Workshop Notes

The IJCAI-09 Workshop on

Automated Reasoning about Context and Ontology Evolution

ARCOE-09

July 11-12, 2009
Pasadena, California, USA

held at the

International Joint Conference on Artificial Intelligence

Co-Chairs

Alan Bundy
Jos Lehmann
Guilin Qi
Ivan José Varzinczak

AAAI Press

Introduction to the Notes of the IJCAI-09 Workshop ARCOE-09

Automated Reasoning about Context and Ontology Evolution

Alan Bundy[a], **Jos Lehmann**[a], **Guilin Qi**[b], **Ivan José Varzinczak**[c]

[a] School of Informatics, University of Edinburgh
[b] Institute AIFB, Universitaet Karlsruhe
[c] Meraka Institute, Pretoria, South Africa

Methods of automated reasoning have solved a large number of problems in Computer Science by using formal ontologies expressed in logic. Over the years, though, each problem or class of problems has required a different ontology, and sometimes a different version of logic. Moreover, the processes of conceiving, controlling and maintaining an ontology and its versions have turned out to be inherently complex. All this has motivated much investigation in a wide range of disparate disciplines about how to relate ontologies to one another.

The IJCAI-09 Workshop ARCOE-09 brings together researchers and practitioners from core areas of Artificial Intelligence (e.g. Knowledge Representation and Reasoning, Contexts, and Ontologies) to discuss these kinds of problems and relevant results.

Historically, there have been at least three different, yet interdependent motivations behind this type of research: providing support to ontology engineers, especially in modeling Common Sense and Non-Monotonic Reasoning; defining the relationship between an ontology and its context; enhancing problem solving and communication for software agents, often by allowing for the evolution of the very basis of their ontologies or ontology languages.

ARCOE Call for Abstracts has been formulated against such historical background. Submissions to ARCOE-09 have been reviewed by two to three Chairs or PC members and ranked on relevance and quality. Approximately seventy-five percent of the submissions have been selected for presentation at the workshop and for inclusion in these Workshop Notes. Accordingly, the abstracts are here grouped in three sections and sequenced within each section by their order of submission.

Common Sense and Non-Monotonic Reasoning Ontology engineers are not supposed to succeed right from the beginning when (individually or collaboratively) developing an ontology. Despite their expertise and any assistance from domain experts, revision cycles are the rule. Research on the automation of the process of engineering an ontology has improved efficiency and reduced the introduction of unintended meanings by means of interactive ontology editors. Moreover, ontology matching has studied the process of manual, offline alignment of two or more known ontologies. The following works focus on the development of revision techniques for logics with limited expressivity.

Ribeiro, Wasserman. *AGM Revision in Description Logics*.

Wang, Wang, Topor. *Forgetting for Knowledge Bases in DL-Lite*$_{bool}$.

Moguillansky, Wassermann *Inconsistent-Tolerant DL-Lite Reasoning: An Argumentative Approach*.

Booth, Meyer, Varzinczak. *First Steps in $\mathcal{EL}$ Contraction*.

Context and Ontology Most application areas have recognized the need for representing and reasoning about knowledge that is distributed over many resources. Such knowledge depends on its context, i.e., on the syntactic and/or semantic structure of such resources. Research on information integration, distributed knowledge management, the semantic web, multi-agent and distributed reasoning have pinned down different aspects of how ontologies relate to and/or develop within their context. The following works concentrate on the relationship between contexts and ontologies.

Ptaszynski, Dybala, Shi, Rzepka, Araki. *Shifting Valence Helps Verify Contextual Appropriateness of Emotions*.

Kutz, Normann. *Context Discovery via Theory Interpretation*.

Redavid, Palmisano, Iannone. *Contextualized OWL-DL KB for the management of OWL-S effects*.

Sboui, Bédard, Brodeur, Badard *Modeling the External Quality of Context to Fine-tune Context Reasoning in Geo-spatial Interoperability*.

Qi, Ji, Haase. *A Conflict-based Operator for Mapping Revision*.

Automated Ontology Evolution Agents that communicate with one another without having full access to their respective ontologies or that are programmed to face new non-classifiable situations must change their own ontology dynamically at run-time – they cannot rely on human intervention. Research on this problem has either concentrated on non-monotonic reasoning and belief revision or on changes of signature, i.e., of the grammar of the ontology's language, with a minimal disruption to the original theory. The following works concentrate on Automated Ontology Evolution, an area which in recent years has been drawing the attention of Artificial Intelligence and Knowledge Representation and Reasoning.

Bundy. *Unite: A New Plan for Automated Ontology Evolution in Physics*.

Chan, Bundy. *An Architecture of GALILEO: A System for Automated Ontology Evolution in Physics*.

Lehmann. *A Case Study of Ontology Evolution in Atomic Physics as the Basis of the Open Structure Ontology Repair Plan*.

Jouis, Habib, Liu. *Atypicalities in Ontologies: Inferring New Facts from Topological Axioms*.

Thanks to the invaluable and much appreciated contributions of the Program Committee, the Invited Speakers and the authors, ARCOE-09 provides participants with an opportunity to position various approaches with respect to one another. Hopefully, though the workshop and these Notes will also start a process of cross-pollination and set out the constitution of a truly interdisciplinary research-community dedicated to automated reasoning about contexts and ontology evolution.

(Edinburgh, Karlsruhe, Pretoria – May 2009)

Table of Contents

Inconsistent-Tolerant *DL-Lite* Reasoning: An Argumentative Approach

Martín O. Moguillansky
Scientific Research's National Council (CONICET)
AI Research and Development Lab (LIDIA)
Department of Computer Science and Eng. (DCIC)
Universidad Nacional del Sur (UNS), ARGENTINA.
mom@cs.uns.edu.ar

Renata Wassermann
Department of Computer Science (DCC)
Institute of Mathematics and Statistics (IME)
University of São Paulo (USP), BRAZIL.
renata@ime.usp.br

1 Introduction

This article is devoted to the problem of reasoning over inconsistent ontologies expressed through the *description logic* DL-Lite [Calvanese *et al.*, 2007]. A specialized argumentation machinery is proposed through which DL-Lite *knowledge bases* (KB) may be reinterpreted *à la* argumentation. Such machinery arises from the reification to DL-Lite of the argumentation framework introduced in [Moguillansky *et al.*, 2008a], which in turn was formalized on top of the widely accepted *Dung's argumentation framework* [Dung, 1995]. A preliminary investigation to handle ontology debugging and consistent ontology evolution of $\mathcal{ALC}$ through argumentation was presented in [Moguillansky *et al.*, 2008a]. But this proposal aims at handling ontology evolution with no need of consistency restoration. Argumentation techniques applied to reasoning over ontologies appear as a promissory fusion to work in certain domains in which it is mandatory to avoid loosing any kind of knowledge disregarding inconsistencies. Therefore, we provide the theory for an inconsistent-tolerant argumentation DL-Lite reasoner, and finally the matter of ontology dynamics is reintroduced.

2 DL-Lite Brief Overview

Next we describe in a very brief manner the language *DL-Lite*$_\mathcal{A}$ used to represent DL-Lite knowledge. In the sequel we will write $\phi \in$ *DL-Lite*$_\mathcal{A}$, or $\Sigma \subseteq$ *DL-Lite*$_\mathcal{A}$, to identify a DL-Lite assertion ϕ, and a DL-Lite knowledge base Σ, respectively. For full details about DL-Lite please refer to [Calvanese *et al.*, 2007]. Consider the DL-Lite grammar:

$$B \longrightarrow A|\exists R \quad C \longrightarrow B|\neg B \quad R \longrightarrow P|P^- \quad E \longrightarrow R|\neg R$$

where A denotes an *atomic concept*, P an *atomic role*, P^- the *inverse* of the atomic role P, and B a *basic concept* that is either atomic or conforming to $\exists R$, where R denotes a *basic role* that is either atomic or its inverse. Finally, C denotes a *(general) concept*, whereas E denotes a *(general) role*.

A KB Σ details the represented knowledge in terms of the intensional information described in the TBox $\mathcal{T}$, and the extensional, in the ABox $\mathcal{A}$. A TBox is formed by a finite set of *inclusion assertions* of the form $B \sqsubseteq C$ for concepts, and $R \sqsubseteq E$ for roles; and a finite set of *functional restrictions* of the form $(\mathrm{funct}R)$. Assuming the sets $\mathrm{N_V}$ of variables and $\mathrm{N_C}$ of constant names, an ABox is composed by a finite set of *membership assertions* on atomic concepts and atomic roles, of the form $A(a)$ and $P(a,b)$, where $\{a,b\} \subseteq \mathrm{N_C}$.

As usual in DLs, semantics is given in terms of *interpretations* $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$. *Reasoning services* (RS) represent *logical implications of KB assertions* verifying:

(subsumption) $\Sigma \models C_1 \sqsubseteq C_2$ or $\Sigma \models E_1 \sqsubseteq E_2$;
(functionality) $\Sigma \models (\mathrm{funct}R)$ or $\Sigma \models \neg(\mathrm{funct}R)$; and
(query answering) $\Sigma \models q(\bar{x})$.

A *conjunctive query* $q(\bar{x})$, with a tuple $\bar{x} \in (\mathrm{N_V})^n$ of arity $n \geq 0$, is a non empty set of atoms $C(z)$, $E(z_1, z_2)$, $z_1 = z_2$, or $z_1 \neq z_2$, where C and E are respectively a general concept and a general role of Σ, and some of the names $\{z, z_1, z_2\} \subseteq \mathrm{N_C} \cup \mathrm{N_V}$ are considered in $\bar{x}$. We will use the function $\mathtt{var} : (\mathrm{N_V})^n \longrightarrow 2^{\mathrm{N_V}}$ to identify the variables in $\bar{x}$. When $\bar{x}$ is the empty tuple, no free variables are considered and the query is identified as *boolean*. Intuitively, $q(\bar{x})$ represents the conjunction of its elements. Let $\mathcal{I}$ be an interpretation, and $\mathfrak{m} : \mathtt{var}(\bar{x}) \cup \mathrm{N_C} \longrightarrow \Delta^\mathcal{I}$ a total function. If $z \in \mathrm{N_C}$ then $\mathfrak{m}(z) = z^\mathcal{I}$ otherwise $\mathfrak{m}(z) = a \in \Delta^\mathcal{I}$. We write $\mathcal{I} \models^\mathfrak{m} C(z)$ if $\mathfrak{m}(z) \in C^\mathcal{I}$, $\mathcal{I} \models^\mathfrak{m} E(z_1, z_2)$ if $(\mathfrak{m}(z_1), \mathfrak{m}(z_2)) \in E^\mathcal{I}$, $\mathcal{I} \models^\mathfrak{m} (z_1 = z_2)$ if $\mathfrak{m}(z_1) = \mathfrak{m}(z_2)$, and $\mathcal{I} \models^\mathfrak{m} (z_1 \neq z_2)$ if $\mathfrak{m}(z_1) \neq \mathfrak{m}(z_2)$. If $\mathcal{I} \models^\mathfrak{m} \phi$ for all $\phi \in q(\bar{x})$, we write $\mathcal{I} \models^\mathfrak{m} q(\bar{x})$ and call $\mathfrak{m}$ a *match* for $\mathcal{I}$ and $q(\bar{x})$. We say that $\mathcal{I}$ satisfies $q(\bar{x})$ and write $\mathcal{I} \models q(\bar{x})$ if there is a match $\mathfrak{m}$ for $\mathcal{I}$ and $q(\bar{x})$. If $\mathcal{I} \models q(\bar{x})$ for all models $\mathcal{I}$ of a KB Σ, we write $\Sigma \models q(\bar{x})$ and say that Σ entails $q(\bar{x})$. Note that the well known *instance checking* RS ($\Sigma \models C(a)$ or $\Sigma \models E(a,b)$) is generalized by query answering. Finally, the *knowledge base satisfiability* RS (whether the KB admits at least one model) will be discussed in Sect. 4.

3 The Argumentation DL-Lite Reasoner

We are particularly interested in handling the reasoning services presented before. This will be achieved through the claim of an argument. Intuitively, an argument may be seen as a set of interrelated pieces of knowledge providing support to a claim. Hence, the claim should take the form of any possible RS, and the knowledge inside an argument will be represented through *DL-Lite*$_\mathcal{A}$ assertions. The notion of argument will rely on the *language for claims*:

Written during the first author's visit to USP, financed by LACCIR.

$\mathcal{L}_{\mathtt{cl}} \longrightarrow C_1 \sqsubseteq C_2 | E_1 \sqsubseteq E_2 | (\mathrm{funct}R) | \neg(\mathrm{funct}R) | q(\bar{x})$

An argument is defined as a structure implying *the claim* from a minimal consistent subset of Σ, namely *the body*.

Definition 1 (Argument) *Given a KB $\Sigma \subseteq$ DL-Lite$_\mathcal{A}$, **an argument** $\mathcal{B}$ is a structure $\langle \Delta, \beta \rangle$, where $\Delta \subseteq \Sigma$ is the **body**, $\beta \in \mathcal{L}_{\mathtt{cl}}$ the **claim**, and it holds (1) $\Delta \models \beta$, (2) $\Delta \not\models \bot$, and (3) $\nexists X \subset \Delta : X \models \beta$. We say that $\mathcal{B}$ **supports** β. The **domain of arguments** from Σ is identified through the set $\mathbb{A}_\Sigma$.*

Consider a KB Σ and an RS $\alpha \in \mathcal{L}_{\mathtt{cl}}$, the *argumentative DL reasoner* verifies $\Sigma \models \alpha$ if there exists $\langle \Delta, \beta \rangle \in \mathbb{A}_\Sigma$ such that β *unifies with* α and $\langle \Delta, \beta \rangle$ is warranted. An argument is *warranted* if it ends up *accepted* (or *undefeated*) from the argumentation interplay. This notion will be made clear in the sequel. *Primitive arguments* are those whose body consists of the claim itself, for instance $\langle \{A(a)\}, \{A(a)\} \rangle$. $\Sigma \models (\mathrm{funct}R)$ is simply verified through a primitive argument $\langle \{(\mathrm{funct}R)\}, (\mathrm{funct}R) \rangle$. Besides, functional assertions $(\mathrm{funct}R)$ can be part of an argument's body. For instance, $\Sigma \models \neg P(a, c)$ is verified through either an argument $\langle \{P(a, b), (\mathrm{funct}P)\}, \{\neg P(a, z), z \neq b\} \rangle$, or $\langle \{P(b, c), (\mathrm{funct}P^-)\}, \{\neg P(z, c), z \neq a\} \rangle$, with $z \in \mathrm{N_V}$. Observe that $z \neq b$ and $z \neq a$ are deduced from (2) in Def. 1.

Once an argument supporting the required RS is identified, the argumentation game begins: a repeated interchange of arguments and *counterarguments* (*i.e.*, arguments whose claim poses a justification to disbelieve in another argument). From the standpoint of logics, this is interpreted as a contradiction between an argument $\mathcal{B}_1$ and a counterargument $\mathcal{B}_2$. Such contradiction could appear while considering the claim of $\mathcal{B}_2$ along with some piece of information deduced from $\mathcal{B}_1$. Thereafter, $\mathcal{B}_1$ and $\mathcal{B}_2$ are referred as a *conflictive pair*.

Definition 2 (Conflict) *Two arguments $\langle \Delta, \beta \rangle \in \mathbb{A}_\Sigma$ and $\langle \Delta', \beta' \rangle \in \mathbb{A}_\Sigma$ are **conflictive** iff $\Delta \models \neg\beta'$. Argument $\langle \Delta', \beta' \rangle$ is identified as the **counterargument**.*

Let us analyze the formation of counterarguments. Consider the argument $\langle \{A \sqsubseteq B, B \sqsubseteq C, C \sqsubseteq D\}, A \sqsubseteq D \rangle$, since $A \sqsubseteq C$ is inferred from its body a possible counterargument could support an axiom like $\neg(A \sqsubseteq C)$. But how could negated axioms be interpreted in DLs? In [Flouris *et al.*, 2006], *negation of general inclusion axioms* was studied. In general, for an axiom like $B \sqsubseteq C$, the *consistency-negation* is $\neg(B \sqsubseteq C) = \exists(B \sqcap \neg C)$ and the *coherency-negation* $\sim(B \sqsubseteq C) = B \sqsubseteq \neg C$. For the former, although the existence assertion $\exists(B \sqcap \neg C)(x)$ falls out of DL-Lite$_\mathcal{A}$, it could be rewritten as a query $q(\langle x \rangle) = \{B(x), \neg C(x)\}$. For instance, the counterargument $\langle \{B(a), A(a), A \sqsubseteq \neg C\}, \{B(a), \neg C(a)\} \rangle$ supports $q(\langle x \rangle)$ with $x = a$. On the other hand, coherency-negation is solved by simply looking for an argument supporting $B \sqsubseteq \neg C$. Recall that an *inconsistent ontology* is that which has no possible interpretation, whereas an *incoherent ontology* [Flouris *et al.*, 2006] is that containing at least one empty named concept. Observe that incoherence does not inhibit the ontology from being satisfiable.

We extend negation of axioms to functional assertions, interpreting $\neg(\mathrm{funct}R)$ as a role R that does not conform to the definition of a function. The only option to form an argument supporting such negation is through extensional information (ABox). For instance, the argument $\langle \{P(a, b), P'(a, c), P' \sqsubseteq P\}, \neg(\mathrm{funct}P) \rangle$. Finally, coherency-negation of functional assertions is not allowed, whereas negation of queries is only allowed for singletons, *i.e.*, $|q(\bar{x})| = 1$.

Comparing the arguments involved in a conflictive pair leads to the notion of *attack* which usually relies on an *argument comparison criterion*. Through such criterion, it is decided if the counterargument prevails from a conflict, and herein the attack ends up identified. A variety of alternatives appears to specify such criterion. In general, the quantity and/or quality of knowledge is somehow weighted to obtain a confidence rate of an argument. The exhaustive analysis goes beyond the scope of this article, hence an abstract argument comparison criterion "$\succ$" will be assumed. The notion of attack is based on the criterion "$\succ$" over conflictive pairs.

Definition 3 (Attack) *Given a conflictive pair of arguments $\mathcal{B}_1 \in \mathbb{A}_\Sigma$ and $\mathcal{B}_2 \in \mathbb{A}_\Sigma$, $\mathcal{B}_2$ **defeats** $\mathcal{B}_1$ iff $\mathcal{B}_2$ is the counterargument and $\mathcal{B}_2 \succ \mathcal{B}_1$ holds. Argument $\mathcal{B}_2$ is said a **defeater** of $\mathcal{B}_1$ (or $\mathcal{B}_1$ is defeated by $\mathcal{B}_2$), noted as $\mathcal{B}_2 \hookrightarrow \mathcal{B}_1$.*

As said before, the reasoning methodology we propose is based on the analysis of the warrant status of the arguments giving support to the required RS. This is the basis of dialectical argumentation [Chesñevar and Simari, 2007]. An *argumentation line* may be seen as the repeated interchange of arguments and counterarguments resembling a dialogue between two parties, formally:

Definition 4 (Argumentation Line) *Given the arguments $\mathcal{B}_1, \ldots, \mathcal{B}_n$ from $\mathbb{A}_\Sigma$, an **argumentation line** λ is a non-empty sequence of arguments $[\mathcal{B}_1, \ldots, \mathcal{B}_n]$ such that $\mathcal{B}_i \hookrightarrow \mathcal{B}_{i-1}$, for $1 < i \leq n$. We will say that λ is **rooted** in $\mathcal{B}_1$, and that $\mathcal{B}_n$ is **the leaf** of λ. Arguments placed on even positions are referred as **con**, whereas those on odd positions will be **pro**. The domain of every argumentation line formed through arguments from $\mathbb{A}_\Sigma$ is noted as $\mathbb{L}_\Sigma$.*

We will consider $\mathbb{L}_\Sigma$ to contain only argumentation lines that are *exhaustive* (lines only end when the leaf argument has no identifiable defeater from $\mathbb{A}_\Sigma$), and *acceptable* (lines whose configuration is compliant with the dialectical constraints). In dialectical argumentation, the notion of *dialectical constraints* (DC) is introduced to state an acceptability condition among the argumentation lines. The DCs assumed in this work will include *non-circularity* (no argument is reintroduced in a same line), and *concordance* (the set of bodies of pro (resp., con) arguments in the same line is consistent).

A *dialectical tree* appears when several dialogues about a common issue (*i.e.*, the root of the tree) are set together. Thus, from a particular set of argumentation lines (namely *bundle set*) the dialectical tree is formalized. A *bundle set* for $\mathcal{B}_1$ is the maximal set $\mathcal{S}(\mathcal{B}_1) \subseteq \mathbb{L}_\Sigma$ (wrt. set inclusion) of exhaustive and acceptable argumentation lines such that every $\lambda \in \mathcal{S}(\mathcal{B}_1)$ is rooted in $\mathcal{B}_1$. Finally, a dialectical tree is:

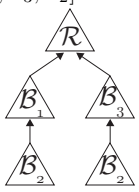
Definition 5 (Dialectical Tree) *Given a KB $\Sigma \subseteq$ DL-Lite$_\mathcal{A}$, a **dialectical tree** $\mathcal{T}(\mathcal{R})$ rooted in an argument $\mathcal{R} \in \mathbb{A}_\Sigma$ is determined by a bundle set $\mathcal{S}(\mathcal{R}) \subseteq \mathbb{L}_\Sigma$ such that $\mathcal{B}$ is an **inner node** in a branch $[\mathcal{R}, \ldots, \mathcal{B}]$ of $\mathcal{T}(\mathcal{R})$ (resp., **the root argument**) iff each **child** of $\mathcal{B}$ is an argument $\mathcal{D} \in [\mathcal{R}, \ldots, \mathcal{B}, \mathcal{D}, \ldots] \in \mathcal{S}(\mathcal{R})$ (resp.,$\mathcal{D} \in [\mathcal{R}, \mathcal{D}, \ldots] \in \mathcal{S}(\mathcal{R})$). **Leaves** in $\mathcal{T}(\mathcal{R})$ and each line in $\mathcal{S}(\mathcal{R})$, coincide. The domain of all dialectical trees from Σ will be noted as $\mathbb{T}_\Sigma$.*

With a little abuse of notation, we will overload the membership symbol writing $\mathcal{B} \in \lambda$ to identify $\mathcal{B}$ from the argumentation line λ, and $\lambda \in \mathcal{T}(\mathcal{R})$ when the line λ belongs to the bundle set associated to the tree $\mathcal{T}(\mathcal{R})$. Dialectical trees allow to determine whether the root node of the tree is to be accepted (ultimately *undefeated*) or rejected (ultimately *defeated*) as a rationally justified belief. Therefore, given a KB $\Sigma \subseteq$ *DL-Lite*$_{\mathcal{A}}$ and a dialectical tree $\mathcal{T}(\mathcal{R})$, the argument $\mathcal{R} \in \mathbb{A}_\Sigma$ is *warranted* from $\mathcal{T}(\mathcal{R})$ *iff* $\mathtt{warrant}(\mathcal{T}(\mathcal{R})) = true$. The *warranting function* $\mathtt{warrant} : \mathbb{T}_\Sigma \longrightarrow \{true, false\}$ will determine the status of the tree from the mark of the root argument. This evaluation will be obtained by weighting all the information present in the tree through the *marking function* $\mathtt{mark} : \mathbb{A}_\Sigma \times \mathbb{L}_\Sigma \times \mathbb{T}_\Sigma \longrightarrow \mathbb{M}$. Such function defines the acceptance criterion applied to each individual argument by assigning to each argument in $\mathcal{T}(\mathcal{R})$ a marking value from the domain $\mathbb{M} = [D, U]$. This is more likely to be done by obtaining the mark of an inner node of the tree from its children (*i.e.*, its defeaters). Once each argument in the tree has been marked (including the root) the warranting function will determine the root's acceptance status from its mark. Hence,

$$\mathtt{warrant}(\mathcal{T}(\mathcal{R})) = true \ \textit{iff} \ \mathtt{mark}(\mathcal{R}, \lambda, \mathcal{T}(\mathcal{R})) = U.$$

DeLP (*Defeasible Logic Programming*) [García and Simari, 2004], is an argumentative machinery for reasoning over defeasible logic programs. In this article we will assume the DeLP marking criterion. That is, (1) all leaves are marked U and (2) every inner node $\mathcal{B}$ is marked U *iff* every child of $\mathcal{B}$ is marked D, otherwise, $\mathcal{B}$ is marked D.

Example 1 *Consider the KB Σ and arguments leading to the dialectical tree depicted below to answer $\Sigma \models B(a)$.*
$\Sigma = \{A \sqsubseteq B, A \sqsubseteq C, C \sqsubseteq \neg B, D \sqsubseteq A, A(a), C(b), D(b)\}$
$\mathcal{R} = \langle\{A \sqsubseteq B, A(a)\}, \{B(a)\}\rangle$
$\mathcal{B}_1 = \langle\{C \sqsubseteq \neg B, A \sqsubseteq C, A(a)\}, \{\neg B(a)\}\rangle$
$\mathcal{B}_2 = \langle\{D \sqsubseteq A, A \sqsubseteq B, C(b), D(b)\}, \{C(b), B(b)\}\rangle$
$\mathcal{B}_3 = \langle\{A \sqsubseteq C, C \sqsubseteq \neg B\}, A \sqsubseteq \neg B\rangle$



Assume $\lambda_1 = [\mathcal{R}, \mathcal{B}_1, \mathcal{B}_2]$ and $\lambda_2 = [\mathcal{R}, \mathcal{B}_3, \mathcal{B}_2]$. Observe that, $\mathcal{B}_3$ is a counterargument of $\mathcal{B}_2$ (as well as $\mathcal{B}_2$ of $\mathcal{B}_3$) but since $\mathcal{B}_2 \succ \mathcal{B}_3$ is deduced from λ_2, $\mathcal{B}_3$ cannot attack $\mathcal{B}_2 \in \lambda_1$. Regarding λ_2, another justification to avoid $\mathcal{B}_3$ attacking $\mathcal{B}_2$ is that non-circularity would be violated. Following the marking function, since $\mathcal{B}_1 \in \lambda_1$ and $\mathcal{B}_3 \in \lambda_2$ are defeated, the root $\mathcal{R}$ is marked undefeated. Thus, the tree ends up warranting the argument $\mathcal{R}$ which supports the RS and therefore $\Sigma \models B(a)$ holds.

4 Concluding Remarks and Future Work

A novel DL-Lite reasoner based on argumentation techniques was proposed. The machinery presented here is defined to support the usual DL-Lite reasoning services, including an extension of the conjunctive queries presented in [Calvanese *et al.*, 2007]. In particular, the notion of satisfiability deserves special attention since its meaning requires to be reinterpreted. Given that the argumentation machinery would answer consistently disregarding the potentially inconsistent KB, a KB that is unsatisfiable for standard DL reasoners would be satisfiable from our theory. On the other hand, what is exactly a satisfiable KB for the proposed reasoner? If there is a warranted argument supporting an RS α and there is another warranted argument supporting $\neg\alpha$, then the KB would be unsatisfiable. An argumentation system free of such drawbacks would depend on the appropriate definition of the comparison and warranting criteria. The required analysis is part of the future work in this direction.

Argumentation was studied in [Williams and Hunter, 2007] to harness ontologies for decision making. Through such argumentation system they basically allow the aggregation of defeasible rules to enrich the ontology reasoning tasks. In our approach an argumentative methodology is defined on top of the DL-reasoner aiming at reasoning about inconsistent ontologies. However our main objective goes further beyond: to manage ontology evolution disregarding inconsistencies. To such purpose, a theory of change is required to be applied over argumentation systems. In [Moguillansky *et al.*, 2008b], a theory capable of handling dynamics of arguments applied over defeasible logic programs (DeLP) [García and Simari, 2004] was presented under the name of *Argument Theory Change* (ATC). Basically, dialectical trees are analyzed to be altered in a form such that the same root argument ends up warranted from the resulting program. Ongoing work also involves the study of ATC on top of the model presented here.

References

[Calvanese *et al.*, 2007] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable Reasoning and Efficient Query Answering in Description Logics: The DL-Lite family. *JAR*, 39(3):385–429, 2007.

[Chesñevar and Simari, 2007] C. Chesñevar and G. Simari. A Lattice-based Approach to Computing Warranted Belief in Skeptical Argumentation Frameworks. In *IJCAI*, pages 280–285, 2007.

[Dung, 1995] P. Dung. On the Acceptability of Arguments and its Fundamental Role in Nonmonotonic Reasoning and Logic Programming and n-person Games. *Artif. Intell.*, 77:321–357, 1995.

[Flouris *et al.*, 2006] G. Flouris, Z. Huang, J. Pan, D. Plexousakis, and H. Wache. Inconsistencies, Negations and Changes in Ontologies. In *AAAI*, pages 1295–1300, 2006.

[García and Simari, 2004] A. García and G. Simari. Defeasible Logic Programming: An Argumentative Approach. *TPLP*, 4(1-2):95–138, 2004.

[Moguillansky *et al.*, 2008a] M. Moguillansky, N. Rotstein, and M. Falappa. A Theoretical Model to Handle Ontology Debugging & Change Through Argumentation. In *IWOD*, 2008.

[Moguillansky *et al.*, 2008b] M. Moguillansky, N. Rotstein, M. Falappa, A. García, and G. Simari. Argument Theory Change Applied to Defeasible Logic Programming. In *AAAI*, pages 132–137, 2008.

[Williams and Hunter, 2007] M. Williams and A. Hunter. Harnessing Ontologies for Argument-Based Decision-Making in Breast Cancer. *ICTAI*, 2:254–261, 2007.

Forgetting for Knowledge Bases in DL-Lite$_{bool}$

Zhe Wang, Kewen Wang and Rodney Topor
Griffith University, Australia

Abstract

We address the problem of term elimination in DL-Lite ontologies by adopting techniques from classical forgetting theory. Specifically, we generalize our previous results on forgetting in DL-Lite$_{core}$ TBox to forgetting in DL-Lite$_{bool}$ KBs. We also introduce query-based forgetting, a parameterized definition of forgetting, which provides a unifying framework for defining and comparing different definitions of forgetting in DL-Lite ontologies.

1 Introduction

An ontology is a formal description of the terminological information of an application domain. An ontology is usually represented as a DL knowledge base (KB), which consists of a TBox and an ABox. As ontologies in Semantic Web applications are becoming larger and more complex, a challenge is how to construct and maintain large ontologies efficiently. Recently, ontology reuse and merging have received intensive interest, and different approaches have been proposed. Among several approaches, the forgetting operator is particularly important, which concerns the elimination of terms in ontologies. Informally, forgetting is a particular form of reasoning that allows a set of attributes F (such as propositional variables, predicates, concepts and roles) in a KB to be discarded or hidden in such a way that future reasoning on information irrelevant to F will not be affected. Forgetting has been well investigated in classical logic [Lin and Reiter, 1994; Lang *et al.*, 2003] and logic programming [Eiter and Wang, 2008; Wang *et al.*, 2005].

Efforts have also been made to define forgetting in DL-Lite [Kontchakov *et al.*, 2008; Wang *et al.*, 2008], a family of lightweight ontology languages. However, a drawback in these approaches is that forgetting is defined only for TBoxes. Although a syntactic notion of forgetting in ontologies is discussed in [Wang *et al.*, 2008], no semantic justification is provided. In most applications, an ontology is expressed as a KB, which is a pair of a TBox and an ABox. We believe that forgetting should be defined for DL-Lite KBs rather than only for TBoxes. Although it is not hard to extend the definitions of forgetting to KBs, our efforts show that it is non-trivial to extend results of forgetting in TBoxes to forgetting in KBs, due to the involvement of ABoxes.

In this paper, we investigate the issue of semantic forgetting for DL-Lite KBs. The main contributions of this paper can be summarized as follows:

- We introduce a model-based definition of forgetting for DL KBs. We investigate some reasoning and expressibility properties of forgetting, which are important for DL-Lite ontology reuse and merging.
- We provide a resolution-like algorithm for forgetting about concepts in DL-Lite$_{bool}$ KBs. The algorithm can also be applied to KB forgetting in DL-Lite$_{horn}$, and to any specific query-based forgetting. It is proved that the algorithm is complete.
- As a general framework for defining and comparing various notions of forgetting, we introduce a parameterized forgetting based on query-answering, by which a given collection of queries determines the result of forgetting. Thus, our approach actually provides a hierarchy of forgetting for DL-Lite.

2 Forgetting in DL-Lite$_{bool}$ Knowledge Bases

In this section, we define the operation of forgetting a signature from a DL-Lite$_{bool}$ KB. We assume the readers' familiarity with DL-Lite$_{bool}$. For the syntax and semantics details of DL-Lite$_{bool}$, the readers should refer to [Artale *et al.*, 2007].

Let $\mathcal{L}$ and $\mathcal{L}_h$, respectively, denote the languages DL-Lite$_{bool}$ and DL-Lite$_{horn}$. Without special mentioning, we use $\mathcal{K}$ to denote a KB in $\mathcal{L}$ and $\mathcal{S}$ a signature (a finite set of concept names and role names) in $\mathcal{L}$. Our model-based definition of forgetting in DL-Lite is analogous to the definition for forgetting in classical logic [Lin and Reiter, 1994; Lang *et al.*, 2003].

Let $\mathcal{I}_1, \mathcal{I}_2$ be two interpretations of $\mathcal{L}$. Define $\mathcal{I}_1 \sim_{\mathcal{S}} \mathcal{I}_2$ iff

1. $\Delta^{\mathcal{I}_1} = \Delta^{\mathcal{I}_2}$, and $a^{\mathcal{I}_1} = a^{\mathcal{I}_2}$ for each individual name a.
2. For each concept name A not in $\mathcal{S}$, $A^{\mathcal{I}_1} = A^{\mathcal{I}_2}$.
3. For each role name P not in $\mathcal{S}$, $P^{\mathcal{I}_1} = P^{\mathcal{I}_2}$.

Clearly, $\sim_{\mathcal{S}}$ is an equivalence relation.

Definition 1 *We call KB $\mathcal{K}'$ a* result of model-based forgetting *about $\mathcal{S}$ in $\mathcal{K}$ if:*

- $\mathsf{Sig}(\mathcal{K}') \subseteq \mathsf{Sig}(\mathcal{K}) - \mathcal{S}$,
- $\mathsf{Mod}(\mathcal{K}') = \{\mathcal{I}' \mid \exists \mathcal{I} \in \mathsf{Mod}(\mathcal{K})$ *s.t.* $\mathcal{I} \sim_{\mathcal{S}} \mathcal{I}'\}$.

It follows from the definition that the result of forgetting about $\mathcal{S}$ in $\mathcal{K}$ is unique up to KB equivalence. So we will use $\mathsf{forget}(\mathcal{K}, \mathcal{S})$ to denote the result of forgetting about $\mathcal{S}$ in $\mathcal{K}$ throughout the paper.

Example 1 *Let $\mathcal{K}$ consist of the following axioms:*

$Lecturer \sqsubseteq \geqslant 2\ teaches$, $\exists teaches^- \sqsubseteq Course$,
$\exists teaches \sqsubseteq Lecturer \sqcup PhD$,
$Lecturer \sqcap Course \sqsubseteq \bot$, $PhD \sqcap Course \sqsubseteq \bot$.
$Lecturer(John)$, $teaches(John, AI)$.

Suppose we want to forget about $\{Lecturer, PhD\}$, then $\mathsf{forget}(\mathcal{K}, \{Lecturer, PhD\})$ consists of the following axioms:

$\exists teaches^- \sqsubseteq Course$, $\exists teaches \sqcap Course \sqsubseteq \bot$,
$\geqslant 2\ teaches(John)$, $\neg Course(John)$,
$teaches(John, AI)$.

The result of forgetting possesses several desirable properties. In particular, it preserves reasoning properties of the KB.

Proposition 1 *We have*

1. $\mathsf{forget}(\mathcal{K}, \mathcal{S})$ *is consistent iff $\mathcal{K}$ is consistent.*
2. *for any inclusion or assertion α with $\mathsf{Sig}(\alpha) \cap \mathcal{S} = \emptyset$, $\mathsf{forget}(\mathcal{K}, \mathcal{S}) \models \alpha$ iff $\mathcal{K} \models \alpha$.*
3. *for any grounded query q with $\mathsf{Sig}(q) \cap \mathcal{S} = \emptyset$, $\mathsf{forget}(\mathcal{K}, \mathcal{S}) \models q$ iff $\mathcal{K} \models q$.*

The following property is useful for ontology partial reuse and merging.

Proposition 2 *Let $\mathcal{K}_1, \mathcal{K}_2$ be two $\mathcal{L}$-KBs. Suppose $\mathsf{Sig}(\mathcal{K}_1) \cap \mathsf{Sig}(\mathcal{K}_2) \cap \mathcal{S} = \emptyset$, then we have:*

$$\mathsf{forget}(\mathcal{K}_1 \cup \mathcal{K}_2, \mathcal{S}) \equiv \mathsf{forget}(\mathcal{K}_1, \mathcal{S}) \cup \mathsf{forget}(\mathcal{K}_2, \mathcal{S}).$$

An interesting special case is $\mathsf{Sig}(\mathcal{K}_2) \cap \mathcal{S} = \emptyset$. In this case, it is safe to forget about $\mathcal{S}$ from $\mathcal{K}_1$ before merging $\mathcal{K}_1$ and $\mathcal{K}_2$.

The following proposition shows that the forgetting operation can be divided into steps.

Proposition 3 *Let $\mathcal{S}_1, \mathcal{S}_2$ be two signatures. Then we have*

$$\mathsf{forget}(\mathcal{K}, \mathcal{S}_1 \cup \mathcal{S}_2) \equiv \mathsf{forget}(\mathsf{forget}(\mathcal{K}, \mathcal{S}_1), \mathcal{S}_2).$$

In the following part of the section, we introduce a syntactic algorithm for computing the results of forgetting about concepts in DL-Lite$_{bool}$ KBs. The algorithm first transforms a DL-Lite$_{bool}$ KB into a equivalent normal form, and then eliminates concepts from the KB.

We call a basic concept or its negation a *literal concept*.

Definition 2 *$\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is in* normal form *if:*

- *All the inclusions in $\mathcal{T}$ are of the form $B_1 \sqcap \ldots \sqcap B_m \sqsubseteq B_{m+1} \sqcup \ldots \sqcup B_n$ where $0 \leq m \leq n$ and $B_1, \ldots, B_n$ are basic concepts such that $B_i \neq B_j$ for all $i < j$.*
- *$\mathcal{A} = \{C_1(a_1), \ldots, C_s(a_s)\} \cup \mathcal{A}_r$, where $s \geq 0$, satisfies the following conditions:*
  1. *$\mathcal{A}_r$ contains only role assertions,*
  2. *$a_i \neq a_j$ for $1 \leq i < j \leq s$, and*
  3. *C_i is in disjunction of conjunctions of literal concepts (DNF), for each $1 \leq i \leq s$.*

We can show that every KB in $\mathcal{L}$ can be equivalently transformed into its normal form.

Now we present in Algorithm 1 how to compute the result of forgetting about a set of concept names in a $\mathcal{L}$-KB.

Algorithm 1 (Compute the result of forgetting a set of concept names in a DL-Lite$_{bool}$ KB)
Input: A DL-Lite$_{bool}$ KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ and a set $\mathcal{S}$ of concept names.
Output: $\mathsf{forget}(\mathcal{K}, \mathcal{S})$.
Method:
Step 1. Transform $\mathcal{K}$ into its normal form.
Step 2. For each pair of inclusions $A \sqcap C \sqsubseteq D$ and $C' \sqsubseteq A \sqcup D'$ in $\mathcal{T}$, where $A \in \mathcal{S}$, add inclusion $C \sqcap C' \sqsubseteq D \sqcup D'$ to $\mathcal{T}$ if it contains no concept name A' appearing on both sides of the inclusion.
Step 3. For each concept name $A \in \mathcal{S}$ occurring in $\mathcal{A}$, add $A \sqsubseteq \top$ and $\bot \sqsubseteq A$ to $\mathcal{T}$;
Step 4. For each assertion $C(a)$ in $\mathcal{A}$, each inclusion $A \sqcap D_1 \sqsubseteq D_2$ and each inclusion $D_3 \sqsubseteq A \sqcup D_4$ in $\mathcal{T}$, where $A \in \mathcal{S}$, add $C'(a)$ to $\mathcal{A}$, where C' is obtained by replacing each occurrence of A in C with $\neg D_1 \sqcup D_2$ and $\neg A$ with $\neg D_3 \sqcup D_4$, and C' is transformed into its DNF.
Step 5. Remove all inclusions of the form $C \sqsubseteq \top$ or $\bot \sqsubseteq C$, and all assertions of the form $\top(a)$.
Step 6. Remove all inclusions and assertions that contain any concept name in $\mathcal{S}$.
Step 7. Return the resulting KB as $\mathsf{forget}(\mathcal{K}, \mathcal{S})$.

Figure 1: Forget concepts in a DL-Lite$_{bool}$ KB.

In Algorithm 1, Step 2 generates all the inclusions in $\mathsf{forget}(\mathcal{K}, \mathcal{S})$ in a resolution-like manner. Step 3 is to make the specification of Step 4 simpler. In Step 4, each positive occurrence of A in the ABox is replaced by its 'supersets', and each negative occurrence by its 'subsets'.

Algorithm 1 always terminates. The following theorem shows that it is sound and complete with respect to the semantic definition of forgetting.

Theorem 1 *Let $\mathcal{S}$ be a set of concept names. Then $\mathsf{forget}(\mathcal{K}, \mathcal{S})$ is expressible in $\mathcal{L}$, and Algorithm 1 always returns $\mathsf{forget}(\mathcal{K}, \mathcal{S})$.*

Suppose $\mathcal{S}$ is fixed. When the input $\mathcal{K}$ is in normal form, Algorithm 1 takes only polynomial time to compute $\mathsf{forget}(\mathcal{K}, \mathcal{S})$.

3 Query-Based Forgetting for DL-Lite Knowledge Bases

In this section, we introduce a parameterized definition of forgetting for DL-Lite$_{bool}$ KBs, which can be used as a unifying framework for forgetting in DL-Lite$_{bool}$ KBs.

We assume that $\mathcal{Q}$ is a query language for DL-Lite$_{bool}$. The notion of query is very general here. A query can be an assertion, an inclusion, or even a formula in a logic language.

Definition 3 (query-based forgetting) *We call KB $\mathcal{K}'$ a* result of $\mathcal{Q}$-forgetting about $\mathcal{S}$ in $\mathcal{K}$ *if the following three conditions are satisfied:*

- $\mathsf{Sig}(\mathcal{K}') \subseteq \mathsf{Sig}(\mathcal{K}) - \mathcal{S}$,
- $\mathcal{K} \models \mathcal{K}'$,
- *for any grounded query q in $\mathcal{Q}$ with $\mathsf{Sig}(q) \cap \mathcal{S} = \emptyset$, we have $\mathcal{K} \models q$ implies $\mathcal{K}' \models q$.*

The above definition implicitly introduces a family of forgetting in the sense that each query language determines a notion of forgetting for DL-Lite KBs.

The results of $\mathcal{Q}$-forgetting are not necessarily unique, We denote the set of all results of $\mathcal{Q}$-forgetting about $\mathcal{S}$ in $\mathcal{K}$ as $\mathsf{Forget}^{\mathcal{Q}}(\mathcal{K}, \mathcal{S})$.

As model-based forgetting requires preserving model equivalence, it is easy to see that the result of model-based forgetting is also a result of $\mathcal{Q}$-forgetting for any query language $\mathcal{Q}$. In this sense, the model-based forgetting is the strongest notion of forgetting for DL-Lite$_{bool}$.

Theorem 2 *We have*

1. $\mathsf{forget}(\mathcal{K}, \mathcal{S}) \in \mathsf{Forget}^{\mathcal{Q}}(\mathcal{K}, \mathcal{S})$*;*
2. *for each $\mathcal{K}' \in \mathsf{Forget}^{\mathcal{Q}}(\mathcal{K}, \mathcal{S})$, $\mathsf{forget}(\mathcal{K}, \mathcal{S}) \models \mathcal{K}'$.*

This theorem shows that Algorithm 1 can also be used to compute a result of $\mathcal{Q}$-forgetting for any $\mathcal{Q}$.

Now we consider how to characterize model-based forgetting by query-based forgetting. The results of model-based forgetting may not be expressible in DL-Lite$_{bool}$. The major reason for this, as our efforts show, is that DL-Lite$_{bool}$ does not have a construct to represent the cardinality of a concept.

For this reason, we extend $\mathcal{L}$ to $\mathcal{L}^c$ by introducing new concepts of the form $\geqslant n\ u.C$, where C is a $\mathcal{L}$-concept and n is a natural number. Given an interpretation $\mathcal{I}$, $(\geqslant n\ u.C)^{\mathcal{I}} = \Delta^{\mathcal{I}}$ if $\sharp(C^{\mathcal{I}}) \geq n$ and $(\geqslant n\ u.C)^{\mathcal{I}} = \emptyset$ if $\sharp(C^{\mathcal{I}}) \leq n-1$, where $\sharp(S)$ denotes the cardinality of set S.

We are interested in the query language $\mathcal{Q}^c_{\mathcal{L}}$, which is the set of concept inclusions and (negated) assertions in $\mathcal{L}^c$, and possibly their unions. We can show that $\mathcal{Q}^c_{\mathcal{L}}$-forgetting is equivalent to model-based forgetting.

Theorem 3 *KB $\mathcal{K}'$ is a result of $\mathcal{Q}^c_{\mathcal{L}}$-forgetting about $\mathcal{S}$ in $\mathcal{K}$ iff $\mathcal{K}' \equiv \mathsf{forget}(\mathcal{K}, \mathcal{S})$.*

Example 2 *Recall the KB $\mathcal{K}$ in Example 1. We have $\mathsf{forget}(\mathcal{K}, \{teaches\})$ is not expressible in $\mathcal{L}$. However, it is expressible in $\mathcal{L}^c$, and it consists of the following axioms:*

$Lecturer \sqcap Course \sqsubseteq \bot$, $PhD \sqcap Course \sqsubseteq \bot$,
$Lecturer \sqsubseteq \geqslant 2\ u.Course$,
$Lecturer(John)$, $Course(AI)$.

4 Related Work and Conclusions

The works that are close to this paper are [Kontchakov *et al.*, 2008; Wang *et al.*, 2008]. One major difference of our work from them is that we investigate forgetting for KBs while [Kontchakov *et al.*, 2008; Wang *et al.*, 2008] are only restricted to TBoxes. Such an extension (from TBoxes to KBs) is non-trivial because the involvement of ABoxes makes things more complex. This can be seen from the algorithms and proofs of corresponding results. Our parameterized forgetting generalizes the definition of forgetting (uniform interpolation) in [Kontchakov *et al.*, 2008] in two ways (although it is not technically hard): (1) our definition is defined for KBs and (2) our definition is defined for arbitrary query languages.

Conservative extension and module extraction have some similarity with forgetting but they are different in that the first two approaches support only removing axioms, but cannot modify them. Update and erasure operations in DL-Lite are discussed in [Giacomo *et al.*, 2007]. Although both erasure [Giacomo *et al.*, 2007; Liu *et al.*, 2006] and forgetting are concerned with eliminating information from an ontology, they are quite different. When erasing an assertion $A(a)$ from a DL KB $\mathcal{K}$, only the membership relation between individual a and concept A is removed, while concept name A is not necessarily removed from $\mathcal{K}$. Whereas forgetting about A in $\mathcal{K}$ involves eliminating all logical relations (e.g., subsumption relation, membership relation, etc.) in $\mathcal{K}$ that refer to A.

We plan to work in different directions including: (1) To identify a query language that can balance computational complexity of the corresponding notion of forgetting and requirements for forgetting from practical applications, such as ontology reuse, merging and update. (2) To establish a general framework of forgetting for more expressive DLs in terms of query-based forgetting.

References

[Artale *et al.*, 2007] A. Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev. DL-Lite in the light of first-order logic. In *Proc. 22nd AAAI*, pages 361–366, 2007.

[Eiter and Wang, 2006] T. Eiter and K. Wang. Forgetting and conflict resolving in disjunctive logic programming. In *Proc. 21st AAAI*, pages 238–243, 2006.

[Eiter and Wang, 2008] T. Eiter and K. Wang. Semantic forgetting in answer set programming. *Artificial Intelligence*, 14:1644–1672, 2008.

[Giacomo *et al.*, 2007] G. De Giacomo, M. Lenzerini, A. Poggi, and R. Rosati. On the approximation of instance level update and erasure in description logics. In *Proc. 22nd AAAI*, pages 403–408, 2007.

[Kontchakov *et al.*, 2008] R. Kontchakov, F. Wolter, and M. Zakharyaschev. Can you tell the difference between DL-Lite ontologies? In *Proc. 11th KR*, pages 285–295, 2008.

[Lang *et al.*, 2003] J. Lang, P. Liberatore, and P. Marquis. Propositional independence: Formula-variable independence and forgetting. *J. Artif. Intell. Res. (JAIR)*, 18:391–443, 2003.

[Lin and Reiter, 1994] F. Lin and R. Reiter. Forget it. In *Proc. AAAI Fall Symposium on Relevance*, pages 154–159, 1994.

[Liu *et al.*, 2006] H. Liu, C. Lutz, M. Milicic, and F. Wolter. Updating description logic ABoxes. In *Proc. KR*, pages 46–56, 2006.

[Wang *et al.*, 2005] K. Wang, A. Sattar, and K. Su. A theory of forgetting in logic programming. In *Proc. 20th AAAI*, pages 682–687. AAAI Press, 2005.

[Wang *et al.*, 2008] Z. Wang, K. Wang, R. Topor, and J. Z. Pan. Forgetting concepts in DL-Lite. In *Proc. 5th ESWC2008*, pages 245–257, 2008.

AGM Revision in Description Logics*

Márcio Moretto Ribeiro and Renata Wassermann
Department of Computer Science
University of São Paulo
{marciomr, renata}@ime.usp.br

Abstract

The AGM theory for belief revision cannot be directly applied to most description logics. For contraction, the problem lies on the fact that many logics are not compliant with the recovery postulate. For revision, the problem is that several interesting logics are not closed under negation.

In this work, we present solutions for both problems: we recall a previous solution proposing to substitute the recovery postulate of contraction by relevance and we present a construction for revision that does not depend on negation, together with a set of postulates and a representation theorem.

1 Introduction

The development of Semantic Web technologies has atracted the attention of the artificial intelligence community to the importance to represent conceptual knowledge in the web. This development has reached a peak with the adoption of OWL as the standard language to represent ontologies on the web. Since knowledge on the web is not static, another area has gained popularity in the past few years: ontology evolution. The main challenge of ontology evolution is to study how ontologies should behave in a dynamic environment. Belief revision theory has been facing this challenge for propositional logic for more than twenty years and, hence, it would be interesting to try to apply these techniques to ontologies.

In this work we apply the most influential work in belief revision, the AGM paradigm, to description logics. First, in section 2, an introduction to AGM theory is presented. In section 3, we show how to adapt the AGM postulates for contraction so that they can be used with description logics. In section 4, we present a construction for AGM-style revision that does not depend on the negation of axioms. Finally we conclude and point towards future work.

2 AGM Paradigm

In the AGM paradigm [Alchourrón *et al.*, 1985], the beliefs of an agent are represented by a *belief set*, a set of formulas closed under logical consequence. The consequence operator Cn is assumed to be tarskian, compact, satisfy the deduction theorem and supraclassicality. We will sometimes refer to these properties as the *AGM-assumptions*. Three operations are defined: expansion, contraction and revision. Given a belief set K and a formula α, the expansion $K + \alpha$ is defined as $K + \alpha = Cn(K \cup \{\alpha\})$. Contraction consists in removing some belief from the belief set and revision consist in adding a new belief in such a way that the resulting set is consistent. Contraction and revision are not uniquely defined, but are constrained by a set of *rationality postulates*. The AGM basic postulates for contraction are:

(closure) $K - \alpha = Cn(K - \alpha)$

(success) If $\alpha \notin Cn(\emptyset)$ then $\alpha \notin K - \alpha$

(inclusion) $K - \alpha \subseteq K$

(vacuity) If $\alpha \notin K$ then $K - \alpha = K$

(recovery) $K \subseteq K - \alpha + \alpha$

(extensionality) If $Cn(\alpha) = Cn(\beta)$ then $K - \alpha = K - \beta$

Of the six postulates, five are very intuitive and widely accepted. The recovery postulate has been debated in the literature since the very beginning of AGM theory [Makinson, 1987]. Nevertheless, the intuition behind the postulate, that unnecessary loss of information should be avoided, is commonly accepted.

In [Alchourrón *et al.*, 1985], besides the postulates, the authors also present a construction for contraction (*partial-meet contraction*) which, for logics satisfying the AGM assumptions, is equivalent to the set of postulates in the following sense: every partial-meet contraction satisfies the six postulates and every operation that satisfies the postulates can be constructed as a partial-meet contraction.

The operation of revision is also constrained by a set of six basic postulates:

(closure) $K * \alpha = Cn(K * \alpha)$

(success) $\alpha \in K * a$

(inclusion) $K * \alpha \subseteq K + \alpha$

(vacuity) If $K + \alpha$ is consistent then $K * \alpha = K + \alpha$

(consistency) If α is consistent then $K * \alpha$ is consistent.

(extensionality) If $Cn(\alpha) = Cn(\beta)$ then $K * \alpha = K * \beta$

*The first author is supported by FAPESP and the second author is partially supported by CNPq. This work was developed as part of FAPESP project 2004/14107-2.

In AGM theory, usually revision is constructed based on contraction and expansion, using the *Levi Identity*: $K * \alpha = (K - \neg\alpha) + \alpha$. The revision obtained using a partial-meet contraction is equivalent to the six basic postulates for revision.

3 AGM Contraction and Relevance

Although very elegant, the AGM paradigm cannot be applied to every logic. [Flouris *et al.*, 2004] define a logic to be *AGM-compliant* if it admits a contraction operation satisfying the six AGM postulates.

The authors also showed that the logics behind OWL ($\mathcal{SHIF}$(**D**) and $\mathcal{SHOIN}$(**D**)) are not AGM-compliant. For this reason it was proposed in [Flouris *et al.*, 2006] that a new set of postulates for contraction should be defined. A contraction satisfying this new set of postulates should exist in any logic (*existence criteria*) and this new set of postulates should be equivalent to the AGM postulates for every AGM-compliant logic (*AGM-rationality criteria*).

In [Ribeiro and Wassermann, 2006] we have shown a set of postulates that partially fulfills these criteria, the AGM postulates with the *recovery* postulate exchanged by:

(relevance) If $\beta \in K \setminus K - \alpha$, then there is K' s. t. $K - \alpha \subseteq K' \subseteq K$ and $\alpha \notin Cn(K')$, but $\alpha \in Cn(K' \cup \{\beta\})$.

The relevance postulate was proposed in [Hansson, 1989] in order to capture the minimal change intuition, i.e. to avoid unnecessary loss of information. We have proven the following result:

Representation Theorem 3.1 *[Ribeiro and Wassermann, 2006] For every belief set K closed under a tarskian and compact logical consequence, $-$ is a partial meet contraction operation over K iff $-$ satisfies* closure*,* success*,* inclusion*,* vacuity*,* extensionality *and* relevance*.*

From this theorem, it follows that the *existence* criteria for this set of postulates is valid for tarskian and compact logics.

Corollary 3.2 *[Ribeiro and Wassermann, 2006] Every tarskian and compact logic is compliant with the AGM postulates if* recovery *is substituted by* relevance*.*

Furthermore, since partial meet contraction is equivalent to the AGM postulates for every logic that satisfies the AGM assumptions [Hansson, 1999], we have the following weaker version of the *AGM-rationality* criteria:

Corollary 3.3 *For every logic that satisfies the AGM assumptions* relevance *is equivalent to* recovery *in the presence of the other AGM postulates.*

4 AGM Revision without Negation

In [Ribeiro and Wassermann, 2008] we pointed out that the main problem to apply AGM revision to description logics is the absence of negation of axioms in some logics, since constructions for revision are usually based on the Levi identity. For example: there is not a consensus on which should be the negation of $R \sqsubseteq S$ in $\mathcal{SHOIN}$(**D**) . For this reason we should try to define constructions that do not depend on the negation of axioms. In [Ribeiro and Wassermann, 2008] we have proposed such constructions for belief bases (sets not necessarily closed under logical consequence). In this section we define a construction for revision without negation that can be used for revising belief sets.

Satisfying the AGM postulates for revision is quite easy. For example if we get $K * \alpha = K + \alpha$ if $K + \alpha$ is consistent and $K * \alpha = Cn(\alpha)$ otherwise, we end up with a construction that satisfies all the AGM postulates for revision.

This happens because there is no postulate in AGM revision that guaranties minimal loss of information We can define a minimality criterium for revision using a postulate similar to the relevance postulate for contraction:

(relevance) If $\beta \in K \setminus K * \alpha$ then there is K' such that $K \cap (K * \alpha) \subseteq K' \subseteq K$ and $K' \cup \{\alpha\}$ is consistent, but $K' \cup \{\alpha, \beta\}$ is inconsistent.

In order to define a construction that satisfies relevance, we will use the definition of maximally consistent sets with respect to a sentence, which are the maximal subsets of K that, together with α, are consistent:

Definition 4.1 (Maximally consistent set w.r.t α) *[Delgrande, 2008] $X \in K \downarrow \alpha$ iff: i.$X \subseteq K$. ii. $X \cup \{\alpha\}$ is consistent. iii. If $X \subset X' \subseteq K$ then $X' \cup \{\alpha\}$ is inconsistent.*

A selection function chooses at least one element of this set:

Definition 4.2 (Selection function) *[Alchourrón* et al.*, 1985] A* selection function *for K is a function γ such that: If $K \downarrow \alpha \neq \emptyset$, then $\emptyset \neq \gamma(K \downarrow \alpha) \subseteq K \downarrow \alpha$, otherwise, $\gamma(K \downarrow \alpha) = \{K\}$.*

We define partial-meet revision without negation by the intersection of the elements chosen by the selection function followed by an expansion by α:

Definition 4.3 (Revision without negation)

$$K *_\gamma \alpha = \bigcap \gamma(K \downarrow \alpha) + \alpha$$

The revision presented above satisfies the six AGM postulates for revision plus relevance:

Theorem 4.4 *$K *_\gamma \alpha$ satisfies the six basic AGM postulates for revision and relevance.*

Proof: *Closure*, *success* and *inclusion* follow directly from the construction. *Vacuity* follows from the fact that if $K + \alpha$ is consistent then $K \downarrow \alpha = \{K\}$. *Extensionality* follows from the fact that if $Cn(\alpha) = Cn(\beta)$ then for all sets X, $X \cup \{\alpha\}$ is inconsistent iff $X \cup \{\beta\}$ is also inconsistent. To prove *consistency*, assume that α is consistent and that $\bigcap \gamma(K \downarrow \alpha) + \alpha$ is inconsistent. Then since $\bigcap \gamma(K \downarrow \alpha) \subseteq X \in K \downarrow \alpha$, by monotonicity $X \cup \{\alpha\}$ is inconsistent, which contradicts the definition. For *relevance*, let $\beta \in K \setminus K *_\gamma \alpha$. Then there exists $X \in \gamma(K \downarrow \alpha)$ such that $\beta \notin X + \alpha$. We also know that for all $X' \in \gamma(K \downarrow \alpha)$, it holds that $\bigcap \gamma(K \downarrow \alpha) + \alpha \subseteq X' + \alpha$ and hence, $K \cap (\bigcap \gamma(K \downarrow \alpha) + \alpha) \subseteq K \cap (X' + \alpha)$. This holds in particular for X as above. Take $K' = K \cap (X + \alpha)$. □

It also satisfies a postulate that for classical logics follows from the AGM postulates, but not in the general case:

(uniformity) If for all $K' \subseteq K$, $K' \cup \{\alpha\}$ is inconsistent iff $K' \cup \{\beta\}$ is inconsistent then $K \cap K * \alpha = K \cap K * \beta$

In order to prove the representation theorem, we need to make use of two properties of consequence operations:

1. Inconsistent explosion: Whenever K is inconsistent, then for all formulas α, $\alpha \in Cn(K)$
2. Distributivity: For all sets of formulas X, Y and W, $Cn(X \cup (Cn(Y) \cap Cn(W))) = Cn(X \cup Y) \cap Cn(X \cup W)$

We also need the following lemmas:

Lemma 4.5 *For any monotonic and compact logic if $X \subseteq K$ and $X \cup \{\alpha\}$ is consistent then there is X' s.t. $X \subseteq X' \in K \downarrow \alpha$.*[1]

Lemma 4.6 *[Delgrande, 2008]$K \downarrow \alpha = K \downarrow \beta$ iff for all $X \subseteq K$, $X \cup \{\alpha\}$ is inconsistent iff $X \cup \{\beta\}$ is inconsistent.*

Lemma 4.7 *If Cn satisfies distributivity and * satisfies success, inclusion and consistency, then $(K \cap K * \alpha) + \alpha = K + \alpha \cap K * \alpha + \alpha = K * \alpha$*

The representation theorem states the equivalence between the construction and a set of postulates:

Representation Theorem 4.8 (Revision without negation) *For any monotonic and compact logic that satisfies inconsistent explosion and distributivity, a revision operator $*$ is a revision without negation $\bigcap \gamma(K \downarrow \alpha) + \alpha$ for some selection function γ if and only if it satisfies closure, success, inclusion, consistency, relevance and uniformity.*

Proof: (construction ⇒ postulates):
The AGM postulates and relevance are satisfied, as shown in Theorem 4.4. For *uniformity*, note that if it holds that for all $K' \subseteq K$, $K' \cup \{\alpha\}$ is inconsistent iff $K' \cup \{\beta\}$ is inconsistent, then by Lemma 4.6, $K \downarrow \alpha = K \downarrow \beta$, and hence, $\bigcap \gamma(K \downarrow \alpha) = \bigcap \gamma(K \downarrow \beta)$.

(postulates ⇒ construction):
Let $*$ be an operator satisfying the six postulates and let $\gamma(K \downarrow \alpha) = \{X \in K \downarrow \alpha | K \cap (K * \alpha) \subseteq X\}$ if α is consistent and $\gamma(K \downarrow \alpha) = \{K\}$ otherwise. We have to prove that: 1) $\gamma(K \downarrow \alpha)$ is a selection function and 2) $K *_\gamma \alpha = K * \alpha$.

1. First we need to prove that γ is well defined, i.e., that if $K \downarrow \alpha = K \downarrow \beta$ then $\gamma(K \downarrow \alpha) = \gamma(K \downarrow \beta)$. This follows directly from Lemma 4.6 and *uniformity*.

To prove that γ is a selection function we need to prove that if $K \downarrow \alpha \neq \emptyset$, then $\emptyset \neq \gamma(K \downarrow \alpha) \subseteq K \downarrow \alpha$. If α is consistent then it follows from *consistency* that $K * \alpha$ is consistent. In this case, because of *closure* and *success* $(K * \alpha) + \alpha$ is consistent. It follows that $(K \cap K * \alpha) \cup \{\alpha\}$ is consistent and by Lemma 4.5 there is X s.t. $K \cap (K * \alpha) \subseteq X \in K \downarrow \alpha$. Hence $X \in \gamma(K \downarrow \alpha)$.

2. If α is inconsistent, it follows from *closure*, *success*, and inconsistent explosion that both $K * \alpha$ and $K *_\gamma \alpha$ are the unique inconsistent belief set.

[1] This is a generalization of the *Upper Bound Property* used in [Alchourrón *et al.*, 1985]

Otherwise, $K \cap (K * \alpha) \subseteq X$ for every $X \in \gamma(K \downarrow \alpha)$. It follows that $K \cap (K * \alpha) \subseteq \bigcap \gamma(K \downarrow \alpha)$. By monotonicity, $(K \cap (K * \alpha)) + \alpha \subseteq \bigcap \gamma(K \perp_\alpha) + \alpha$ and by Lemma 4.7, $K * \alpha \subseteq K *_\gamma \alpha$.

To prove that $K *_\gamma \alpha \subseteq K * \alpha$, we will show that $\bigcap \gamma(K \downarrow \alpha) \subseteq K * \alpha$. Let $\beta \in \bigcap \gamma(K \downarrow \alpha) \setminus K * \alpha$. Since $\bigcap \gamma(K \downarrow \alpha) \subseteq K$, by *relevance*, there is K' such that $K \cap (K * \alpha) \subseteq K' \subseteq K$ and $K' \cup \{\alpha\}$ is consistent, but $K' \cup \{\alpha, \beta\}$ is inconsistent. Since $K' \subseteq K$ and $K' \cup \{\alpha\}$ is consistent, by Lemma 4.5 we know that there is X such that $\beta \notin X$, $K' \subseteq X \in K \downarrow \alpha$. Since $K \cap (K * \alpha) \subseteq X$, by the construction of γ, $X \in \gamma(K \downarrow \alpha)$ and hence, $\beta \notin \bigcap \gamma(K \downarrow \alpha)$. □

5 Conclusion and Future Work

Although it is not possible to apply the AGM paradigm directly to description logics, it is possible to adapt it. In the case of the contraction, that means a small change in the postulates. In the case of the revision, we need to adapt both construction and postulates.

As future work we plan to compare our set of postulates for contraction with the one proposed in [Flouris *et al.*, 2006] and study the relation between the contraction and revision operators proposed. We also plan to apply the solution for contraction and revision to other logics where negation is problematic, such as Horn logic.

References

[Alchourrón *et al.*, 1985] C. Alchourrón, P. Gärdenfors, and D. Makinson. On the logic of theory change. *J. of Symbolic Logic*, 50(2):510–530, 1985.

[Delgrande, 2008] J. P. Delgrande. Horn clause belief change: Contraction functions. In G. Brewka and J. Lang, editors, *Proceedings of KR2008*, pages 156–165. AAAI Press, 2008.

[Flouris *et al.*, 2004] G. Flouris, D. Plexousakis, and G. Antoniou. Generalizing the AGM postulates. In J.P. Delgrande and T. Schaub, editors, *Proceedings of (NMR-04)*, pages 171–179, 2004.

[Flouris *et al.*, 2006] G. Flouris, D. Plexousakis, and G. Antoniou. On generalizing the AGM postulates. In *Proceedings of STAIRS*, 2006.

[Hansson, 1989] S. O. Hansson. New operators for theory change. *Theoria*, 55:114–132, 1989.

[Hansson, 1999] S. O. Hansson. *A Textbook of Belief Dynamics*. Kluwer Academic Publishers, 1999.

[Makinson, 1987] D. Makinson. On the status of the postulate of recovery in the logic of theory change. *Journal of Philosophical Logic*, 16:383–394, 1987.

[Ribeiro and Wassermann, 2006] M. M. Ribeiro and R. Wassermann. First steps towards revising ontologies. In *Proceedings of WONTO*, 2006.

[Ribeiro and Wassermann, 2008] M. M. Ribeiro and R. Wassermann. Base revision for ontology debugging. *Journal of Logic and Computation*, to appear, 2008.

First Steps in $\mathcal{EL}$ Contraction*

Richard Booth
Mahasarakham University
Thailand
richard.b@msu.ac.th

Thomas Meyer
Meraka Institute, CSIR and
School of Computer Science
University of Kwazulu-Natal
South Africa
tommie.meyer@meraka.org.za

Ivan José Varzinczak
Meraka Institute
CSIR, South Africa
ivan.varzinczak@meraka.org.za

Abstract

In this work we make a preliminary investigation of belief contraction in the $\mathcal{EL}$ family of description logics. Based on previous work on propositional Horn logic contraction, here we state the main definitions for contraction in a description logic extending it, and point out the main issues involved when addressing belief set contraction in $\mathcal{EL}$.

1 Motivation

Belief change is a subarea of knowledge representation concerned with describing how an intelligent agent ought to change its beliefs about the world in the face of new and possibly conflicting information. Arguably the most influential work in this area is the so-called AGM approach [Alchourrón *et al.*, 1985; Gärdenfors, 1988] which focuses on two types of belief change: *revision*, in which an agent has to keep its set of beliefs consistent while incorporating new information into it, and *contraction*, in which an agent has to give up some of its beliefs in order to avoid drawing unwanted conclusions.

Our primary reason for focusing on this topic is because of its application to constructing and repairing ontologies in description logics (DLs) [Baader *et al.*, 2003]. A typical scenario involves an ontology engineer teaming up with an expert to construct an ontology related to the domain of expertise of the latter with the aid of an ontology engineering tool such as SWOOP [`http://code.google.com/p/swoop`] or Protégé [`http://protege.stanford.edu`]. An example of this is the medical ontology SNOMED [Spackman *et al.*, 1997], whose formal substratum is the less expressive description logic $\mathcal{EL}$ [Baader, 2003].

One of the principal methods for testing the quality of a constructed ontology is for the domain expert to inspect and verify the computed *subsumption hierarchy*. Correcting such errors involves the expert pointing out that certain subsumptions are missing from the subsumption hierarchy, while others currently occurring in the subsumption hierarchy ought not to be there. A concrete example of this involves SNOMED, which classifies the concept `Amputation-of-Finger` as being subsumed by the concept `Amputation-of-Arm`. Finding a solution to problems such as these is known as *repair* in the DL community [Schlobach and Cornet, 2003], but it can also be seen as the problem of *contracting* by the subsumption statement $\texttt{Amputation-of-Finger} \sqsubseteq \texttt{Amputation-of-Arm}$.

The scenario also illustrates why we are concerned with belief contraction of belief *sets* (logically closed theories) and not *belief base* contraction [Hansson, 1999]. In practice, ontologies are not constructed by writing DL axioms, but rather using ontology editing tools, from which the axioms are generated automatically. Because of this, from the knowledge engineer's perspective, it is the belief set and not the axioms from which the theory is generated that is important.

2 Classical Belief Contraction

Before establishing our constructions, we give a very brief summary of classical belief contraction. For that we assume the reader is familiar with Classical Propositional Logic. Given a finitely generated propositional language $\mathcal{L}_\mathsf{P}$, if $X \subseteq \mathcal{L}_\mathsf{P}$, the set of sentences logically entailed by X is denoted by $Cn(X)$. A *belief set* is a logically closed set, i.e., for a belief set X, $X = Cn(X)$.

AGM [Alchourrón *et al.*, 1985] is the best-known approach to contraction. It gives a set of postulates characterizing all rational contraction functions. The aim is to describe belief contraction on the *knowledge level* of how beliefs are represented. Belief states are modelled by *belief sets* in a logic with a Tarskian consequence relation including classical propositional logic. The *expansion* of K by φ, $K + \varphi$, is defined as $Cn(K \cup \{\varphi\})$. *Contraction* is intended to represent situations in which an agent has to give up information from its current beliefs. Formally, belief contraction is a (partial) function from $\mathscr{P}(\mathcal{L}_\mathsf{P}) \times \mathcal{L}_\mathsf{P}$ to $\mathscr{P}(\mathcal{L}_\mathsf{P})$: the contraction of a belief set by a sentence yields a new set.

The AGM approach to contraction requires that the following set of postulates characterize *basic* contraction.

(K–1) $K - \varphi = Cn(K - \varphi)$
(K–2) $K - \varphi \subseteq K$
(K–3) If $\varphi \notin K$, then $K - \varphi = K$
(K–4) If $\not\models \varphi$, then $\varphi \notin K - \varphi$
(K–5) If $\varphi \equiv \psi$, then $K - \varphi = K - \psi$

*This paper is based upon work supported by the National Research Foundation under Grant number 65152.

$(K\text{–}6)$ If $\varphi \in K$, then $(K - \varphi) + \varphi = K$

Various methods exist for constructing basic AGM contraction. In this paper we focus on the use of *remainder sets*.

Definition 2.1 *For a belief set K, $X \in K \downarrow \varphi$ iff (i) $X \subseteq K$, (ii) $X \not\models \varphi$, and (iii) for every X' s.t. $X \subset X' \subseteq K$, $X' \models \varphi$. We call the elements of $K \downarrow \varphi$* remainder sets *of K w.r.t. φ.*

Remainder sets are belief sets, and $K \downarrow \varphi = \emptyset$ iff $\models \varphi$. AGM presupposes the existence of a suitable selection function for choosing between possibly different remainder sets.

Definition 2.2 *A selection function σ is a function from $\mathscr{P}(\mathscr{P}(\mathcal{L}_{\mathsf{P}}))$ to $\mathscr{P}(\mathscr{P}(\mathcal{L}_{\mathsf{P}}))$ such that $\sigma(K \downarrow \varphi) = \{K\}$, if $K \downarrow \varphi = \emptyset$, and $\emptyset \neq \sigma(K \downarrow \varphi) \subseteq K \downarrow \varphi$ otherwise.*

Selection functions provide a mechanism for identifying the remainder sets judged to be most appropriate, and the resulting contraction is then obtained by taking the intersection of the chosen remainder sets.

Definition 2.3 *For σ a selection function, $-_\sigma$ is a partial meet contraction iff $K -_\sigma \varphi = \bigcap \sigma(K \downarrow \varphi)$.*

One of the fundamental results of AGM contraction is a representation theorem which shows that partial meet contraction corresponds exactly with the six basic AGM postulates.

Theorem 2.1 ([Gärdenfors, 1988]) *Every partial meet contraction satisfies $(K\text{–}1)$–$(K\text{–}6)$. Conversely, every contraction function satisfying $(K\text{–}1)$–$(K\text{–}6)$ is a partial meet contraction.*

Two subclasses of partial meet deserve special mention.

Definition 2.4 *Given a selection function σ, $-_\sigma$ is a* maxichoice *contraction iff $\sigma(K \downarrow \varphi)$ is a singleton set. It is a* full meet *contraction iff $\sigma(K \downarrow \varphi) = K \downarrow \varphi$ whenever $K \downarrow \varphi \neq \emptyset$.*

Clearly full meet contraction is unique, while maxichoice contraction usually is not.

3 The Description Logic $\mathcal{EL}$

The language of $\mathcal{EL}$ ($\mathcal{L}_{\mathcal{EL}}$) is a sublanguage of the description logic $\mathcal{ALC}$ [Baader *et al.*, 2003] built upon a set of atomic *concept names* $\{A, B, \ldots\}$ and a set of *role names* $\{R, S, \ldots\}$, together with the constructors $\sqcap$ and $\exists$. Among the concept names are the distinguished concepts $\top$ and $\bot$. Complex concepts are denoted by $C, D, \ldots$ and are constructed according to the rule

$$C ::= A \mid \top \mid \bot \mid C \sqcap C \mid \exists R.C$$

The semantics of $\mathcal{EL}$ is the standard set theoretic Tarskian semantics. An *interpretation* is a structure $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$, where $\Delta^{\mathcal{I}}$ is a non-empty set called the *domain*, and $\cdot^{\mathcal{I}}$ is an *interpretation function* mapping concept names A to subsets $A^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$ and mapping role names R to binary relations $R^{\mathcal{I}}$ over $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$:

$$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}},\ R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}},$$
$$\top^{\mathcal{I}} = \Delta^{\mathcal{I}},\ \bot^{\mathcal{I}} = \emptyset$$

Given an interpretation $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$, $\cdot^{\mathcal{I}}$ is extended to interpret complex concepts in the following way:

$$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}},$$
$$(\exists R.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \exists b.(a, b) \in R^{\mathcal{I}} \text{ and } b \in C^{\mathcal{I}}\}$$

Given $\mathcal{EL}$ concepts C, D, $C \sqsubseteq D$ is a *general concept inclusion axiom* (GCI for short). $C \equiv D$ is an abbreviation for both $C \sqsubseteq D$ and $D \sqsubseteq C$. An *$\mathcal{EL}$-TBox* $\mathcal{T}$ is a set of GCIs.

An interpretation $\mathcal{I}$ satisfies an axiom $C \sqsubseteq D$ (noted $\mathcal{I} \models C \sqsubseteq D$) iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. $\mathcal{I} \models C \equiv D$ iff $C^{\mathcal{I}} = D^{\mathcal{I}}$.

An interpretation $\mathcal{I}$ is a *model* of a TBox $\mathcal{T}$ (noted $\mathcal{I} \models \mathcal{T}$) iff $\mathcal{I}$ satisfies every axiom in $\mathcal{T}$. An axiom $C \sqsubseteq D$ is a *consequence* of a TBox $\mathcal{T}$, noted $\mathcal{T} \models C \sqsubseteq D$, iff for every interpretation $\mathcal{I}$, if $\mathcal{I} \models \mathcal{T}$, then $\mathcal{I} \models C \sqsubseteq D$. Given a TBox $\mathcal{T}$, $Cn(\mathcal{T}) = \{C \sqsubseteq D \mid \mathcal{T} \models C \sqsubseteq D\}$. An *$\mathcal{EL}$ belief set* is then a TBox $\mathcal{T} = Cn(\mathcal{T})$.

For more details on $\mathcal{EL}$, the reader is referred to the work by Baader [2003].

4 $\mathcal{EL}$ Contraction

In this section we give the basic definitions for contraction in $\mathcal{EL}$. The presentation follows that for Horn contraction by Delgrande [2008] and Booth *et al.* [2009].

We want to be able to contract an $\mathcal{EL}$ TBox $\mathcal{T}$ with a set of GCIs Φ. The goal of contracting $\mathcal{T}$ with Φ is the removal of at least one of the axioms in Φ. For full propositional logic, contraction with a set of sentences is not particularly interesting since it amounts to contracting by their conjunction. For $\mathcal{EL}$ it is interesting though, since the conjunction of the axioms in Φ is not always expressible as a single $\mathcal{EL}$ axiom. Our starting point for defining contraction is in terms of remainder sets.

Definition 4.1 ($\mathcal{EL}$ Remainder Sets) *For a belief set $\mathcal{T}$, $X \in \mathcal{T}{\downarrow_{\mathcal{EL}}}\, \Phi$ iff (i) $X \subseteq \mathcal{T}$, (ii) $X \not\models \Phi$, and (iii) for every X' s.t. $X \subset X' \subseteq \mathcal{T}$, $X' \models \Phi$. We refer to the elements of $\mathcal{T}{\downarrow_{\mathcal{EL}}}\, \Phi$ as the $\mathcal{EL}$* remainder sets *of $\mathcal{T}$ w.r.t. Φ.*

$\mathcal{EL}$ remainder sets exist since $\mathcal{EL}$ is compact [Sofronie-Stokkermans, 2006] and has a Tarskian consequence relation. It is easy to verify that all $\mathcal{EL}$ remainder sets are belief sets. Also, $\mathcal{T}{\downarrow_{\mathcal{EL}}}\, \Phi = \emptyset$ iff $\models \Phi$.

We now proceed to define selection functions to be used for $\mathcal{EL}$ partial meet contraction.

Definition 4.2 ($\mathcal{EL}$ Selection Functions) *A partial meet $\mathcal{EL}$ selection function σ is a function from $\mathscr{P}(\mathscr{P}(\mathcal{L}_{\mathcal{EL}}))$ to $\mathscr{P}(\mathscr{P}(\mathcal{L}_{\mathcal{EL}}))$ s.t. $\sigma(\mathcal{T}{\downarrow_{\mathcal{EL}}}\, \Phi) = \{\mathcal{T}\}$ if $\mathcal{T}{\downarrow_{\mathcal{EL}}}\, \Phi = \emptyset$, and $\emptyset \neq \sigma(\mathcal{T}{\downarrow_{\mathcal{EL}}}\, \Phi) \subseteq \mathcal{T}{\downarrow_{\mathcal{EL}}}\, \Phi$ otherwise.*

Using these selection functions, we define partial meet $\mathcal{EL}$ contraction.

Definition 4.3 (Partial Meet $\mathcal{EL}$ Contraction) *Given a partial meet $\mathcal{EL}$ selection function σ, $-_\sigma$ is a partial meet $\mathcal{EL}$ contraction iff $\mathcal{T} -_\sigma \Phi = \bigcap \sigma(\mathcal{T}{\downarrow_{\mathcal{EL}}}\, \Phi)$.*

We also consider two special cases.

Definition 4.4 ($\mathcal{EL}$ Maxichoice and Full Meet) *Given a partial meet $\mathcal{EL}$ selection function σ, $-_\sigma$ is a* maxichoice $\mathcal{EL}$ contraction *iff $\sigma(\mathcal{T}{\downarrow_{\mathcal{EL}}}\, \Phi)$ is a singleton set. It is a* full meet $\mathcal{EL}$ contraction *iff $\sigma(\mathcal{T}{\downarrow_{\mathcal{EL}}}\, \Phi) = \mathcal{T}{\downarrow_{\mathcal{EL}}}\, \Phi$ when $\mathcal{T}{\downarrow_{\mathcal{EL}}}\, \Phi \neq \emptyset$.*

Example 4.1 *Let $\mathcal{T} = Cn(\{A \sqsubseteq B, B \sqsubseteq \exists R.A\})$. Then $\mathcal{T}{\downarrow_{\mathcal{EL}}}\, \{A \sqsubseteq \exists R.A\} = \{\mathcal{T}_1', \mathcal{T}_2'\}$, where $\mathcal{T}_1' = Cn(\{A \sqsubseteq B\})$, and $\mathcal{T}_2' = Cn(\{B \sqsubseteq \exists R.A, A \sqcap \exists R.A \sqsubseteq B\})$. So contracting with $\{A \sqsubseteq \exists R.A\}$ yields either $\mathcal{T}_1'$, $\mathcal{T}_2'$, or $\mathcal{T}_1' \cap \mathcal{T}_2' = Cn(\{A \sqcap \exists R.A \sqsubseteq B\})$.*

As pointed out by Booth *et al.* [2009], maxichoice contraction is not enough for logics that are less expressive than full propositional logic, like e.g. Horn logic. The same argument holds here for $\mathcal{EL}$, as the following example shows.

Example 4.2 *Recalling Example 4.1, when contracting* $\{A \sqsubseteq \exists R.A\}$ *from* $\mathcal{T} = Cn(\{A \sqsubseteq B, B \sqsubseteq \exists R.A\})$*, full meet yields* $\mathcal{T}_{fm} = Cn(\{A \sqcap \exists R.A \sqsubseteq B\})$*, while maxichoice yields either* $\mathcal{T}_{mc}{}' = Cn(\{A \sqsubseteq B\})$ *or* $\mathcal{T}_{mc}{}'' = Cn(\{B \sqsubseteq \exists R.A, A \sqcap \exists R.A \sqsubseteq B\})$*. Now consider the belief set* $\mathcal{T}_i = Cn(\{A \sqcap \exists R.A \sqsubseteq B, A \sqcap B \sqsubseteq \exists R.A\})$*. Clearly* $\mathcal{T}_{fm} \subseteq \mathcal{T}_i \subseteq \mathcal{T}_{mc}{}''$*, but no partial meet yields* $\mathcal{T}_i$*.*

Our contention is that $\mathcal{EL}$ contraction should include cases such as $\mathcal{T}_i$ above. Given that full meet contraction is deemed to be appropriate, it stands to reason that any belief set $\mathcal{T}_i$ bigger than it should also be seen as appropriate, *provided* that $\mathcal{T}_i$ does not contain any irrelevant additions. But since $\mathcal{T}_i$ is contained in some maxichoice contraction, $\mathcal{T}_i$ cannot contain any irrelevant additions. After all, the maxichoice contraction contains only relevant additions, since it is an appropriate form of contraction. Hence $\mathcal{T}_i$ is also an appropriate result of $\mathcal{EL}$ contraction.

Definition 4.5 (Infra-Remainder Sets) *For belief sets* $\mathcal{T}$ *and* X*,* $X \in \mathcal{T}\Downarrow_{\mathcal{EL}} \Phi$ *iff there is some* $X' \in \mathcal{T}\downarrow_{\mathcal{EL}} \Phi$ *s.t.* $(\bigcap \mathcal{T}\downarrow_{\mathcal{EL}} \Phi) \subseteq X \subseteq X'$*. We refer to the elements of* $\mathcal{T}\Downarrow_{\mathcal{EL}} \Phi$ *as the* infra-remainder sets *of* $\mathcal{T}$ *w.r.t.* Φ*.*

Note that all remainder sets are also infra-remainder sets, and so is the intersection of any set of remainder sets. Indeed, the intersection of any set of infra-remainder sets is also an infra-remainder set. So the set of infra-remainder sets contains *all* belief sets between some remainder set and the intersection of all remainder sets.

Definition 4.6 ($\mathcal{EL}$ Contraction) *An infra-selection function* τ *is a function from* $\mathscr{P}(\mathscr{P}(\mathcal{L}_{\mathcal{EL}}))$ *to* $\mathscr{P}(\mathcal{L}_{\mathcal{EL}})$ *s.t.* $\tau(\mathcal{T}\Downarrow_{\mathcal{EL}} \Phi) = \mathcal{T}$ *whenever* $\models \Phi$*, and* $\tau(\mathcal{T}\Downarrow_{\mathcal{EL}} \Phi) \in \mathcal{T}\Downarrow_{\mathcal{EL}} \Phi$ *otherwise. A contraction function* $-_\tau$ *is an* $\mathcal{EL}$ *contraction iff* $\mathcal{T} -_\tau \Phi = \tau(\mathcal{T}\Downarrow_{\mathcal{EL}} \Phi)$*.*

5 A Representation Result

Our representation result makes use of $\mathcal{EL}$ versions of all of the basic AGM postulates, except for the Recovery Postulate $(K-6)$. It is easy to see that $\mathcal{EL}$ contraction does not satisfy Recovery. As an example, take $\mathcal{T} = Cn(\{A \sqsubseteq C\})$ and let $\Phi = \{A \sqcap B \sqsubseteq C\}$. Then $\mathcal{T} - \Phi = Cn(\emptyset)$ and so $(\mathcal{T} - \Phi) + \Phi = Cn(\{A \sqcap B \sqsubseteq C\}) \neq \mathcal{T}$. In place of Recovery we have a postulate that closely resembles Hansson's [1999] Relevance Postulate, and a postulate handling the case when trying to contract with a tautology.

$(\mathcal{T}-1)$ $\mathcal{T} - \Phi = Cn(\mathcal{T} - \Phi)$

$(\mathcal{T}-2)$ $\mathcal{T} - \Phi \subseteq \mathcal{T}$

$(\mathcal{T}-3)$ If $\Phi \not\subseteq \mathcal{T}$, then $\mathcal{T} - \Phi = \mathcal{T}$

$(\mathcal{T}-4)$ If $\not\models \Phi$, then $\Phi \not\subseteq \mathcal{T} - \Phi$

$(\mathcal{T}-5)$ If $Cn(\Phi) = Cn(\Psi)$, then $\mathcal{T} - \Phi = \mathcal{T} - \Psi$

$(\mathcal{T}-6)$ If $\alpha \in \mathcal{T} \setminus (\mathcal{T} - \Phi)$, then there is a $\mathcal{T}'$ such that $\bigcap(\mathcal{T}\downarrow_{\mathcal{EL}} \Phi) \subseteq \mathcal{T}' \subseteq \mathcal{T}, \mathcal{T}' \not\models \Phi$, and $\mathcal{T}' + \{\alpha\} \models \Phi$

$(\mathcal{T}-7)$ If $\models \Phi$, then $\mathcal{T} - \Phi = \mathcal{T}$

Postulates $(\mathcal{T}-1)$–$(\mathcal{T}-5)$ are analogues of $(K$–$1)$–$(K$–$5)$, while $(\mathcal{T}-6)$ states that all sentences removed from $\mathcal{T}$ during a contraction of Φ must have been removed for a reason: adding them again brings back Φ. $(\mathcal{T}-7)$ simply states that contracting with a (possibly empty) set of tautologies leaves the initial belief set unchanged. We remark that $(\mathcal{T}-3)$ is actually redundant in the list, since it can be shown to follow mainly from $(\mathcal{T}-6)$.

Theorem 5.1 *Every* $\mathcal{EL}$ *contraction satisfies* $(\mathcal{T}-1)$–$(\mathcal{T}-7)$*. Conversely, every contraction function satisfying* $(\mathcal{T}-1)$–$(\mathcal{T}-7)$ *is an* $\mathcal{EL}$ *contraction.*

6 Concluding Remarks

Here we have laid the groundwork for contraction in the $\mathcal{EL}$ family of description logics by providing a formal account of *basic* contraction of a TBox with a set of GCIs.

Here we focus only on *basic* contraction. For future work we plan to investigate $\mathcal{EL}$ contraction for *full* AGM contraction, obtained by adding the *extended* postulates. We also plan to pursue our future investigations on repair in more expressive DLs.

References

[Alchourrón *et al.*, 1985] C. Alchourrón, P. Gärdenfors, and D. Makinson. On the logic of theory change: Partial meet contraction and revision functions. *J. of Symbolic Logic*, 50:510–530, 1985.

[Baader *et al.*, 2003] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors. *Description Logic Handbook*. Cambridge University Press, 2003.

[Baader, 2003] F. Baader. Terminological cycles in a description logic with existential restrictions. In *Proc. IJCAI*, 2003.

[Booth *et al.*, 2009] R. Booth, T. Meyer, and I. Varzinczak. Next steps in propositional Horn contraction. In *Proc. IJCAI*, 2009.

[Delgrande, 2008] J. Delgrande. Horn clause belief change: Contraction functions. In *Proc. KR*, pages 156–165, 2008.

[Gärdenfors, 1988] P. Gärdenfors. *Knowledge in Flux: Modeling the Dynamics of Epistemic States*. MIT Press, 1988.

[Hansson, 1999] S. Hansson. *A Textbook of Belief Dynamics*. Kluwer, 1999.

[Schlobach and Cornet, 2003] S. Schlobach and R. Cornet. Non-standard reasoning services for the debugging of DL terminologies. In *Proc. IJCAI*, pages 355–360, 2003.

[Sofronie-Stokkermans, 2006] V. Sofronie-Stokkermans. Interpolation in local theory extensions. In *Proc. IJCAR*, 2006.

[Spackman *et al.*, 1997] K.A. Spackman, K.E. Campbell, and R.A. Cote. SNOMED RT: A reference terminology for health care. *Journal of the American Medical Informatics Association*, pages 640–644, 1997.

Shifting Valence Helps Verify Contextual Appropriateness of Emotions

Michal Ptaszynski Pawel Dybala Wenhan Shi Rafal Rzepka Kenji Araki
Graduate School of Information Science and Technology, Hokkaido University
{ptaszynski,paweldybala,shibuka,kabura,araki}@media.eng.hokudai.ac.jp

Abstract

This paper presents a method for estimating contextual appropriateness of speaker's emotions, supported with the analysis of Contextual Valence Shifters (CVS), which determine the semantic orientation of the valence of emotive expressions. In the proposed method a Web mining technique is used to verify the contextual appropriateness of the emotions recognized by a system for affect analysis supported with CVS. A conversational agent equipped with this system can choose an appropriate conversational procedure. The proposed method is evaluated using two conversational agents. The use of CVS has shown an improvement in the method.

1 Introduction

In recent years there has been a rising tendency in Artificial Intelligence research to enhance Human-Computer Interaction by implementing human factors in machines [Treur, 2007]. One of the most important human factors is expressing and understanding emotions - a vital part of human intelligence [Salovey and Mayer, 1990]. The field of AI embracing this subject, Affective Computing focuses on recognizing the emotions of users in human-computer interaction from: facial expressions or voice. However, such methods, based on behavioral approach, ignore the pragmatics and context dependence of emotions.

This led to creation of Affect Analysis - a field focused on developing natural language processing techniques for estimating the emotive aspect of text. For the Japanese language, which this paper focuses on e.g., Ptaszynski et al. [2008] tried to estimate emotive aspect of utterances with lexical representations of emotive information in speech or Shi et al. [2008] used a Web mining technique. However, obtaining information about the emotion expressed by a user does not tell us anything about whether it is appropriate or not to express that emotion in a given situation e.g., expressing joy during a funeral would be very inappropriate. Salovey and Mayer [1990] proposed a framework of human Emotional Intelligence (EI). Its first part assumes acquiring by a person the abilities to **a)** identify emotions and **b)** discriminate between appropriate and inappropriate expression of emotions. The few attempts to implement the EI Framework into machines [Picard *et al.*, 2001] eventually still do not go beyond the first basic step - recognition of emotions and none of the present methods is capable to perform contextual affect analysis.

We present a method not only specifying what type of emotion was expressed, but also determining whether the emotion is appropriate for the context it appears in. We use Shi's method for gathering emotive associations from the Web and Ptaszynski's system for affect analysis of utterances [Ptaszynski *et al.*, 2008]. One of the problems with this system was confusing the valence polarity of emotive expressions in the last step of the analysis. To solve this problem we applied the analysis of Contextual Valence Shifters.

2 ML-Ask

Ptaszynski's et al. [2008] ML-Ask (Emotive Elements / Emotive Expressions Analysis System) performs affect analysis of utterances in two steps: 1) Analyzing the general emotiveness of utterances; 2) Recognizing the particular emotion types. The system is based on Ptaszynski's idea of two-part analysis of realizations of emotions in language into: **Emotive elements**, indicating that emotions have been conveyed, but not detailing what specific emotions there are, e.g. interjections, mimetic expressions, or vulgarities; **Emotive expressions**, parts of speech or phrases, that in emotive sentences describe emotional states, e.g. nouns, verbs or adjectives.

The emotive element database was divided into interjections, mimetics, endearments, vulgarities and representations of non-verbal emotive elements, such as exclamation mark or ellipsis [907 items in total]. The database of emotive expressions contains Nakamura's [1993] collection [2100 items in total]. For a textual input utterance provided by the user, the system searches in order for: 1) emotive elements to determine the emotiveness; 2) emotive expressions to determine the specific types of the conveyed emotions. The system uses Nakamura's 10 type-classification of emotions said to be the most appropriate for the Japanese language: [joy], [anger], [sadness], [fear], [shame], [fondness], [dislike], [excitement], [relief] and [surprise].

However, keyword-based extraction of emotive expressions caused misinterpretations in valence polarity determination. To solve this problem, we applied an analysis of Contextual Valence Shifters.

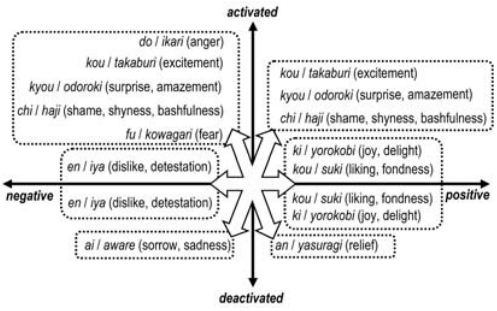


Figure 1: Nakamura's emotions on Russell's space.

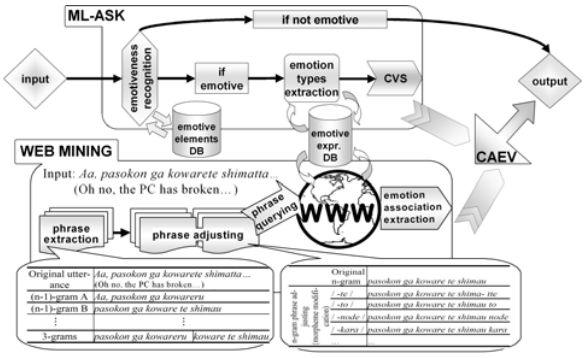


Figure 2: Flow chart of the CAEV procedure.

3 Contextual Valence Shifters in ML-Ask

The idea of Contextual Valence Shifters (CVS) was proposed by Polanyi and Zaenen [2004]. They distinguish 2 kinds of CVS: negations (changing the valence polarity of semantic orientation of an evaluative word) and intensifiers (intensifying the semantic orientation). Examples of CVS negations in Japanese are: *amari -nai* (not quite-), *-to wa ienai* (cannot say it is-), or *-te wa ikenai* (cannot+[verb]-). Intensifiers are: *sugoku-* (-a lot) or *kiwamete-* (extremely).

When a CVS structure is discovered, ML-Ask changes the valence polarity of the detected emotion. To specify the emotion types afterwards, we applied the 2-dimensional model of affect [Russell, 1980] which assumes that all emotions can be described in 2 dimensions: valence polarity (positive/negative) and activation (activated/deactivated). Nakamura's emotion types were mapped on the 2-dimensional model of affect and their affiliation to one of the spaces determined. The appropriate emotion after valence shifting is determined as the one with valence polarity and activation parameters different to the contrasted emotion (see Figure 1).

4 Web Mining Technique

As a verifier of appropriateness of the speaker's emotions recognized by ML-Ask we apply Shi's Web mining technique for extracting emotive associations from the Web [Shi *et al.*, 2008]. This technique performs common-sense reasoning about what emotions are the most natural to appear in a context of the utterance or which emotions should be associated with it. Emotions expressed, which are unnatural for the context are perceived as inappropriate. The technique is made up of three stages: **a)** extracting from an utterance n-gram phrases (of the length from one whole utterance to trigrams); **b)** modification of phrases ending with adjectives and verbs by the use of causality morphemes; and **c)** extraction of emotion associations from the Web using the prepared n-gram phrases as query inputs in Google search engine and cross-referencing them with the emotive expressions database.

5 Appropriateness Verification Procedure

The two described above systems are combined as follows. When ML-Ask discovers an emotive utterance and successfully specifies an emotion type, the Web mining technique verifies whether the expressed emotion type is appropriate for the context. The emotions commonsensically associating with the context of the utterance are extracted from the Web and the emotions discovered by ML-Ask in user's utterance are compared to the Web mining procedure results. If the two match, the emotions expressed by the user are determined as appropriate. In such situations, a conversational agent equipped with this method can choose an appropriate dialogue strategy (2 examples below).

Positive-appropriate emotion:
User: I'm so happy I passed the exam!
`ML-Ask:joy; Web mining:joy, excitement`
Agent: Yeah! That's great! `[sympathy]`
Negative-appropriate emotion:
User: I hate him for making a fool of me in front of everyone.
`ML-Ask:dislike; Web mining:anger, dislike`
Agent: You have a reason to be angry. `[empathy]`

However, if the verification procedure indicates that the expressed emotion is inappropriate for the context, the agent could undertake different measures e.g., helping the user to manage their emotions (2 examples below).

Positive-inappropriate emotion:
User: I'd be happy if he was hit by a car!
`ML-Ask:joy; Web mining:fear, sadness`
Agent: Is that what you really feel? `[counselling]`
Negative-inappropriate emotion:
User: I won the prize, but I feel so bored.
`ML-Ask:dislike, depression; Web mining:excitement, joy`
Agent: You should be happy! `[consolation]`

The flow chart of this Contextual Appropriateness of Emotions Verification (CAEV) procedure is presented in Figure 2.

6 Evaluation

In the evaluation experiment we used two conversational agents: 1) **Modalin**, a non-task oriented keyword-based conversational agent, which uses modality to enhance Web-based propositions for dialogue [Higuchi *et al.*, 2008]; 2) **Pundalin**, also a non-task oriented conversational agent, but equipped with Dybala's Pun generating system [Dybala *et al.*, 2009] to enhance interaction with a user.

13 users performed a ten turn conversation with both agents. All conversations were analyzed by ML-Ask (both without and with CVS). 6 of 26 conversations contained no specified emotional states and were excluded from the further

Table 1: 3 examples of the results of emotion appropriateness verification procedure (CAEV) with a separate display of the examples showing the improvement of the procedure after applying CVS.

Part of conversation in Japanese (English translation)	**ML-Ask output**		**Web Mining**	**CAEV**
USER: *Konpyuuta wa omoshiroi desu ne.* (Computers are so interesting!)	**positive** [joy]		**positive** [joy]	**appro-priate**
SYSTEM: *Sore wa oishii desu ka.* (Is it tasty?) [about instant noodles]	×		×	×
USER: *Oishii kedo, ore wa akita kana.* (Its tasty, but I've grown tired of it.)	**negative** [dislike]		**negative** [dislike]	**appro-priate**
Part of conversation in Japanese (English translation)	**ML-Ask baseline**	**with CVS**	**Web Mining**	**CAEV**
SYSTEM: *Sore wa omoshiroi tte* (Its so interesting!) [about conversation]	×	×	×	×
USER: *Sore hodo omoshiroku mo nakatta yo.* (It wasn't that interesting.)	positive [joy]	**negative** [dislike]	**negative** [fear], [sad]	**appro-priate**

evaluation. For the rest, the Web mining procedure determined whether the emotions expressed by the user were contextually appropriate. After that every conversation set was evaluated by 10 other people regarding what was the valence and the specific emotion types conveyed in conversations, and whether they were contextually appropriate.

6.1 Results and Discussion

We assumed that the results are correct when at least 4 participants of 10 per 1 set agree with the system. This is a fair ratio when we consider that it means that at least four people of ten provided exactly the same results as the system, which is a difficult thing to achieve in emotion research. The survey provided many positive results. In most cases affect analysis results were confirmed by humans, which proves the system is implementational in practice. ML-Ask supported with the CVS procedure acquired 85% of accuracy in recognition of particular emotion types (10% of improvement to the same method without CVS) and 90.0% of accuracy in emotion valence recognition (15% of improvement). Applying CVS analysis procedure improved also the performance of the contextual appropriateness verification procedure on the level of emotion types from 45.0% to 50.0% and valence from 50% to 55%. Some of the successful examples as well as the ones showing the improvement after applying CVS are shown in Table 1.

Since one of the agents was using humorous responses we also checked whether the jokes influenced the human-computer interaction. 67% of the emotions expressed in the conversations with Pundalin were positive whereas for Modalin 75% of the emotions were negative, which confirms that users tend to be positively influenced by the use of jokes in conversational agents [Dybala *et al.*, 2009].

7 Conclusions and Future Work

The paper presents one of the means to enhance a novel method performing and automated reasoning about whether the emotions expressed in a conversation are appropriate for the context. The system uses affect analysis to determine emotions conveyed by the speaker and a Web-based method performing approximate reasoning about which emotions should-, or are thought as natural (or commonsensical) to associate with the contents of the utterance. We enhanced the emotion types extraction in the baseline affect analysis system with Contextual Valence Shifters, which determine the semantic orientation of the valence of emotive expressions. We also applied a two-dimensional model of affect to determine which types of emotions are most probable to appear instead of the contrasted ones. Our system can provide a conversational agent with hints about what communication strategy would be the most desirable at a certain moment.

We are planning to use the system to gather a large ontology of actions/events with emotions associating with them divided according to their appropriateness. This task should be helpful in improving the newly created Japanese WordNet by supplementing it with Japanese version of WordNet Affect.

We were able to show that computing contextual appropriateness of emotions is a feasible task. Although the system's components (ML-Ask, Web mining) need improvement, it defines a new set of goals for Affective Computing. Contextual Affect Analysis including proper valence shifting is the next step towards practical implementation of Emotional Intelligence Framework in machines.

Acknowledgments

This research is partially supported by a Research Grant from the Nissan Science Foundation and The GCOE Program from Japan's Ministry of Education, Culture, Sports, Science and Technology.

References

[Dybala *et al.*, 2009] Pawel Dybala, Michal Ptaszynski, Rafal Rzepka and Kenji Araki. Humoroids - Talking Agents That Induce Positive Emotions with Humor, In *Proceedings of AAMAS'09*, 2009. (in print)

[Higuchi *et al.*, 2008] Shinsuke Higuchi, Rafal Rzepka and Kenji Araki. A Casual Conversation System Using Modality and Word Associations Retrieved from the Web. In *Proceedings of the EMNLP 2008*, pages 382-390, 2008.

[Nakamura, 1993] Akira Nakamura. *Kanjo hyogen jiten* [Dictionary of Emotive Expressions] (in Japanese), Tokyodo, 1993.

[Picard *et al.*, 2001] R. W. Picard, E. Vyzas, J. Healey. Toward Machine Emotional Intelligence: Analysis of Affective Physiological State, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 23, No. 10, pp. 1175-1191. 2001.

[Polanyi and Zaenen, 2004] L. Polanyi and A. Zaenen. Contextual Valence Shifters, *AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications*, 2004.

[Ptaszynski *et al.*, 2008] Michal Ptaszynski, Pawel Dybala, Rafal Rzepka and Kenji Araki. Effective Analysis of Emotiveness in Utterances Based on Features of Lexical and Non-Lexical Layers of Speech. In *Proceedings of NLP2008*, pages 171-174. 2008.

[Russell, 1980] James A. Russell. A circumplex model of affect. *J. of Personality and Social Psychology*, 39(6):1161-1178. 1980.

[Shi *et al.*, 2008] Wenhan Shi, Rafal Rzepka and Kenji Araki. Emotive Information Discovery from User Textual Input Using Causal Associations from the Internet. *FIT-08*,pp.267-268,2008.

[Salovey and Mayer, 1990] Peter Salovey and John D. Mayer. Emotional intelligence, *Imagination, Cognition, and Personality*, 9:185-211, 1990.

[Treur, 2007] Jan Treur. On Human Aspects in Ambient Intelligence. In *Proceedings of The First International Workshop on Human Aspects in Ambient Intelligence*, pages 5-10, 2007.

Context Discovery via Theory Interpretation

Oliver Kutz[1] and **Immanuel Normann**[2]
[1]Research Center on Spatial Cognition (SFB/TR 8), University of Bremen, Germany
[2]Department of Linguistics and Literature, University of Bremen, Germany
okutz@informatik.uni-bremen.de, normann@uni-bremen.de

Abstract

We report on ongoing work to apply techniques of automated theory morphism search to ontology matching and alignment problems. Such techniques are able to discover 'structural similarities' across different ontologies by providing theory interpretations of one ontology into another. In particular, we discuss two such scenarios: one where the signatures and logics of the component ontologies fit enough to directly translate one ontology into the other, called *stringent contexts*, and one where we need to lift the ontologies to first-order logic, possibly extended by definitonal axioms introducing extra non-logical symbols, called *conforming contexts*. We also sketch the techniques currently available for automating the task of finding theory interpretations in first-order logic and discuss possible extensions.

1 Introduction and Motivation

The problem of finding semantically well-founded correspondences between ontologies, possibly formulated in different logical languages, is a pressing and challenging problem. Ontologies may be about the same domain of interest, but may use different terms; one ontology might go into greater detail than another, or they might be formulated in different logics, whilst mostly formalising the same conceptualisation of a domain, etc. To allow re-use of existing ontologies and to find overlapping 'content', we need means of identifying these 'overlapping parts'.

Often, ontologies are mediated on an ad-hoc basis. Clearly, any approach relying exclusively on lexical heuristics or manual alignment is too error prone and unreliable, or does not scale. As noted for instance by [Meilicke *et al.*, 2008], even if a first matching is realised automatically using heuristics, a manual revision of such candidate alignments is still rather difficult as the semantics of the ontologies generally interacts with the semantics given to alignment mappings.

A new approach, that we currently explore, is to apply methods of automated theory interpretation search to the realm of ontologies. Such methods have been mainly developed for the application to formalised mathematics, and some of the techniques are specialised for theories formulated in first-order logic. Theory interpretations have a long history in mathematics generally, and are probably employed by any 'working mathematician' on a daily basis; the basic idea is the following: given two theories T_1 and T_2, find a mapping of terms of T_1 to terms of T_2 (a signature morphism, typically expected to respect typing) such that all translations of axioms of T_1 become provable from T_2. If such a theory interpretation is successfully provided, all the knowledge that has already been collected w.r.t. T_1 can be re-used from the perspective of T_2, using the translation (see [Farmer, 1994] for some examples from the history of mathematics). In this case, in mathematical jargon, we might say that T_2 **carries the structure** of T_1.

An abundance of notions of context have been studied in the literature (see e.g. [Serafini and Bouquet, 2004]). We here propose to use a notion of context, more precisely contextual interpretation of an ontology, inspired by the notion of theory interpretation from mathematics, which in practice is used as a *structuring device* for mathematical theories.

Certain, very basic structures, are found everywhere in mathematics. The most obvious example might be group theory. The basic abstract structure of a group can be re-interpreted in a more concrete setting, giving the group in question additional structure (think of the natural numbers, rings, vector-spaces, etc.). Re-using the metaphor mentioned above, we say that an ontology O_2 *carries the structure of* O_1, if the latter can be re-interpreted, by an appropriate translation σ, into the language of O_2 such that all translations of its axioms are entailed by O_2. In this case, informally, we consider the pair $\langle O_2, \sigma\rangle$ a **context** for O_1.

2 Ontology Interpretations and Context

For simplicity and lack of space, we here focus on the case of ontologies formulated in first-order logic (**FOL**) or standard description logics (DLs), and omit some of the technical details. More precisely, we limit the investigation here to the case of **FOL** and to DLs that have a *standard translation* into first-order logic (which of course are conservative).

In this setting, given an ontology O, i.e. in the case of DL a set of Abox, Tbox, and Rbox statements, the **signature** of O, denoted $\mathbf{Sig}(O)$ is the set of *non-logical symbols*, i.e. object, concept, and role names found in O, i.e. the set of nullary (constants), unary, and binary (or in general n-ary) predicates, when seen from the first-order perspective. Given two ontologies O_1, O_2, an **ontology signature morphism** (mop for short) is any map $\sigma : \mathbf{Sig}(O_1) \to \mathbf{Sig}(O_2)$ respecting typing, i.e. mapping concept names to concept names, role names to role names, and object names to object names. If such a σ exists, we call the signatures **fitting**, written $O_1 \boxdot\!\!\to O_2$.

By the **logic** of O, written $\mathcal{L}(O)$, we mean the set of *logical symbols* used in O (and thus provided by the underlying DL or **FOL**), and by the **language** of O we mean the set of all well-formed formulae that can be build from the signature $\mathbf{Sig}(O)$ using the logic $\mathcal{L}(O)$.

We distinguish between directly and indirectly interpretable ontologies. An ontology O_1 is **directly interpretable** into an ontology O_2 if $\mathcal{L}(O_1) \subseteq \mathcal{L}(O_2)$ (i.e. the set of logical symbols used in O_1 are a subset of those used in O_2),[1] written $O_1 \odot\!\!\rightarrow O_2$, and $\mathbf{Sig}(O_1), \mathbf{Sig}(O_2)$ are fitting, i.e. $O_1 \boxdot\!\!\rightarrow O_2$. Otherwise, we call them **indirectly interpretable**. To illustrate this concept, if for instance we have $\mathcal{L}(O_1) = \{\forall^{\mathbf{DL}}, \sqcap\}$ just using universal restrictions and conjunctions (where the underlying DL of O_1 is $\mathcal{ALC}$), and $\mathcal{L}(O_2) = \{\exists^{\mathbf{FOL}}, \vee\}$ just using existential quantification and disjunction (where **FOL** is the underlying logic), O_1 is only indirectly interpretable in O_2, because, although the latter is of course strictly more expressive, it requires a definition of the logical operators of $\mathcal{ALC}$ within **FOL**, accomplished via the standard translation into **FOL**.

The significance of these distinctions can be seen from:

Definition 1 (Canonical Sentence Translation) *Let O_1, O_2 be ontologies, and assume O_1 is directly interpretable into O_2. Then every* mop *σ will, by a straightforward structural induction over the grammar of that DL (or* **FOL***), yield a* ***sentence translation*** *$\hat{\sigma}$ of the axioms of O_1 along σ into the language of O_2.*

However, whenever either the logics or the signatures of the ontologies involved do not directly fit, there are a number of possible solutions to choose from (we can just extend the logic in question, we can extend definitionally the signature, or both).[2] We here provide a uniform solution as follows:

Definition 2 (Derived Sentence Translation) *Suppose O_1 is only indirectly interpretable in O_2. Let λ_i denote the standard translation from $\mathcal{L}(O_i)$ into* **FOL***, $O_i' = \lambda_i(O_i)$, and let $\mathcal{S} \supseteq \mathbf{Sig}(O_2')$ such that $\mathbf{Sig}(O_1') \boxdot\!\!\rightarrow \mathcal{S}$ for a signature morphism $\widetilde{\sigma}$ such that $\widetilde{\sigma}|_{\mathbf{Sig}(O_2')} =$ id (this always exists). Let $\widetilde{O_2'}$ result from O_2' by adding, for each element of $\mathcal{S} \setminus \mathbf{Sig}(O_2')$ a definitional axiom in the language of* **FOL***. By construction, $O_1' \odot\!\!\rightarrow \widetilde{O_2'}$. Now, define the* ***derived sentence translation*** *$\widetilde{\sigma}$ as the canonical sentence translation map in* **FOL***, induced by $\widetilde{\sigma}$.*

The situation in Def. 2 is illustrated in Fig. 1. We can now define the notion of *ontology interpretation*:

Definition 3 (Stringent Interpretations and Context) *Let O_1, O_2 be ontologies, suppose O_1 is directly interpretable into O_2, and let $\sigma : \mathbf{Sig}(O_1) \to \mathbf{Sig}(O_2)$ be a* mop*.*

$\sigma : O_1 \to O_2$ is called a ***stringent ontology interpretation*** *(*sop*) if $O_2 \models \hat{\sigma}(O_1)$. In this case, $\langle \sigma, O_2 \rangle$ is called a* ***stringent context*** *for O_1.*

Stringent interpretations cover the case where we can 'embed', within the same logic, one ontology into another, thus 'strictly aligning' the resp. terminologies. Let us next look at the more complex heterogeneous case of only indirectly interpretable ontologies.

Definition 4 (Conforming Interpretations and Context) *Let O_1, O_2 be ontologies, and suppose O_1 is only indirectly interpretable into O_2. Let σ be a maximally partial signature morphism, and assume $\widetilde{\sigma} : \mathbf{Sig}(O_1') \to \mathbf{Sig}(\widetilde{O_2'})$ be a* mop *in* **FOL***, where $\mathfrak{S}$ is the set of additional definitional axioms.*

$\widetilde{\sigma} : O_1 \to O_2$ is called a ***conforming ontology interpretation*** *(*cop*) if $\widetilde{O_2'} \models \widetilde{\sigma}(O_1')$. In this case, $\langle \widetilde{\sigma}, \mathfrak{S}, O_2 \rangle$ is called an O_1-****conforming context****.*

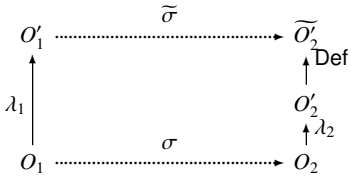


Figure 1: Ontology Interpretations

Note that the notion of a conforming context is closely related to the *heterogeneous refinements* defined in [Kutz *et al.*, 2008]: namely, given any O_1-conforming context $\langle \widetilde{\sigma}, \mathfrak{S}, O_2 \rangle$, O_2 is a heterogeneous refinement of O_1 where the transition from O_2 to $\widetilde{O_2'}$ is not only conservative but in fact definitonal.

Here is an illustrative example from mathematics:

Example 5 (Lattices and Partial Orders) *Consider P as the theory of partial-orders with $\mathbf{Sig}(P) = \{\leq\}$ and let L be the theory of lattices with $\mathbf{Sig}(L) = \{\sqcap, \sqcup\}$. These are both first-order theories, so we have $P \odot\!\!\rightarrow L$. However, the signatures obviously do not fit as L has no binary relations, so we have $P \not\boxdot\!\!\rightarrow L$. Extend the signature of L by a binary relation symbol $\sqsubseteq$ (which makes the signatures fit by the mapping $\widetilde{\sigma} :\leq \mapsto \sqsubseteq$), and define $\mathfrak{S} = \{\forall a, b. a \sqsubseteq b \leftrightarrow a \sqcup b = a\}$. This is a definitional axiom. It can now be seen that $L \cup \mathfrak{S} \models \widetilde{\sigma}(P)$, i.e. $\widetilde{\sigma}$ is a* cop*, and thus $\langle \widetilde{\sigma}, \mathfrak{S}, L \rangle$ is a P-conforming context.*

Thus, we may say that lattices carry the structure of partial orders. It should be obvious that both these theories also define central structures for ontology design.

3 Automated Discovery of Contexts

The goal of discovering ontology interpretations may be rephrased as the problem of finding all those ontologies in a large repository $\mathfrak{R}$ that could serve as a (stringent or conforming) context for a given ontology O_1. More formally, given O_1, we are looking for the sets

$$\{O_2 \in \mathfrak{R} \quad | \quad \langle \sigma, O_2 \rangle \text{ stringent context for } O_1\}$$

and

$$\{O_2 \in \mathfrak{R} \quad | \quad \langle \widetilde{\sigma}, \mathfrak{S}, O_2 \rangle \text{ conforming context for } O_1\}$$

In case of ontologies formalised in **FOL**, this task is undecidable, whereas for ontologies formalised in DL it is generally decidable. I.e., given the ontologies O_1, O_2, and a signature morphism σ from O_1 to O_2, it is decidable whether the σ-translated axioms of O_1 are entailed by O_2. However, the combinatorial explosion yielded by trying to find all possible symbol mappings between two given ontologies makes such a brute force approach unpractical.

To obtain one of the answer sets above in reasonable time (i.e seconds or minutes), we necessarily have to relax our initial goal towards an approximation of the set of all possible contexts for a given ontology. In summary, our approach for

[1] This is a purely syntactic and somewhat lax definition which we adopt here only for lack of space; a more elaborate definition would be defined via a notion of logic translation using e.g. institution comorphims, see [Kutz *et al.*, 2008].

[2] E.g. the OneOf constructor found in many description logics allowing a finite enumeration of the elements of a concept is also expressible as a disjunction of nominals, and conversely. Such translations/simulations can be handled by a library of logic translations.

the first-order case[3] is based on formula matching modulo an equational theory—elaborated in detail in [Normann, 2009]. We want to outline this in the following.

Suppose we are given a source ontology O_1 and a target ontology O_2, which we assume have been translated to first-order via the standard translations. In the first step, we normalise each sentence of these ontologies according to a fixed equational theory. The underlying technique basically stems from term-rewriting: rewrite rules represent an equational theory such that all sentence transformations obtained through these rules are in fact equivalence transformations, e.g. such as $\neg A \sqcap \neg B \mapsto \neg(A \sqcup B)$. A normal form of a convergent rewrite system is then the unique representative of a whole equivalence class of sentences. The goal of normalisation is thus to identify (equivalent) expressions such as $\neg(\exists R.A \sqcap B)$ and $\neg B \sqcup \forall R.\neg A$.

In the next step, we try to translate each normalised axiom φ from O_1 into O_2, i.e. we seek a sentence ψ in O_2 and a translation σ such that $\sigma(\varphi) = \psi$. Note that potentially each axiom can be translated to several target sentences via different signature morphisms. To translate all axioms of O_1 into O_2, there must be a combination of *compatible* signature morphisms[4] determined from the previous, single sentence matchings. This task is also known as (consistent) many-to-many formula matching. In fact many-to-many formula matching modulo some equational theory is already applied in automated theorem proving (ATP) [Graf, 1996]. However, our approach is different in a crucial aspect: it allows for significant search speed up. We are normalising all ontologies as soon as they are inserted into the repository, i.e. not at cost of query time. Only the normalisation of the query ontology is at query time. Moreover, the normal forms not just allow for matching modulo some equational theory, but also enable a very efficient matching pre-filter based on skeleton comparison. A sentence skeleton is an expression where all (non-logical) symbols are replaced by placeholders. E.g., $\square \sqsubseteq \square \sqcup \square$ is the skeleton of $A \sqsubseteq B \sqcup C$. Obviously, two sentences can only match if they have an identical skeleton. Since syntactic identity can be checked in constant time, a skeleton comparison is a very efficient pre-filter for sentence matching.

All the presented techniques were developed in the context of formalised mathematics and a tool for the automated discovery of theory interpretations in first-order logic has already been implemented [Normann, 2009], and is currently being integrated into the Hets system [Mossakowski *et al.*, 2007]. This has been used for experiments on a **FOL** version of the Mizar library [MizarKB] that contains about 4.5 million formulae distributed in more than 45.000 theories, and thus is the world's largest corpus of formalised mathematics. Experiments where each theory was used as source theory for theory interpretation search in the rest of the library demonstrated the scalability of our approach. On average, a theory interpretation search takes about one second and yields 60 theory interpretations per source theory.

4 Discussion and Future Work

Because of the encouraging results in formalised mathematics, we are currently adopting and modifying these techniques for the application in the realm of ontologies. The techniques we have sketched above are directly applicable to two of the cases we have discussed: to searching for stringent contexts in the case where both ontologies are formalised in **FOL**, and to the case of conforming context where the logics can be in a DL or **FOL**, but where no definitional axioms are required, or where they are added manually. Automated search of such definitions is not yet supported.

Of course, there is no guarantee that what is successful for mathematical theories is equally successful for formal ontologies, and some of the characteristics and features regularly found in ontologies are problematic. For instance, ontologies are often formalised in DL as opposed to first- or higher-order logics used in formal mathematics. Hence, formal mathematical theories are in general constituted by much more complex axioms than formal ontologies (many ontologies have no other axioms than is-a hierarchies). The lower complexity of ontology axioms has the effect that many axioms share the same skeleton. This makes skeletons a less effective pre-filter, which means that the reduction of the search space for candidate signature morphisms will be less significant.

Initial experiments on DL ontologies already suggested some ideas on how to overcome these problems in future work:

- Interactive search space reduction: the user should be able to enforce some mappings of non-logical symbols—often some mappings are explicitly intended.
- Exploitation of the decidability of DLs.
- Specialised normal forms designed for various DLs.

[3] In principle, these methods can be applied to any formalised content as long as the entailment relation obeys certain properties (as specified e.g. in entailment systems [Meseguer, 1989]).

[4] Two signature morphisms are compatible if they translate all their common symbols equally.

Acknowledgements

We gratefully acknowledge the financial support of the European Commission through the OASIS project (Open Architecture for Accessible Services Integration and Standardisation) and the Deutsche Forschungsgemeinschaft through the Research Center on Spatial Cognition (SFB/TR 8). The authors would like to thank Joana Hois for fruitful discussions.

References

W. M. Farmer. Theory Interpretation in Simple Type Theory. In *Higher-Order Algebra, Logic, and Term Rewriting*, volume 816 of *LNCS*, pages 96–123. Springer, 1994.

P. Graf. *Term Indexing*, volume 1053 of *Lecture Notes in Computer Science*. Springer, 1996.

O. Kutz, D. Lücke, and T. Mossakowski. Heterogeneously Structured Ontologies—Integration, Connection, and Refinement. In *Advances in Ontologies (KROW-08)*, volume 90 of *CRPIT*, pages 41–50. ACS, 2008.

C. Meilicke, H. Stuckenschmidt, and A. Tamilin. Reasoning Support for Mapping Revision. *Journal of Logic and Computation*, 2008.

J. Meseguer. General logics. In *Logic Colloquium 87*, pages 275––329. North Holland, 1989.

Mizar Mathematical Library. `http://www.mizar.org/library`.

T. Mossakowski, C. Maeder, and K. Lüttich. The Heterogeneous Tool Set. In Orna Grumberg and Michael Huth, editors, *TACAS 2007*, volume 4424 of *LNCS*, pages 519–522. Springer, 2007.

I. Normann. *Automated Theory Interpretation*. PhD thesis, Department of Computer Science, Jacobs University, Bremen, 2009.

L. Serafini and P. Bouquet. Comparing Formal Theories of Context in AI. *Artificial Intelligence*, 155:41–67, 2004.

Contextualized OWL-DL KB for the management of OWL-S effects

Domenico Redavid
Department of Computer Science
University of Bari
redavid@di.uniba.it

Ignazio Palmisano
Department of Computer Science
University of Liverpool
ignazio@csc.liv.ac.uk

Luigi Iannone
Department of Computer Science
University of Manchester
iannone@cs.man.ac.uk

Abstract

Description Logics are monotonic, therefore an OWL DL knowledge base has no primitives for retraction. This absence is main obstacle for the implementation of effective Semantic Web Services (SWS) platforms that allow for composition and, in general, orchestration of services. This paper explores a possible workaround based on a notion of *context* developed for OWL-DL knowledge bases.

1 Introduction

In order to accomplish automated SWS [McIlraith *et al.*, 2001] composition, a reasoning system must combine Web Services that collectively achieve the user's goal. This involves resolving constraints between Web Service Inputs, Outputs, Preconditions and Results (IOPRs), where a Result can be an output and/or an effect. Moreover, the composition problem as defined in [Sycara *et al.*, 2003] distinguishes beetwen information-providing and world-altering services. The latter perform world alterations as side effects. Managing such modifications in a OWL knowledge base is an open issue. Let us suppose that we have a SWS S whose *effect* is to change the state of a process activity A. An example of ontology describing the knowledge base that models *Activities* and *States* follows:

```
Ontology(<http://www.owl-ontologies.com/StateOntology.owl>
Namespace
  (a = <http://www.owl-ontologies.com/StateOntology.owl#>)
 ObjectProperty(a:hasState Functional
        domain(a:Activity)range(a:State))
 Class(a:Activity   partial)
 Class(a:Delivery   partial a:Activity)
 Class(a:Finished   partial a:State)
 Class(a:InProgress partial a:State)
 Class(a:Scheduled  partial a:State)
 Class(a:State      partial)
 Individual(a:DeliveryOrder1  type(a:Delivery)
      value(a:hasState a:ScheduledType1))
 Individual(a:FinishedType1   type(a:Finished))
 Individual(a:InProgressType1 type(a:InProgress))
 Individual(a:ScheduledType1  type(a:Scheduled))
 DisjointClasses(a:Finished   a:Scheduled)
 DisjointClasses(a:InProgress a:Scheduled)
 DisjointClasses(a:Finished   a:InProgress))
```

The functional property *hasState* indicates the state of an activity: *Scheduled*, *InProgress* or *Finished*. In the example ontology there is an activity called *DeliveryOrder1* that is set to *ScheduledType1* (Fig. 1 (left)). When *DeliveryOrder1* goes in production, the service S is invoked in order to change the state of this activity to *InProgressType1*. This needs a corresponding modification in the knowledge base. However, OWL DL is monotonic, hence it is not possible to modify an *assertion*, but only to add new ones. Therefore, the execution of the service generates the situation reported in Fig. 1 (right), where the same activity has two states. Currently, a reasoner would interpret this as an extra assertion of the *hasState* property, while it should be interpreted as a modification of the assertion itself. Furthermore, we now have two distinct values assigned to a functional property for the same instance, i.e., an inconsistency. This simple example shows how the absence of retraction primitives makes the management of common situations very hard. There is ongoing work on a language extension that could make these situations manageable [Baader *et al.*, 2003], but, in order to fit this solution for the Semantic Web, a revision of the current OWL specification[1] is required. In this paper we propose a temporary solution that can be used to avoid inconsistencies caused by the service effects during the application of an existing composition method based on OWL-S[2]. This solution relies on the notion of RDF[3] contexts adapted to OWL-DL and detailed in the following.

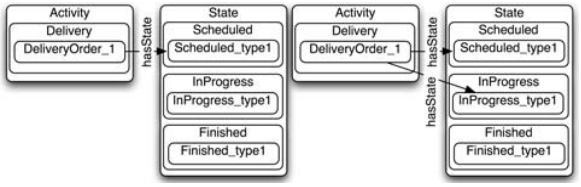


Figure 1: The initial value of hasState property (left) and the KB after the execution of service S (right).

2 OWL-S Composer

In this section we present a platform for service composition that makes use of our solution to overcome the monotonicity of OWL DL as opposed to the non monotonic nature of world altering services. The OWL-S composition method [Redavid *et al.*, 2008] is based on Semantic Web Rule Language

[1] http://www.w3.org/TR/owl-ref/
[2] http://www.w3.org/Submission/OWL-S/
[3] http://www.w3.org/TR/rdf-concepts/

(SWRL) [Horrocks *et al.*, 2005]. SWRL extends the set of OWL axioms to include *Horn-like* rules. The proposed rules are in the form of implications between an antecedent (body) and consequent (head); both consist of zero or more conjunctive atoms. The intended meaning can be read as: whenever the conditions specified in the antecedent hold, then the conditions specified in the consequent must also hold. A rule with conjunctive consequent can be transformed into multiple rules each with an atomic consequent (Lloyd-Topor transformations [Lloyd, 1987]). In order to achieve a SWRL based service composition mechanism two steps are required:

1. Encoding of the OWL-S services process model descriptions into SWRL rules of the form: $BODY\ (\ inCondition\ \wedge\ Preconditions\ )$ and $HEAD\ (\ \{output\}\ \wedge\ Effects\ )$
2. Use of a SWRL composer that takes as Input a set of SWRL rules and a goal identified by a SWRL atom, and returns every possible path (plan) built combining the available SWRL rules in order to achieve the input goal.

The computed plans can then be re-encoded as OWL-S composite processes or into another XML-based orchestration languages (e.g., BPEL[4]).

2.1 Contextualization of Knowledge Bases

Contextualization of knowledge bases has many different definitions that disagree on what a context actually is. The definition we refer to [Stoermer *et al.*, 2007] is:

> "A *context* is a set of properties or of parameters (e.g., property values) that hold for a specific subset of information in a knowledge base."

Contextualized RDF knowledge bases may be used to represent OWL axioms and assertions, since OWL has a standard RDF mapping[5]. Aspects such as temporary evolution of a knowledge base, relevance and trust, require a model that does not consist only of a set of universally true statements, without any reference to a situation or a point in time. This limitation depends on a design decision, i.e., the fact that RDF has been designed to represent binary relations; the representation of n-ary relations[6], makes use of anonymous nodes to attach the extra parameters. The proposed solution to these issues is the use of *contexts* to separate statements that refer to different contextual information (parameters) [Bouquet *et al.*, 2005]; parameters cannot explicitly be tied to the statements because of the simplicity of the RDF model.

Contextualization of a knowledge base amounts therefore to the identification of sections of the knowledge base owned or generically used by a specific software entity. In this acception, each agent or each Web Service we consider owns a context, in which its knowledge holds true; it is possible that contexts for which different parameters are specified contain contradictory information, e.g., a knowledge base may be in a situation where multiple different values for functional properties are specified, while the single contexts are consistent and correct on their own.

In the example presented in Sec. 1, one such situation is sketched: values for a functional property are being generated by SWRL rules, while there is no way to remove values previously specified and therefore already in the knowledge base; the ability to specify which assertion is valid according to external conditions, in this case the execution of a service, would give us a solution to the problem of how to consistently evolve the knowledge base. Our proposal is to solve this specific problem by introducing the time parameter, which will enable us to split the knowledge base in contexts where each context records the time at which its assertions are valid. How to do this efficiently is explained in Sec. 2.2.

The same Section shows how to relate the contexts through *Compatibility Relations* (CR) [Stoermer *et al.*, 2007], that formalize under which circumstances knowledge from other contexts becomes relevant.

2.2 OWL-S Effects Management

In this paragraph we show how RDF contexts could be used during the application of the OWL-S composition method presented in Sec. 2 so as to avoid inconsistencies caused by a particular type of OWL-S effects. In detail, the considered effects consist of a single SWRL Atom that can assume a finite number of values having an order relation. Consider the following OWL-S sample service.

```
SERVICE NAME ChangeStateService
  DOMAIN ONTOLOGY
    http://www.owl-ontologies.com/StateOntology.owl
  INPUTS:  I1 = {Activity}, I2 = {State}
  OUTPUTS: O1 = {State}
  PRECONDITION
      P1 = {hasState(I1,?y)  and  hasNextState(?y,I2)}
  INCONDITION
    InC1 = {Activity(I1)     and  State(I2) and
            hasState(I1,?y)  and  hasNextState(?y,I2)}
  EFFECT   E1 = {hasState(I1,I2)}
```

This service changes the state of an activity and is based on the ontology reported in Sec. 1, extended with the property *hasNextState*. This property, whose both domain and range are the *State* class, specifies the order of the states: *Scheduled* → *InProgress* → *Finished*; the successor of *Finished* is itself.

Its process model has the following characteristics:

- **Inputs.** $I1$ identifies an *Activity* and $I2$ identifies the required *State* for $I1$.
- **Output.** The current state of the *Activity*.
- **Precondition.** Check that the state for the input activity is legal, e.g., if an activity is not *Scheduled*, it cannot become *InProgress*.
- **InCondition.** Requirements over inputs in order to have output and effect specified for the service.
- **Effect.** The change in the activity state due to the execution of the service.

As described above, we encode *Effects* as the *head* of a SWRL rule, therefore we obtain the following:

$$Activity(?I1) \wedge State(?I2) \wedge hasState(?I1,?y) \wedge hasNextState(?y,?I2) \rightarrow hasState(?I1,?I2)$$

[4] http://www.ibm.com/developerworks/webservices/library/ws-bpel/

[5] http://www.w3.org/TR/owl-semantics/mapping.html

[6] http://www.w3.org/TR/swbp-n-aryRelations/

This rule, together with the ontology representing its semantics, is added to the global Knowledge Base containing other services. Applying our composition method, a plan containing the added rule is created. Notice that there are two axioms involving the *hasNext* property, namely its definition; and the rule axiom. The rule asserts a new state based on the previous one. This would make the KB inconsistent, as explained in Sec. 1, because the *hasNext* property is functional. In this case, we can avoid the inconsistency applying the contextualization method. Analyzing the Knowledge Base in the example, we could have three contexts with an order relation from *Scheduled* to *Finished* as in Fig. 2. This order relation can be properly modeled with the Context Relation EXTENDS [Stoermer *et al.*, 2007]. Initially, we have a base context C_1 where the State of a given activity is set to *scheduled*; the second context C_2 EXTENDS C_1 and it has the State set to *InProgress*; finally, the third context C_3 EXTENDS C_2 and it has the State of the activity set to *Finished*.

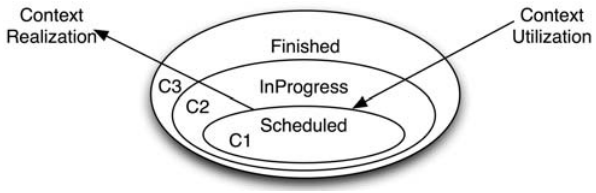


Figure 2: The contexts for the state example

Therefore, during the composition process it is necessary to keep separated the *hasState* property assertion appearing in the body, which refers to the current context, from one appearing in the head, which refers to the next context.

More in general, this kind of problem can be summarized as caused by rules which generate assertions that the ontology somehow limits; in our example, *hasNext* is functional, therefore there is a limit to the number of assertions of *hasNext* for the same individual. This pattern, that spans both the description of the service and the ontology, may be used in order to foresee, during the planning, where inconsistencies may raise, and therefore contextualize the knowledge base to avoid the problem, without waiting for the inconsistencies to manifest themselves, which could be very time consuming.

The procedure can be then resumed as follows:

1. While no problem patterns are identified, the reasoning can be applied on the current KB;
2. When an *Effect* is detected as possible origin for inconsistency, there are two cases: a) the *Effect* is not already treated during the composition process. In order to explicit all possible contexts (*Realization* in Fig. 2) the splitting mechanism for inconsistency treatment is applied for each SWRL atom value that generates potentially inconsistent assertions. The last generated context becomes the current KB; b) the *Effect* is already treated in the composition process. To comply with the SWRL atom values order relation, the next context becomes the current KB (*Utilization* in Fig. 2).

A composition will be valid if and only if all the contexts are expected to be applied. Furthermore, since the composition method is based on a backward search algorithm, the first context to be considered during execution must be the last context obtained with the CR EXTENDS during planning. Therefore, a composition is valid if all the contexts belonging to the hierarchy are used exactly in the inverse of the order obtained during the splitting process.

Conclusions and Future Works

The proposed method is a possible way to prevent inconsistencies caused by the absence of retraction primitives, through the use of patterns. The discussed example shows how an inconsistency can be generated by means of a particular type of OWL-S effects. The current implementation needs to be verified in larger deployment studies in order to evaluate its viability; it is also necessary to discover other situations that might create inconsistency problems, so that suitable patterns can be devised and implemented.

References

[Baader *et al.*, 2003] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, 2003.

[Bouquet *et al.*, 2005] Paolo Bouquet, Luciano Serafini, and Heiko Stoermer. Introducing Context into RDF Knowledge Bases. In Paolo Bouquet and Giovanni Tummarello, editors, *SWAP*, volume 166 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2005.

[Horrocks *et al.*, 2005] Ian Horrocks, Peter F. Patel-Schneider, Sean Bechhofer, and Dmitry Tsarkov. OWL rules: A proposal and prototype implementation. *J. of Web Semantics*, 3(1):23–40, 2005.

[Lloyd, 1987] John W. Lloyd. *Foundations of Logic Programming* 2^{nd} *Edition*. Springer, 1987.

[McIlraith *et al.*, 2001] Sheila A. McIlraith, Tran Cao Son, and Honglei Zeng. Semantic Web Services. *IEEE Intelligent Systems*, 16(2):46–53, 2001.

[Redavid *et al.*, 2008] Domenico Redavid, Luigi Iannone, Terry R. Payne, and Giovanni Semeraro. OWL-S Atomic Services Composition with SWRL Rules. In Aijun An, Stan Matwin, Zbigniew W. Ras, and Dominik Slezak, editors, *ISMIS*, volume 4994 of *Lecture Notes in Computer Science*, pages 605–611. Springer, 2008.

[Stoermer *et al.*, 2007] Heiko Stoermer, Paolo Bouquet, Ignazio Palmisano, and Domenico Redavid. A Context-Based Architecture for RDF Knowledge Bases: Approach, Implementation and Preliminary Results. In Massimo Marchiori, Jeff Z. Pan, and Christian de Sainte Marie, editors, *RR*, volume 4524 of *Lecture Notes in Computer Science*, pages 209–218. Springer, 2007.

[Sycara *et al.*, 2003] Katia P. Sycara, Massimo Paolucci, Anupriya Ankolekar, and Naveen Srinivasan. Automated discovery, interaction and composition of Semantic Web services. *J. Web Sem.*, 1(1):27–46, 2003.

Modeling the External Quality of Context to Fine-tune Context Reasoning in Geospatial Interoperability

Tarek Sboui[12], Yvan Bédard[12], Jean Brodeur[31], Thierry Badard[1]
[1] Department of Geomatic Sciences, Université Laval, Quebec, Qc, G1K 7P4, Canada
[2] NSERC Industrial Research Chair in Geospatial Databases for Decision-Support
[3] Natural Resources Canada, CIT, 2144-010 King West St, Sherbrooke, Qc, J1J 2E8, Canada
Tarek.Sboui.1@ulaval.ca, Yvan.Bedard@scg.ulaval.ca, brodeur@nrcan.gc.ca,
Thierry.Badard@scg.ulaval.ca

Abstract

Context reasoning is the process of drawing conclusions from existing context information, and is considered crucial to geospatial interoperability. However, such reasoning still remains a challenge because context may be incomplete or not appropriate for a specific use. Thus, evaluating and modeling the external quality (fitness-for-use) of context information is important for reasoning about context. In this paper, we propose an ontology-based approach to model the external quality of context information. This approach aims at fine-tuning context reasoning and hence enhancing geospatial interoperability.

1 Introduction

Geospatial interoperability has been widely recognized as an efficient paradigm for reusing geospatial data [Bishr, 1998; Brodeur, 2004]. However, geospatial interoperability still faces problems due to the heterogeneity of semantically related data (e.g., synonyms and homonyms). In order to deal with such heterogeneity, we need to reason about context information associated with semantically related data [Bishr, 1998]. Context information may be used in several ways to capture data semantics. We distinguish between two kinds of context: *production context* and *use context*. Production context is any information that can be specified explicitly or implicitly to indicate the circumstances in which data was created and intended to be used (e.g., the method of data collection). When it is explicitly represented, production context is referred to as metadata. On the other hand, use context is any information about the circumstances that surround user's application (e.g., reference system, scale). Context can be thematic (e.g., data acquisition method), spatial (e.g., spatial reference system) or temporal (e.g., time of data acquisition).

Production context may be incomplete or inappropriate for a specific use (use context). This may represent a risk that affects the context reasoning process [Henricksen and Indulska, 2004; Bikakis *et al.*, 2007]. In order to manage such a risk, we need to verify and compare the degree of appropriateness of the production context with regard to a given application (use context). This degree can be indicated by the external quality of context production (i.e. its fitness-for-use). Consequently, evaluating and modeling the external quality of production context is important for context reasoning in geospatial interoperability. For example, evaluating the external quality can help to choose between two heterogeneous data elements[1] (i.e. the element fitting better for a given application) and hence, avoid the risk of choosing an inappropriate element.

In previous work we proposed a set of indicators for the external quality of the explicit production context (i.e., metadata) and a method to evaluate those indicators in the context of geospatial semantic interoperability [Sboui *et al.*, 2009]. In this paper, we propose an ontology-based approach to model the external quality of the production context with respect to the application for which the interoperability is carried out. This approach aims at fine-tuning context reasoning and hence, enhancing geospatial interoperability.

In the next section, we briefly present the existing approaches to reason about context. In Section 3, we present a set of external quality indicators for the production context; and we propose a model to represent and reason about the external quality of the production context. We conclude and present further works in Section 4.

2 Approaches for context modeling and reasoning

Context modeling and reasoning has attracted the attention of many researchers [Giunchiglia, 1993; Benerecetti *et al.*, 2000; Gu *et al.*, 2004; Frank, 2007]. Giunchiglia [1993] emphasized that effective context reasoning can be build based on a manageable subset of that context. Benerecetti *et al.* [2000] provided a theory foundation for reasoning about

[1] An element can be a word/phrase, a set of graphic symbols, or a combination of both.

context using partial, approximate, and perspectival representations of the world. I recent years, several approaches for context reasoning have been based on different techniques (e.g., ontology, predefined sets of rules, and probability). Ontology-based and rule-based reasoning are the two major approaches [Gu *et al.*, 2004; Bikakis *et al.*, 2007].

Ontology-based reasoning

Ontology, as a formal and explicit specification of a shared conceptualization, is considered as an efficient technique for modeling context enabling software agents to interpret and reason about context information [Gu *et al.*, 2004; Frank, 2007]. Ontology-based approaches use Semantic Web technologies (e.g., RDF(S) and OWL) to model and reason about context information. These approaches are the most commonly used thanks to their formal structure and high expressiveness.

Rule-based reasoning

These approaches are based on predefined sets of rules that aim at verifying the consistency of context information [Bikakis *et al.*, 2007]. They typically provide a formal model for context reasoning and can be integrated with the ontology-based reasoning approaches.

Both approaches focus on verifying the internal quality of the context information (i.e., the extent to which context information is free from errors and inconsistency). They pay less or no attention to the external quality of context information (i.e., fitness-for-use). However, in semantic interoperability, context reasoning needs to take into account the quality of context with regard to the application for which the interoperability is carried out. Consequently, evaluating and modeling such quality is important to enhance context reasoning.

3 Evaluating and Modeling the external quality of production context

3.1. Evaluating the external quality of context

In previous work [Sboui *et al.*, 2009], we proposed a restricted set of indicators and a method for evaluating the external quality of metadata with regard to a specific use. These indicators are: *convenience of language*, *completeness*, *trust*, and *freshness*. Each indicator is evaluated according to a function. The resulting quality value is within the interval [0, 1]. The value 1 indicates perfect quality while the value 0 indicates completely poor quality. Based on this value, a qualitative value (i.e., "good", "medium" or "poor") is assigned to the external quality of both explicit and implicit production context.

1- Convenience of language. It indicates the convenience of using a given language to represent the production context of geospatial data. For example, the convenience of a free natural language for a novice user is "medium".

2- Completeness. It shows the quantity of the production context with regards to user's requirements. We recognize thematic, spatial, and temporal completeness. For example, the spatial completeness of a context that does not contain information about reference system is "poor".

3- Trust. It describes the degree of faith that we have about the production context transmitted in a chain of interveners in semantic interoperability of geospatial data. For example, the faith we have about data precision is "medium".

4- Freshness. This quality indicator shows the degree of rationalism related to the use of context information at a given time. The value of freshness is determined by the age and lifetime of the context information. For example, a context defined in 2008 is fresh (i.e., the freshness is "good").

We should notice that these indicators do not aim at being complete or precise but rather at making agents globally aware of the external quality of the production context.

3.2. Modeling the quality of context

We propose a formal model based on ontology using Web Ontology Language (OWL) to represent the previously defined indicators of external quality and facilitate context reasoning. The model embeds both the production context and the information about its external quality using OWL classes and properties. The choice of ontology technique is motivated by the fact that ontology provides: (1) a flexible structure with explicit vocabulary to represent concepts and relations of both a context and its external quality, (2) logic reasoning mechanisms which are necessary to verify the context external quality, and (3) a common structure to exchange data between interveners in the geospatial interoperability.

Figure 1 shows a simplified view of the model that includes a set of OWL classes such as *ContextElement*, *ExternalQuality*, *Indicator* and *Value. ContextElement* class allows representing any context element (spatial, thematic or temporal). *ExternalQuality* class allows specifying various qualities with different external quality indicators. *Indicator* class defines the indicators of the external quality of context in a specific use. *LanguageConvenience*, *Completeness*, *Trust* and *Freshness* are subclasses of *Indicator*. Each indicator has a *Value* that can be *Good*, *Medium* or *Poor*.

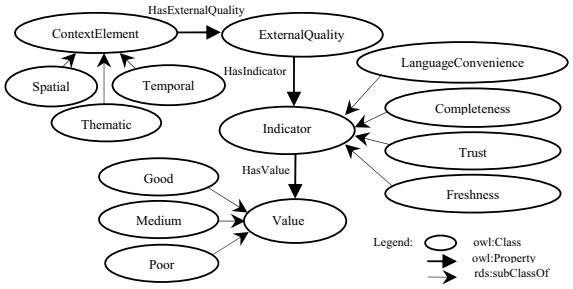


Figure 1. OWL-based context external quality model

The semantics of OWL ontology are derived from Description Logics (DL) [Baader, 2003]. We use the description logics ALC as an example. The descriptions expressible by ALC are: bottom-concept ($\bot$), top-concept ($\top$), A ($A \in N_C$), conjunction ($C \sqcap D$), disjunction ($C \sqcup D$), negation ($\neg C$),

existential restriction ($\exists R.C$), and value restriction ($\forall R.C$). Where N_C is the set of primitive concept names, A is a concept name, C and D are concept descriptions, and R is a role description. The expression of the form $A \equiv C$ assigns A to C, and is called *concept definition*.

In the proposed model, we defined the following sets of primitive concepts and primitive roles:

Primitive concepts: N_C = {ContextElement, ExternalQuality, Indicator, LanguageConvenience, Completeness, Trust, Freshness, Value, Good, Medium, Poor, SpatialContextElement, TemporalContextElement, ThematicContextElement}

Primitive roles: N_R = {HasQuality, HasInternalQuality, HasExternalQuality, HasIndicator, HasValue}

Using the above sets, we can define additional concepts and roles. For example the concepts GoodExternalQuality and BadExternalQuality can be defined as follows:

GoodExternalQuality ≡ ExternalQuality ⊓ ∀HasIndicator. (Indicator ⊓ ∃HasValue.Good)

That is, a good external quality is an external quality that has Good value for all its indicators.

BadExternalQuality ≡ ExternalQuality ⊓ ∀HasIndicator. (Indicator ⊓ ∃HasValue.Poor)

Also, in order to facilitate the comparison of heterogeneous context elements, we define the following additional concepts: ExcellentContextElement and BadContextElement. These two concepts indicate, respectively, that a context element has a good and bad quality for all indicators. They can be represented as follows:

ExcellentContextElement ≡ ContextElement ⊓ ∀HasExternalQuality.GoodExternalQuality

BadContextElement ≡ ContextElement ⊓ ∀HasExternalQuality.BadExternalQuality

The above examples show that the proposed model allows inferring conclusions not only from existing production context, but also from information about the external quality of this context. Based on such conclusions, interveners in the interoperability process (i.e., agent systems or humans) will be able to appropriately reason about the context of geospatial data and make appropriate decisions about the semantic heterogeneity of geospatial data. For instance, if interveners have to choose between two heterogeneous elements, they will be invited to choose the element having a better external quality of production context (i.e. the element fitting better for the current use). For example, in an interoperability, aiming at integrating data from different provinces for rail traffic analysis in Canada (use context), we may need to choose between the almost-equivalent concepts *railroad* and *railway*. Knowing that the first concept has a GoodExternalQuality, while the second has a BadContextElement, the interveners are invited to choose the first concept.

4 Conclusion

In this paper, we proposed an approach based on ontology to model and reason about context taking into account the external quality (fitness-for-use) of the production context. Such model aims at helping interveners to appropriately reason about context information and hence, fine-tuning geospatial interoperability. Indeed, the proposed model provides relevant information that can be used in making appropriate decisions (e.g., comparing two heterogeneous context elements and considering one of them, or ignoring a context that has a poor external quality).

Further research is undergoing to define additional indicators such as relevancy and granularity of the production context. Then, a prototype will be implemented and tested using two heterogeneous datasets related to the road accident and the density of population. We will first show that the interoperability faces problems related to the semantic heterogeneity. Then, we will show how the proposed approach supports context reasoning based on the external quality of production context and, hence, fine-tunes the semantic interoperability of real geospatial datasets.

Acknowledgments

We acknowledge the contribution of the NSERC Research Chair in Geospatial Databases for Decision Support.

References

[Baader, 2003] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P.F. Patel-Schneider. *The description logic handbook: theory, implementation, and applications*. Cambridge University Press, London, 2003.

[Benerecetti *et al.*, 2000] M. Benerecetti, P. Bouquet, and C. Ghidini. Contextual reasoning distilled. *JETAI*, 12(3):279–305, 2000.

[Bikakis *et al.*, 2007] A. Bikakis, T. Patkos, G. Antoniou, and D. Plexousakis. A Survey of Semantics based Approaches for Context Reasoning in Ambient Intelligence. In *Proceedings of the AmI'07*, pages 15–24, 2007.

[Bishr, 1998] Y. Bishr. Overcoming the semantic and other barriers to GIS interoperability. *Int. Journal of GIS*, 12(4):299–314, 1998

[Brodeur, 2004] J. Brodeur. *Interopérabilité des données géospatiales: élaboration du concept de proximité géosémantique*. Ph.D. Dissertation. U. Laval, 2004.

[Frank, 2007] A.U. Frank. Data Quality Ontology: An Ontology for Imperfect Knowledge. *COSIT 2007, LNCS 4736*, pages 406–420, 2007.

[Giunchiglia, 1993] F. Giunchiglia. Contextual reasoning. *Epistemologia*, Special issue on *I Linguaggie le Macchine* (XVI) pages 345–364, 1993.

[Gu *et al.*, 2004] T. Gu, X.H. Wang, H.K. Pung, and D.Q. Zhang. An Ontology-based Context Model in Intelligent Environments. In *Proceedings of CNDS*, 2004.

[Henricksen and Indulska, 2004] K. Henricksen and J. Indulska. Modelling and Using Imperfect Context Information. In *Proceedings of PERCOMW*, 2004.

[Sboui *et al.*, 2009] T. Sboui, M. Salehi and Y. Bédard. Towards a Quantitative Evaluation of Geospatial Metadata Quality in the Context of Semantic Interoperability. To appear in the *proceedings of ISSDQ*, 2009.

A Conflict-based Operator for Mapping Revision

Guilin Qi
AIFB
University of Karlsruhe
76128, Karlsruhe

Qiu Ji
AIFB
University of Karlsruhe
76128, Karlsruhe

Peter Haase
AIFB
University of Karlsruhe
76128, Karlsruhe

Abstract

In this paper, we propose a novel method for mapping revision. We first propose a conflict-based mapping revision operator and show that it can be characterized by a set of logical properties. We then provide an iterative algorithm for mapping revision by using an ontology revision operator and show that this algorithm defines a conflict-based mapping revision operator.

1 Introduction

One of the major challenges in managing these distributed and dynamic ontologies is to handle potential inconsistencies introduced by integrating multiple distributed ontologies. However, there is little work done on handling inconsistency in distributed ontologies connected by mappings, where a mapping between two ontologies is a set of correspondences between entities in the ontologies. Given a distributed system which is a triple consisting of two ontologies and a mapping between them, correspondences in the mapping can have different interpretations. For example, in Distributed Description Logics (DDL) [Borgida and Serafini, 2003], a correspondence in a mapping is translated into two weighted *bridge rules* that describes the "flow of information" from one ontology to another one. The weight attached to a bridge rule is often considered as degree of uncertainty of the bridge rule. In [Meilicke *et al.*, 2007], the authors deal with the problem of mapping revision in DDL by removing some bridge rules which are responsible for the inconsistency. The idea of their approach is similar to that of the approaches for debugging and repairing terminologies in a single ontology. However, existing methods lack of rationality analysis w.r.t. logical properties. In this paper, we propose a novel method for mapping revision. We first propose a conflict-based mapping revision operator and show that it can be characterized by a set of important logical properties. We then provide an iterative algorithm for mapping revision by using a revision operator in description logics and show that this algorithm defines a conflict-based mapping revision operator.

2 Preliminaries

We assume the reader is familiar with Description Logics (DLs). Given two ontologies O_1 and O_2, describing the same or largely overlapping domains of interest, we can define correspondences between elements of the ontologies.

Definition 1 *[Euzenat and Shvaiko, 2007] Let O_1 and O_2 be two ontologies, Q be a function that defines sets of mappable elements $Q(O_1)$ and $Q(O_2)$. A correspondence is a 4-tuple $\langle e, e', r, \alpha\rangle$ such that $e \in Q(O_1)$ and $e' \in Q(O_2)$, r is a semantic relation, and α is a confidence value from a suitable structure $\langle D, \leq\rangle$, such as a lattice. Suppose $\mathcal{M}$ is a set of correspondences, then $\mathcal{M}$ is a mapping between O_1 and O_2 iff for all correspondences $\langle e, e', r, \alpha\rangle \in \mathcal{M}$ we have $e \in Q(O_1)$ and $e' \in Q(O_2)$.*

In Definition 1, there is no restriction on function Q, semantic relation r and domain D. In the mapping revision scenario, we often consider correspondences between concepts and restrict r to be one of the semantic relations from the set $\{\equiv, \sqsubseteq, \sqsupseteq\}$, and let $D = [0.0, 1.0]$. A mapping is a set of correspondences whose elements are mappable.

Given a mapping between two ontologies O_1 and O_2, we can define the notion of a distributed system.

Definition 2 *A distributed system is a triple $\mathcal{D} = \langle O_1, O_2, \mathcal{M}\rangle$, where O_1 and O_2 are ontologies and $\mathcal{M}$ is a mapping between them. We call O_1 the source ontology and O_2 the target ontology.*

Definition 3 *[Meilicke* et al.*, 2007] Let $\mathcal{D} = \langle O_1, O_2, \mathcal{M}\rangle$ be a distributed system. The* union *$O_1 \cup_{\mathcal{M}} O_2$ of O_1 and O_2 connected by $\mathcal{M}$ is defined as $O_1 \cup_{\mathcal{M}} O_2 = O_1 \cup O_2 \cup \{t(m) : m \in \mathcal{M}\}$ with t being a translation function that converts a correspondence into an axiom in the following way:*

$$t(\langle C, C', r, \alpha\rangle) = CrC'$$

That is, we first translate all the correspondences in the mapping $\mathcal{M}$ into DL axioms, then the union of the two ontologies connected by the mapping is the set-union of the two ontologies and the translated axioms. Given $\mathcal{D} = \langle O_1, O_2, \mathcal{M}\rangle$, we use $Union(\mathcal{D})$ to denote $O_1 \cup_{\mathcal{M}} O_2$.

Definition 4 *[Meilicke and Stuckenschmidt, 2007] Given a mapping $\mathcal{M}$ between two ontologies[1] O_1 and O_2, $\mathcal{M}$ is consistent with O_1 and O_2 iff there exists no concept C in O_i with $i \in \{1, 2\}$ such that C is satisfiable in O_i but unsatisfiable in $O_1 \cup_{\mathcal{M}} O_2$. Otherwise, $\mathcal{M}$ is inconsistent. A distributed system $\mathcal{D} = \langle O_1, O_2, \mathcal{M}\rangle$ is inconsistent if $\mathcal{M}$ is inconsistent with O_1 and O_2.*

[1] We assume that O_1 and O_2 use different name spaces.

An inconsistent mapping is a mapping such that there is a concept that is satisfiable in a mapped ontology but unsatisfiable in the union of the two ontologies together with the mapping.

Definition 5 *A mapping revision operator $\circ$ is a function $\circ\langle O_1, O_2, \mathcal{M}\rangle = \langle O_1, O_2, \mathcal{M}'\rangle$ such that $\mathcal{M}' \subseteq \mathcal{M}$, where O_1 and O_2 are two ontologies and $\mathcal{M}$ is a mapping between them.*

Our definition of a mapping revision operator is similar to the definition of a revision function given in [Meilicke *et al.*, 2008a].

3 A Conflict-based Mapping Revision Operator

In this section, we propose a method for mapping revision based on the idea of kernel contractions defined by Hansson in [Hansson, 1994].

We adapt the notion of a minimal conflict set of a distributed system given in [Meilicke *et al.*, 2007] as follows.

Definition 6 *Let $\langle O_1, O_2, \mathcal{M}\rangle$ be a distributed system. A subset $\mathcal{C}$ of $\mathcal{M}$ is a* conflict set *for a concept A in O_i ($i = 1, 2$) if A is satisfiable in O_i but unsatisfiable in $O_1 \cup_{\mathcal{C}} O_2$. $\mathcal{C}$ is a* minimal conflict set *(MCS) for A in O_i if $\mathcal{C}$ is a conflict set for A and there exists no $\mathcal{C}' \subset \mathcal{C}$ which is also a conflict set for A in O_i.*

A minimal conflict set for a concept in one of the ontologies is a minimal subset of the mapping that, together with the local ontologies, is responsible for the unsatisfiability of the concept in the distributed system. It is similar to the notion of a kernel in [Hansson, 1994]. Note that if O_i ($i = 1, 2$) is incoherent, then every MCS for any concept is an empty set. We use $MCS_{O_1,O_2}(\mathcal{M})$ to denote the set of all the minimal conflict sets for all unsatisfiable concepts in $O_1 \cup_{\mathcal{C}} O_2$. It corresponds to the notion of a *kernel set* in [Hansson, 1994].

Hansson's kernel contraction removes formulas in a knowledge base through an *incision function*, which is a function that selects formulas to be discarded. However, we cannot apply the notion of an incision function to mapping revision directly because the mapping to be revised is dependent of ontologies in the distributed system. Therefore, the problem of mapping revision is not exactly the same as the problem of belief revision where the two knowledge bases may come from different sources. Furthermore, each correspondence in the mapping carries a confidence value which can be used to guide the revision.

Definition 7 *An* incision function *σ is a function*[2] *that for each distributed system $\langle O_1, O_2, \mathcal{M}\rangle$, we have*

(i) $\sigma(MCS_{O_1,O_2}(\mathcal{M})) \subseteq \bigcup(MCS_{O_1,O_2}(\mathcal{M}))$;

(ii) if $\emptyset \neq \mathcal{C} \in MCS_{O_1,O_2}(\mathcal{M})$, *then* $\mathcal{C} \cap \sigma(MCS_{O_1,O_2}(\mathcal{M})) \neq \emptyset$;

(iii) if $m = \langle C, C', r, \alpha\rangle \in \sigma(MCS_{O_1,O_2}(\mathcal{M}))$, *then there exists* $\mathcal{C} \in MCS_{O_1,O_2}(\mathcal{M})$ *such that* $\alpha = min\{\alpha_i : \langle C_i, C'_i, r_i, \alpha_i\rangle \in \mathcal{C}\}$.

[2] A function is often assumed to be single-valued. However, in our definition, we do not follow this assumption.

The first two conditions say that an incision function selects from each kernel set at least one element. The third condition says that if a correspondence is selected by an incision function, then there must exist a MCS $\mathcal{C}$ such that its confidence value is the minimal confidence value of correspondences in $\mathcal{C}$. Note that we do not assume that a function is single-valued because it is a too strong requirement that an incision function always selects the same set of correspondences to remove when it is applied to the same distributed system twice.

We define our mapping revision operator based on an incision function.

Definition 8 *A mapping revision operator $\circ$ is called a* conflict-based mapping revision operator *if there exists an incision function σ such that:*

$$\circ\langle O_1, O_2, \mathcal{M}\rangle = \langle O_1, O_2, \mathcal{M} \setminus \sigma(MCS_{O_1,O_2}(\mathcal{M}))\rangle.$$

We provide the representation theorem for conflict-based mapping revision. Before that, we need to define the notion of an inconsistency degree of a distributed system for a concept. Given a distributed system $\mathcal{D} = \langle O_1, O_2, \mathcal{M}\rangle$, a concept A in O_i ($i = 1, 2$) is unsatisfiable in $\mathcal{D}$ if A is unsatisfiable in $O_1 \cup_{\mathcal{M}} O_2$.

Definition 9 *Given a distributed system $\mathcal{D} = \langle O_1, O_2, \mathcal{M}\rangle$, the β-cut (resp. strict β-cut) set of $\mathcal{D}$, denoted as $\mathcal{D}_{\geq\beta}$ (resp. $\mathcal{D}_{>\beta}$), is defined as $\mathcal{D}_{\geq\beta} = \langle O_1, O_2, \{\langle C, C', r, \alpha\rangle \in \mathcal{M} : \alpha \geq \beta\}\rangle$ (resp. $\mathcal{D}_{>\beta} = \langle O_1, O_2, \{\langle C, C', r, \alpha\rangle \in \mathcal{M} : \alpha > \beta\}\rangle$).*

The β-cut set of $\mathcal{D}$ is the union of O_1, O_2 and axioms translated from correspondences in the mapping whose confidence values are greater than or equal to β. It is adapted from the notion of cut set in possibilistic DLs in [Qi *et al.*, 2007].

Definition 10 *Given a distributed system $\mathcal{D} = \langle O_1, O_2, \mathcal{M}\rangle$, the inconsistency degree of $\mathcal{D}$ for a concept A in O_i ($i = 1, 2$), denoted by $Inc(\mathcal{D})_A$, is defined as $Inc(\mathcal{D})_A = max\{\alpha : A \text{ is unsatisfiable in } \mathcal{D}_{\geq\alpha}\}$. The inconsistency degree of $\mathcal{D}$, denoted as $Inc(\mathcal{D})$, is defined as $Inc(\mathcal{D}) = max\{\alpha : \text{there exists an unsatisfiable concept in } \mathcal{D}_{\geq\alpha}\}$.*

It is easy to check that the inconsistency degree of a distributed system is the maximum confidence value α such that the α-cut set of $\mathcal{D}$ is inconsistent under some conditions.

Proposition 1 *Given a distributed system $\mathcal{D} = \langle O_1, O_2, \mathcal{M}\rangle$ where at least one of O_i ($i = 1, 2$) is coherent, suppose $Inc(\mathcal{D})$ is its inconsistency degree, then we have $Inc(\mathcal{D}) = max\{\alpha : \mathcal{D}_{\geq\alpha} \text{ is inconsistent}\}$.*

Theorem 1 *The operator $\circ$ is a conflict-based mapping revision operator if and only if it satisfies the following condition:*

(Relevance) suppose $\circ\langle O_1, O_2, \mathcal{M}\rangle = \langle O_1, O_2, \mathcal{M}'\rangle$, if $m = \langle C, C', r, \alpha\rangle \in \mathcal{M}$ and $m \notin \mathcal{M}'$, then there exist a concept A in O_i ($i = 1, 2$) and a subset $\mathcal{S}$ of $\mathcal{M}$ such that A is satisfiable in $\langle O_1, O_2, \mathcal{S}\rangle$ but is unsatisfiable in $\langle O_1, O_2, \mathcal{S} \cup \{m\}\rangle$ and $Inc(\langle O'_1, O'_2, \mathcal{S} \cup \{m\}\rangle)_A = \alpha$.

4 An Algorithm for Mapping Revision

We describe the idea of our algorithm (Algorithm 1) as follows. Given a distributed system $\mathcal{D} = \langle O_1, O_2, \mathcal{M}\rangle$, if

Algorithm 1: An iterated algorithm for mapping revision

Data: A distributed system $\mathcal{D} = \langle O_1, O_2, \mathcal{M}\rangle$ and a revision operator $\star$
Result: A repaired distributed system $\mathcal{D}_\star = \langle O_1, O_2, \mathcal{M}_\star\rangle$

begin
  if *either* O_1 *or* O_2 *is incoherent* **then**
    return $\mathcal{D}$
  Rearrange the weights in $\mathcal{M}$ such that $\beta_1 > \beta_2 > ... > \beta_l > 0$;
  $S_i := \{t(\langle C, C', r, \alpha\rangle) : \langle C, C', r, \alpha\rangle \in \mathcal{M}, \alpha = \beta_i\}, i = 1, ..., l$;
  while $\mathcal{M}$ *in* $\mathcal{D}$ *is inconsistent* **do**
    if $\beta_k = Inc(\mathcal{D})$ **then**
      $S_t := S_k \setminus (S_k \star (Union(\mathcal{D})_{>\beta_k}))$;
      $\mathcal{M} := \mathcal{M} \setminus \{\langle C, C', r, \alpha\rangle : t(\langle C, C', r, \alpha\rangle) \in S_t, \alpha = \beta_k\}$;
  return $\mathcal{D}$
end

either O_1 or O_2 is incoherent, then we take $\mathcal{D}$ as the result of revision. That is, no change is needed. Suppose $\mathcal{M} = \{\langle C_i, C_i', r_i, \alpha_i\rangle : i = 1, ..., n\}$ where n is the number of correspondences in $\mathcal{M}$. Let us rearrange the weights of axioms in $\mathcal{M}$ such that $\beta_1 > \beta_2 > ... > \beta_l > 0$, where β_i $(i = 1, ..., l)$ are all the distinct weights appearing in $\mathcal{M}$. For each $i \in \{1, ..., l\}$, S_i consists of translated axioms of correspondences in $\mathcal{M}$ which have the confidence value β_i. Suppose $Inc(\mathcal{D}) = \beta_k$. We revise S_k by $Union(\mathcal{D}_{>\beta_k})$. Suppose S_t is the axioms in S_k that are removed after revision of S_k by $Union(\mathcal{D}_{>\beta_k})$ using the operator $\star$. We then remove the correspondences in $\mathcal{M}$ that have confidence values β_k and are mapped to axioms in S_t by the translation function t. We iterate the revision process until the mapping becomes consistent.

In Algorithm 1, we need to compute the inconsistency degree of a distributed system. This can be easily done by adapting the algorithm for computing the inconsistency degree in [Qi *et al.*, 2007] so we do not bother to provide it here.

We have not specified a revision operator in Algorithm 1. However, we require that the revision operator $\star$ used in the algorithm satisfies the following properties which are similar to the postulates *Inclusion*, *Success* and *core-retainment* for kernel revision operator given in [Hansson, 1994]:

- Inclusion: $O \star O' \subseteq O \cup O'$;
- Success: $O' \subseteq O \star O'$;
- Core-retainment: if $\phi \in O$ and $\phi \notin O \star O'$, then there exist a concept A in $O \cup O'$ and a subset O_s of O, such that A is satisfiable in $O_s \cup O'$ but is unsatisfiable in $O_s \cup O' \cup \{\phi\}$.

Theorem 2 *Suppose $\star$ satisfies Inclusion, Success and Core-retainment, and $\circ$ is a mapping revision operator such that, for any distributed system $\mathcal{D}$, $\circ(\mathcal{D})$ is the result of Algorithm 1 with $\star$ as an input parameter, then $\circ$ is a conflict-based mapping revision operator.*

We give a simple revision operator. By SORT we denote a procedure that for each ontology O, arbitrarily ranks its elements as an ordered sequence $(\phi_1, ..., \phi_n)$. Let O and O' be two ontologies, and let $\text{SORT}(O) = \{\phi_1, ..., \phi_n\}$, the random linear base revision operator, denoted as $\circ_{linear}$, is defined inductively as follows $O \circ_{linear} O' = O' \cup S_1 \cup ... \cup S_n$, where S_i is defined by $S_0 = O'$, $S_i = \{\phi_i\}$ if $\{\phi_i\} \cup O' \cup \bigcup_{j=1}^{i-1} S_j$ is coherent, $\emptyset$ otherwise, for $i \geq 1$. It is easy to check that this revision operator satisfies conditions Inclusion, Success and Core-retainment.

Proposition 2 *For any distributed system $\mathcal{D} = \langle O_1, O_2, \mathcal{M}\rangle$ where both O_1 and O_2 are coherent, suppose $\mathcal{D}_{\circ_{linear}}$ is the result of revision by Algorithm 1, then $\mathcal{D}_{\circ_{linear}}$ can be obtained by the algorithm given in [Meilicke* et al.*, 2008b] as well, and vice versa.*

5 Conclusions

In this paper, we discussed the problem of repairing inconsistent mappings in distributed systems. We first defined a conflict-based mapping revision operator and provided a representation theorem for it. We then presented an iterated algorithm for mapping revision in a distributed system based on a revision operator in DLs and showed that this algorithm resulting in a conflict-based mapping revision operator.

References

[Borgida and Serafini, 2003] A. Borgida and L. Serafini. Distributed description logics: Assimilating information from peer sources. *Journal of Data Semantics*, 1:153–184, 2003.

[Euzenat and Shvaiko, 2007] J. Euzenat and P. Shvaiko. *Ontology Matching*. Berlin; Heidelberg, Springer, 2007.

[Hansson, 1994] S. Ove Hansson. Kernel contraction. *Journal Symbolic Logic*, 59(3):845–859, 1994.

[Meilicke and Stuckenschmidt, 2007] C. Meilicke and H. Stuckenschmidt. Applying logical constraints to ontology matching. In *Proc. of KI*, pages 99–113, 2007.

[Meilicke *et al.*, 2007] C. Meilicke, H. Stuckenschmidt, and A. Tamilin. Repairing ontology mappings. In *Proc. of AAAI*, pages 1408–1413, 2007.

[Meilicke *et al.*, 2008a] C. Meilicke, H. Stuckenschmidt, and A. Tamilin. Reasoning support for mapping revision. *Journal of Logic and Computation*, 2008.

[Meilicke *et al.*, 2008b] C. Meilicke, J. Völker, and H. Stuckenschmidt. Learning disjointness for debugging mappings between lightweight ontologies. In *Proc. of EKAW*, pages 93–108, 2008.

[Nebel, 1994] B. Nebel. Base revision operations and schemes: Semantics, representation and complexity. In *Proc. of ECAI*, pages 341–345, 1994.

[Qi *et al.*, 2007] G. Qi, J. Z. Pan, and Qiu Ji. Extending description logics with uncertainty reasoning in possibilistic logic. In *Proc. of ECSQARU*, pages 828–839, 2007.

Unite: A New Plan for Automated Ontology Evolution in Physics*

Alan Bundy
School of Informatics, University of Edinburgh,
A.Bundy@ed.ac.uk

Abstract

We are developing a novel technique for ontology evolution, which we call *ontology repair plans*. Our development case studies are drawn from instances in the history of physics where experimental observation contradicted current physical theories, which then had to be evolved. In particular, it was often necessary to evolve the *representation language* of these theories, and not just the physical laws. To date, we have implemented one ontology repair plan for splitting a function into three and another for adding an additional argument to a function. In this paper, we describe a new ontology repair plan *Unite*, for equating two previously distinct functions.

1 Ontology Repair Plans

We are developing a series of *ontology repair plans* which operate simultaneously on a small set of modular higher-order[1] ontologies, e.g., one representing an initial theory of physics, another representing a particular experimental set-up [Bundy and Chan, 2008]. Each repair plan has a trigger formula and some actions: when the trigger is matched, the actions are performed. The actions modify both the signatures[2] and the axioms of the old ontologies to produce new ones. The repair plans have been implemented in the GALILEO system (Guided Analysis of Logical Inconsistencies Leads to Evolved Ontologies) using λProlog [Miller and Nadathur, 1988] as our implementation language, because it provides a polymorphic, higher-order logic programming language.

We have so far developed two repair plans, which we call *Where's my stuff?* (WMS) and *Inconstancy*. These roughly correspond to the operations of splitting a function and adding an argument, respectively. We have found multiple examples of these repairs across the history of physics.

*I'm grateful to Michael Chan and Jos Lehmann for comments on an earlier draft, and to both Michael and Alan Smaill for programming assistance.

[1]The physics domain requires higher-order logic: both at the object-level, to describe things like planetary orbits and calculus, and at the meta-level, to describe the ontology repair operations.

[2]A signature describes the representation language of an ontology, e.g., its functions and their types.

The WMS repair plan aims at resolving contradictions arising when the predicted value returned by a function does not match the observed value. This is modelled by having two ontologies, corresponding to the prediction and the observation, with different values for this function. To break the inconsistency, the conflicted function is split into three new functions: *visible*, *invisible* and *total*. The conflicted function becomes the total function in the predictive theory and the visible function in the observation theory[3]. The invisible function is defined as the difference between them, and this new definition is added to the predictive theory. The intuition behind this repair is that the discrepancy arose because the function was not being applied to the same *stuff* in the predictive and the observational ontologies — the invisible stuff was not observed.

WMS has been successfully applied to conflicts between predictions of and observations of the following functions: the temperature of freezing water; the energy of a bouncing ball; the graphs relating orbital velocity of stars to distance from the galactic centre in spiral galaxies; and the precession of the perihelion of Mercury. In these examples the role of the invisible stuff is played by: the latent heat of fusion, elastic potential energy, dark matter and an additional planet, respectively.

The Inconstancy repair plan is triggered when there is a conflict between the predicted independence and the observed dependence of a function on some parameter, i.e., the observed value of a function unexpectedly varies when it is predicted to remain constant. This generally requires several observational ontologies, each with different observed values of the function, as opposed to the one observational theory in the WMS plan. To effect the repair, the parameter causing the unexpected variation is first identified and a new definition for the conflicted function is created that includes this new parameter. The nature of the dependence is induced from the observations using curve-fitting techniques.

Inconstancy has been successfully applied to the following conflicts between predictions and various observations: the ratio of pressure and volume of a gas; replacement of Aristotle's concept of instantaneous light travel with a finite (but fast) light speed; and again the graphs relating orbital velocity of stars to distance from the galactic centre in spiral galaxies. The unexpected parameter of the function is: the temperature

[3]There are situations in which these roles are inverted.

of the gas; the distance between the source and target of a flash of light; and the acceleration between the stars, respectively. The first of these repairs generalises Boyle's Law to the Ideal Gas Law, the second generalises a moment of light travel to an interval and the third generalises the Gravitational Constant to Milgrom's MOND (MOdified Newtonian Dynamics). Interestingly, WMS and Inconstancy produce the two main rival ontologies on the spiral galaxy anomaly, namely dark matter and MOND. Since this is still an active controversy, its unfolding will help us develop mechanisms to choose between rival theory repairs.

The merging of functions and the dropping of arguments are identified as two common forms of *abstraction* in [Giunchiglia and Walsh, 1992]. The abstractions and their inverses, the *refinements* of *splitting* functions *adding* arguments, were used as the basis for ontology evolution in ORS [McNeill and Bundy, 2007]. The GALILEO work addresses one of the key outstanding issues in ORS: the essential ambiguity of refinement. When a function is split it is necessary to decide which occurrences of the old function map to which new function. Similarly, it is necessary to decide the value of each occurrence of a new argument. GALILEO addresses this problem by making uniform mappings within each ontology.

Function merging and argument dropping are examples of *signature* evolution, i.e., a change in the underlying language of an ontology. This is complementary to *belief revision*, which manages axiom evolution within a fixed signature. Both are needed. Below, we explore an ontology repair plan that creates a new axiom.

2 The Unite Ontology Repair Plan

We now explore the *converse* of the WMS ontology repair plan, which we will call *Unite*. The idea of the Unite plan is to take two different functions and equate them. The classic example of the need for this repair is in the identification of the Evening Star and the Morning Star as two manifestations of Venus. In this case the functions are all nullary, i.e., constants, but we will also give a non-nullary example.

2.1 Defining Properties

If two different terms refer to the same thing then they should yield the same value for each of their properties. Unfortunately, this is impractical as a trigger for the repair plan, because, in practice, the values of *all* properties of a thing are unlikely to be known — indeed, some of the properties themselves may not yet be known. Fortunately, there are properties which *alone* are sufficient. I will call these *defining properties*. For instance, for physical objects, such as heavenly bodies, their *orbit* is a defining property, which we can summarise as saying that two different objects cannot be at the same place at the same time. Note that I mean 4D orbits, not mere 3D ones, i.e., it is not just that the orbits occupy the same path in 3D space, but that each moment of time defines the same position in both orbits.

We will use $DefProp(dp, \tau)$ to represent that property dp is a defining property of objects of type τ, i.e.,

$$DefProp(dp, \tau) \quad ::= \quad \forall x, y{:}\tau.\ dp(x) = dp(y) \rightarrow x = y$$

We are now ready to formally define the Unite repair plan.

2.2 The Plan Formalism

Trigger: If $stuff_1$ and $stuff_2$ are functions of the same type, not already known to be equal, that are observed to take the same value for some defining property, dp, for functions of type τ then the following formula will be triggered.

$$\begin{array}{lll} O_t & \not\vdash & stuff_1 = stuff_2, \\ O_t & \vdash & stuff_1 : \tau \wedge stuff_2 : \tau \wedge DefProp(dp, \tau), \\ O_s & \vdash & dp(stuff_1) = dp(stuff_2). \end{array} \quad (1)$$

O_t is the theoretical ontology, O_s is the local ontology describing a particular set of experiments.

Create New Ontologies: The repair is to add an equality between $stuff_1$ and $stuff_2$ as a new axiom to O_t. O_s is unchanged.

$$Ax(\nu(O_t)) \quad ::= \quad \{stuff_1 = stuff_2\} \cup Ax(O_t) \quad (2)$$

where $Ax(O)$ is the set of axioms of ontology O and $\nu(O)$ is the ontology resulting from repairing O.

2.3 The Plan Implementation

Unite has been implemented in the GALILEO system. This implementation consists of λProlog code for:

- an additional clause in the generic `Repair` function, that takes two ontologies and outputs their repaired axioms;
- a function `unite_trigger` that checks that the trigger formula holds; and
- a function `changeUnite` that adds the additional axiom to the theoretical ontology.

Note that the physical formulae are represented using a *deep embedding*, in which `applic` applies a function to an argument and `lambda` abstracts a variable in a function. `turnstile O T` represents $O \vdash T$. `equal` is defined as a unary function on a list of equal objects. In λProlog, variables start with an upper case letter; constants with lower case[4].

```
% Unite repair
repair Ot Os NAt As :-
 unite_trigger Ot Os S1 S2,
 changeUnite Ot S1 S2 NAt,
 axioms Os As.

% Unite trigger:
unite_trigger Ot Os S1 S2 :-
 not turnstile Ot (applic equal (S1::S2::nil)),
 turnstile Ot (applic (applic isa S1) T),
 turnstile Ot (applic (applic isa S2) T),
 turnstile Ot (applic (applic defprop DP) T),
 turnstile Os (applic equal ((applic DP S1)
  ::(applic DP S2)::nil )).

% Unite repair:
changeUnite O S1 S2 ((applic equal
   (S1::S2::nil))::A) :-
 axioms O A.
```

[4] Unfortunately, the opposite of the normal mathematical conventions used elsewhere in this paper.

3 Case Studies

We now describe two case studies that were used as the development set for Unite, and have been successfully implemented and evaluated.

3.1 Example: The Morning and Evening Stars

Because Venus is closer to the Sun than the Earth, it becomes visible either just before dawn or just after sunset, when it is the brightest heavenly object after the Moon. These two kinds of appearance were not originally identified as coming from the same object. Both the Ancient Egyptians and the Ancient Greeks thought there were two objects. The pre-dawn appearances were identified as the Morning Star and the post-sunset ones as the Evening Star. It was only with the quantification of astronomy that the orbits of these 'two' 'stars' were calculated and seen to be the same (up to experimental error).

We can use the Unite repair plan to emulate this episode as follows. The trigger formulae are:

$$\begin{array}{lll} O_t & \nvdash & MS = ES \\ O_t & \vdash & MS : obj \wedge ES : obj \wedge DefProp(orbit, obj) \\ O_s & \vdash & orbit(MS) = orbit(ES). \end{array}$$

where MS and ES are constants standing for "Morning Star" and "Evening Star", respectively. These formulae match with (1).

The repair is then:

$$Ax(\nu(O_t)) \quad ::= \{MS = ES\} \cup Ax(O_t)$$

from (2), as required. When the objects being equated are constants, as in this example, we might want to go further and replace both old constants with a new one, e.g. $Venus$, but this is beyond the scope of this paper.

3.2 Example: The Earth as a Sphere

Pythagoras was one of the first astronomers to realise that the Earth was a sphere. He gathered evidence to support this theory from various sources, but in this paper we will consider only his observations of lunar eclipses. He noticed that the edge of the shadow that the Earth cast on the Moon was always circular. He reasoned that the only 3D shape that always casts circular shadows is a sphere.

This reasoning is also an example of the Unite repair plan. This time the terms being equated are compound: $Shape(Earth)$ and $Shape(Ball)$, where $Ball$ is some imaginary spherical object in the same orbit as the Earth. The defining property is $\lambda v, t.\ project(v, Sun, Moon, t)$: the projection of a volume v from the Sun onto the Moon. The idea is that if two 3D objects always have the same 2D projections then they have the same shape, i.e., $DefProp(\lambda v, t.\ project(v, Sun, Moon, t), vol)$.

Note that multiple, independent projections are required. A cylinder also projects as a circle along its axis, but most of its projections are not circular, so one projection is not enough. The abstraction over time supplies these. Pythagoras could not, of course, observe all possible lunar eclipses, so an element of induction is required in the observational ontology O_s. Note also that the 'observed' projections of $Ball$ are a thought experiment.

The application of Unite proceeds as follows. The trigger formulae are:

$$\begin{array}{lll} O_t & \nvdash & Shape(Earth) = Shape(Ball) \\ O_t & \vdash & Shape(Earth) : vol \wedge Shape(Ball) : vol \\ & & \quad \wedge DefProp(\lambda v, t.\ project(v, Sun, Moon, t), vol), \\ O_s & \vdash & \lambda t.\ project(Shape(Earth), Sun, Moon, t) \\ & & \quad = \lambda.\ project(Shape(Ball), Sun, Moon, t) \end{array}$$

where

$$\begin{array}{rl} & \lambda t.\ project(Shape(Earth), Sun, Moon, t) \\ = & \lambda t.\ project(Shape(Ball), Sun, Moon, t) \end{array}$$

is Pythagoras' generalisation from his observations of lunar eclipses and his thought experiments. The repair to O_t is to add the new axiom $Shape(Earth) = Shape(Ball)$.

4 Conclusion

In this paper we have described Unite, a converse repair plan to WMS, which equates two different terms. Its central notion is of a defining property, i.e., a property whose value is unique for things of a specific type. We have given a formal definition of the trigger formulae and of the repair operations. This repair plan has been implemented in the GALILEO ontology evolution system. We have illustrated the operation of the Unite plan with two examples:

- the identification of the Morning Star and the Evening Star, where the defining property is their orbit; and
- the identification of the shape of the Earth as that of a sphere, where the defining property is the projection of a 3D volume onto a 2D surface.

These two examples have been successfully evaluated in GALILEO. In future work, we plan to evaluate Unite on a test set of case studies from the history of physics. Since the repair is to add a new law to a physical theory, it has the potential to emulate many major advances in physical ontology evolution.

References

[Bundy and Chan, 2008] A. Bundy and M. Chan. Towards ontology evolution in physics. In W. Hodges, editor, *Procs. Wollic 2008*. LNCS, Springer-Verlag, July 2008.

[Giunchiglia and Walsh, 1992] F. Giunchiglia and T. Walsh. A theory of abstraction. *Artificial Intelligence*, 56(2–3):323–390, 1992.

[McNeill and Bundy, 2007] F. McNeill and A. Bundy. Dynamic, automatic, first-order ontology repair by diagnosis of failed plan execution. *IJSWIS*, 3(3):1–35, 2007. Special issue on ontology matching.

[Miller and Nadathur, 1988] D. Miller and G. Nadathur. An overview of λProlog. In R. Bowen, editor, *Proceedings of the Fifth International Logic Programming Conference/ Fifth Symposium on Logic Programming*. MIT Press, 1988.

An Architecture of GALILEO: A System for Automated Ontology Evolution in Physics*

Michael Chan and **Alan Bundy**
School of Informatics
University of Edinburgh
{M.Chan, A.Bundy}@ed.ac.uk

Abstract

The GALILEO system implements novel mechanisms for automated ontology evolution in physics. These mechanisms, called *ontology repair plans*, resolve logical conflicts between several modular ontologies. To demonstrate that ontology evolution can be automated using ontology repair plans, we propose a flexible architecture for the implementation. The support for inference of formulae that trigger repair plans and modularisation of ontologies is central to the design. Huet's zipper data structure is to be used to combine the benefits of shallow and deep embeddings. For a high degree of modularity, the management of a collection of ontologies is handled by development graphs.

1 Introduction

Visions of applications involving multi-agent systems, e.g., the Semantic Web, have intensified the need for mechanisms for automated ontology evolution. We address the issue by developing *ontology repair plans* and applying them to records of ontology evolution in the history of physics. The implemented programs form the core of the GALILEO system (Guided Analysis of Logical Inconsistencies Leads to Evolved Ontologies). Ontology repair plans enable agents to autonomously update their own ontologies by applying diagnosis and repair operations [Bundy and Chan, 2008]. As a step towards the demonstration that automated ontology evolution via ontology repair plans is computationally feasible, the repair plans and the example ontologies are implemented in Teyjus, an implementation of λProlog [Miller and Nadathur, 1988]. Higher-order abstract syntax is used to enable the meta-logic to recurse over the syntactical structure of the object-logic. Unfortunately, this makes reasoning about whether some ontology has a theorem that infers the triggering formulae of a repair plan more difficult. One consequence is that theorems are forced to be coded as axioms. This implementation is clearly unnatural, but has been kept simple because it is a mere proof-of-concept. A more flexible architecture is essential in order to go further.

*We're grateful to Jos Lehmann for feedbacks on an earlier draft.

The goals underlying the design of the architecture of the new implementation are two-fold:

- Higher-order reasoning engines must be incorporated to infer formulae that trigger ontology repair plans.
- The representation of knowledge as ontologies should be modular.

In the current pure λProlog implementation, the inference is left to the programmer and the encoding of the ontologies for repair is tailored to the specific triggering formulae of the repair plans. This issue can be addressed by adopting a reasoner, such as a theorem prover. Only higher-order reasoners are required because many physics properties are more naturally represented as higher-order objects. For instance, a star's orbit is more realistically represented as a function object that returns the 3-D spatial position of the star for a given time moment. If it were represented as a non-functional object, then much of the description and expressivity would be lost and the representation would be much closer to that of static, physical entities, e.g., a dog or a house.

The idea of making the knowledge-base modular has received much attention from the knowledge representation community. For instance, it is the basis of Sowa's notion of "knowledge soup" [Sowa, 2006]; it suggests that because the knowledge in a person's head is of a disorganised and dynamically changing nature, it consists of many self-consistent chunks of knowledge. Similarly in Cyc [Lenat, 1995], different contexts are represented as separate *microtheories*; each is represented by a set of assertions that corresponds to the relevant facts and assumptions valid in the particular context. Our current representation is modular as the predictive and observational ontologies are already separate representations [Bundy, 2008]. Each of these is internally consistent, but possibly inconsistent with one another [Bundy and Chan, 2008]. We will extend this principle by modularising other contents as well, e.g., foundational theories of physics and mathematics.

2 Reasoning for Ontology Evolution

Physics ontologies are represented at the object-level, quantifying over physics functions, objects, and measurements. Object-level reasoning is therefore needed to, e.g., derive the law of universal gravitation from an ontology containing the

axioms of the Newtonian theory. Repair plans are designed to guide the ontology repair procedure, so they need to be represented at the meta-level quantifying over the object-level expressions. Meta-level reasoning is therefore clearly distinct from that of object-level; for instance, it reasons about whether an ontology can trigger a repair plan by recurring over the syntactic structure. Moreover, ontology manipulation is also guided by the meta-level, e.g., the change of signatures and the addition of axioms.

Some higher-order theorem provers, such as Isabelle, offer a clear separation of the logic into meta- and object-levels. We shall discuss the possibility of using a theorem prover as the sole basis for the implementation.

2.1 Effects of Shallow and Deep Embeddings

There are generally two techniques of encodings: shallow and deep embeddings. In a shallow embedding, only the semantics of the logic needs to be defined. As the syntactic structures are not represented, theorems about the syntax cannot be proven. If the two logics are both shallowly-embedded, it is then difficult for the meta-logic to reason about the syntactical structure of the object-level formalism, e.g., matching a theorem to some trigger formulae. Syntax then cannot be deconstructed by pattern matching because higher-order unification attempts to instantiate some variable to the term in question. We do not want a variable to be instantiated when almost any term is pattern matched against it.

In a deep embedding, both the (abstract) syntax and the semantics of the object-logic must be defined. The syntax is typically defined by an inductive definition. In effect, statements of an object-logic are represented as objects of a data type of the meta-logic. For instance, $f(x)$ becomes `(applic f x)`, where `applic` is of type `(A -> B) -> A -> B` and both `f` and `x` are constants. If inference is to be enabled, the unification algorithm may need to be reimplemented. Note that the current λProlog implementation uses deep embedding to represent the two logics as one.

Neither a pure shallow nor deep embedding appears to enable both inference and the matching and instantiation of trigger formulae. Our solution involves a kind-of hybrid approach.

2.2 Huet's Zipper Data Structure

Due to the rigorous formalisation required, the effort for producing a deep embedding that allows inference to be done can be substantial. Our solution aims at the accommodation of both the shallowly-embedded formulae and their parse trees. It is based on using Huet's *zipper* data structure [Huet, 1997], a data structure that can deconstruct any list- and tree-like data structure and requires a constant time to move the focus position in any location.

The creation of a zipper from the parse tree of a formula resembles the translation from shallow to deep embedding. Of course, it is not true deep embedding since the zipper does not carry information about the semantics, but the representation of the parse tree in a zipper is sufficient for our needs. Navigating around a zipper naturally corresponds to moving around terms, so operations for term manipulation can then become intuitive. We will then have the best of both deep and shallow embeddings, because the zipper itself containing the parse tree can be used for syntactic analysis and the original formula can be recovered for inference.

3 Modularising Knowledge

Currently, the theoretical and sensory ontologies are treated as separate interacting logic theories, e.g., the *Inconstancy* ontology repair plan resolves conflicts between a theoretical and multiple sensory ontologies [Chan and Bundy, 2008]. A major theoretic advantage of having multiple, internally consistent ontologies is the better management of logical inconsistency. Since inconsistent theories can prove all formulae, all instantiations of the triggers of all repair plans would also be provable. Keeping the theories consistent only internally avoids the combinatorial explosion of instantiations, yet a conflict can still be detected at the global level. A high degree of modularity also makes the integration of new ontologies more natural. Instead of adding axioms and changing type declarations to existing ontologies, which can invalidate previous proofs, a new ontology can be wholly introduced to the collection. This is in fact close to a central principle of object-oriented programming; that is, the workings of an object are decoupled from the rest of the system and the addition of a class does not affect other parts of the program.

To further modularise the existing knowledge representation, the physics and mathematical theories can be partitioned into small ontologies. Each ontology resembles a small context, e.g., there could be separate ontologies for a theory of Newtonian mechanics and for a theory of geometry. Certainty factors can be assigned to an ontology, determining the vulnerability of the theory or experiment to repair. For instance, the ontology for a controversial theory should be valued at a lower confidence than an established one. The factors could be expressed using a discrete representation, e.g., `definitely true` and `true by default`, as once represented in Cyc[1].

The collection of ontologies can be translated to form a structured logical representation called a *development graph* [Autexier *et al.*, 1999], in which a node corresponds to an ontology. A node, thus an ontology, can be defined to import signatures and theorems from other nodes. Such morphisms representing relationships between theories are formalised using *definition links*.

For the implementation of ontology repair, repaired ontologies are inserted into the development graph as new nodes, replacing those that correspond to the original ontologies. The new nodes should have the same morphisms as the old, so the morphisms of the replaced nodes are preserved. Clearly, some of the old morphisms may no longer be desirable depending on the type of repair performed and the notion of minimal change adopted.

A development graph is already implemented in HETS [Mossakowski *et al.*, 2007] and MAYA [Autexier *et al.*, 2002], which are systems for managing evolutionary development and for analysis and proof management, respectively. In both systems, a development graph can be created by translating

[1] It is unclear whether the same representation is still being used.

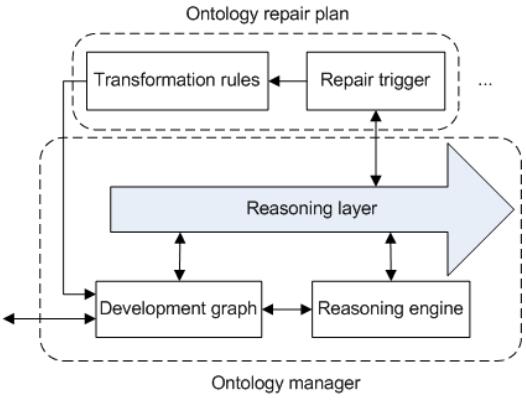


Figure 1: A block diagram of the proposed architecture.

an input HASCASL specification, a higher-order extension of CASL.

4 Putting the Pieces Together

The high-level structure of the architecture is illustrated in Figure 1. To leverage an existing implementation of a development graph, we are currently experimenting with HETS because the implementation is better maintained. Unfortunately, HETS has no support for evolutionary change management. We will likely migrate to a new system to be called DocTip [Krieg-Bruckner *et al.*,], the successor of HETS and MAYA, once it becomes available. Its role in the architecture is depicted as *ontology manager* in Figure 1. Both DocTip and HETS can be connected to a range of theorem provers, including Isabelle. So, the trigger formulae of a repair plan can be inferred from the conflicting ontologies if the ontologies are encoded by using a shallow embedding.

The repair plan programs are responsible for directing the modification of the development graph, e.g., adding and deleting nodes and morphisms. These therefore need to communicate to the ontology manager via an API. Each repair plan needs to pass its trigger formulae to the ontology manager for matching via a *reasoning layer*, which prepares the way for reasoning. A polymorphic, higher-order logic programming language, e.g., λProlog, is needed to encode trigger formulae and transformation rules.

It is, however, not yet clear how to recurse over the syntactical structure of the object-level formulae within the architecture. Future work will involve the investigation of the use of zippers in HETS or DocTip or in the reasoning layer.

5 Conclusion

The current λProlog implementation of GALILEO deeply-embeds the meta- and object-logics into one, but this approach limits the ability to perform inference on the trigger formulae. As an alternative, we have described the possibility of using Huet's zipper data structure to encode the parse tree of a formula. Our new architecture accommodates the use of various reasoning engines, e.g., Isabelle, to prove theorems of ontologies. Since partitioning the knowledge-base into small ontologies helps control logical inconsistency and provides easier maintenance, we go beyond just separating the predictive and sensory ontologies. Development graphs are used for the management of the highly modularised ontologies.

References

[Autexier *et al.*, 1999] S. Autexier, D. Hutter, H. Mantel, and A. Schairer. Towards an Evolutionary Formal Software-Development Using CASL. *Lecture Notes In Computer Science; Vol. 1827*, pages 73–88, 1999.

[Autexier *et al.*, 2002] S. Autexier, D. Hutter, T. Mossakowski, and A. Schairer. The development graph manager MAYA. In *Proceedings of the 9th International Conference on Algebraic Methodology and Software Technology*, pages 495–501. Springer-Verlag London, UK, 2002.

[Bundy and Chan, 2008] A. Bundy and M. Chan. Towards ontology evolution in physics. In W. Hodges, editor, *Procs. Wollic 2008*. LNCS, Springer-Verlag, July 2008.

[Bundy, 2008] Alan Bundy. *Automating Signature Evolution in Logical Theories*, volume 5144/2008, pages 333–338. Springer Berlin / Heidelberg, Jul 2008.

[Chan and Bundy, 2008] M. Chan and A. Bundy. Inconstancy: An ontology repair plan for adding hidden variables. In S. Bringsjord and A. Shilliday, editors, *Symposium on Automated Scientific Discovery*, number FS-08-03 in Technical Report, pages 10–17. AAAI Press, November 2008. ISBN 978-1-57735-395-9.

[Huet, 1997] G. Huet. The zipper. *Journal of Functional Programming*, 7(5):549–554, 1997.

[Krieg-Bruckner *et al.*,] B. Krieg-Bruckner, D. Hutter, M. Kohlhase, C. Luth, T. Mossakowski, L. Schroder, and W. Stephan. Formal Development for Safe Robotics. Available from http://www.dfki.de/sks/formalsafe.

[Lenat, 1995] D.B. Lenat. CYC: a large-scale investment in knowledge infrastructure. *Commun. ACM*, 38(11):33–38, November 1995.

[Miller and Nadathur, 1988] D. Miller and G. Nadathur. An overview of λProlog. In R. Bowen, editor, *Proceedings of the Fifth International Logic Programming Conference/ Fifth Symposium on Logic Programming*. MIT Press, 1988.

[Mossakowski *et al.*, 2007] Till Mossakowski, Christian Maeder, and Klaus Lüttich. The Heterogeneous Tool Set. In Orna Grumberg and Michael Huth, editors, *TACAS 2007*, volume 4424 of *Lecture Notes in Computer Science*, pages 519–522. Springer-Verlag Heidelberg, 2007.

[Sowa, 2006] J.F. Sowa. The challenge of knowledge soup. In J. Ramadas and S. Chunawala, editors, *Research Trends in Science, Technology and Mathematics Education*, Mumbai, 2006. Homi Bhabha Centre.

A Case Study of Ontology Evolution in Atomic Physics as the Basis of the Open Structure Ontology Repair Plan*

Jos Lehmann

School of Informatics, University of Edinburgh
jlehmann@inf.ed.ac.uk

Abstract

In the GALILEO project a number of *Ontology Repair Plans* are being developed and implemented in higher-order logic. These plans resolve a contradiction between two or more ontologies that represent the domain of physics. In this abstract, the transition from Thomson's *plum pudding model* of the atom to Rutherford's *planetary model* is used as the main inspiration of a new ontology repair plan called *Open Structure*

1 Introduction

In the framework of the GALILEO project (Guided Analysis of Logical Inconsistencies Leads to Evolved Ontologies) a number of *Ontology Repair Plans* (ORPs) are being developed and implemented in higher-order logic. For reasons of space, the interested reader is referred to [Bundy and Chan, 2008] to find general indications on how the overall GALILEO approach, and thus what presented in this abstract, relates to analogues in the literature. It should be noted, though, that GALILEO's future research objectives do include a more precise positioning of the notion of ORP with respect to existing formal-ontological and epistemological approaches. ORPs resolve a contradiction between two or more ontologies that represent the domain of physics. In ORPs developed thus far, one of the ontologies represents a physical theory – its theorems are interpreted as expectations or predictions; a second ontology represents a sensory or experimental set-up for that theory – its theorems are interpreted as observations. When the sensory ontology generates a theorem that contradicts a theorem of the theoretical ontology – i.e., when an observation contradicts an expectation – an ORP kicks in and amends the two ontologies. Such repair enables the theoretical ontology to make correct predictions and updates the experimental ontology with the new theory. ORPs may act either as theory revision mechanisms or as signature revision mechanisms – in the latter case they bring about an evolution of the ontologies by changing their representation language.

The development of ORPs is inspired by cases in the history of physics where the transition from an old theory to a new one was forced by a contradiction between observations and predictions. For instance, the discovery of Latent Heat inspired the ORP called *Where is my stuff?* (WMS). This plan changes the signature of the ontologies by splitting a function into a *visible* and an *invisible* part; such repair accounts for the impossibility, at a given moment of the history of physics, to fully measure a given quantity (e.g. heat or presently mass, as Dark Matter has never been measured directly). The transition from Boyle' Law to the Ideal Gas Law inspired the ORP called *Inconstancy*, which changes the signature of the ontologies by adding an argument to a function; it may, for instance, make a constant (e.g. the volume to pressure ratio of a gas) dependent on an quantity (e.g. temperature). The case of the ancient distinction between the Morning Star and the Evening Star, later identified by astronomers as the same planet Venus, inspired the recently proposed ORP called *Unite*. This plan operates as a theory revision mechanism which equates two functions.

The present abstract provides an example of a modeling exercise, from historical analysis to formalization. The transition from Thomson's *plum pudding model* of the atom to Rutherford's *planetary model* is used as source of inspiration of a new ORP called *Open Structure* (OP) and of its inverse, not presented here, called *Close Structure* (CS). OS and CS operate as theory revision mechanisms, which delete one of the cases of a function, thus restructuring a given entity (e.g. an atom) by distributing a given quantity differently (e.g., the work excerted by an atom's positive electric field on a particle orbiting around it)[1]. Section 2 presents part of the historical analysis that led to *Open Structure* and presents the formulae that may be subject to ontological evolution. Section 3 presents *Open Structure* and its application to the case study. Section 4 sets further research objectives.

2 From Thomson's to Rutherford's Atom

Thomson's Atom Around 1904 Thomson believed that the atom was a uniform sphere of positive charge, as represented in Fig. 1 (left), with electrons rotating and oscillating in it. Negative charges were placed on concentric rings or at the

*I'm grateful to Alan Bundy, Michael Chan and the ARCOE-09 reviewers for comments on earlier drafts.

[1] As noted by Alan Bundy during a recent private exchange, in general mathematical terms it may be said that OS and CS let a non-monotonic function evolve into a monotonic one.

center. The number of (electrons on) the rings would increase monotonically while the number of charges at the center would periodically increase and decrease [Baily, 2008].

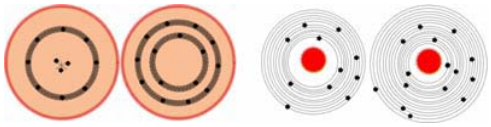


Figure 1: Left: Thomson's atom (11 electron atom and 15 electron atom). Right: Rutherford's atom (11 electron atom 15 electron atom). Black dots and circumferences represent negative charges and their orbits; red circles represent positive charge (more intense where darker).

Scattering Experiment Together with Geiger and Marsden, around 1911 Rutherford learnt how to control a heavy emission named α-particle, i.e. doubly positively charged helium atoms (He2+). Using the apparatus shown in Fig. 2[2], a stream of alpha particles originating from source R was shot at a thin foil of gold atoms (F). Scintillations, which were due to the particles scattering against a rotating screen (S), were observed through a microscope (M) mounted behind the screen. The chamber was evacuated and could be rotated around the foil.

Based on Thomson's model, Rutherford expected the sphere

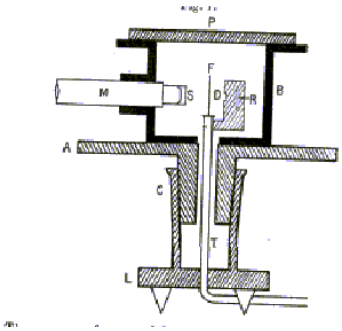


Figure 2: Rutherford's scattering apparatus

of positive charge and the electrons in it to offer virtually no resistance to the passage of α-particles. Given their high speed, their momentum would overcome the repulsion excerted by the positively charged sphere. Thus the particles would be deflected slightly (just like, on left side of Fig. 3, the trajectories with impact parameter around $\pm\ b'$) or go straight through (like virtually all other trajectories). Also, since α-particles are 7000 times heavier than electrons, their trajectory would never be curved by electrons (that is why no negative charges are represented in Fig. 3). Contrary to the expectations, three degrees of scattering were observed:

- most α-particles passed straight through the gold foil *without any or with very little deflection* (like most particles with impact parameters between $\pm\ b$ and $\pm\ b'$ on right side of Fig. 3);
- some α-particles were deflected through *large angles* (i.e. particles with impact paraments around $\pm\ b$);
- a few α-particles *rebounded completely* (i.e. particles with impact parameter around 0).

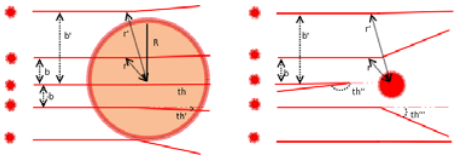


Figure 3: Left: expected scattering. Right: observed scattering. Small red spots represent α-particles; red lines represent particle's paths; half-dashed thin red lines represent ideal undeflected paths; b's and $-b$'s are impact parameters, i.e. perpendicular distances between a particle's velocity vector and the centre of the target atom ($b\ =\ 0$ for third particle); r's are distances between a point of the atom's electric field and the atom's center; R is the atom's radius; th's are scattering angles.

Rutherford's Atom Rutherford analyzed the results of the experiment in terms of Newtonian mechanics and obtained a formula to calculate the *differential scattering cross-section* of the α-particles, i.e. the probability, given the impact parameter, of an α-particle to be deflected through a given angle. This made it possible to derive the following conclusions:

1. Since most of the α-particles were not deflected, there is a lot of empty space in an atom.
2. Since some of the α-particles were deflected, there is a centre of positive charge in an atom.
3. Since very few α-particles rebounded, the nucleus is very dense and hard.
4. Since 1 to 3, the structure of an atom is comparable to the Solar System's structure, with the nucleus forming the main mass and the electrons revolving around it.

Evolution between the Two Atoms As shown in [Zoli, 1998], the differences between the scattering of α-particles in Thomson's and Rutherford's atoms may be expressed in various ways. One way is to define two distinct *scattering potential functions* (like 1 below for Thomson and 2 for Rutherford) to calculate the amount of work excerted by the electric fields of the two atoms when deflecting incident particles placed at a distance r from their centers. Note that both potentials are central but, while 1 is both non-Coulombic (i.e. for values of r lower than the atom's radius R, the potential is directly proportional to r) and Coulombic (i.e. for values of r higher than R, the potential is inversely proportional to r),

[2] Originally in [Geiger and Marsden, 1913], downloaded from: galileo.phys.virginia.edu/classes/252/Rutherford_/Scattering/Rutherford_Scattering.html

2 is only Coulombic (R needs not to be considered).

$$V(r)_T = \begin{cases} \frac{Q_A Q_B}{4\pi\epsilon_0}\frac{1}{r} & R \leq r \\ \frac{Q_A Q_B}{4\pi\epsilon_0}\frac{1}{2R^3}(3R^2 - r^2) & 0 \leq r \leq R \end{cases} \quad (1)$$

$$V(r)_R = \frac{Q_A Q_B}{4\pi\epsilon_0}\frac{1}{r} \quad (2)$$

where Q_A is the charge of incident particle, Q_B is the charge of the target atom, $1/4\pi\epsilon_0$ is the Coulomb constant, r is the distance between the incident particle and the centre of the target atom, R is the radius of the target atom.

3 Open Structure Ontology Repair Plan

In this section the evolution from $V(r)_T$ to $V(r)_R$ is modeled by *Open Structure* (3 below). The idea of the plan is to evolve a function that has values that are partly directly proportional and partly inversely proportional to its argument into a function that is only inversely proportional to its argument. In this way the physical structure of the entity described by the function loses an internal and/or external boundary.

Just like all ORPs, OS models the function that is subject to evolution as *stuff*. V is *stuff*. Just like V ranges over the type dis of distances r's, *stuff* ranges over a type δ. The distance R plays the role of cut-off point of V. So *stuff*'s domain contains a cut-off point, *cop*. As the relation between all other quantities in V is constant, these are modeled as K.

Two cases of contradiction between O_t and O_s trigger OS:

1. In O_t, for all arguments below the cut-off point, the value of *stuff* is directly proportional to the argument, while for all other arguments the value of *stuff* is inversely proportional to the argument. In contrast with this, in O_s the value of *stuff* is always inversely proportional to its argument. In this case, the solution of the contradiction by OS results in the structure of the entity described by the function to lose a physical internal boundary (where the function is at its maximum).
2. In O_t, for all arguments below the cut-off point, the value of *stuff* is inversely proportional to the argument while, for all other arguments, the value of *stuff* is directly proportional to the argument. O_s is the same as in the first case above. Here, after applying OS, the structure of the entity described by the function loses a physical internal boundary (where the function is at its minimum) as well as an external boundary (where the function is at its maximum).

In all cases the contradiction is repaired according to what dictated by O_s (4 through 6 below). This is also true for the inverse of OS, Closed Structure, but here O_s forces the acquisition of an external boundary.

OS may be formlized and applied to the case study as below.

Trigger

$$\begin{aligned} &O_t \vdash d_4 > d_3 \geq cop \geq d_2 > d_1 \wedge \\ &\quad ((stuff(d_2) > stuff(d_1) \wedge stuff(d_3) > stuff(d_4)) \vee \\ &\quad (stuff(d_1) > stuff(d_2) \wedge stuff(d_4) > stuff(d_3))). \\ &O_s \vdash \forall d, d' : \delta.\ d' > d \rightarrow stuff(d) > stuff(d'). \end{aligned} \quad (3)$$

Open Structure

$$\nu(stuff) ::= \lambda d : \delta.\ K/d. \quad (4)$$

Create New Axioms

$$\begin{aligned} &Ax(\nu(O_t)) ::= Ax(O_t) \setminus \\ &\quad \{stuff ::= \lambda d : \delta.\ (cop > d \wedge Kd) \vee K/d\} \cup \\ &\quad \{\nu(stuff) ::= \lambda d : \delta.\ K/d\}. \end{aligned} \quad (5)$$

$$\begin{aligned} &Ax(\nu(O_s)) ::= Ax(O_s) \setminus \\ &\quad \{stuff ::= \lambda cop, d : \delta.\ (cop > d \wedge Kd) \vee K/d\} \cup \\ &\quad \{\nu(stuff) ::= \lambda d : \delta.\ K/d\}. \end{aligned} \quad (6)$$

The following substitutions in the atom case study would yield the repair below:

$$\left\{ V/stuff, d_i/r_i, cop/R, K/\frac{Q_A Q_B}{4\pi\epsilon_0} \right\}$$

$$\begin{aligned} &Ax(\nu(O_t)) ::= Ax(O_t) \setminus \\ &\quad \{V ::= \lambda r : dis.\ (R > r \wedge Kr) \vee K/r\} \cup \\ &\quad \{\nu(V) ::= \lambda r : dis.\ K/r\}. \\ &Ax(\nu(O_s)) ::= Ax(O_s) \setminus \\ &\quad \{V ::= \lambda r : dis.\ (R > r \wedge Kr) \vee K/r\} \cup \\ &\quad \{\nu(V) ::= \lambda r : dis.\ K/r\}. \end{aligned}$$

4 Conclusion

The transition from Thomson's *plum pudding model* of the atom to Rutherford's *planetary model* was used as the basis of a new ontology repair plan called *Open Structure* and of its inverse, *Close Structure*. Future research will focus on:

- the interpretation of the second case of OS: what are the examples of the physical structure represented by O_t in the OS's second case?;
- the interpretation of CS, OS's inverse: what are the examples of the physical structure represented by O_s in CS?
- the alternative to OS and CS: what would be the advantages of modeling the evolution between the two atoms in terms of their different scattering angle functions (i.e. by comparing alternative trajectories of particles that have the same impact paramater)?

References

[Baily, 2008] C. Baily. Atomic modeling in the early 20th century: 1904-1913. Technical report, University of Colorado, Boulder - Department of Physics, 2008.

[Bundy and Chan, 2008] A. Bundy and M. Chan. Towards ontology evolution in physics. *W. Hodges, editor, Procs. Wollic 2008. LNCS Springer-Verlag*, 2008.

[Geiger and Marsden, 1913] H. Geiger and E. Marsden. The laws of deflection of alpha particles through large angles. *Philosophical Magazine (6), 25, 507*, 1913.

[Zoli, 1998] M. Zoli. Alpha-particle scattering in the thomson and rutherford atomic models. *European Journal of Physics 19 187-193*, 19:187–193, 1998.

Atypicalities in Ontologies: Inferring New Facts from Topological Axioms

Christophe Jouis
LIP6 - University Paris 6
Paris, France
cjouis@univ-paris3.fr

Bassel Habib
LIP6 – University Paris 6
Paris, France
Bassel.Habib@lip6.fr

Jie Liu
LIP6 – University Paris 6
Paris, France
Jie.Liu@lip6.fr

Abstract

This paper is a contribution to formal ontology study: the problem of atypical entities. Some individual entities are attached to classes when they do not check all the properties of the class. We introduce the topological operators of interior, border, closure and exterior. These operators allow us to describe whether an entity belonging to a class is typical or not. We define a system of relations of inclusion and membership by adapting the topological operators, based on a precise axiomatic. In this paper, we propose to better formalize these topological relations of inclusion and membership based on the mathematical properties of topological operators. However, there are properties of combining operators of interior, border, closure and exterior allowing the definition of an algebra. We propose to use these mathematical properties as a set of axioms. This set of axioms allows us to establish properties of relations of topological inclusion and membership.

1 Introduction

In a first paper modeling entities in atypical ontologies [Jouis, Bourdaillet 2008], we introduced the topological operators of interior, border, closure and exterior. These operators enabled us to describe whether an entity belonging to a class is typical or not. Some individual entities are attached to classes when they do not check all the properties of the class. We then developed a system of relations of inclusion and membership by adapting topological operators. But the properties of these relations had been introduced in an intuitive way, without relying on a precise axiomatic. In this paper, we propose to better formalize topological relations of inclusion and membership based on the mathematical properties of topological operators. However, there are properties of combining topological operators that allow the definition of an algebra [Kuratowski, 1958]. We propose to use these mathematical properties as a set of axioms. This set of axioms allows us to establish relationships properties of topological inclusion and membership. The paper is organized as follows: In section 2 we recall the issue of atypical entities, existing solutions and why we chose the topology to solve the problem. We recall, in section 3, the main results of basic topological operations. We then define the six relations of topological inclusion and membership. Next section focuses on the properties of the six relations of topological inclusion and membership. Finally, we present interpretations using the relations of topological inclusion and membership to illustrate the phenomenon of typicality identified in section 2. We show the possible inferences.

2 Problematic and Related Works

2.1 Problem

Some entities belong more or less to a class. In particular, some individual entities are attached to classes whereas they do not check all the properties of the class. To illustrate this phenomenon, let us consider the ontological network above (see Figure 1).

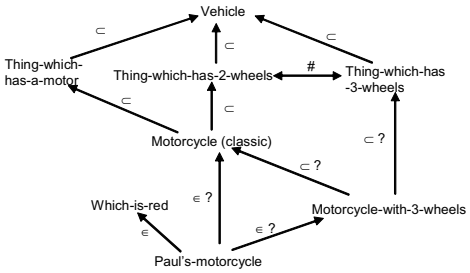


Figure 1: the individual entity [Paul's motorcycle] does not satisfy all the properties of the class [Motorcycle]. The subclass [Motorcycle-with-3-wheels] does not satisfy all the properties of the class [Motorcycle].

This network corresponds to the ten following declarative statements:

(1) A thing which has an engine is a vehicle;
(2) A thing which has two wheels is a vehicle;
(3) A thing which has three wheels is a vehicle;
(4) A motorcycle has two wheels;
(5) A motorcycle has an engine;
(6) There are motorcycles with three wheels;

(7) A thing can not simultaneously have two wheels and three wheels;
(8) Paul's motorcycle is a motorcycle;
(9) Paul's motorcycle has three wheels;
(10) Paul's motorcycle is red.
Because "Paul's motorcycle is a motorcycle", as introduced in the statement (8), it inherits all the typical properties of [Motorcycle] class, in particular, [Thing-which-has-2-wheels]. A paradox is introduced by the statement (7) because "A motorcycle has two wheels" is a general fact but not a global one. The statement (4) "A motorcycle has two wheels" means that "in general, a motorcycle has two wheels but there are some exceptions to this law".
The same phenomenon is observed with distributive classes. Some subclasses are attached more or less to a general class because some of theirs elements may not check all the properties of this general class. To illustrate this phenomenon, let us consider again the ontological network above (see Figure 1): for identical reasons to the first example, the statement (6) introduces a paradox.
We can consider that an individual entity is typical of a class if it checks all the properties of this class. Similarly, we can consider that a class is a typical subclass of another class if all its typical elements check the properties of the more general class. For example, the [Motorcycle] class is a typical subclass of [Thing-which-has-2-wheels] class.
We note that an individual entity is atypical of a class if it does not check all the properties of the class. For example, [Paul's-motorcycle] is an atypical entity of [Motorcycle] class. Similarly, we consider a class as atypical subclass of a more general class if all its typical elements do not check all the properties of the more general class. For example, [Motorcycle-with-3-wheels] class is an atypical subclass of [Motorcycle] class.
It is important to distinguish between atypical individual entity and atypical class. When we talk about [Motorcycle-with-3-wheels], we do not distinguish a particular object, but an undetermined number of objects that have typical properties.
In fact, we do not talk here about a real object, but about a "typical object" that does not exist in reality, but that checks all the properties of the concept by definition: using natural language, we express that by "motorcycle with three wheels". That is what we call the intension vision of a concept. The class associated with a concept includes all the objects, at a certain moment t, checking all the properties of this concept. That is what we call the extension vision of a concept. This set of objects may vary over time, while typical objects of a concept do not change: using natural language, we express that by "the motorcycles with three wheels… (Which are currently on the beach at Daytona…)".
In contrast, an individual entity identifies a particular object that has its own properties in addition to those of the class to which it is attached. For instance, the individual entity [Paul's-motorcycle], which is an entity that inherits properties of the [Motorcycle] class but is also assigned some other specific properties: [Which-is-red], [Motorcycle-which-has-3-wheels], etc.
We thus make successive determinations from the typical object in order to have an object completely determined: an individual entity.
The example of "Paul's motorcycle, which has three wheels" is a particular property of the individual entity [Paul's-motorcycle]. In the case of [Paul's-motorcycle], the property is not considered as a general property. Thus, we must distinguish between two types of atypical properties, which are not similar: those on a concept, i.e. that are valid for all typical elements of the class associated with the concept; and those on a particular object, i.e. "Paul's motorcycle is red".

2.2 Related Works

In Artificial Intelligence, the solution for this kind of problem is default reasoning: an individual A belonging to a concept F inherits concepts subsuming F except contrary indications. This technique of default reasoning led for example Reiter [Reiter 1980] to propose Default Logic. Many other works focus on non-monotonic logics: for instance, the work on Auto epistemic Logic [Moore, 1985], on Belief Revision [Alchourron *et al.*, 1985] and on Circumscription [McCarthy, 1986] among other works.

3 Topological Relations

We postulate that networks of concepts and semantic relationships between concepts can be represented on a plan D. Thus, instances are points of the plan, while classes are demarcated areas of the plan, which consist of: (a) an interior (the typical elements belonging to the class), (b) an exterior (the elements that are not in the class), (c) a border (atypical elements that do not check all the properties of the class, i.e. atypical elements that are neither within nor outside the class).
The properties of the four topological operators of interior (noted i), exterior (noted e), border (noted b) and closure (noted f) are used as a set of axioms.

3.1 Reminder of the remarkable properties of one of the topological operations

Topological operations are applications of the set 2^D in 2^D ($i : 2^D \rightarrow 2^D$).

We have, for instance, for the interior of a set F, noted iF:
1. F open $\Leftrightarrow$ iF = F (by definition);
2. iiF = iF (idempotence);
3. iF $\subset$ F;
4. a) i(F$\cap$G) = iF $\cap$ iG and b) iF$\cup$iG $\subset$ i(F$\cup$G);
5. F$\subset$G $\Rightarrow$ iF$\subset$iG (monotony).

We have, as well, properties for eF, bF et fF.

3.2. Definition of the six topological relations

X represents any given point and {X} a singleton (i.e. the smallest neighbourhood containing X). We note S as the set of singletons of D. F,G, H represent any parts of D that are

not singletons. Those are sets that represent the extensions of concepts.

3.2.1. Membership at the interior of a class (noted $\in$i)
We define (X $\in$i F) if and only if X inherits all the properties of F : X $\in$i F iff {X} $\subset$ iF.
3.2.2. Membership at the exterior of a class (noted $\in$e)
We define (X $\in$e F) if and only if X cannot belong neither the interior nor the border of F (and in the same way recursively for the subclasses of F): X $\in$e F iff {X} $\subset$ eF.
3.2.3. Membership at the border of a class (noted $\in$b)
We define (X $\in$b F) if and only if X is an atypical individual entity of F: X $\in$b F iff {X} $\subset$ bF.
3.2.4. Inclusion at the interior of a class (noted $\subset$i)
We define (F $\subset$i G) if and only if F is a typical subclass of G: F $\subset$i G iff F $\subset$ iG.
3.2.5. Inclusion at the exterior of a class (noted $\subset$e)
We define (F $\subset$e G) iff F cannot be a subclass neither at the interior nor at the border of G: F $\subset$e G iff F $\subset$ eG.
3.2.6. Inclusion at the border of a class (noted $\subset$b)
We define (F $\subset$b G) if and only if F is an atypical subclass of the class G: F $\subset$b G iff F $\subset$ bG.
We note that $\in$I $\in$b et $\in$e are subsets of the Cartesian product S × D, while $\subset$i, $\subset$b et $\subset$e are subsets of the Cartesian product D × D.
Using properties of i, e, b and f operators, we can deduce inference rules of the relations of inclusion and membership. We identified thirteen rules. In particular, we have:

- A4: (X$\in$i G) $\wedge$ (G $\subset$i C) $\Rightarrow$ (X $\in$i H)
- A5: (X$\in$i G) $\wedge$ (G $\subset$b H) $\Rightarrow$ (X $\in$b H)
- f1: (G $\subset$e F) $\wedge$ (H $\subset$i G) $\Rightarrow$ (H $\subset$e F)
- f2: (A $\subset$e B) $\wedge$ (C $\in$i A) $\Rightarrow$ (C $\in$e B)

4 Topological interpretation of the example

Figure 2 represents an interpretation of the example using our topological relations where dotted arrows represent some possible deductions thanks to the rules of combination we defined in the previous section. In particular, we notice that the [Paul's motorcycle] is an atypical element of the class [Motorcycle]. Furthermore, we notice that the class [Motorcycle-with-3-wheels] is an atypical subclass of the class [Motorcycle].

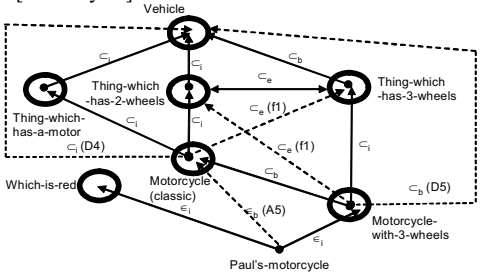


Figure 2: some possible deductions

(11) The class [Motorcycle-with-3-wheels] cannot be nor a typical subclass nor an atypical subclass of [Thing-which-has-2-wheels] (statements 6 and 7 and rule F1);
(12) The class [Motorcycle] is a typical subclass of the class [Vehicle] (statements 2 and 4 and rule D4);
(13) The class [Motorcycle-with-3-wheels] is an atypical subclass of the class [Vehicle] (statements 3, 6 and rule D5);
(14) [Paul's-motorcycle] cannot belong neither the interior nor the border of the class [Thing-which-has-2-wheels] class (statements 9 and 11 and rule F2).

5 Conclusions

In this paper, the topological concepts of interior, border, closure and exterior are used to specify whether an individual entity belonging to a class is typical or not. By adapting these operators, a system of relations is defined. We proposed to formalize the topological relations of inclusion and membership based on the mathematical properties of topological operators. We used the properties of combination of these operators that allow the definition of an algebra. We proposed as well to use these mathematical properties as a set of axioms. This set of axiom allows us to establish the properties of the relations of topological inclusion and membership. This model has been implemented in ANSPROLOG* [Baral, 2003]. This language used to describe directly the facts (i.e. the initial network) and inference rules in the same formalism as PROLOG. It has the ability to represent normative statements, exceptions, and default statements, and is able to reason with them.

References

[Alchourron, et al., 1985] Alchourron C.E., Gardenfors, P., and Makinson, D. 1985. *On the logic of theory change: partial meet functions for contraction and revision*. In Journal of Symbolic Logic, 50:510-530, 1985.

[Baral, 2003] Baral, C. Knowledge representation, reasoning and declarative problem solving, Cambridge University Press, 2003.

[Jouis, Bourdaillet, 2008] Jouis, C. & Bourdaillet, J. (2008), *Exceptions in Ontologies: when Topology meets Typicality*, In FLAIRS-21 (The 21st International Florida Artificial Intelligence Research Society Conference", Coconut Grove, Florida, USA, May 14-16, 2008, published by the AAAI, to appear.

[Kuratowski, 1958] Kuratowski, C. *Topologie*. 2 vol. Panstwowe Wydawnie two Naukowe, Varsovie 1958.

[McCarthy, 1986] McCarthy, J. *Application of circumscription formalizing common-sense knowledge*. Artificial Intelligence. 28:86-116. 1986.

[Moore, 1985] Moore, R. *Semantical considerations on nonmonotonic reasoning.* Artificial Intelligence. 25(1):75-94, 1985.

[Reiter, 1980] Reiter, R. *A logic for default reasoning.* Artificial Intelligence, 13:81-132, 1980.