of *blutooth devices and buddy lists,* it can also be recognized if a buddy is close by. We thus also have the *meeting buddy, leaving buddy, staying in company of a buddy* as well as the *being alone event.*

It is obvious that the *leaving and entering a region events* are easy to model with ABox rules as follows: the general event rule pattern is used, but additional constraints on the states are imposed, for example, (?s1 ?r in-region) and (neg (?s2 ?r in-region)) in case of the leaving a region event, and vice versa for the entering a region event. Maximum duration and homogeneity of events are harder to enforce. Let us consider the staying-in-region event. Assume that ?r is the region in which the agent is currently staying. To enforce maximum duration of the interval to the left, we require that ?s1 does not have a previous state which is also contained in ?r, and similar for ?s2 and next. Homogeneity can be expressed as well – between ?s1 and ?s2 there shall be no states ?s3 in which (?s3 ?r in-region) does *not* hold. This gives us the additional conjuncts:

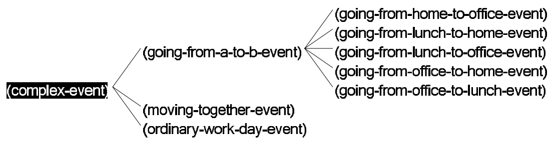

Fig. 2. Taxonomy of Complex Thematic Events

```
(and (neg (project-to (?s3 ?r)
              (and (?s3 ?s4 next) (?s4 ?r in-region))))
     (neg (project-to (?s2 ?r)
              (and (?s12 ?s2 next) (?s12 ?r in-region))))
     (neg (project-to (?s2 ?s3 ?r)
              (and (?s2 ?sx future) (?sx ?s3 future)
                   (neg (?sx ?r in-region))))))
```

The analog social event, *staying in company of a buddy,* is even more complicated, since here one has to relate states of histories of two different agents in order to detect the constancy; note that the `in-buddy-proximity` relation holds between *states* of agents.

Having recognized the simple generic events, we can specialize these to *thematic events,* for example, a `leaving-home-event` is a special `leaving-region-event`. In some cases, simple concept definitions are sufficient for recognition, but in other cases, rules are needed again.

Complex and Very High Level Events We then continue and define *complex event* that consist of several subevents. As with the simple events, we distinguish *generic and thematic complex events.* An important generic complex event is the *going from A to B event.* This event is neither maximal nor homogeneous; instead, it is well known that going from A to B eventually means that one fist has to go from A to C, and then from C to B. Such recursive event rules can become very complex.

So, what are reasonable complex *thematic* events in our case study scenario? Given the *typical working day scenario,* we primarily consider further specializations of the *going from A to B event* which takes the thematic types of the origin and destination regions into account. For example, a *going from office to lunch event* is recognized if the destination region is a restaurant, and if the source region is the work office of the agent. Moreover, such an event has to overlap the *lunch time* individual interval. The introduced complex thematic events are visualized in Fig. 2.

Finally, we can define *very high level complex events.* An *ordinary working day event* is assumed to consist of the following consecutive *sequence of events:* `going-from-home-to-office-event, working-event, going-from-office-to-lunch-event, lunch-event, going-from-lunch-to-office-event, working-event, going-from-office-to-home-event`. If such a sequence of events `?e1` to `?e7` is found, all belonging to the same agent, such that `(?e[n] ?e[n+1] meets)` holds for all `n` from 1 to 6, then a complex event of type `ordinary-working-day-event` is constructed, and the seven subevents are connected to it using the `has-subevent` role.

4 A Complex Example

A complex history ABox is visualized in Fig. 3. The histories of the three agents `A1, A2, A3` are shown. Circles denote states, and containment of a state in a region (the

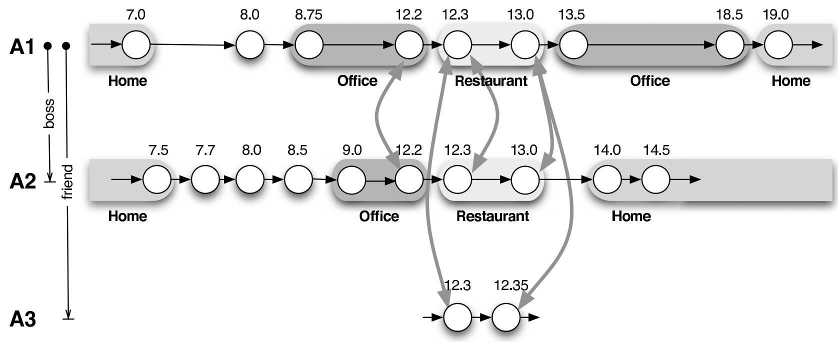

Fig. 3. History ABox for Agents `A1, A2, A3`

`in-region` relationship) is depicted with the help of the state enclosing gray shaded boxes (visualizing the regions). The `home` regions of `A1` and `A2` are different, but the `restaurant` and `office` boxes visualize the same region. The bold gray arrows visualize the `in-buddy-proximity` relationship. The values of the `time` attributes are shown as well (in decimal coding).

The scenario modeled in the example ABox goes as follows: Agent `A2` is the boss of `A1`, and `A3` is a friend of `A1`. `A1` leaves its home at 7.0 and enters his office at 8.75; in the meantime he was *on the road.* He stayed in the office (presumably working) until 12.20 (we are omitting the concrete times in the remaining description). He is then in buddy proximity with his boss. Both are leaving the office together and are entering the restaurant, where they are having lunch. In the restaurant, `A1` meets a friend (`A3`) for a couple of minutes. After staying a while in the restaurant, `A1` and `A2` are leaving the restaurant together. `A1` goes back to office and stays there until he leaves the office in the evening, heading towards home. In contrast, `A2` goes home after lunch.

This history ABox induces a complex event structure. After the rules no more apply, the following complex thematic events have been constructed for `A1`. The following list is the result of a NRQL query; for each binding `?x` to a complex thematic event we are also including its start and end time as well as its *most specific types.* For the events, a `<event-name><start-state><end-state>` naming schema is used, and `<state-number><agent>` for the states:

```
(((?x ordinary-working-day-s1a1-s9a1) (7.0) (19.0)
     (ordinary-working-day-event long-interval))

 ((?x going-from-a-to-b-s1a1-s3a1) (7.0) (8.75)
     (going-from-home-to-office-event region-event short-interval))

 ((?x going-from-a-to-b-s4a1-s5a1) (12.2) (12.3)
     (going-from-office-to-lunch-event moving-together-event region-event))

 ((?x going-from-a-to-b-s6a1-s7a1) (13.0) (13.5)
     (going-from-lunch-to-office-event region-event))

 ((?x going-from-a-to-b-s8a1-s9a1) (18.5) (19.0)
     (going-from-office-to-home-event region-event)))
```

Thus, as expected, `A1` has experienced an arbitrary working day. However, this is neither the case for `A2` nor for `A3`. Note that some more events have been recognized, but they are not “complex”, e.g., the events `staying-in-region-of-office-of-a1-s3a1-s4a1` from

8.75 to 12.2 of type `(working-event long-interval)`, `in-company-a3-s5a1-s6a1` from 12.3 to 13.0 of type `(in-company-event meeting-friend-event))` (which is a "social event" due to the bluetooth proximity with friend `A3`), and `staying-in-region-of-restaurant-1-s5a1-s6a1` from 12.3 to 13.0 of type `(lunch-with-boss-event)`.

5 Conclusion & Future Work

We have proposed a practical and working event model methodology in the RACERPRO DL system. The long term research goal of this work is to enhance the spatial, temporal and dynamic awareness of the ContextWatcher application framework. The principle feasibility of the approach has been demonstrated with a case study. A drawback of the proposed model is the slightly non-declarative semantics shown by some rules, especially those that create new individuals. Recently it has been shown that *abduction* – which is a non-deductive inference process – has the potential to deliver hypotheses and can thus also be used to hypothesize the assertions which we have constructed simply by means of rules [8,9]. How to apply this abduction framework is future work.

It should be stressed that the proposed model only works with RACERPRO, since current W3C Semantic Web standards (OWL, SPARQL, SWRL etc.) do not offer the required expressivity for the formulation of rules (e.g., negation as failure, closed domain universal quantification, creation of new individuals, concrete domain reasoning). It is clear that corresponding concept constructors easily lead to undecidability. But *pragmatic solutions* have to be developed for practical applications, as we have demonstrated.

References

1. Koolwaaij, J., Tarlano, A., Luther, M., et al.: ContextWatcher – Sharing Context Information in Everyday Life. In: Proc. of the IASTED Int. Conference on Web Technologies, Applications, and Services (WTAS'06). (2006)
2. Luther, M., Fukazawa, Y., Wagner, M., Kurakake, S.: Situational Reasoning for Task-oriented Mobile Service Recommendation. The Knowledge Engineering Review **Special Issue on Contexts and Ontologies: Theory, Practice and Applications** (2007) To Appear.
3. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P., eds.: The Description Logic Handbook – Theory, Implemenation and Applications. Cambridge University Press (2003)
4. Wessel, M., Möller, R.: A High Performance Semantic Web Query Answering Engine. In: Proc. Int. Workshop on Description Logics (DL '05). (2005)
5. McCarthy, J., Hayes, P.J.: Some Philosophical Problems from the Standpoint of Artificial Intelligence. In Meltzer, B., Michie, D., eds.: Machine Intelligence 4. Edinburgh University Press (1969) 463–502
6. Kowalski, R., Sergot, M.: A Logic-based Calculus of Events. New Gen. Computing **4** (1986)
7. Allen, J.F.: Maintaining Knowledge about Temporal Intervals. Communications of the ACM **26** (1983) 832–843
8. Espinosa, S., Kaya, A., Melzer, S., Möller, R., Wessel, M.: Towards a Foundation for Knowledge Management: Multimedia Interpretation as Abduction. In: Proc. Int. Workshop on Description Logics (DL '07). (2007)
9. Möller, R., Neumann, B.: Ontology-based Reasoning Techniques for Multi-Media Interpreation and Retrieval. In Hobbson, P., Kompatsiaris, Y., eds.: Semantic Multi-Media and Ontolgies: Theory and Applications. (2007) To appear.

Modeling Adaptive Behavior with Conceptual Spaces

Michael Cebulla

Technische Universität Berlin, Fakultät für Elektrotechnik und Informatik,
Institut für Softwaretechnik und Theoretische Informatik,
Franklinstr. 28/29, 10587 Berlin
mce@cs.tu-berlin.de

Abstract. We discuss a membrane-based calculus for the combination of conceptual spaces during runtime. Since our goal is to support emergent properties of behavior (and due to the fact that it is not possible to define a complete calculus for all situations) we introduce the notion of self-modification. Terms from situational description can evolve according to simple rules thus providing various possibilities for reactions.

1 Introduction

In this paper some of the problems which are connected with context-dependent behavior are developed as a general problem of combining conceptual spaces. Examples from the field of medical workflows are given. The approach which is discussed in this paper relies on the notion of conceptual integration which was developed in the field of cognitive semantics [2]. This integration can be simulated by the rule-based integration of ontologies.

The framework described in this paper partly follows previous proposals for the formalization of contextual reasoning. While classifications [1] heavily rely on universal algebra or category theory [3] we focus on an operational treatment using membrane computing [4].

In this paper we make an attempt to bridge the gap between highly reactive behavior during runtime and the need for highly abstract and meaningful concepts for context-awareness. Especially we propose to integrate highly abstract forms of common sense reasoning (as proposed by [1]) with membrane computing (as proposed by [4]) in order to support a way of runtime reasoning whose robustness is comparable to human reasoning. By this proposal we extend previous suggestions concerning high-level and intuitive specifications (cf. [5]). Especially we propose to exploit common sense reasoning for the robustness of context-aware behavior in distributed systems.

2 Context and Behavior

Simplified models of medical workflows are employed in this paper as examples for the treatment of adaptive behavior.

Example 1 (Intubation: A Medical Workflow) *The activity of intubation is considered with represents a specific part of a medical operation. Although there is certainly a definition of the process (i.e. a pattern) the exact shape of the final activity highly* *depends on the context in which this pattern is activated. In this paper we propose to represent the definition of the process (the pattern) as well as the situation as input spaces (in the sense of [2]). We develop an emergent calculus which establishes links between these spaces.* ♢

The Intubation Space. The conceptual space containing the process of intubation contains a constraint-based description of this process (cf. Figure 2). The important actions are described with their causal relationships as well as constraints which have to hold in certain states. Especially three subtasks can be identified *(Preparation, Laryngoscopy and Introduction)* which have different relevance values for the overall process. Since the intubation space contains a pattern which is described in this space there are many variables which have to be bound to actual values from a specific situation. For instance, agents are represented by *roles* which have to be bound to real agents taken from another conceptual space. In a similar way constraints which are specified over objects or states are applied to elements from other spaces.

The Situation Space. While the intubation space contains *roles* for the agents which are responsible for certain actions the situation space is populated by (entities representing) real agents and resources. In addition in this conceptual space specific relations and circumstances can be described which are of informal nature but which heavily influence the shape of the resulting process. As an example a relation of informal hierarchy is given which may hold between an experienced nurse and a less experienced anesthesist.

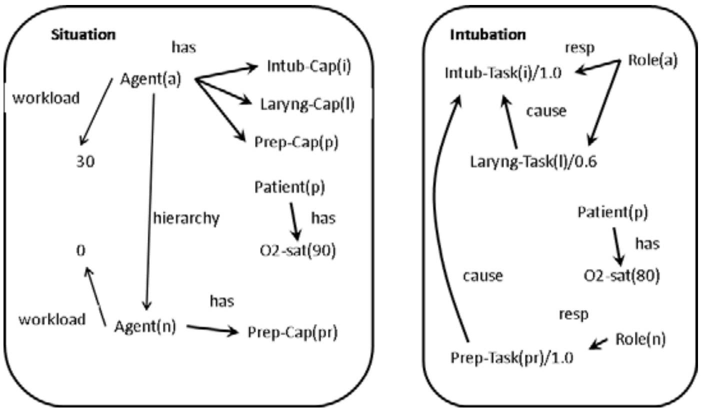

Fig. 1. Conceptual Spaces

Cross-Space Mapping. The combination of conceptual spaces is triggered by cross-space mappings. Cross-space mappings are enabled by morphisms between ontologies. Morphisms represent background knowledge for combining conceptual spaces. In our example relevant morphisms are:

```
mapping intsit from Intubation to Situation
   sort Intubation-Task Intubation-Capability
```

As we will see the background knowledge is used to establish infomorphisms between the conceptual spaces. As we will see there are multiple possibilities to establish these infomorphisms. One of our main points consist in the claim that the adaptive or self-configuring capabilities of complex systems (like human teams in the operation theatre) can be simulated by an adequate selection of the best possibility.

Generally the process of blending results in the creation of a blended space. Due to space restrictions we concentrate on the establishment of vital relations in this paper.

3 Operational Treatment of Vital Relations

Under operational aspects we represent conceptual spaces as P-systems P_{CS}. Basically a conceptual space is enclosed by a membrane. These entities which are contained in the space are mapped to components of P-systems (cf. [4]). While concepts are mapped to molecules, individuals and relations are mapped to labeled membranes.

Definition 1 (P-System P_{CS}) *The P-System for the representation of conceptual spaces P_{CS} is defined by the tuple $\langle V, L, \mu, w_i, Rn_i \rangle$ where V is the terminology of the classifications (containing concept names) and the label algebra L (containing individuals, situations and the concatenation operator ","). μ is the structure of membranes containing the multi-fuzzy sets w_i. The rules contained in Rn_i are discussed below.* □

One of our central goals is to support self-organizational capabilities in the dynamic composition of conceptual spaces. This is especially due to the fact that it is impossible to foresee every possible combination of situations. Since we do not want to define a uniform rigid calculus which is restricted to a certain set of known combinations we take the opposite approach which promises a more flexible solution. This means that we allow the terms to evolve in a solution and to look for possible combinations by themselves. This decentralized approach is robust against local evolutions and to unforeseen changes.

We proceed in two steps. Firstly we have to map context descriptions to membrane structures. This can be easily done by mapping individuals and situations to labeled membranes and concepts to molecules floating in a solution. In the same way we have to represent ontology morphisms by membrane structures. In a second step we give the rules for the evolution of these structures and for the establishment of valid combinations of contexts.

Airlock Rules. In our membrane-based approach molecules are enclosed by membranes. In order to make reactions possible however they have be able to leave their membranes. This is defined by the airlock rule. We introduce an extended version (EAL) which enables molecules to cross multiple membranes.

$$\begin{array}{ll} \text{(AL)} & [_a C_1]_a \leftrightharpoons C_1 \lhd_{\langle a \rangle} [_a]_a \\ \text{(EAL)} & [_b C_1 \lhd_{\langle L,a \rangle} [_a]_a]_b \leftrightharpoons C_1 \lhd_{\langle L,a,b \rangle} [_b[_a]_a]_b \end{array}$$

Intuitively we enable the molecules to travel through the membrane structure keeping track of the membranes they crossed in a list which is an annotation of the airlock-operator.

Interaction. The main goal is to find and encourage possible interactions. Especially molecules from situations should react with molecules from morphisms. Such reactions are only possible because both situations and morphisms evolve according to the airlock rules. The rule for interaction can be given as follows.

$$\text{(INT)} \qquad C_1^+ \lhd_{\langle src,mor\rangle} [_{mor}[_{src}]_{src}]_{mor}, C_1^- \lhd_{\langle a,s1\rangle} [_{s1}[_{a}]_{a}]_{s1} \rightarrow$$
$$[_{\langle mor,s1\rangle}[_{\langle src,s1,a\rangle}C_1]_{\langle src,s1,a\rangle}]_{\langle mor,s1\rangle}[_{\langle s1,mor\rangle}[_{\langle a,mor\rangle}C_1]_{\langle a,mor\rangle}]_{\langle s1,mor\rangle}$$

Intuitively the reaction between the molecules is recorded by the labels. Thus the labels of the morphism are added to the labels of the situation. In the same way the labels of the morphism are extended. Note that we only treat the matching of the source ontology of the morphism. We presume that the molecules of the morphism are charged negatively while the molecules of the situation are charged positively. Note that there can be many different reactions between situations and morphisms because many copies of the structures are floating in the solution.

Completing the Infomorphism. An ontology morphism is completely bound when two molecules from two context description have been bound to its source and target ports. Since the knowledge about the creation of the bindings is contained in the labels the information is present which which completes an infomorphism (i.e. the relation between the individuals which is contravariant to the original ontology morphism).

Compression. Elements from the input spaces which are connected by a vital relation (i.e. infomorphisms) are projected into the blend. The resulting individuals which are created in the blend can be considered a tuple-valued individuals which establish a connection between the original tokens (or es equivalence classes). We cannot deepen these issues due to space restrictions.

4 Outlook

We discussed a membrane-based calculus for the creation of infomorphisms between conceptual spaces during runtime. We consider this line of research as a contribution to the exploration of adaptive and context-aware behavior in distributed systems. The treatment of infomorphisms is the strategic foundation for the integration of more advanced formal constructs from common sense reasoning.

References

1. J. Barwise and J. Seligman. *Information Flow: the logic of distributed Systems.* Cambridge Univ. Press, 1997.
2. G. Fauconnier and M. Turner. *The Way we Think.* Basic Books, 2002.
3. J. Goguen. Information integration in institutions. In *Memorial Volume for Jon Barwise.* Indiana University Press, Indiana, 2005.
4. G. Păun. *Membrane Computing. An Introduction.* Springer, Berlin, 2002.
5. P. Pepper, M. Cebulla, K. Didrich, and W. Grieskamp. From program languages to software languages. *The Journal of Systems and Software*, 60, 2002.

Opinion nets for reasoning with uncertain context information

Yves Vanrompay and Yolande Berbers

Department of Computer Science
Katholieke Universiteit Leuven
Belgium

Abstract. Context-aware systems must be able to deal with uncertain context information. We propose a generic context architecture and representation that incorporates the uncertainty of context elements in terms of upper and lower bounds of probabilities. It is shown how opinion nets can be used to reason with these upper and lower bound probabilities. In this way it is possible to combine ambiguous or conflicting context information that comes from different sources. Moreover, information coming from different sources can be combined with experience learned from the past in a clean way.

1 Introduction

Pervasive systems that can adapt to changing environments and availability of resources must be aware of their context. These systems sense and react to context. Most systems make the assumption that the context they use is completely accurate. However, the information about context may not come from a reliable source, may be out dated, not available or may be erroneous. Firstly, a context-aware system senses its context via a network of sensors working together. The resolutions, accuracies and formats of these sensors can differ from each other. The resulting sensed values can have conflicts and ambiguities. The second cause of uncertainty are the current limitations of the underlying reasoning systems that deduce high-level context information from low-level sensor data. Lastly, due to the asynchronicity of context acquisition and use of context we must deal with the imperfection and aging of the context information. A challenge for the development of real-life and commercial context-aware systems is therefore the ability to handle uncertain and ambiguous context information.

We propose a generic context architecture consisting of context synthesizers, providers and consumers. Context elements are represented as predicates, with which are associated upper and lower bound probabilities. Then opinion nets are used to reason with these probabilities. If the context comes from different sources contradictions and ambiguities can arise. It is shown how opinion nets can resolve conflicts and ambiguities by combining several probabilistic inputs to a single output.

This paper is organized as follows. In section 2 the context architecture and representation are described. In section 3 we introduce opinion nets and show

how they can be of great value to deal with uncertainty in context-aware systems. Section 4 gives an overview of related work and compares it with the presented approach. Finally, section 5 gives an outline of future work and draws conclusions.

2 Context architecture and representation

The context architecture is a generic infrastructure inspired by Gaia [2]. It supports gathering contextual information from sensors, inferring higher level context and delivering context information to the correct entities. A context provider provides context information in a synchronous way. A context consumer or context synthesizer can invoke the provider in order to acquire information about context. A context synthesizer is an aggregator of context information from different sources. A context consumer is an entity that needs context data. A context consumer can retrieve context information by sending a request to the context provider. Every component can play more than one role. A component can be a context provider if it provides context data about a specific domain and at the same time a consumer if the component also needs data from other domains.

The general uncertainty model is based on predicates representing context elements or facts with associated confidence values. The predicate name describes the context element. The arguments are mostly of the form subject-object or subject-verb-object, e.g. *location(John, in, room3, lower, upper)* or *activity(room7, conference, lower, upper)*. The confidence values of the predicates are expressed as upper and lower bounds of probabilities. Alternatively we could express confidence values as a probabilistic value together with an accuracy measurement of the probability. We will call the confidence value an opinion.

3 Opinion nets

In a simple approach we could work with a singular probabilistic value to indicate the frequency that a predicate is true. However, in opinion nets [1] , each opinion is translated into a range of probability numbers. That range is specified as an upper and a lower bound on the probability of the predicate to be true.

Opinions coming from different sources can be tied together in several ways. The different sources could for example be one or more context providers and context synthesizers. The combining of the inputs to one output can be done in a context provider, synthesizer or consumer, depending on the requirements and structure of the application. The inputs of a context provider or synthesizer can also be put together with historical information that is learned from the past. In that way history can be taken into account and easily incorporated in the opinion net approach.

The advantage of working with upper and lower bounds is that we can work with imprecise probabilities. When there is not enough information to give an exact probability, but if the system knows enough to say that the probability is definitely between 0.4 and 0.7, we can capture this with opinions. Also, ambiguous information can be presented in a straightforward way. Lastly, conflicting information that comes from different sources can be combined in a clean way by using opinion nets.

Figure 1 shows the different opinions and how they are put together, for clarity only for the lower bounds. The opinions can be propagated through opinion nets. The boxes are called constraint boxes, and they can be *and* or *or* boxes. The following constraint equations govern the action of the or boxes. A and B represent inputs and A *or* B represents an output. Then, *l(A)*, *l(B)* and *l(A or B)* are the lower bounds of the probabilities. Similarly *u(A)*, *u(B)* and *u(A or B)* represent the upper bounds of the probabilities.

$$
\begin{aligned}
u(A) &\leq u(AorB) \\
l(A) &\geq l(AorB) - u(B) \\
u(B) &\leq u(AorB) \\
l(B) &\geq l(AorB) - u(A) \\
u(AorB) &\leq u(A) + u(B) \\
l(AorB) &\geq max[l(A), l(B)]
\end{aligned}
\tag{1}
$$

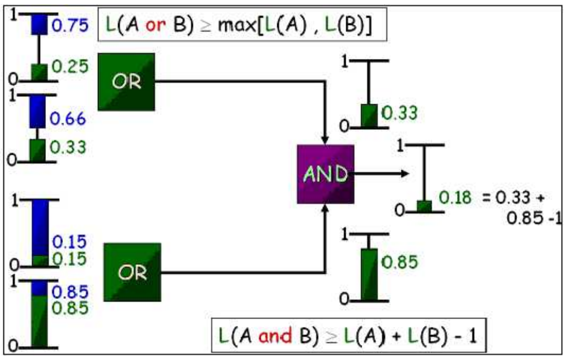

Fig. 1. Forward propagation of lower bounds in an opinion net

The equations for the *and* operator are similar. The combination of these assertions and boxes is an opinion net. An opinion net is thus a numeric constraint net in which it is possible to keep track of a conclusion's probability.

4 Related work

There exist several approaches for dealing with uncertain, ambiguous and inconsistent context information. The Integrated Context Model proposed by Truong

[3] allows to construct a Bayesian Network for reasoning with context information. Although this technique deals with uncertain context information, there is no straightforward way to combine conflicting and ambiguous information that comes from different sources, which is possible with opinion nets.

Gaia [2] is a prototype pervasive computing middleware system that allows to reason about uncertainty. Several mechanisms like probabilistic logic, fuzzy logic and Bayesian Networks can be plugged in. Context information is represented as predicates. Each predicate is described as a class in an ontology defined in DAML+OIL. A confidence value between 0 and 1 is attached to a predicate. Since our context architecture and representation is based on Gaia, opinion nets could be plugged in as a reasoning mechanism in Gaia.

Several techniques can be used to deal with imperfect context. However, Dey and Mankoff [4] argue that in realistic scenarios not all ambiguity in the data can be removed. Moreover, certain human aspects of context cannot be sensed or inferred by technological means. Their proposal is to involve end users in removing the remaining ambiguities through a process called mediation. The uncertainty inherent to the context information is explicitly presented to the user. Mediation can be fitted well in our approach. The opinion of the user can be treated as an opinion next to that of the system or next to an opinion inferred from experience learned from the past. Then these opinions can be combined using opinion nets.

5 Future work and Conclusions

To validate our approach a simulation environment will be developed. Experiments with context information coming from different sources have to be carried out. A test scenario can be the introduction of a vague concept like proximity as a context element. Context-aware systems such as a portable touristic city guide are location-aware and can suggest a tourist to visit a touristic attraction that is nearby. Proximity however is a subjective measure of distance depending on the context of the tourist. How close an attraction is depends amongst others on whether the tourist is on foot or by car, what his mood is, whether he is really interested to see the attraction and so on. Opinions concerning the proximity of a location that come from different sources can be combined by an opinion net to a single output of a probability range.

Based on a context architecture and representation, we proposed the use of opinion nets to deal with uncertain, ambiguous and conflicting context information. This approach allows to resolve ambiguities and conflicts arising from information that comes from different sources in a natural way. Moreover, the reasoning mechanism with opinion nets is general in the sense that it can be plugged in into every context system that represents information with probabilities and accuracies. We believe that further research will show the usefullness of opinion nets for reasoning with uncertain context information.

References

1. Patrick Henry Winston. Artificial Intelligence. Addison-Wesley, third edition, 1993 234–241
2. M. Roman et al.: A Middleware Infrastructure for Active Spaces. IEEE Pervasive Computing **1-4** (2002) 74–83
3. B. A. Truong, Y.-K. Lee, S.-Y. Lee: Modeling Uncertainty in Context-aware Computing. 4th Annual ACIS International Conference on Computer and Information Science (ICIS'05)
4. A. Dey, J. Mankoff: Designing Mediation for Context-Aware Applications. ACM Transactions on Computer-Human Interaction, Vol. 12, No. 1, March 2006, 53–80

Ontologies as Contexts for Constraint-Based Reasoning

Richard J. Wallace, Tomas E. Nordlander, and Ioannis Dokas

Cork Constraint Computation Centre and Department of Computer Science
University College Cork, Cork, Ireland
email: {r.wallace,t.nordlander,i.dokas}@4c.ucc.ie

Abstract. We give a brief description of a system we are currently building that fulfils the promise of our title. This description also indicates why we are interested in the topic of contexts and ontologies.

1 Introduction

Many problems involve finding a solution subject to certain constraints. Constraint programming is an important technique for modelling such problems, which supports combinations of inference and search that allow more efficient solving. However, this cleverness is obtained at a cost, since these methods normally required highly skilled specialists as well as a lengthy period of time for modelling and crafting of solvers.

The question we are interested in here is whether these methods can be embedded so that a layperson using such a system can make use of these methods without having to deal with them directly. In this case it is necessary that a constraint model - with at least a simple form - be built automatically, and that the solving process proceed without the user being aware of the particular kind of representation involved. What the user is aware of is the original problem, stated in everyday terms that he or she is familiar with.

In order to bring about this result, we are interesting in using ontologies to represent the problem in a simple, qualitative manner, which can also serve as the basis for deducing a more abstruse CSP model. The ontology, therefore, provides the context for constraint modelling and constraint-based reasoning. (In fact, it can provide many contexts as the following section will show.)

2 Description of an Ontology-and-Constraint Based System

We are presently trying to implement our ideas in a product selection system called the Matchmaker. Product selection is carried out using a form of interaction in which the system presents individual products and the user critiques them. (This is called a "suggestion/correction cycle".) Internally, the problem of product selection is represented as a constraint satisfaction problem, but with each critique the representation is updated to conform to the additional constraints revealed on this cycle. (This scheme is presented in more detail in [1].)

The Matchmaker program is written in Java, and Protege-OWL is being used to build the product ontology. In its final form, the system will be web-based and the system therefore falls under the heading of e-commerce.

Conceptually, a product is not only an item of interest but is also possibly an object, an artefact, an organisation, etc., and is associated with some type of setting, or even a specific location. In order to carry out the matchmaking process in a 'knowledgeable' manner, a product must be viewed in all of these ways. At the same time, there are restrictions on the degree of detail or the level of technical sophistication that can be handled efficiently; hopefully, there are also limits on what the user needs to know in order to make a satisfactory selection.

The resulting conceptualisation of a product can be thought of as a set of contexts in which the item in question is embedded. A straightforward and efficient way to represent these various contexts is through an ontology. In this case, if we start with a particular product type, we can reach associated concepts by means of a class hierarchy of the usual type, as well as deducing relevant relations ("properties" in the usual parlance) involving these classes. Our expectation is that such context-supported interaction will help us achieve the kind of interaction envisaged for the Matchmaker system in a sufficiently compelling fashion.

3 Ontologies to CSPs

Another major challenge necessitated by our approach is connecting the ontology to the constraint solver in a way that supports the computational requirements, while being transparent to the user. This is related to another motivation for using ontologies, which is to allow users to work with familiar concepts, without having to think in terms of the rather abstruse language of constraint programming and constraint satisfaction. In the implementation we are building, this is done by translating relevant ontology classes into variables, following rules that support the CSP formalism, in particular, the rule of single assignments to a variable.

An alternative approach, that we will consider in the future, is to use a CSP ontology to support the mapping between subject-matter concepts and the CSP representation. Some work along these lines has been done by other authors [2].

4 Future Vistas

The use of ontologies to provide conceptual contexts for constraint solving and to 'shield' users from constraint programming details is a general idea that may be useful in a number of application areas. Obvious candidates are systems for scheduling and configuration.

References

1. Freuder, E.C., Wallace, R.J.: Suggestion strategies for constraint-based matchmaker agents. International Journal on Artificial Intelligence Tools **11** (2002) 3–18
2. Sleeman, D., Chalmers, S.: Assisting domain experts to formulate and solve constraint satisfaction problems. In: Proc. 15th Internat. Conf. on Knowledge Engineering and Knowledge Management, LNCS No. 4248, Berlin, Springer (2006) 27–34