



Inference in Computational Semantics ICoS-5

Buxton, England

April 20–21, 2006



Workshop Proceedings

Johan Bos and Alexander Koller (Eds.)

About ICoS

Natural Language Processing has reached a stage where the exploration and development of inference is one of its most pressing tasks. On the theoretical side, it is clear that inference plays a key role in such areas as semantic construction and the management of discourse and dialogue. On the practical side, the use of sophisticated inference methods is expected to play a crucial role in such application areas as natural language generation, automatic question answering, and spoken dialogue systems. At the same time, inference tools and resources have matured to the point that they can become useful for actual applications. Automated theorem provers and other inference tools are becoming increasingly robust and efficient. And the world knowledge bottleneck is being addressed from different angles, by the manual development of knowledge resources, the merging of existing such resources, and the automated extraction of world knowledge from corpora.

The Inference in Computational Semantics (ICoS) workshops are intended to bring together researchers from areas such as Computational Linguistics, Artificial Intelligence, Computer Science, Formal Semantics, and Logic, to discuss approaches to, and applications of, inference in natural language semantics. ICoS-1 took place in Amsterdam, the Netherlands, on August 15, 1999. ICoS-2 was organised in Dagstuhl Castle, Germany, on July 29–30, 2000. ICoS-3 was co-located with the International Joint Conference on Automated Reasoning (IJCAR 2001), which took place on June 18–23, 2001 at Siena, Italy. ICoS-4 took place in Nancy, France, on September 25–26, 2003.

Welcome to ICoS-5

ICoS-5 is organised as a two-day event at the University of Derby College, Buxton, England, taking place on April 20–21. The programme features three invited presentations (by Christian Ebert, Patrick Pantel, and Stephen Pulman). In addition, we have selected twelve regular papers from 24 submissions, which span topics ranging from the use of inference techniques for formal and computational semantics, over new methods for knowledge extraction from corpora, to NLP applications that use inference tools.

We have accepted six of the remaining submissions as “short papers”. These papers will be presented as posters and demos at the workshop, and we believe that this new format will be a worthwhile experience both for authors and for participants.

We would like to thank the members of the programme committee, who produced highly informative reviews and made it possible to set up the broad and strong program for ICoS-5. Finally, we are very grateful to the local organisers, Ian Pratt-Hartmann and Allan Ramsay, who did a flawless job of preparing what promises to be a wonderful workshop location for us.

Rome and Saarbrücken, March 2006
Johan Bos & Alexander Koller (co-chairs)

Workshop Organization

Programme Committee

<i>Carlos Areces</i>	INRIA Lorraine
<i>Peter Baumgartner</i>	National ICT Australia
<i>Christoph Benzmueller</i>	University of the Saarland
<i>Raffaella Bernardi</i>	Free University of Bozen-Bolzano
<i>Patrick Blackburn</i>	INRIA Lorraine
<i>Johan Bos (co-chair)</i>	University of Roma “La Sapienza”
<i>Harry Bunt</i>	Tilburg University
<i>Ann Copestake</i>	University of Cambridge
<i>Dick Crouch</i>	PARC
<i>Ido Dagan</i>	Bar Ilan University
<i>Kees van Deemter</i>	University of Aberdeen
<i>Nissim Francez</i>	Technion
<i>Claire Gardent</i>	CNRS/LORIA
<i>Alexander Koller (co-chair)</i>	University of the Saarland
<i>Shalom Lappin</i>	King’s College London
<i>Alex Lascarides</i>	University of Edinburgh
<i>Bernardo Magnini</i>	ITC-Irst
<i>Katja Markert</i>	University of Leeds
<i>Dan Moldovan</i>	University of Texas at Dallas
<i>Jeff Pelletier</i>	Simon Fraser University
<i>Maarten de Rijke</i>	University of Amsterdam
<i>Michael Schiehlen</i>	University of Stuttgart
<i>Matthew Stone</i>	Rutgers
<i>Bonnie Webber</i>	University of Edinburgh

Local Organizers



The University
of Manchester

Ian Pratt-Hartmann School of Computer Science, University of Manchester
Allan Ramsay School of Informatics, University of Manchester

Table of Contents

Abstracts of Invited Talks

Expressive Power and Complexity of Underspecified Representations	1
<i>Christian Ebert</i>	
Knowledge Harvesting and Fusion from Small and Large Corpora	3
<i>Patrick Pantel</i>	
Bridging the gap between formal and computational semantics	5
<i>Stephen Pulman</i>	

Regular Papers

Anaphora Resolution and Minimal Models	7
<i>Ariel Cohen</i>	
Extracting Formal Specifications from Natural Language Regulatory Documents	17
<i>Nikhil Dinesh, Aravind Joshi, Insup Lee and Bonnie Webber</i>	
How to change a person's mind: Understanding the difference between the effects and consequences of speech acts	27
<i>Debora Field and Allan Ramsay</i>	
Towards a redundancy elimination algorithm for underspecified descriptions	37
<i>Alexander Koller and Stefan Thater</i>	
Quantifiers in Dependency Tree Semantics	47
<i>Leonardo Lesmo, Livio Robaldo, and Jelle Gerbrandy</i>	
Controlled Language for Geographical Information System Queries	57
<i>Sela Mador-Haim, Yoad Winter and Anthony Braun</i>	
Computing relative polarity for textual inference	67
<i>Rowan Nairn, Cleo Condoravdi, and Lauri Karttunen</i>	
Using Answer Set Programming in an inference-based approach to Natural Language Semantics	77
<i>Farid Nouioua and Pascal Nicolas</i>	
A Bootstrapping Algorithm for Automatically Harvesting Semantic Relations	87
<i>Marco Pennacchiotti and Patrick Pantel</i>	
Concepts across categories	97
<i>Hilke Reckman and Crit Cremers</i>	
Multi-dimensional Temporal Logic for Events and States	107
<i>Satoshi Tojo</i>	
Considerations on the nature of metaphorical meaning arising from a computational treatment of metaphor interpretation	117
<i>A.M. Wallington, R. Agerri, J.A. Barnden, S.R. Glasbey, and M.G. Lee</i>	

Short Papers

Supporting temporal question answering: strategies for offline data collection	127
<i>David Ahn, Steven Schockaert, Martine De Cock, and Etienne Kerre</i>	
Formal semantics of verbs for knowledge inference	133
<i>Igor Boyko</i>	
Ingredients of a first-order account of bridging	139
<i>Philipp Cimiano</i>	
A Computational Theory of Inference for Arithmetic Explanation	145
<i>Albert Goldfain</i>	
Towards a Logical Foundation of Semantic Networks - A Typology of Descriptive Means for Semantic Inference	151
<i>Hermann Helbig and Ingo Glöckner</i>	
The Alligator theorem prover for dependent type systems: Description and proof samples	157
<i>Paul Piwek</i>	

Expressive Power and Complexity of Underspecified Representations

Christian Ebert
Fakultät für Linguistik und Literaturwissenschaft
Universität Bielefeld

My talk will be about two requirements on *Underspecified Representation Formalisms* in the context of underspecification of scope. The requirement on *partial disambiguation*, stating that partially disambiguated ambiguities need to be represented, does not carry much content unless it has become clear, exactly what those ambiguities are. In line with König and Reyle [1999], I will argue that *all* theoretically possible patterns of ambiguity, i.e. subsets of readings, can occur in natural language, in particular when underspecified representations are assumed to represent patterns of ambiguity that arise through disambiguation by discourse or even by world knowledge. Therefore an underspecified representation formalism can only be regarded as *expressively complete*, if it provides representations for all potential subsets of readings. Taking a closer look at recent prominent approaches to scope underspecification, namely Hole Semantics [Bos, 2002], Minimal Recursion Semantics [Copestake et al., 1999], and Normal Dominance Constraints [Koller, 2004], it turns out that none of these formalisms is expressively complete. Furthermore, these incompleteness results allow for a straightforward comparison of the discussed approaches with respect to expressive power.

The second requirement is the *avoidance of combinatorial explosion*. I will argue that the decisive process that determines the efficiency of an underspecification approach is the construction phase of the representations and not so much the check for satisfiability or enumeration of readings. Thus the desired avoidance of combinatorial explosion comes down to a requirement on a feasible construction procedure and hence to a requirement on the maximal 'size' of the representations, which can only be fulfilled if the involved representations are in some sense more *compact* than the mere listing of the available readings. Unfortunately it turns out that due to the rapid growth of the number of potential patterns of ambiguity, the two requirements of compactness and expressive completeness cannot be fulfilled at the same time. In other words, I will show that any underspecified representation formalism must necessarily miss out on some of the potential patterns of ambiguity or run into the combinatorial explosion problem [Ebert, 2005].

References

- Johan Bos. *Underspecification and Resolution in Discourse Semantics*. PhD thesis, Universität des Saarlandes, 2002.
- Ann Copestake, Dan Flickinger, and Ivan A. Sag. Minimal Recursion Semantics – An Introduction. Technical report, CSLI, Stanford University, 1999. (Draft).
- Christian Ebert. *Formal Investigations of Underspecified Representations*. PhD thesis, King's College London, 2005.
- Alexander Koller. *Constrained-Based And Graph-Based Resolution of Ambiguities in Natural Language*. PhD thesis, Universität des Saarlandes, July 2004.
- Esther König and Uwe Reyle. A General Reasoning Scheme for Underspecified Representations. In *Logic, Language and Reasoning. Essays in Honour of Dov Gabbay.*, pages 251–277. Kluwer, 1999.

Knowledge Harvesting and Fusion from Small and Large Corpora

Patrick Pantel

ISI, University of Southern California

Inferencing requires large collections of axioms and knowledge bases of interrelated concepts and instances. In the past decade, researchers have explored many approaches to automatically learn such knowledge from textual corpora. In this two-part talk, we will discuss the challenges of harvesting knowledge from various corpus sizes and we will look at some recent attempts at fusing the knowledge into a single semantic repository. On the harvesting front, we will focus on the challenges posed by three types of corpora: i) the Web (order of 10^{12} words) – pattern learning algorithms, models of reliability, and scalable computing; ii) large corpora such as newswire collections (order of 10^9 words) – amenable to complex natural language processing and clustering; and iii) small corpora such as college textbooks (order of 10^5 words) – in which low redundancy requires leveraging external resources such as the Web and ontologies.

This multitude of harvested knowledge inevitably overlaps and is partially inconsistent and incompatible since information can be expressed across data sources in so many ways. However, little effort has been spent on fusing such harvested knowledge into an overarching consistent semantic repository. We will present a recent algorithm for linking semantic resources using grammatical templates. We will also present an algorithm to automatically induce conceptual relations between mid-level ontological concepts and then disambiguate instances with regard to the most appropriate senses in the ontology, using the induced conceptual relations.

Bridging the gap between formal and computational semantics

Stephen Pulman

Computational Linguistics Group, Oxford University.

The literature in formal linguistic semantics contains a wealth of fine grained and detailed analyses of many linguistic phenomena. But very little of this work has found its way into implementations, despite a widespread feeling (among linguists at least) that this can't be very difficult in principle: you just fix a grammar to produce the right logical forms and hook them up to a theorem prover, don't you? In this talk I take a representative analysis of adjectival comparatives and ask what steps one might actually have to go through so as to use this analysis in a realistic question-answering setting. I then try to identify some general conclusions that can be drawn from this exercise.

Anaphora Resolution and Minimal Models

Ariel Cohen

Ben-Gurion University, Israel

arikk@bgu.ac.il

Abstract

Some anaphora resolution algorithms are based on model builders, and use the fact that they generate minimal models: only those elements that are necessary are postulated to exist in the model. In this way, such systems have the desirable property that if anaphora can be resolved to a linguistically available antecedent, this resolution applies, and only if there is no suitable antecedent, a deictic reading is generated.

In this paper I formalize the entailments that follow from such anaphora resolution algorithms. In particular, I will suggest a simple, linguistically motivated, underspecified representation for anaphora—DRT, and place the burden of the resolution of anaphora and its consequences on an independently motivated logic for default reasoning—Default Logic.

1 Introduction

Consider a simple case of ambiguous anaphoric reference:

- (1) I had gone to see John before I visited Bill and Mary. He doesn't want to speak with her.

What can we say about the resolution of the anaphora? The pronoun *her* probably refers to Mary; the pronoun *he* is ambiguous between John and Bill, but most likely refers to John. And either pronoun (or both) may be used deictically, referring to some other individual that is not denoted by a linguistic antecedent. What we would like is a system that allows us to represent all these options, pick those we consider plausible, and draw some inferences even in the absence of a clear resolution.

Intuitively, the deictic interpretation is dispreferred; we will assume it only if there is no suitable linguistic antecedent. An elegant explanation of this fact can be provided by anaphora resolution algorithms that use *domain building* techniques (e.g., [1,10,11]). Model builders receive as input a set of propositions, and produce a model for them if such exists. Typically, the models so generated are minimal, i.e. models whose domain is only as large as it needs to

be. Thus, if the referent of the pronoun can be identified with a linguistic antecedent, no additional elements need to be postulated. Only if this turns out to be impossible, will an additional element be added to the model, resulting in the deictic reading.

The goal of this paper is not to propose new algorithms, but to formalize the idea of using minimal models to resolve (pronominal) anaphora, and the conclusions that can be drawn by such a system in case the anaphora is not resolved.¹

2 An Underspecified Representation for Anaphora

As the discourse in (1) exemplifies, anaphora is often ambiguous. Moreover, the deictic possibility always exists, so it is always possible, in principle, that what we had identified as the antecedent of a pronoun actually is not, and the pronoun is used deictically. In the case of (1), since we have two pronouns, one with three possible interpretations (John, Bill, or the deictic use) and the other with two (Mary or deictic), we will have six potential interpretations. We need to be able to represent the ambiguity, but still draw inferences as best we can on the basis of what we know. This calls for some sort of underspecified representation, and some inference mechanism to derive conclusions from it.

Many special formalisms have been proposed, whose sole purpose is to allow efficient representation of and reasoning with underspecification. I will not, however, go down this road, for several reasons. A formalism that is not independently motivated on linguistic grounds, and whose sole justification is to represent underspecification, may work in a practical system, but its explanatory adequacy from a linguistic point of view would be dubious. To give one example, recall that deictic readings of a pronoun are always possible, and this is the case across languages. Why is this? Why don't we have languages where pronouns are restricted to linguistic antecedents only, and deictic readings are indicated only by, say, demonstratives? A formalism that is only geared toward underspecification would be quite adequate if pronouns could only refer to linguistic antecedents, and it is hard to see why it would necessitate the availability of deictic readings. It is, of course, preferable to have the possibility of deictic readings follow directly from the representation, thus explaining the puzzle.

Furthermore, a nonstandard representation will typically require nonstandard inference methods, especially tailored for the representation.² Again, these inference methods would not be independently justified, unlike rules of common-sense inference that must, in one way or another, be used in order to understand natural language.

¹ While this paper only deals with pronominal anaphora, the approach may be extended to handle definite descriptions—see [6,7]) for an account based on model building.

² Though see [15], who uses a nonstandard representation of anaphora, but applies Default Logic to generate its perceived readings.

An additional reason for keeping the representation as simple and as close to standard linguistic representations as possible is the fact that it is not likely to be replaced by a fully specified representation during the interpretation process. Normally, one uses an underspecified representation in the hope that, in the fullness of time, or as the need arises, it will be fully specified. In this sense, an underspecified representation is only a “temporary measure.” However, because it is always possible to interpret pronouns deictically, we can never fully specify the representation. The possibility always exists that we will receive later some information that will force us to interpret the pronoun deictically and undo our previous resolution. Hence, the representation of anaphora cannot be treated as a temporary measure, and must be as close as possible to the fully motivated representation.

In this paper I am going to suggest that we don’t need to look far for a representation and its associated inference method. A standard, linguistically motivated representation, without special machinery for underspecification, will do.³ For concreteness, I choose Discourse Representation Theory ([9]). Thus, for example, the discourse in (1) will be represented by the following DRS (here and elsewhere, ignoring tense and the possibility of a collective reading of the conjunction):

	x y z u v
(2)	John(x)
	Bill(y)
	Mary(z)
	go-to-see(I,x)
	visit(I,y)
	visit(I,z)
	male(u)
	female(v)
	\neg want-to-speak(u,v)

Note that this DRS does not resolve the anaphora. In this representation, u and v are subject to existential closure, and all we know is that *some* antecedent exists. So, in effect, the DRS (2) is an underspecified representation, containing all the possible ways of resolving the anaphora. Any specific resolution of the anaphora results in the addition of equalities identifying the referents of the pronouns. For example:

³ Of course, it may be the case that some sort of special underspecified representation is needed for other reasons, e.g., to represent scope ambiguities. All I claim is that such special representations are not necessitated by the need to represent anaphora.

(3)

x y z u v
John(x)
Bill(y)
Mary(z)
go-to-see(I,x)
visit(I,y)
visit(I,z)
male(u)
female(v)
\neg want-to-speak(u,v)
$u=x$
$v=z$

The problem of anaphora resolution now becomes the problem of inferring the necessary equalities from the representation. Of course, DRT places some constraints on acceptable antecedents—they have to be accessible. Accessibility constraints can be modeled simply as inequalities between all inaccessible pairs of discourse referents. Additional constraints come from our world knowledge. For example, if we know that *her* must refer to a female individual, and that John is not female, we know that John cannot be a suitable antecedent.

3 Default Logic

Inferring the equalities identifying pronoun with antecedent must be nonmonotonic: we may later find that our anaphora resolution was wrong, and revise it. Some form of nonmonotonic reasoning, attempting to derive consistent conclusions from an incomplete description of the world, is independently necessary for any kind of system that attempts to draw inferences from natural language texts. Thus, instead of devising a special form of inference mechanism for our underspecified representation, I will use well studied and independently motivated mechanisms for nonmonotonic reasoning (cf. [12,15]).

Specifically, I choose Default Logic ([17]). Default Logic is one of the most widely used nonmonotonic formalisms and may be the only one that has a clearly useful contribution to the wider field of computer science through logic programming and database theory.

A *default theory* is a pair (D, A) , where D is a set of defaults and A is a set of first-order sentences (axioms). Defaults are expressions of the form

$$(4) \quad \frac{\alpha(x) : \beta_1(x), \dots, \beta_m(x)}{\gamma(x)},$$

where $\alpha(x), \beta_1(x), \dots, \beta_m(x), m \geq 1$, and $\gamma(x)$ are formulas of first-order logic whose free variables are among $x = x_1, \dots, x_n$. A default is *closed* if none of $\alpha, \beta_1, \dots, \beta_m$, and γ contains a free variable. Otherwise it is *open*.

Roughly speaking, the intuitive meaning of a default is as follows. For every n -tuple of objects $t = t_1, \dots, t_n$, if $\alpha(t)$ is believed, and the $\beta_i(t)$ s are consistent with one's beliefs, then one is permitted to deduce $\gamma(t)$.

Crucial to the interpretation of Default Logic is the notion of an *extension*. Roughly speaking, an extension of a default theory is a set of statements containing all the logical entailments of the theory, plus as many of the default inferences as can be consistently believed. Sometimes a default theory has more than one extension, as in the well known *Nixon diamond*. Suppose we have the following set of defaults:

$$(5) \quad \left\{ \frac{\mathbf{Quaker}(x) : \mathbf{pacifist}(x)}{\mathbf{pacifist}(x)}, \frac{\mathbf{Republican}(x) : \neg\mathbf{pacifist}(x)}{\neg\mathbf{pacifist}(x)} \right\}.$$

If Nixon is both a Quaker and a Republican, in one extension he will be pacifist, and in another he won't be. So, is Nixon a pacifist or isn't he?

When faced with multiple extensions, there are two general strategies we can use to decide which conclusions to accept: skeptical or credulous reasoning. Skeptical reasoning means taking only what is true in all extensions. In the case of the Nixon diamond, we will believe neither that Nixon is a pacifist, nor that he is not a pacifist. Credulous reasoning means picking one extension, based on whatever principles one deems appropriate, and accepting its conclusions. This means we will pick one extension, perhaps using our knowledge of Nixon's statements and actions, and based on this extension, conclude whether he is a pacifist or not.

4 Equality by Default

4.1 A default rule for equality

Resolving anaphora means generating an equality between two discourse referents. I suggest that we will generate such an equality by default: we assume that two elements are equal if they cannot be proved to be different. The idea underlying this notion has been proposed, though not formalized, in [2]. Charniak's approach is further explored in [5], and formalized more fully in [3,4].

The idea of equality by default can be implemented in Default Logic very simply, by adding the following default:

$$(6) \quad \frac{: x = y}{x = y}$$

This default rule means that whenever it is consistent to assume that two

elements are the same, conclude that they are.⁴ What does it mean to say that it is consistent to assume $x = y$? It means that it is not known that $x \neq y$. From the axioms of equality it follows that this is equivalent to saying that there is no property ϕ s.t. we know $\phi(x)$ but we also know $\neg\phi(y)$.

4.2 Minimality of models

In order to explain what it means for the models of our theory to be minimal, we will need some definitions. In particular, since (6) is an open default, we need to provide a semantic definition of extensions of open default theories. Since model builders generate what are, in essence, Herbrand models, it seems natural to assume that the theory domain is a Herbrand universe (cf. [14, Chapter 1, §3]). Fortunately, such a definition has already been proposed ([13,8]), and I will follow it here.

Suppose we have a first order language \mathcal{L} , and we augment it with a set of new constants, b , calling the resulting language \mathcal{L}_b . The set of all closed terms of the language \mathcal{L}_b is called the *Herbrand universe* of \mathcal{L}_b and is denoted $\mathbf{T}_{\mathcal{L}_b}$.

A *Herbrand b -interpretation* is a set of closed atomic formulas of \mathcal{L}_b .

Let w be a Herbrand b -interpretation and let φ be a closed formula over \mathcal{L}_b . We say that w *satisfies* φ , denoted $w \models \varphi$, if the following holds:

- (i) If φ is an atomic formula, then $w \models \varphi$ if and only if $\varphi \in w$;
- (ii) $w \models \varphi \rightarrow \psi$ if and only if $w \not\models \varphi$ or $w \models \psi$;
- (iii) $w \models \neg\varphi$ if and only if $w \not\models \varphi$; and
- (iv) $w \models \forall x\varphi(x)$ if and only if for each $t \in \mathbf{T}_{\mathcal{L}_b}$, $w \models \varphi(t)$.

For a Herbrand b -interpretation w , the \mathcal{L}_b -*theory* of w , denoted $\mathbf{Th}_{\mathcal{L}_b}(w)$, is the set of all closed formulas of \mathcal{L}_b satisfied by w . For a set of Herbrand b -interpretations W , the \mathcal{L}_b -*theory* of W , denoted $\mathbf{Th}_{\mathcal{L}_b}(W)$, is the set of all closed formulas of \mathcal{L}_b satisfied by all elements of W .

Let E be a set of closed formulas over \mathcal{L}_b . We say that w is a *Herbrand b -model* of E , denoted by $w \models E$, if $E \subseteq \mathbf{Th}_{\mathcal{L}_b}(w)$.

Extensions of open default theories are then defined as follows:

Definition 1 (cf. [8, Definition 27]) *Let b be a set of new constant symbols and let (D, A) be a default theory. For any set of Herbrand b -interpretations W let $\Delta_{(D,A)}^b(W)$ be the largest set V of Herbrand b -models of A that satisfies the following condition.*

For any default $\frac{\alpha(x) : \beta_1(x), \beta_2(x), \dots, \beta_m(x)}{\gamma(x)} \in D$ and any tuple t of elements of $\mathbf{T}_{\mathcal{L}_b}$ if $V \models \alpha(t)$ and $W \not\models \neg\beta_i(t)$, $i = 1, 2, \dots, m$, then $V \models \gamma(t)$.

A set of sentences E is called a b -extension for (D, A) if $E = \mathbf{Th}_{\mathcal{L}_b}(W)$

⁴ Note that this is, in a sense, the opposite of the Unique Name Assumption ([16]). The uniqueness of names can still be ensured, by following standard DRT practice and defining appropriate external anchors.

for some fixpoint W of $\Delta_{(D,A)}^b$.

It has been shown ([4]) that if E is a b -extension for the default theory $\left(\left\{\frac{x=y}{x=y}\right\}, A\right)$, and w is a Herbrand b -model of E , then w is *minimal*. That is to say, there is no Herbrand b -model w' of E such that

$$(7) \quad \{\langle t_1, t_2 \rangle : w \models t_1 = t_2\} \subset \{\langle t_1, t_2 \rangle : w' \models t_1 = t_2\}.$$

In other words, the proposed default theory minimizes the number of different elements in the models, as desired.

4.3 Deictic interpretations

It turns out that using Herbrand models has a consequence that is particularly important for our purposes. Note that the new elements introduced in b , by being new, are equal by default to any term. In particular, they are equal by default to any pronoun; this is the reason why deictic interpretations of pronouns are always possible. Hence, we have a logical explanation for a linguistic phenomenon—the universal availability of deictic readings.

Note that this theory allows deictic readings, but only as a last resort, when no other readings are possible. Given the discourse in (1), we have a good reason to believe that *her* refers to Mary, i.e. $v = z$. It is true that we have in the Herbrand model additional new terms, but this does not negate the minimality of the model. Since these terms are new, nothing is known about them and consequently it is consistent to assume that, for any such new term n_i , $v = n_i$. It is also consistent to hold the conjunction of all these beliefs, namely the belief that $n_1 = n_2 = n_3 = \dots = v$. So, the model is, indeed, minimal; the addition of new constants does not mean that they denote additional entities. Thus, we capture the intuition that deictic readings are dispreferred, and are only available when no suitable antecedent is available. Note that if we didn't have this requirement of minimality, deictic readings would be on an equal footing with anaphoric readings.

If necessary, however, we *can* get a deictic interpretation, i.e. equate the pronoun with an element that is different from all other discourse referents. This happens when no possible antecedent is available, i.e. for every discourse referent t other than v , we know, or can deduce $v \neq t$. Then, we will have an extension where for some new term n_i , $v = n_i$. By the axioms of equality, n_i will not be equal to any of the other discourse referents, hence the domain will not be minimal. Of course, we may have an extension where the new terms are equal to other terms, but none will be equal to v ; but this extension will not constitute resolution of the anaphora, and will therefore be ruled out.

5 Inference

Let us see the kinds of inferences that this theory gives rise to. First, note that, although we are quite liberal in our assumption of equality, we can still rule

out inappropriate antecedents. Recall that antecedents that are not accessible, in the DRT sense, will be explicitly stated to be different from the pronoun. Hence, obviously, it will not be consistent to assume that they are, so such equalities will arise in any extension.

We can also rule out antecedents that are semantically incompatible. For example, if we know that **male**(u) but \neg **male**(z), we cannot assume $u = z$; this is because if u is male and z is female, they have to be different, by the axioms of equality.

But suppose we have two acceptable antecedents for a pronoun u : in our example, it is possible that $u = x$ (John), but it is also possible that $u = y$ (Bill). If we know that they are different people, we know $x \neq y$, so it is impossible to believe both $u = x$ and $u = y$. We will therefore have two extensions: in one of them, the pronoun is equated with John, and in the other, with Bill.

How do we deal with these extensions? If we prefer one antecedent over the other, for reasons of pragmatic plausibility or salience, we apply credulous reasoning and pick the appropriate extension. In this extension, the pronoun will be equated with the chosen antecedent; hence, by the nature of equality, all properties of the antecedent will also hold of the pronoun.

At other times, however, the anaphora may be genuinely ambiguous, and we may have no reason to prefer one reading over the other. In this case, it makes sense to apply skeptical reasoning, and accept only what is true in all candidate extensions.

Consider, for example, the following discourse:

(8) John met Bill at the ice cream parlor. He was upset.

In this case, the pronoun may be equated with either John or Bill, and there are no good grounds, without further context, to decide between them. Yet, we do know something about the antecedent of the pronoun: he was at the ice cream parlor. We know this because we know that both John and Bill were there, and the pronoun refers to one of them. Skeptical reasoning will, indeed, give us precisely this result, since in both extensions, the pronoun has the properties that its antecedent has.

But now suppose that one possible antecedent has a property that the other one lacks:

(9) John walked along the sidewalk and saw Bill inside the ice cream parlor. He was upset.

In this case, Bill has the property of being inside the ice cream parlor, but John does not. Thus, in one extension, the pronoun will have this property, and in another—its negation. If we have no reason to prefer one extension over the other, we will apply skeptical reasoning, and will not conclude of the referent either that he is or that he is not inside the ice cream parlor. This appears intuitively correct.

Now suppose we know that some property holds of one potential antecedent, but we don't know whether it holds of another:

- (10) While eating ice cream, John saw Bill at the ice cream parlor. He was upset.

We know that John was eating ice cream, but we do not know whether Bill was eating ice cream too or not. In this case, intuitively, we cannot conclude about the antecedent of the pronoun that he was eating ice cream, although this is consistent with him being either John or Bill. Indeed, the proposed system conforms with this judgment. This is because in one extension, the one where the pronoun is associated with John, the property of eating ice cream is predicated of the discourse referent corresponding to the pronoun. But in the other extension, neither this property nor its negation will be so predicated. So, in this extension it will not be true that “he” is eating ice cream, hence skeptical reasoning will not license this inference.

Note that I have ignored here the addition of the new terms. The reason is simple: since they are new, they do not make a difference to the inference patterns discussed above. Consider, for example, the inference associated with (8) again. Suppose we have a new term n_i . So long as it is possible to find at least one antecedent to the pronoun, a model for the deictic reading, i.e. where the pronoun is equated with n_i but with no other element, will not be minimal, hence it will not be the model of any extension. In every extension, then, the pronoun u will be equated with some discourse referent x . Now, suppose $n_i = x$. In this case, by the axioms of equality, n_i will also have the property of being at the ice cream parlor, hence skeptical reasoning will still conclude that the pronoun has this property. Alternatively, suppose $n_i \neq x$ (perhaps because it is associated deictically with another pronoun). Now, it follows that $n_i \neq u$, so whether or not n_i was at the ice cream parlor should have no effect on whether “he” was.

6 Conclusion

I have proposed a theory of the representation of anaphora, based on the assumption that if two elements cannot be proved to be different, then they can be assumed to be equal. This assumption is implemented using a standard linguistic representation (DRT) and a standard default reasoning system (Default Logic), and this requires no special mechanisms for representation or inference. Yet this conceptually simple theory appears to produce exactly the sort of inferences regarding anaphora that are intuitively desirable.

References

- [1] Baumgartner, P. and M. Kühn, *Abducing coreference by model construction*, Journal of Language and Computation **1** (2000), pp. 175–190.

- [2] Charniak, E., *Motivation analysis, abductive unification and nonmonotonic equality*, Artificial Intelligence **34** (1988), pp. 275–295.
- [3] Cohen, A., M. Kaminski and J. A. Makowsky, *Indistinguishability by default*, in: S. Artemov, H. Barringer, A. S. d’Avila Garcez, L. C. Lamb and J. Woods, editors, *We Will Show Them: Essays in Honour of Dov Gabbay*, College Publications, 2005 pp. 415–428.
- [4] Cohen, A., M. Kaminski and J. A. Makowsky, *Applying default logic to anaphora, vagueness, and uncertain reasoning* (in preparation).
- [5] Cohen, A. and J. A. Makowsky, *Two approaches to nonmonotonic equality*, Technical Report CIS-9317, Technion—Israel Institute of Technology (1993).
- [6] Gardent, C. and K. Konrad, *Definites and the proper treatment of rabbits*, in: C. Monz and M. de Rijke, editors, *Proceedings of the 1st Workshop on Inference in Computational Semantics, ICOS-1*, 1999, pp. 53–69.
- [7] Gardent, C. and K. Konrad, *Interpreting definites using model generation*, Journal of Logic, Language and Information **1** (2000), pp. 193–209.
- [8] Kaminski, M., *A comparative study of open default theories*, Artificial Intelligence **77** (1995), pp. 285–319.
- [9] Kamp, H. and U. Reyle, “From Discourse to Logic,” Kluwer Academic Publishers, Dordrecht, 1993.
- [10] Kohlhase, M., *Model generation for discourse representation theory*, in: W. Horn, editor, *Proceedings of the 14th European Conference on Artificial Intelligence*, 2000, pp. 441–445.
- [11] Konrad, K., “Model Generation for Natural Language Interpretation and Analysis,” Ph.D. thesis, University of Saarlandes, Saarbrücken (2000).
- [12] Lascarides, A. and N. Asher, *Temporal interpretation, discourse relations and common sense entailments*, Linguistics and Philosophy **16** (1993), pp. 437–493.
- [13] Lifschitz, V., *On open defaults*, in: J. Lloyd, editor, *Computational Logic: Symposium Proceedings* (1990), pp. 80–95.
- [14] Lloyd, J., “Foundation of logic programming, second extended edition,” Springer–Verlag, Berlin, 1993.
- [15] Poesio, M., *Semantic ambiguity and perceived ambiguity*, in: K. van Deemter and S. Peters, editors, *Semantic Ambiguity and Underspecification*, CSLI, Stanford, 1996 pp. 159–201.
- [16] Reiter, R., *Equality and domain closure in first order databases*, Journal of the ACM **27** (1980), pp. 235–249.
- [17] Reiter, R., *A logic for default reasoning*, Artificial Intelligence **13** (1980), pp. 81–132.

Extracting Formal Specifications from Natural Language Regulatory Documents

Nikhil Dinesh, Aravind Joshi, and Insup Lee

*Department of Computer Science, University of Pennsylvania,
Philadelphia, PA - 19104 USA
nikhild, joshi, lee@cis.upenn.edu*

Bonnie Webber

*University of Edinburgh, Edinburgh, EH8 9LW Scotland
bonnie@inf.ed.ac.uk*

Abstract

Formal verification techniques provide a way to determine whether regulatory documents are consistent and whether implementations conform to them. To apply these techniques a formal description of the regulation needs to be extracted. We present a framework, under which NLP techniques can be brought to bear, to aid a requirements engineer in extracting the formal description.

1 Introduction

Regulatory documents, which include the vast bodies of legislation, operating procedures and organizational policy, are meant to be accessible to the people affected by them. Hence, they have to be in natural language (NL). On the other hand, regulations are expected to be consistent, and the governed entities/events are expected to conform to the regulation.

For example, the Food and Drug Administration's Code of Federal Regulations (FDA CFR) governs the bloodbanks in America.¹ The bloodbanks perform safety-critical functions like the testing of blood for communicable disease agents (like HIV). It is highly desirable to determine whether (a) the CFR is consistent, and (b) a bloodbank's implementation of such a function conforms to the CFR.

* This research was supported in part by NSF CCF-0429948 and ARO 911NF-05-1-0158

¹ <http://www.gpoaccess.gov/cfr/index.html>

The problem of creating descriptions of regulation which can be checked for consistency has been explored by several authors [1,8], but the challenge of checking an implementation for conformance has not been addressed, and this is the main goal of our work. The conformance guarantees can be obtained if formal descriptions of regulation and implementations are available, and if verification techniques [4] can be applied to these descriptions. But extracting a formal description of regulation is expensive, as regulatory bases like the CFR are large (about a million words) and complex.

Formal descriptions of regulation are usually extracted by an individual who has a background in logic, e.g., [1,8]. We will call this individual *the requirements engineer*. In this paper, we describe a framework to assist a requirements engineer in extracting a formal description of regulation for use in conformance checking.

An overview of the framework, the theoretical background and the various constraints that apply is given in Section 2. This lets us determine the nature of the description that needs to be extracted from the regulation. We then turn to the question of how these descriptions might be composed. In Section 3, we attempt to map the denotations of sentences assigned by Kratzer [12] to a form that can be used for the task at hand. Some difficulties arise in this mapping, mainly because notions of *obligation* (that which is required) and *permission* (that which is allowed) are not captured in the denotations. We argue that an account of these notions is essential to the task at hand. Section 4 describes a semantic representation, and composition procedure to assist the requirements engineer in extracting the required description. By treating obligations and permissions as different dimensions of the description computed, the difficulties encountered in Section 3 are addressed.

The approach is motivated by our case study of the FDA CFR, and we use (1) and (2) as examples through the course of this paper.² (1) conveys an obligation to perform a test for HIV and Hepatitis B, and (2) conveys a permission not to test source plasma (a blood component) for Hepatitis B.

- (1) Except as specified in (2), you must test each donation of human blood or blood component, for evidence of infection due to the Human immunodeficiency virus, and the Hepatitis B virus.
- (2) You are not required to test donations of Source Plasma for evidence of infection due to the Hepatitis B virus.

2 A Framework

To determine whether an implementation (bloodbank) conforms to the regulation (CFR), we extract specifications in the Computation Tree Logic (CTL) from the CFR. Then, given a description of a bloodbank's procedure (as a finite transition system, or model) there is an efficient search procedure to

² (1) and (2) are modified versions of sentences that appear in the FDA CFR 610.40. The actual sentences are very long, and the modifications are made in the interests of space.

determine if the model conforms to the CTL specification [3]. This is known as temporal model checking [2,13]. The problem of conformance checking is thus split into three steps:

(1) Extract CTL specifications from the regulation - This is done by a requirements engineer, and our goal is to assist her. We use CTL as the specification language, because it allows for efficient model checking [3].

(2) Obtain a model of an implementation - We assume the availability of models. There are tools that aid in extracting models from software [5], and in creating models if they cannot be extracted directly [11].

(3) Apply model checking to determine if the model conforms to the CTL specification.

Formally, a model can be defined as follows:

Definition 2.1 A model M is the five-tuple (S, I, δ, π, Π) , where:

(a) S is a set of states, $I \subseteq S$ is a non-empty set of initial states,

(b) $\delta \subseteq S \times S$ is a total transition relation (that is, $\forall s \in S : [\exists t \in S : (s, t) \in \delta]$),

(c) π is a set of propositions (with power set 2^π), and

(d) $\Pi : S \rightarrow 2^\pi$ is a function from states to sets of propositions. $\Pi(s)$ for $s \in S$ can be thought of as the propositions true at s .

Figure 1(a) and 1(b) show models of two different bloodbanks. The left-most state is the initial state. Each state is labeled with $\Pi(s)$. The propositions have the following interpretation: d' is true ($d' \in \Pi(s)$) iff a donation of blood or blood component is being processed, sp' is true iff a donation of source plasma is being processed, $thiv'$ is true iff a test for HIV has been performed, and $thepb'$ is true iff a test for Hepatitis B has been performed. The use of the propositions deo (denoting *deontic accessibility*) and app_1 (denoting the application of a permission) is explained in later sections.

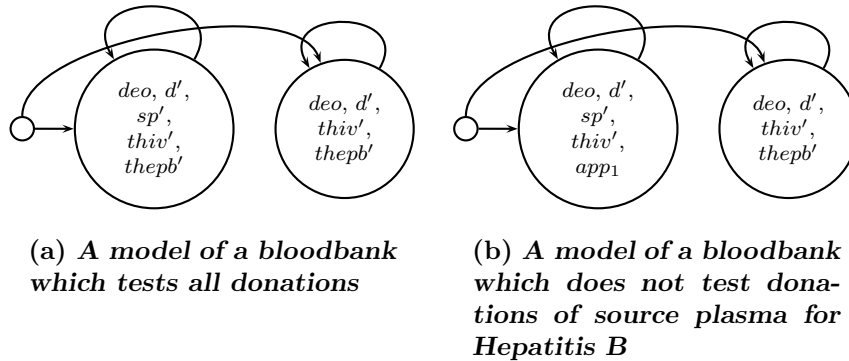


Fig. 1. Two models of bloodbanks

Definition 2.2 Given a finite set of propositions π , CTL formulas are defined inductively as follows:

(a) $p \in \pi$ is a formula,

(b) Boolean combinations and negations of formulas are formulas,

(c) if ϕ , and ψ are formulas, then $AG(\phi)$ (on all paths, globally ϕ), $AX(\phi)$ (on all paths, at the next state ϕ), and $\phi AU\psi$ (on all paths, ϕ until ψ) are formulas.

The only temporal operator in CTL that we use is AG (for reasons that we describe below), and hence rather than define the interpretation formally, we will give some examples. Let M_1 be the model in Figure 1(a), and M_2 be the model in Figure 1(b). The CTL specification $AG(deo \rightarrow (d' \rightarrow thiv'))$ holds of both models, since on all paths (from the initial state, the leftmost one in Figures 1(a), and 1(b)), globally, in all deontically accessible states deo , if a donation of blood or blood component is being processed d' , it is tested for HIV $thiv'$. Hence, we write $M_1 \models AG(deo \rightarrow (d' \rightarrow thiv'))$, and $M_2 \models AG(deo \rightarrow (d' \rightarrow thiv'))$. Also, $M_1 \models AG(deo \rightarrow (sp' \rightarrow thepb'))$. But, $M_2 \not\models AG(deo \rightarrow (sp' \rightarrow thepb'))$ (since there is a state s with $sp' \in \Pi(s)$, and $thepb' \notin \Pi(s)$).

2.1 Approaches to extracting specifications

The central problem we face is that CTL and other temporal logics that lend themselves to model checking are not expressive enough for a compositional semantic procedure to be defined for natural language. One reason is that CTL, like propositional logic, cannot express relations between entities.

There are several routes one might take to address this problem, i.e., design more expressive logics that allow for tractable model checking, focus on a subset of NL from which an automatic translation is guaranteed, or make the procedure machine-assisted. While the design of more expressive logics makes the composition of specifications easier, using them for model checking needs the creation of more expressive models (which requires more effort). As a result, there is a trade-off between amount of effort spent in obtaining models, and that in obtaining the specifications. Our decision to work with less expressive models is motivated by the extensive tool support available for creating and extracting such models [5,11]. Further, subsets of NL for which automatic translation is guaranteed, such as the one derived by Holt and Klein [10], assume (among other things) that references are resolved and hence cannot be directly applied to regulatory documents. We are thus left with the choice of making the procedure machine-assisted.

There have been two kinds of machine-assisted approaches to extracting temporal logic specifications: (a) composing the semantics in a general semantic framework which is then mapped to temporal logic [7], and (b) attempting to compose the semantics in the temporal logic directly [6]. In the latter approach, a human specifies denotations for a portion of the sentence, and the rest of the composition happens automatically. We attempt to compose the semantics in a temporal logic directly like [6], as it lends itself to defining semantic representations with which a requirements engineer can interact in well-defined ways.

2.2 Constraints on the CTL specifications

We apply two constraints to the CTL specifications:

- (i) The specifications extracted should hold of all and only the valid mod-

els. There may be several implementations that aim to conform to a single base of regulation. Given (1) and (2), the models in Figures 1(a) and 1(b) are both valid. This is an important difference from the NL sentences considered in previous approaches, which were elicited from appropriate users by presenting them with a single model. For example, Holt and Klein [10] obtained specifications by asking users to describe a particular timing diagram.

(ii) To account for the variation between models, all temporal information about the governed entities/events is modelled through propositions. The only use of the temporal operators in CTL is to obtain a quantification over paths and states. A mapping will need to be performed so that the propositions used in the specifications can be evaluated at a states in different models, and the critical assumption is that this mapping will be very easy to specify.

3 From Sets of Worlds to Sets of Models

Several approaches in formal semantics take sentences to denote sets of worlds. For normative statements, we assume (following Kratzer [12]) that worlds are connected by an accessibility relation. Consider (1) in Section 1 which among other things *requires a test for Hepatitis B if no exceptions apply*. A denotation of this requirement is given in (3), and is the set of worlds w_0 , such that for every deontically accessible world w , for every entity x such that x is a donation in that world $d'(x, w)$, if no exception holds of that donation $\neg e'(x, w)$, a test for Hepatitis B is carried out for that donation $thepb'(x, w)$. We will assume that negation has the highest precedence. Therefore $\neg a \rightarrow b \equiv (\neg a) \rightarrow b$, and brackets are used to resolve other ambiguities.

$$(3) \quad \lambda w_0. \forall w : (w \in deo(w_0) \rightarrow (\forall x : (d'(x, w) \rightarrow (\neg e'(x, w) \rightarrow thepb'(x, w))))))$$

A difference between worlds in Kratzer's denotations and states in a model is that: in a state there is no notion of entities and relations between them. All that is available at a state s is the set of propositions which are true at that state $\Pi(s)$. To map (3) to a form that is useful for checking conformance, we need two assumptions.

First, we assume that regulation denotes the *set of models* that conform to it. Intuitively speaking, w_0 in (3) can be thought of as a model in its entirety, and $w \in deo(w_0)$ correspond to special states in the model. A universal quantification over accessible worlds can be replaced with the CTL *AG* operator. We then obtain the denotation in (4), read as : *on every path in M, if a state is deontically accessible, for each donation x at that state, if no exception holds, a test is carried out*. In a model, only special states (like when the bloodbank has finished processing all the donations it has received) need to conform to the regulation, and *deo* can be thought of as marking those states.

$$(4) \quad \lambda M. M \models AG(deo \rightarrow (\forall x : (d'(x) \rightarrow (\neg e'(x) \rightarrow thepb'(x))))))$$

(4) is still not in CTL because of the universal quantification over entities x at a state. The universal quantifier can be eliminated by assuming a

serial processing model. This has the effect that at the deontically accessible states, exactly one donation is under consideration (e.g. the models in Figures 1(a) and 1(b)). In the sections of the CFR that we examined, a universal quantification over entities is absolutely essential when these entities correspond to inputs of an implementation. This assumption lets us tie the inputs to states, and use the quantification over states to achieve the quantification over entities. Thus (4) can be reduced to (5).

$$(5) \quad \lambda M. M \models AG(deo \rightarrow (d' \rightarrow (\neg e' \rightarrow thepb')))$$

A problem that is encountered in taking this approach is that there is no distinction between *obligations*, and *permissions* (both of which stem from the Hohfeldian legal conceptions of right, duty, privilege, and no right [9]). While this did not cause a problem for the obligation in (1), if one were to follow the same procedure for the permission in (2), we would get the denotation in (6).

$$(6) \quad \lambda M. M \models \neg(AG(deo \rightarrow (sp' \rightarrow thepb')))$$

A model satisfies (6) only if *there is some path in which there is a state that is deontically accessible, and if a donation of source plasma is being processed it is not tested*. This is too strong a requirement, because an organization may choose not to do what it is permitted to do. The model in Figure 1(a) is a valid model, which would be declared invalid if (6) were required of it.

Another problem is that it is not clear how one would use (6) in interpreting the exemption e' in (5). A reasonable candidate is $e' \equiv deo \rightarrow (sp' \rightarrow \neg thepb')$. But this is not the exemption because it is true in *every deontically accessible state in which a donation of source plasma is not being processed*. Consider a state s at which $sp' = false$ ($sp' \notin \Pi(s)$). At s , $e' \equiv (deo \rightarrow (false \rightarrow \neg thepb')) \equiv (deo \rightarrow true) \equiv true$. The specification in (5), at s is: $AG(deo \rightarrow (\neg e \rightarrow \neg thepb')) \equiv AG(deo \rightarrow (\neg true \rightarrow \neg thepb')) \equiv AG(deo \rightarrow true) \equiv AG(true) \equiv true$. Therefore, a model that doesn't test any donation for Hepatitis B would conform to (5). We now turn to the task of addressing these problems by revising how the specifications are composed.

4 Extracting the specifications

To aid the requirements engineer in extracting the specifications, the idea is to present her with intermediate semantic representations of the sentence with which she interacts. The intermediate representations that we use fall into the category of *abstract syntax trees (ASTs)*. ASTs are generally used as intermediate representations in compiling code in a high-level programming language to machine dependant code. The internal nodes in ASTs are *operators* (predicates/meta-predicates), the subtrees they dominate are *operands* (arguments), and leaf nodes correspond to variables or constants (the requirements engineer specifies the denotation of the leaves). An AST encodes the resolution of scope ambiguities, i.e., if p_1 dominates p_2 in the AST, then p_1 outscopes p_2 .

Section 4.1 describes some phenomena in natural language that can be used in the construction of the ASTs, and how these ASTs can be interpreted. In Section 4.2, we describe how the ASTs and their interpretation for (1) and (2) (in Figures 3 and 4) address the problems described in Section 3.³

4.1 Abstract Syntax Trees (ASTs) and their interpretation

To capture the distinction between obligations and permissions, the denotation of each node N in an AST is given by the 3-tuple: $[[N]] = \begin{pmatrix} \phi_N \\ \mathcal{O}_N \\ \mathcal{P}_N \end{pmatrix}$, where \mathcal{O}_N is a set of propositional logic formulas which correspond to *the obligations that have been satisfied*, and \mathcal{P}_N is a set of propositional logic formulas that correspond to *the permissions that have been taken*, and ϕ_N is a propositional logic formula which can be thought of as indicating whether N is true at a state. The set of obligations \mathcal{O} obtained from the policy base is the union of the obligations obtained at the root of the AST for each sentence. The denotation of the policy base is then given by: $\lambda M. M \models AG \left(deo \rightarrow \bigwedge_{\phi \in \mathcal{O}} \phi \right)$. We now identify various linguistic constructions that can be used to obtain ASTs.

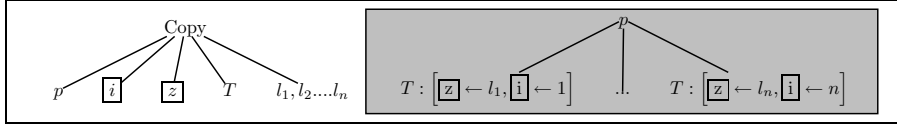


Fig. 2. Semantics of the *Copy* meta-predicate

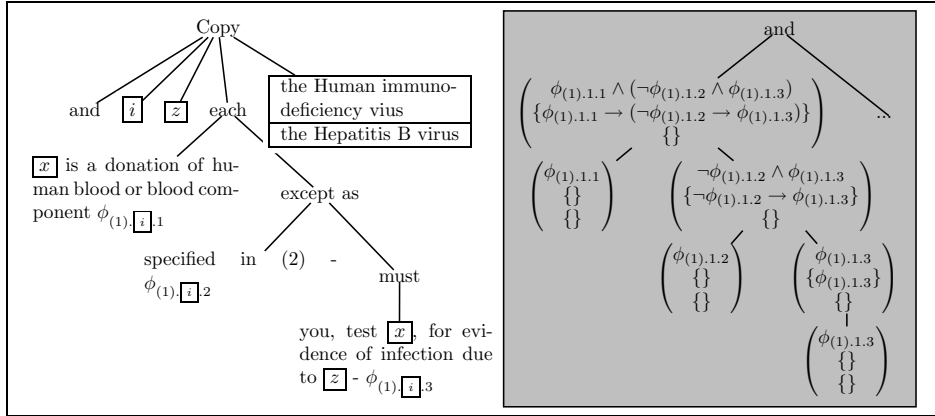


Fig. 3. AST and its interpretation for (1)

Distributive readings and the *Copy* meta-predicate: (1) is ambiguous between a *collective reading* (where there is a single test for both the

³ We assume that obligation and permission denoting categories, e.g. *must*, do not occur in contexts like antecedent clauses of subordinating conjunctions (like *if*), and restrictors of determiners. Handling these cases requires an extension to CTL which is beyond the scope of this paper.

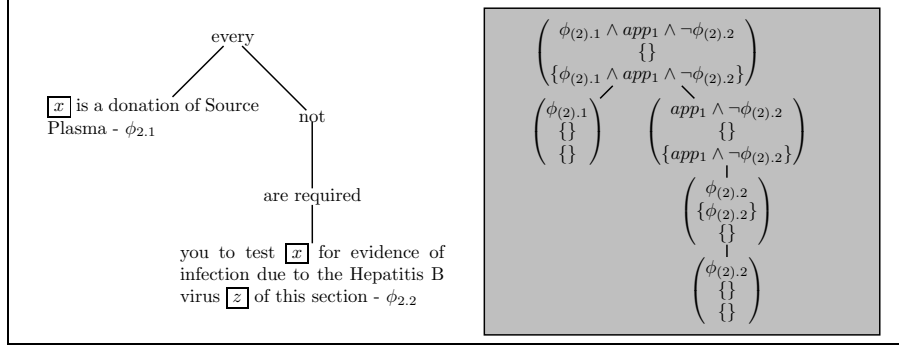


Fig. 4. AST and its interpretation for (2)

diseases), and a *distributive reading* (where there are separate tests for each disease). However, (2) gives an exemption to a test for one of the diseases, and this suggests that a distributive reading may be more appropriate in the specifications extracted, and that the distributivity has scope over the exception. Hence *Copy* dominates *except* in Figure 3.

The interpretation of the *Copy* meta-predicate is given in Figure 2. It is called a meta-predicate because it is a function from an AST to another AST, by simple variable substitution. For the AST for (1) shown in Figure 3, this results in an AST rooted with *and* with subtrees corresponding to each of the tests. The interpretation of *and* in this context is given by:

$$and \left(\begin{array}{c} \phi_A^1 \\ \mathcal{O}_A^1 \\ \mathcal{P}_A^1 \end{array} \right) \dots \left(\begin{array}{c} \phi_A^n \\ \mathcal{O}_A^n \\ \mathcal{P}_A^n \end{array} \right) = \left(\begin{array}{c} \bigwedge_{i=1}^n \phi_A^i \\ \bigcup_{i=1}^n \mathcal{O}_A^i \\ \bigcup_{i=1}^n \mathcal{P}_A^i \end{array} \right)$$

The RHS of the equation corresponds to the denotation of the node labeled *and* in the AST (shaded in gray in Figure 3).

Universally Quantified NPs corresponding to inputs: As mentioned in Section 3, the universal quantification over inputs (donations) is achieved by associating states with unique inputs. The interpretation of the determiner *each* is designed with idea that the obligations will be evaluated at each state.

$$each \left(\begin{array}{c} \phi_A \\ \mathcal{O} \\ \mathcal{P} \end{array} \right) \left(\begin{array}{c} \phi_B \\ \mathcal{O}_B \\ \mathcal{P}_B \end{array} \right) = \left(\begin{array}{c} \phi_A \wedge \phi_B \\ \{\phi_A \rightarrow \phi_{\mathcal{O}.j}^B | \phi_{\mathcal{O}.j}^B \in \mathcal{O}_B\} \\ \{\phi_A \wedge \phi_{\mathcal{P}.j}^B | \phi_{\mathcal{P}.j}^B \in \mathcal{P}_B\} \end{array} \right)$$

The interpretation of the determiner *no* is similar to that of *each/every*, except that a negation needs to be applied to the nuclear scope. We discuss the interpretation of negation in what follows.

Conditional and Exceptive constructions: There are several predicates that denote conditions and exceptions. For example, the subordinating conjunctions *if*, *unless*, and *except as*, coordinating conjunctions like *except that* or *but*. The interpretation of *if* is the same as that for *every*. The interpretation of predicates like *except as*, and *unless* are similar, the only difference being that $\neg \phi_A$ is used instead of ϕ_A in the RHS.

Modals and Negation: The semantics of modals and negation are given below:

$$must \left(\begin{array}{c} \phi_A \\ \mathcal{O} \\ \mathcal{P} \end{array} \right) = \left(\begin{array}{c} \phi_A \\ \{\phi_A\} \\ \mathcal{O} \end{array} \right) \quad may \left(\begin{array}{c} \phi_A \\ \mathcal{O} \\ \mathcal{P} \end{array} \right) = \left(\begin{array}{c} app_i \wedge \phi_A \\ \mathcal{O} \\ \{app_i \wedge \phi_A\} \end{array} \right)$$

$$\text{not} \left(\begin{array}{c} \phi_A \\ \mathcal{O}_A \\ \mathcal{P}_A \end{array} \right) = \left(\begin{array}{c} \phi'_A \\ \{-\phi_{\mathcal{P},j}^A | \phi_{\mathcal{P},j}^A \in \mathcal{P}_A\} \\ \{app_j \wedge \neg\phi_{\mathcal{O},j}^A | \phi_{\mathcal{O},j}^A \in \mathcal{O}_A\} \end{array} \right), \text{ where } \phi'_A = \begin{cases} app_j \wedge \neg\phi_A & \phi_A \equiv \phi_{\mathcal{O},j}^A \in \mathcal{O}_A \\ \neg\phi_A & \text{otherwise} \end{cases}$$

$\text{must}(A)$ results in the interpretation that ϕ_A is an obligation. $\text{may}(A)$ results in the interpretation that $app_i \wedge \phi_A$ is a permission, where app_i is a variable introduced which the implementation must set to true when the permission is applied (we discuss its use in Section 4.2). And intuitively, the interpretation of negation captures the idea that $\text{may}(\neg A) \equiv \text{not}(\text{must}(A))$.

4.2 Discussion

There are two obligations obtained at the root of the AST for (1): $\phi_{(1).1.1} \rightarrow (\neg\phi_{(1).1.2} \rightarrow \phi_{(1).1.3}) \equiv d' \rightarrow (\neg e'_1 \rightarrow \text{thiv}')$ and $\phi_{(1).2.1} \rightarrow (\neg\phi_{(1).2.2} \rightarrow \phi_{(1).2.3}) \equiv d' \rightarrow (\neg e'_2 \rightarrow \text{thepb}')$, where d' is true iff the donation is one of blood or blood component, e'_1 and e'_2 are the exceptions to the required test for each disease, and thiv' and thepb' are true iff tests for HIV and Hepatitis B respectively have been performed. The computation of the second obligation is not shown in Figure 3, and is obtained from the second child of *and* (in the AST shaded in gray). Note that the individual propositions like d' need to be specified by the requirements engineer at the leaf nodes of the AST.

Figure 4 shows the AST and its interpretation for (2). The permission obtained at the root node is: $\phi_{(2).1} \wedge app_1 \wedge \neg\phi_{(2).2} \equiv sp' \wedge app_1 \wedge \neg\text{thepb}'$ where sp' is true iff a donation of source plasma is being processed, and thepb' is true iff a test for the Hepatitis B virus has been carried out.

The use of the app_1 proposition is as follows. It is possible for the regulation to cancel the permission given in (2), but there may be several cases in which permission not to test a donation of source plasma for Hepatitis B is given. Suppose the case under consideration is one where the permission in (2) is cancelled, but the organization doesn't test a donation of source plasma for Hepatitis B because a different permission can be applied. Since the permission being applied sets thepb' to false, and sp' is true, the only way for the implementation to indicate that the permission in (2) is not being applied is by setting app_1 to false. Setting $e'_1 \equiv \text{false}$, and $e'_2 \equiv sp' \wedge app_1 \wedge \neg\text{thepb}'$:

$$\phi_{\mathcal{O},1} \equiv d' \rightarrow (\neg\text{false} \rightarrow \text{thiv}'), \text{ and } \phi_{\mathcal{O},2} \equiv d' \rightarrow (\neg(sp' \wedge app_1 \wedge \neg\text{thepb}') \rightarrow \text{thepb}')$$

Considering just these obligations, the denotation of the regulatory document would be: $\lambda M. M \models AG(\text{deo} \rightarrow (\phi_{\mathcal{O},1} \wedge \phi_{\mathcal{O},2}))$. Therefore, a bloodbank could decide not to test a donation of source plasma for Hepatitis B, but they would always have to test a donation for HIV.

5 Conclusions and Future Work

We have described a framework to assist a requirements engineer in extracting CTL specifications from regulatory documents. An account of obligations and permissions turns out to be essential in composing the specifications. The composition procedure (defined in Section 4) was applied to a large part of

the FDA CFR 610.40. While it does seem to scale well, providing tool support to extract and interact with the ASTs is vital. To this end, we plan to conduct a small scale annotation of ASTs which will let us determine the accuracy with which these representations can be computed. On the user interface side, we are working on ways of presenting the ASTs to the requirements engineer.

References

- [1] Breuker, J. and N. den Haan, *Separating world and regulation knowledge: where is the logic?*, in: M. Sergot, editor, *Proceedings of the third international conference on AI and Law* (1991), pp. 41–51.
- [2] Clarke, E. M. and E. A. Emerson, *Synthesis of synchronization skeletons for branching time temporal logic*, in: *Logic of Programs: Workshop*, 1981.
- [3] Clarke, E. M., E. A. Emerson and A. P. Sistla, *Automatic verification of finite-state concurrent systems using temporal logic specifications*, *ACM Transactions on Programming Languages and Systems* **8** (1986), pp. 244–263.
- [4] Clarke, E. M. and J. M. Wing, *Formal methods: State of the art and future directions*, *ACM Computing Surveys* **28** (1996), pp. 626–643.
- [5] Corbett, J. C., M. B. Dwyer, J. Hatcliff, S. Laubach, C. S. Pasareanu, Robby and H. Zheng, *Bandera: Extracting finite-state models from java source code*, in: *Proceedings of the International Conference on Software Engineering (ICSE)*, 2000.
- [6] Fantechi, A., S. Gnesi, G. Ristori, M. Carenini, M. Marino and M. Moreschini, *Assisting requirements formalization by means of natural language translation*, *Formal Methods in System Design* **4** (1994), pp. 243–263.
- [7] Fuchs, N. and R. Schwitter, *Attempto controlled english (ace)*, in: *First International Workshop on Controlled Language Applications*, 1996.
- [8] Glasse, E., T. V. Engers and A. Jacobs, *Power: An integrated method for legislation and regulations from their design to their use in e-government services and law enforcement*, in: M.-F. Moens, editor, *Digitale Wetgeving, Digital Legislation*, Die Keure Brugge, 2003 pp. 175–204, iISBN 90 5958 039 7.
- [9] Hohfeld, W. N., *Fundamental legal conceptions as applied in judicial reasoning*, *Yale Law Journal* **23** (1913), pp. 16–59.
- [10] Holt, A. and E. Klein, *A semantically-derived subset of English for hardware verification*, in: *37th Annual Meeting of the ACL*, 1999.
- [11] Holzmann, G., *The Spin model checker*, *IEEE Trans. on Software Engineering* **23** (1997), pp. 279–295.
- [12] Kratzer, A., *The notational category of modality*, in: H.-J. Eikmeyer and H. Rieser, editors, *Words, Worlds, and Contexts. New approaches to Word Semantics*, deGruyter, Berlin, 1981 .
- [13] Queille, J. P. and J. Sifakis, *Specification and verification of concurrent systems in CAESAR*, in: *Proceeding of the Fifth ISP*, 1981.

How to change a person's mind: Understanding the difference between the effects and consequences of speech acts

Debora Field and Allan Ramsay

*Computer Science, Univ. of Liverpool, L69 3BX, UK
Informatics, Univ. of Manchester, PO Box 88, M60 1QD, UK
debora@csc.liv.ac.uk, allan.ramsay@manchester.ac.uk*

Abstract

This paper discusses a planner of the semantics of utterances, whose essential design is an epistemic theorem prover. The planner was designed for the purpose of planning communicative actions, whose effects are famously unknowable and unobservable by the doer/speaker, and depend on the beliefs of and inferences made by the recipient/hearer. The fully implemented model can achieve goals that do not match action effects, but that are rather entailed by them, which it does by reasoning about how to act: state-space planning is interwoven with theorem proving in such a way that a theorem prover uses the effects of actions as hypotheses. The planner is able to model problematic conversational situations, including felicitous and infelicitous instances of bluffing, lying, sarcasm, and stating the obvious.¹

1 Introduction

The motivation for this research was the problem of planning the semantics of communicative actions: given that I want you to believe P , how do I choose what meaning to express to you? The well-documented, considerable difficulties involved in this problem include this: a key player in the ensuing evolution of the post-utterance environment is the **hearer** of the utterance.

First, consider an imaginary robot Rob, designed not for communication, but for making tea. Whenever he is in use, Rob's top-level goal is to attain a state in which there is a certain configuration of cups, saucers, hot tea, cold milk, etc. Rob's plans for making tea are made on the strong assumption that at plan execution time, the cups (and other items) will have no desires and opinions of their own concerning which positions they should take up—Rob expects to be the author of the effects of his actions.²

¹ Initially funded by the EPSRC. Recent partial funding under EU-grant FP6/IST No. 507019 (*PIPS*: Personalised Information Platform for Health and Life Services).

² notwithstanding impeding concurrent events, sensor failures, motor failures, *etc.*

In contrast, consider the human John, designed for doing all sorts of things besides making tea, including communicating messages to other humans. Imagine John’s current goal is to get human Sally to believe the proposition *John is kind*. In some respects, John has a harder problem than Rob. Unlike Rob, John has no direct access to the environment he wishes to affect—he cannot simply implant *John is kind* into Sally’s belief state. John knows that Sally has desires and opinions of her own, and that he will have to plan something that he considers might well lead Sally to infer *John is kind*. This means that when John is planning his action—whether to give her some chocolate, pay her a compliment, tell her he is kind, lend her his credit card—he has to consider the many different messages Sally might infer from the one thing John chooses to say or do. Unfortunately, there is no STRIPS operator [13] John can choose that will have his desired effect; he has to plan an action that he expects will **entail** the state he desires.

We considered ‘reasoning-centred’ planning of actions that entailed goals to be an approach that would enable this difficult predicament to be managed, and implemented a model accordingly. Our planner is, in essence, an epistemic theorem prover that hypothesises desirable actions, and is able to plan to achieve goals that do not match action effects, but that are entailed by the final state. Like John, the planner can have particular communicative goals in mind, and knows that the execution of any single plan could have a myriad different effects on *H*’s belief state, depending on what *H* chooses to infer.

1.1 *Bucking the trend*

The main focus of current research in AI planning is on how to reduce the search space required for making plans, and thus, for example, to get Rob the tea-making robot to be able to make his plans fast enough to be of practical use in the real world. Many planners use heuristics, either to constrain the generation of a search space, or to prune and guide the search through the state space for a solution, or both [4,5,18,25]. All such planners succeed by relying on the static effects of actions—on the fact that you can tell by inspection what the effects of an action will be in any situation—which limits their scope in a particular way [4, p. 299]:

“...if one of the actions allows the planner to dig a hole of an arbitrary integral depth, then there are potentially infinitely many objects that can be created... The effect of this action cannot be determined statically ...”

The class of problems that these planners do not attempt to solve—the ability to plan actions whose effects are not determined statically—was the class that particularly interested us.

2 Planning the semantics of utterances

With our attention firmly fixed on the myriad different effects a single communicative act can have on a hearer’s belief state, we concentrated on a (logically)

very simple utterance:

“There’s a/an [some object]!”

We devised situations culminating in this utterance which illustrate **sarcasm**, **stating the obvious**, **bluffing**, and **lying**, and developed a planner which could use these tactics. Here is a much-shortened example of a scenario from the model, which leads to the planning of an instance of sarcasm:^{3 4}

<i>Initial state</i>	John has been bird-watching with Sally for hours, and so far, they have only seen pigeons. John thinks Sally is feeling bored and fed up. John has some chocolate in his bag. John thinks Sally likes chocolate. John knows lots of rules about how conversation works, and what one can expect a hearer to infer under given conditions.
<i>Goal condition</i>	John wants to cheer Sally up.
<i>Solutions</i>	John is just thinking about getting out some chocolate to give her, when yet another pigeon lands in a nearby tree. John sees an opportunity to make Sally laugh by means of a bit of sarcasm, and so plans to say to her, “There’s an albatross!”

John plans (the semantics of) his utterance, expecting that the utterance will have particular ‘effects’ on Sally’s belief state; if John were to perform the utterance, he would not be certain that it had achieved his intention, but he would expect that it probably had. Whether John’s intention would be achieved by this utterance depends on Sally having the ‘right’ set of beliefs (the ones John thinks she has) and making the ‘right’ inferences (the ones John expects her to make).

For example, if John’s utterance “There’s an albatross!” is to be felicitous, the following must happen. Sally must first believe that John has said something that Sally thinks John and Sally mutually believe is false. From this, she must infer that John has flouted a conversational maxim, and consequently that John has attempted to implicate a meaning which is not expressed by the semantics of “There’s an albatross!”. Sally must then infer that the implicature John intends is of humour. Whether or not any of this happens depends on Sally’s beliefs, which John cannot observe, but about which he has beliefs. The formal version of this example contains all the necessary information about the beliefs of John and Sally in this situation for the planner: (i) to be able to plan John’s utterance; and (ii) to additionally deduce whether John’s utterance would be felicitous or infelicitous, if he performed it.

³ The example is an English paraphrase of a task, written in the model in Prolog code.

⁴ An albatross (*Diomedea exulans*) is a huge sea-faring bird, rarely seen from the land.

2.1 Linguistic motivations

Our approach to planning the semantics of utterances was to build on seminal work in speech acts [3,28] and pragmatics [29,15,21]. In contrast to the ‘speech acts with STRIPS’ approach [6,11,1,2], which is fraught with well-documented difficulties [9,16,26,7,27], we aimed to develop a small set of linguistic acts that were unambiguously identifiable purely by surface linguistic form (after [7]), including ‘declare’, ‘request’, and perhaps others—a set of acts with negligible effects (after [27]), and minimal preconditions. We in fact developed a single linguistic act for all contexts.

2.2 Planner design

The planner is essentially an epistemic theorem prover which employs some planning search. The development process we undertook is helpful in understanding the planner’s design:

- A state-space search was implemented that searches backwards in hypothetical time from the goal via STRIPS operators (based on foundational work in classical planning [23,24,14,22,13]);
- A theorem prover for FOL was implemented that constructively proves conjunctions, disjunctions, implications, and negations, and employs modus ponens and unit resolution;
- State-space search and theorem proving were interwoven in such a way that:
 - not only can disjunctions, implications and negations be **proved** true, they can also be **achieved**;
 - not only can a goal Q be **proved** true by proving $(P \Rightarrow Q) \wedge P$, but Q can also be **achieved** by proving $P \Rightarrow Q$ and achieving P ;
 - a goal can be achieved by reasoning with **recursive** domain-specific rules—thus the planner is able to plan to ‘dig holes of arbitrary depths’.
- The theorem prover was transformed into an epistemic theorem prover by incorporating a theory of knowledge and belief suitable for human reasoning about action, so agents make plans according to their beliefs about the world, including their beliefs about others’ beliefs.

A goal is proved by assuming the effect of some action is true, on the grounds that the goal would be true in the situation that resulted from performing that action. Hence, a set of actions is computed that might be useful for achieving a goal by carrying out hypothetical proofs, where the hypotheses are the actions whose effects have been exploited.

Here is a simple, non-dialogue example to aid explanation. Consider the achievement of the goal $above(e,f)$ and $on(e,d)$, where $above$ is the transitive closure of on . First, it is not possible to judge whether the first goal $above(e,f)$ is true by inspecting the current state (which contains $on(-,-)$ facts but no $above(-,-)$ facts), so reasoning is carried out to find out whether it is false. Secondly, in order to achieve $above(e,f)$, something different from an action with an $above(-,-)$ expression in its add list is needed. Placing e onto f , for

example, will make $above(e,f)$ proveable, but it will also make the achievement of $on(e,d)$ impossible. By reasoning with rules that describe the meaning of $above$ as the transitive closure of on , the planner hypothesises that $on(d,f)$ might enable the proof of $above(e,f)$ to be completed, and also knows that $on(d,f)$ is an effect of action $stack(d,f)$. A proof of the preconditions of action $stack(d,f)$ is carried out, and the process continues (with backtracking), until a solution is found.

The preference for a backwards planning search was motivated by a defining quality of the communication problem, as epitomised by utterance planning: there are too many applicable actions to make a forwards search feasible. People generally have the physical and mental capabilities to say whatever they want at any moment. This means that the answer to the question ‘What **can** I say in the current state?’ is something like ‘**Anything**, I just have to decide what I want to say’. A backwards search is far more suitable than a forwards search under conditions like these.

With this ‘reasoning-centred’ design, the planner is able to plan an utterance to achieve a goal, ‘knowing’ that the utterance may or may not achieve the desired effects on H , and that the same utterance can have many different effects, depending on H ’s belief state.

3 Modelling problematic conversations

In the model, utterances are planned according to Grice’s Cooperative Principle [15]. Here is an extract from the CP (*ibid* p. 308):

“[Quantity]

- (i) Make your contribution as informative as is required (for the current purposes of the exchange).
- (ii) Do not make your contribution more informative than is required. . .

[Quality]

- (i) Do not say what you believe to be false.
- (ii) Do not say that for which you lack adequate evidence.”

Grice’s maxims prescribe a standard for speaker behaviour which S can blatantly contravene (‘flout’), thus signalling to H that there is an implicature to be recovered. For instance, in our ‘sarcasm’ scenario, John’s utterance is planned using the following maxim, derived from Grice’s first Quality maxim.⁵ The first line means, ‘If S addresses H by putting Q into the conversational minutes’:

- (1) `minute([S], [H], Q)`
`and believes(S, believes(H, mutuallybelieve(([H], S]), not(Q)))`
`==> believes(S, believes(H, griceuncoop(S, [H], Q))`

⁵ The model embodies a ‘deduction’ model of belief [19], rather than a ‘possible worlds’ model [17,20]. Thus agents are not required to draw **all** logically possible inferences, and are therefore not required to infer an infinite number of propositions from a mutual belief.

Using this maxim, John reasons that he can get Sally to realise he is flouting a maxim in order to generate an implicature (that he is being ‘Grice uncooperative with respect to Q ’). But what is the nature of the implicature? This is dealt with by two additional rules: (2), which describes what John thinks Sally believes about the meaning of this kind of maxim-flouting; and (3), a ‘general knowledge’ rule:

- ```
(2) believes(john,
 believes(sally,
 (griceuncoop(PERSON2, _PERSON1, Q)
 and mutuallybelieve([sally, john]), not(Q)))
 ==> funny(PERSON2, re(Q)))

(3) believes(john,
 believes(sally,
 (funny(PERSON2, re(Q))
 ==> happy(sally))))
```

With these three rules, John can reason that saying something he thinks he and Sally mutually disbelieve will make her laugh, and thus cheer her up, thus achieving his goal. Here is a second maxim from the model, also derived from Grice’s CP:

- ```
(4) minute([S], [H], Q)
      and believes(S, believes(H, mutuallybelieve([H, S]), Q))
      ==> believes(S, believes(H, griceuncoop(S, [H], Q)))
```

Using this maxim, and some additional rules, John can plan to flout Quantity maxim 2, and generate an implicature by ‘stating the obvious’.

3.1 Modelling deception

Grice’s CP seems an excellent formalism for planning and understanding utterances, so long as everyone is committed to obeying it. We know, however, that people **violate** the CP maxims— S contravenes maxims without wanting H to know. For example, lying violates Quality maxim (1), bluffing violates Quality maxim (2), and being economical with the truth violates Quantity maxim (1). However, there is nothing in Grice’s maxims to help H deal with the possibility that S may be trying to deceive her. Our solution is to give S and H some further maxims which legislate for the fact that speakers do not necessarily always adhere to the CP, and which enable S to plan to deceive, and H to detect intended deceptions.

3.1.1 Hearer violation maxims

Given that H admits the possibility that S might be trying to deceive her with his utterance, we consider that there are three strong predictors of how H ’s belief state will change in response to S ’s utterance of the proposition P :

- ```
(5) i What is H ’s view of the proposition P ?
 ii What is H ’s view concerning the goodwill of S ?
 iii What is H ’s view of the reliability of S ’s testimony?
```



Consider, for example, an attempt at bluffing:<sup>6</sup>

|                       |                                                                                                                                                                                                                                                                                                                                                                                       |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>Initial state</i>  | John has gone bird-watching with Sally. John is wearing a warm coat, and he thinks that Sally looks cold. John thinks Sally will be impressed by a chivalrous gesture. John thinks Sally is new to bird-watching, and that she is keen to learn about birds. John knows lots of rules about how conversation works, and what one can expect a hearer to infer under given conditions. |
| <i>Goal condition</i> | John wants Sally to be impressed by him.                                                                                                                                                                                                                                                                                                                                              |
| <i>Solutions</i>      | John is just thinking of offering Sally his coat to wear, when a huge bird lands in a nearby tree. John isn't quite sure what species the bird is, nevertheless, he decides to try and impress Sally with his bird expertise, and plans to say to her,<br><br><p style="text-align: center;"><b>“There’s a dodo!”</b></p>                                                             |

Let us imagine that Sally’s answers to three above questions are as follows. Before John performed his utterance:

- (6) i Sally believed that the proposition  $P$  (“There’s a dodo!”) was false (because she knew the bird was a buzzard).  
 Additionally, she did not believe that John thought that they mutually believed  $P$  was false.  
 ii She believed that John was well-disposed towards her.  
 iii She didn’t know whether John was a reliable source of information or not.

After John has said “There’s a dodo!”, Sally derives the following new set of beliefs from the above set:

- (7) i’ Sally still believes that the proposition  $P$  (“There’s a dodo!”) is false.  
 She **now** believes that John thinks that they mutually believe  $P$  is true.  
 ii’ She still believes that John is well-disposed towards her.  
 iii’ She **now** believes John is an unreliable source of information.

The mapping of belief set (6) into belief set (7) is determined in the model by a ‘hearer violation (HV) maxim’. We call this maxim the ‘infelicitous bluff’ HV maxim. We have so far implemented eight HV maxims, however, there is clearly scope for many more permutations of all the different possible answers to (6). There are obvious additional refinements that should be made, for example, people do not normally consider others to be reliable sources of information on **all** subjects.

### 3.1.2 Speaker violation maxims

If  $S$  is to succeed in his attempt to deceive  $H$ , he will have to take into account how  $H$  is going to try and detect his deception. To represent this in the model,  $S$  has his own ‘speaker violation (SV) maxims’, which concern the same issues as the HV maxims, but from the other side of the table, as it were. What  $S$  plans to say will depend on which answer he selects from each of these four categories:

<sup>6</sup> A dodo is a large flightless bird that is famously extinct.

- (8) i **What is  $S$ 's view of  $H$ 's view of various different propositions?**
- ii **What is  $S$ 's own view of the same propositions?**
- iii **What is  $S$ 's view of  $H$ 's view of the goodwill of  $S$ ?**
- iv **What is  $S$ 's view of  $H$ 's view of the reliability of  $S$  as a source?**

Here is an example of an SV maxim from the model:

```
(9) minute([S], [H], Q)
 and believes(S, believes(H, reliable(S)))
 and believes(S, believes(H, well_disposed_towards(S, [H])))
 and believes(S, believes(H, Q or not(Q)))
==> believes(S, believes(H, gricecoop(S, [H], Q)))
```

Using this maxim, John can reason that Sally will believe he is being Grice-cooperative, which means Sally will believe that what he is saying is true, even if John does not believe it himself. Thus John is able to plan to lie to Sally by using tactics he hopes will prevent Sally from detecting his attempt to deceive.

## 4 Epistemic theorem prover

The planner's theorem prover embodies a constructive/intuitionist logic and it proves theorems by natural deduction, chosen in preference to classical logic and its inferencing methods. The way humans do every-day inferencing is, we consider, quite different from the way inferencing is handled under classical logic. In classical logic, for example, and using our general knowledge, we judge the following formulae to be true:

- (10) Earth has one moon  $\Rightarrow$  Elvis is dead
- (11) Earth has two moons  $\Rightarrow$  Elvis is alive
- (12) Earth has two moons  $\Rightarrow$  Elvis is dead

(10) is true simply because antecedent and consequent are both true formulae. We find this truth odd, however, because of the absence of any discernible relationship between antecedent and consequent. (11) and (12) are true simply because the antecedent is false, which seems very counter-intuitive. Even more peculiarly, the following formula is provable in classical logic in all circumstances:

- (13) (Earth has one moon  $\Rightarrow$  Elvis is dead) or  
(Elvis is dead  $\Rightarrow$  Earth has one moon)

but it feels very uncomfortable to say that it must be the case that one of these implies the other.

In order to avoid having to admit proofs like this, and to be able to do reasoning in a more human-like way, we opted for constructive logic and natural deduction. In order to prove  $P \Rightarrow Q$  by natural deduction, one must show that  $Q$  is true **when**  $P$  is true; if  $P$  is not true, constructive logic does not infer  $P \Rightarrow Q$ . This treatment of implication hints at a relationship between  $P$  and  $Q$  which is absent from material implication.

#### 4.1 Constructive logic and belief

Taking a constructive view allows us to simplify our reasoning about when the hearer believes something of the form  $P \Rightarrow Q$ , and hence (because of the constructive interpretation of  $\neg P$  as  $P \Rightarrow \perp$ ) about whether she believes  $\neg P$ . We will assume that  $believes(H, P)$  means that  $H$  **could** infer  $P$  on the basis of her belief set, not that she already does believe  $P$ , and we will examine the relationship between  $believes(H, P \Rightarrow Q)$  and  $believes(H, P) \Rightarrow believes(H, Q)$ .

Consider first  $believes(H, P) \Rightarrow believes(H, Q)$ . Under what circumstances could you convince yourself that this held?

For a constructive proof, you would have to assume that  $believes(H, P)$  held, and try to prove  $believes(H, Q)$ . So you would say to yourself ‘Suppose I were  $H$ , and I believed  $P$ . Would I believe  $Q$ ?’ The obvious way to answer this would be to try to prove  $Q$ , using what you believe to be  $H$ ’s rules of inference. If you could do this, you could assume that  $H$  could construct a proof of  $P \Rightarrow Q$ , and hence it would be reasonable to conclude  $believes(H, P \Rightarrow Q)$ .

Suppose, on the other hand, that you believed  $believes(H, P \Rightarrow Q)$ , and that you also believed  $believes(H, P)$ . This would mean that you thought that  $H$  had both  $P \Rightarrow Q$  and  $P$  available to her. But if you had these two available to you, you would be able to infer  $Q$ , so since  $H$  is very similar to you she should also be able to infer  $Q$ . So from  $believes(H, P \Rightarrow Q)$  and  $believes(H, P)$  we can infer  $believes(H, Q)$ , or in other words  $(believes(H, P \Rightarrow Q)) \Rightarrow (believes(H, P) \Rightarrow believes(H, Q))$ .

We thus see that if we take  $believes(H, P)$  to mean ‘If I were  $H$  I would be able to prove  $P$ ’, then  $(believes(H, P \Rightarrow Q))$  and  $(believes(H, P) \Rightarrow believes(H, Q))$  are equivalent. This has considerable advantages in terms of theorem proving, since it means that much of the time we can do our reasoning by switching to the believer’s point of view and doing perfectly ordinary first-order reasoning. If, in addition, we treat  $\neg P$  as a shorthand for  $P \Rightarrow \perp$ , we see that  $believes(H, \neg P)$  is equivalent to  $believes(H, P) \Rightarrow believes(H, \perp)$ . If we take the further step of assuming that nobody believes  $\perp$ , we can see that  $believes(H, \neg P) \Rightarrow \neg believes(H, P)$  (though not  $\neg believes(H, P) \Rightarrow believes(H, \neg P)$ ). We cannot, however, always assume that everyone’s beliefs are consistent, so we may not always want to take this further step (note that in possible worlds treatments, we are **forced** to assume that everyone’s beliefs are consistent), but it is useful to be able to use it as a default rule, particularly once we understand the assumptions that lie behind it.

## References

- [1] Allen, J. F. and C. R. Perrault, Analyzing intention in utterances (1980), *AI* 15: 143–78.
- [2] Appelt, D. E., *Planning English referring expressions* (1985), *AI* 26: 1–33.
- [3] Austin, J. L., *How to do things with words* (1962), Oxford: OUP, 2nd edition.
- [4] Blum, A. L. and M. L. Furst, Fast planning through planning graph analysis (1995), in *Proc. 14th IJCAI*, pp. 1636–1642.

- [5] Bonet, B. and H. Geffner, Heuristic Search Planner (2000), *AI Magazine* 21(2).
- [6] Bruce, B. C., Generation as a social action (1975), in B. L. Nash-Webber and R. C. Schank (eds), *Theoretical issues in natural language processing*, pp. 74–7. Cambridge, Massachusetts: ACL.
- [7] Bunt, H., Dialogue pragmatics and context specification (2000), [8], pp. 81–150.
- [8] Bunt, H. and W. Black, (eds), *Abduction, belief and context in dialogue: studies in computational pragmatics* (2000), Philadelphia: John Benjamins.
- [9] Cohen, P. R. and H. J. Levesque, Rational interaction as the basis for communication (1990), [10], pp. 221–55.
- [10] Cohen, P. R., J. Morgan and M. E. Pollack, (eds), *Intentions in communication* (1990), Cambridge, Massachusetts: MIT.
- [11] Cohen, P. R. and C. R. Perrault, Elements of a plan-based theory of speech acts (1979), *Cognitive Science* 3: 177–212.
- [12] Feigenbaum, E. A. and J. Feldman, Editors, *Computers and thought* (1995), Cambridge, Massachusetts: MIT Press. First published 1963 by McGraw-Hill.
- [13] Fikes, R. E. and N. J. Nilsson, STRIPS: A new approach to the application of theorem proving to problem solving (1971), *AI* 2: 189–208.
- [14] Green, C., Application of theorem proving to problem solving (1969), in *Proc. 1st IJCAI*, pp. 219–39.
- [15] Grice, H. P., Logic and conversation (1975), in P. Cole and J. Morgan, (eds), *Syntax and semantics 3: Speech acts*, pp. 41–58. New York: Academic Press.
- [16] Grosz, B. J. and C. L. Sidner, Plans for discourse (1990), [10], pp. 416–44.
- [17] Hintikka, J., *Knowledge and belief: An introduction to the two notions* (1962), New York: Cornell University Press.
- [18] Hoffmann, J. and B. Nebel, The FF planning system: Fast plan generation through heuristic search (2001), *Journal of AI Research* 14: 253–302.
- [19] Konolige, K., *A deduction model of belief* (1986), London: Pitman.
- [20] Kripke, S., Semantical considerations on modal logic (1963), in *Acta Philosophica Fennica* 16: 83–94.
- [21] Lewis, D., Scorekeeping in a language game (1979), *J. Phil. Logic* 8: 339–59.
- [22] McCarthy, J. and P. J. Hayes, Some philosophical problems from the standpoint of artificial intelligence (1969), *Machine Intelligence* 4: 463–502.
- [23] Newell, A., J. C. Shaw and H. A. Simon, Empirical explorations with the logic theory machine (1957), *Proc. Western Joint Computer Conference*, 15: 218–239.
- [24] Newell, A. and H. A. Simon, GPS, a program that simulates human thought (1963), [12], pp. 279–93.
- [25] Nguyen, X. and S. Kambhampati, Reviving partial order planning (2001), in *Proc. IJCAI*, pp. 459–66.
- [26] Pollack, M. E., Plans as complex mental attitudes (1990), [10], pp. 77–103.
- [27] Ramsay, A., Speech act theory and epistemic planning (2000), [8], pp. 293–310.
- [28] Searle, J. R., What is a speech act? (1965), in M. Black, (ed), *Philosophy in America*, pp. 221–39. Allen and Unwin.
- [29] Stalnaker, R., Pragmatics (1972), in D. Davidson and G. Harman, (eds), *Semantics of natural language (Synthese Library, Vol. 40)*, pp. 380–97. Dordrecht, Holland: D. Reidel.

# Towards a redundancy elimination algorithm for underspecified descriptions

Alexander Koller and Stefan Thater

*Department of Computational Linguistics  
Universität des Saarlandes, Saarbrücken, Germany  
{koller,stth}@coli.uni-sb.de*

---

## Abstract

This paper proposes an efficient algorithm for the *redundancy elimination* problem: Given an underspecified semantic representation (USR), compute an USR which has fewer readings, but still describes at least one representative of each semantic equivalence class of the original readings. The algorithm operates on underspecified chart representations which are derived from dominance graphs; it can be applied to the USRs computed by large-scale grammars. To our knowledge, it is the first redundancy elimination algorithm which maintains underspecification, rather than just enumerating non-redundant readings.

---

## 1 Introduction

Underspecification is the standard approach to dealing with scope ambiguities in computational semantics [12,6,7,2]. The basic idea is to not enumerate all possible semantic representations for each syntactic analysis, but to derive a single compact *underspecified representation (USR)*. This simplifies semantics construction, and current algorithms support the efficient enumeration of readings from an USR [10].

In addition, underspecification has the potential for eliminating incorrect or redundant readings by inferences based on context or world knowledge, without even enumerating them. For instance, sentences with scope ambiguities often have readings which are semantically equivalent. In this case, we typically need to retain only one reading from each equivalence class. This situation is illustrated by the following two sentences from the Rondane treebank, which is distributed with the English Resource Grammar (ERG; [5]), a broad-coverage HPSG grammar.

- (1) For travellers going to Finnmark there is a bus service from Oslo to Alta through Sweden. (Rondane 1262)
- (2) We quickly put up the tents in the lee of a small hillside and cook for the first time in the open. (Rondane 892)

For the two example sentences, the ERG (Version 01-2006) derives USRs with seven and six quantifiers, respectively, that correspond to various types of noun

phrases (including proper names and pronouns). The USR for (1) describes 3960 readings, which are all semantically equivalent to each other. On the other hand, the USR for (2) has 480 readings, which fall into two classes of mutually equivalent readings, characterised by the relative scope of “the lee of” and “a small hillside.”

This paper presents an algorithm for the *redundancy elimination* problem: Given an USR, compute an USR which has fewer readings, but still describes at least one representative of each equivalence class – without enumerating any readings. This algorithm computes the one or two representatives of the semantic equivalence classes in the above examples, so subsequent modules don’t have to deal with all the other equivalent readings. It also closes the gap between the large number of readings predicted by the grammar and the intuitively perceived much lower degree of ambiguity of these sentences. Finally, it can be helpful for a grammar designer because it is much more feasible to check whether two readings are linguistically reasonable than 480.

We model equivalence in terms of rewrite rules that permute quantifiers without changing the semantics of the readings. The particular USRs we work with are underspecified chart representations, which can be computed from dominance graphs (or USRs in some other underspecification formalisms) efficiently [10]. The algorithm can deal with many interesting cases, but is incomplete in the sense that the resulting USR may still describe multiple equivalent readings.

To our knowledge, this is the first algorithm in the literature for redundancy elimination on the level of USRs. There has been previous research on *enumerating* only some representatives of each equivalence class [13,4], but these approaches don’t maintain underspecification: After running their algorithms, we have a set of readings rather than an underspecified representation.

*Plan of the paper.* We will first define dominance graphs and review the necessary background theory in Section 2. We will then give a formal definition of equivalence and derive some first results in Section 3. Section 4 presents the redundancy elimination algorithm. Finally, Section 5 concludes and points to further work.

## 2 Dominance Graphs

The basic underspecification formalism we assume here are *labelled dominance graphs* [1]. Dominance graphs are equivalent to leaf-labelled normal dominance constraints [7], which have been discussed extensively in previous literature.

**Definition 2.1** A (*compact*) *dominance graph* is a directed graph  $(V, E \uplus D)$  with two kinds of edges, *tree edges*  $E$  and *dominance edges*  $D$ , such that:

- (i) the graph  $(V, E)$  defines a collection of node disjoint trees of height 0 or 1. We call the trees in  $(V, E)$  the *fragments* of the graph.
- (ii) if  $(v, v')$  is a dominance edge in  $D$ , then  $v$  is a hole and  $v'$  is a root in  $G$ . A node  $v$  is a *root* (in  $G$ ) if  $v$  does not have incoming tree edges; otherwise,  $v$  is a *hole*.

A *labelled dominance graph* over a ranked signature  $\Sigma$  is a triple  $G = (V, E \uplus D, L)$

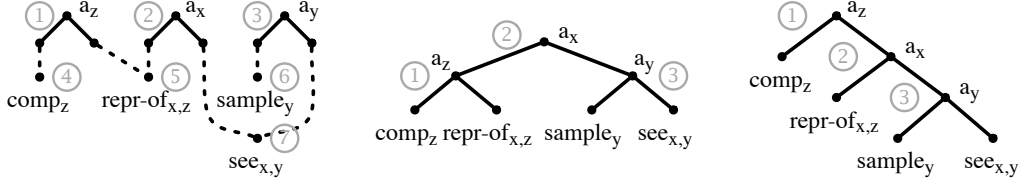


Fig. 1. A dominance graph that represents the five readings of the sentence “a representative of a company saw a sample” (left) and two (of five) configurations.

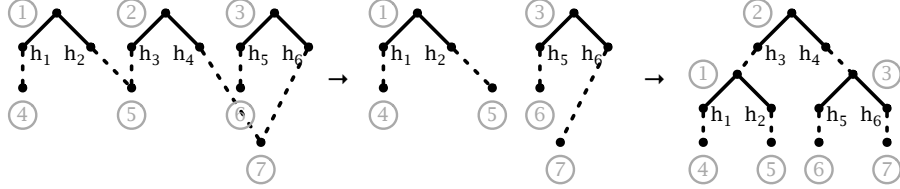


Fig. 2. An example computation of a solved form.

such that  $(V, E \uplus D)$  is a dominance graph and  $L : V \rightsquigarrow \Sigma$  is a partial *labelling function* which assigns a node  $v$  a label with arity  $n$  iff  $v$  is a root with  $n$  outgoing tree edges. Nodes without labels (i.e., holes) must have outgoing dominance edges.

We will write  $v:f(v_1, \dots, v_k)$  for a fragment whose root  $v$  is labelled with  $f$  and whose holes are  $v_1, \dots, v_k$ . We will write  $R(F)$  for the root of the fragment  $F$ , and we will typically just say *graph* instead of *labelled dominance graph*.

An example of a labelled dominance graph is shown to the left of Fig. 1. Tree edges are drawn as solid lines, and dominance edges are drawn as dotted lines, directed from top to bottom. This graph can serve as an *USR* for the sentence “a representative of a company saw a sample” if we demand that the holes are “plugged” by roots while realising the dominance edges as dominance, as in the two (of five) *configurations* shown to the right [7]. Configurations encode semantic representations of the sentence, and we freely read configurations as ground terms over  $\Sigma$ .

## 2.1 Solving dominance graphs

Algorithms for *solving* a dominance graph in order to compute the readings it describes typically compute its *minimal solved forms* [1,3]. In this paper, we restrict ourselves to *hypernormally connected* graphs (defined below), for which one can show that all solved forms are minimal and bijectively correspond to configurations.

Let  $G, G'$  be dominance graphs. We say that  $G$  is *in solved form* iff it is a forest, and  $G$  is a *solved form of*  $G'$  if  $G$  is in solved form and more specific than  $G'$  i.e.,  $G$  and  $G'$  have the same labels and tree fragments, and the reachability relation of  $G$  extends that of  $G'$ .  $G'$  is *solvable* if it has a solved form  $G$ . If  $G'$  is hypernormally connected, then each hole in  $G$  has exactly one outgoing dominance edge, and  $G$  can be mapped to a configuration by identifying the two ends of each dominance edge; conversely, we can find a unique solved form for each configuration. The graph to the left of Fig. 2 shows one of the (minimal) solved forms of the example graph, which corresponds to the configuration in the middle of Fig. 1.

```

COMPUTE-CHART(G)
1 if there is an entry for G in the chart
2 then return true
3 $free \leftarrow$ FREE-FRAGMENTS(G)
4 if $free = \emptyset$
5 then return false
6 if G contains only one fragment
7 then return true
8 for each $F \in free$
9 do $split \leftarrow$ SPLIT(G, F)
10 for each $S \in WCCS(G - F)$
11 do if COMPUTE-CHART(S) = false
12 then return false
13 add ($G, split$) to the chart
14 return true

```

|                             |                                                                       |
|-----------------------------|-----------------------------------------------------------------------|
| $\{1, 2, 3, 4, 5, 6, 7\} :$ | $\langle 1, h_1 \mapsto \{4\}, h_2 \mapsto \{2, 3, 5, 6, 7\} \rangle$ |
|                             | $\langle 2, h_3 \mapsto \{1, 4, 5\}, h_4 \mapsto \{3, 6, 7\} \rangle$ |
|                             | $\langle 3, h_5 \mapsto \{5\}, h_6 \mapsto \{1, 2, 4, 5, 7\} \rangle$ |
| $\{2, 3, 5, 6, 7\} :$       | $\langle 2, h_3 \mapsto \{5\}, h_4 \mapsto \{3, 6, 7\} \rangle$       |
|                             | $\langle 3, h_5 \mapsto \{6\}, h_6 \mapsto \{2, 5, 7\} \rangle$       |
| $\{3, 6, 7\} :$             | $\langle 3, h_5 \mapsto \{6\}, h_6 \mapsto \{7\} \rangle$             |
| $\{2, 5, 7\} :$             | $\langle 2, h_3 \mapsto \{5\}, h_4 \mapsto \{7\} \rangle$             |
| $\{1, 4, 5\} :$             | $\langle 1, h_1 \mapsto \{4\}, h_2 \mapsto \{5\} \rangle$             |
| $\{1, 2, 4, 5, 7\} :$       | $\langle 1, h_1 \mapsto \{4\}, h_2 \mapsto \{2, 5, 7\} \rangle$       |
|                             | $\langle 2, h_3 \mapsto \{1, 4, 5\}, h_4 \mapsto \{7\} \rangle$       |

Fig. 3. The chart solver and an example chart computed for the dominance graph in Fig. 2.

The key concept of the solver we build upon is that of a *free fragment* [3]. A fragment  $F$  in a solvable graph  $G$  is free iff there is a solved form in which  $F$  is at the root. It can be shown that a fragment is free iff it has no incoming dominance edges and its holes are in different biconnected components of the graph i.e., they are disconnected if the root of the fragment is removed from the graph [3]. Removing a free fragment from a graph splits the graph into different weakly connected components (wccs) – one for each hole. Thus each free fragment  $F$  induces a *split* of  $G$ , which consists of a reference to  $F$  and a mapping of the other fragments to the hole to which they are connected. For instance, the example graph has three free fragments: 1, 2, and 3. By removing fragment 2, the graph is decomposed into two wccs, which are connected to the holes  $h_3$  and  $h_4$ , respectively (see Fig. 2).

The solver [10] is shown in Fig. 3. It computes a chart-like data structure which assigns sets of splits to subgraphs. For each subgraph it is called on, the solver computes the free fragments, the splits they induce, and calls itself recursively on the wccs of each split. It records subgraphs and splits in the chart, and will not repeat work for a subgraph it has encountered before. The algorithm returns true iff the original graph was solvable. The chart tells us how to build the minimal solved forms of the graph: For each subgraphs, pick any split, compute a solved form for each wcc recursively, and plug them into the given hole of the split’s root fragment. As an example, the chart for the graph in Fig. 1 is shown to the right of Fig. 3.

Notice that the chart which the solver computes, while possibly exponentially larger than the original graph, is still exponentially smaller than the entire set of readings because common subgraphs (such as  $\{2, 5, 7\}$  in the example) are represented only once. Thus the chart can still serve as an underspecified representation.

## 2.2 Hypernormally connected dominance graphs

A *hypernormal path* [1] in a graph  $G$  is a path in the undirected version  $G_u$  of  $G$  that does not use two dominance edges that are incident to the same hole. We say that  $G$  is *hypernormally connected* (*hnc*) iff each pair of nodes is connected by a simple



hypernormal path in  $G$ . Hnc graphs are equivalent to *chain-connected* dominance constraints [9], and are closely related to *dominance nets* [11]. The results in this paper are restricted to hnc graphs, but this does not limit the applicability of our results: an empirical study suggests that all dominance graphs that are generated by current large-scale grammars are (or should be) hnc [8].

The key property of hnc dominance graphs is that their solved forms correspond to configurations, and we will freely switch between solved forms and their corresponding configurations. Another important property of hnc graphs which we will use extensively in the proofs below is that it is possible to predict which holes of fragments can dominate other fragments in a solved form.

**Lemma 2.2** *Let  $G$  be a hnc graph with free fragment  $F$ . Then all weakly connected components of  $G - F$  are hnc.*

**Proposition 2.3** *Let  $F_1, F_2$  be fragments in a hnc dominance graph  $G$ . If there is a solved form  $S$  of  $G$  in which  $R(F_1)$  dominates  $R(F_2)$ , then there is exactly one hole  $h$  of  $F_1$  which is connected to  $R(F_2)$  by a simple hypernormal path which doesn't use  $R(F_1)$ . In particular,  $h$  dominates  $R(F_2)$  in  $S$ .*

**Proof.** Let's say that  $F_1$  dominates  $F_2$  in some solved form  $S$ . There is a run of the solver which computes  $S$ . This run chooses  $F_1$  as a free fragment before it chooses  $F_2$ . Let's call the subgraph in which the split for  $F_1$  is chosen,  $G'$ .  $G'$  is hnc (Lemma 2.2), so in particular there is a simple hypernormal path from the hole  $h$  of  $F_1$  which is in the same wcc as  $F_2$  to  $R(F_2)$ ; this path doesn't use  $R(F_1)$ . On the other hand, assume there were another hole  $h'$  of  $F_1$  which is connected to  $R(F_2)$  by a path that doesn't use  $R(F_1)$ . Then the path via  $R(F_2)$  would connect  $h$  and  $h'$  even if  $R(F_1)$  were removed, so  $h$  and  $h'$  would be in the same biconnected component of  $G$ , in contradiction to the assumption that  $F_1$  is free in  $G'$ .

For the second result, note that  $F_2$  is assigned to the hole  $h$  in the split for  $F_1$ .  $\square$

The following definition captures the complex condition in Prop. 2.3:

**Definition 2.4** Let  $G$  be a hnc dominance graph. A fragment  $F_1$  in  $G$  is called a *possible dominator* of another fragment  $F_2$  in  $G$  iff it has exactly one hole  $h$  which is connected to  $R(F_2)$  by a simple hypernormal path which doesn't use  $R(F_1)$ . We write  $\text{ch}(F_1, F_2)$  for this unique  $h$ .

### 3 Equivalence

Equivalence is traditionally defined as the relation between formulas which have the same interpretation. However, even first-order equivalence is an undecidable problem, thus an algorithm which checks for semantic equivalence of different configurations of a graph can't possibly be efficient. On the other hand, we do not need to solve the full semantic equivalence problem, as we only want to compare formulas that are readings of the same sentence i.e., different configurations of the same USR. Such formulas only differ in the way that the fragments are combined. We

can therefore approximate equivalence by using a *rewrite system* that permutes fragments and defining equivalence of configurations as mutual rewritability as usual.

By way of example, consider again the two (equivalent) configurations shown in Fig. 1. We can obtain the second configuration from the first one by applying the following rewrite rule, which rotates the nodes 1 and 2:

$$a_x(a_z(P, Q), R) \rightarrow a_z(P, a_x(Q, R)) \quad (3)$$

The formulas on both sides of the arrow are semantically equivalent in first-order logic for any choice of the subformulas  $P$ ,  $Q$ , and  $R$ . Thus the equivalence of the two configurations with respect to our one-rule rewrite system implies that they are also semantically equivalent.

While we will require that the rewriting approximation is *sound* i.e., rewrites formulas into equivalent formulas, we cannot usually hope to achieve *completeness* i.e., there will be semantic equivalences that are not modelled by the rewriting equivalence. However, we believe that the rewriting-based system will still prove to be useful in practical applications, as the permutation of quantifiers is exactly the kind of variability that an underspecified description allows.

We formalise this rewriting-based notion of equivalence as follows. The definition uses the abbreviation  $\overline{x_{[1,k]}}$  for  $x_1, \dots, x_{k-1}$ , and  $\overline{x_{[k,n]}}$  for  $x_{k+1}, \dots, x_n$ .

**Definition 3.1** A *permutation system*  $R$  is a system of rewrite rules over a signature  $\Sigma$  of the following form:

$$f_1(\overline{x_{[1,i]}}), f_2(\overline{y_{[1,k]}}), z, \overline{y_{[k,m]}}), \overline{x_{[i,n]}}) \rightarrow f_2(\overline{y_{[1,k]}}), f_1(\overline{x_{[1,i]}}), z, \overline{x_{[i,n]}}), \overline{y_{[k,m]}})$$

The *permutability relation*  $P(R)$  is the binary relation  $P(R) \subseteq (\Sigma \times \mathbb{N})^2$  which contains exactly the pairs  $((f_1, i), (f_2, k))$  and  $((f_2, k), (f_1, i))$  for each such rewrite rule.

As usual, we say that two terms are *equivalent* with respect to  $R$ ,  $s \approx_R t$ , iff there is a sequence of rewrite steps and inverse rewrite steps that rewrite  $s$  into  $t$ . We say that  $R$  is *sound* with respect to a semantic notion of equivalence  $\equiv$  if  $\approx_R \subseteq \equiv$ . If  $G$  is a graph over  $\Sigma$  and  $R$  a permutation system, then we write  $SC_R(G)$  for the set of equivalence classes  $\text{Conf}(G)/\approx_R$ , where  $\text{Conf}(G)$  is the set of configurations of  $G$ .

A rewrite system (let's call it  $R_{fol}$ ) which is sound for the standard equivalence relation of first-order logic could use rule (3) and the three other permutations of two existential quantifiers, plus the following rule for universal quantifiers:

$$\text{every}_x(X, \text{every}_y(Y, Z)) \rightarrow \text{every}_y(Y, \text{every}_x(X, Z))$$

The other three permutations of universal quantifiers, as well as the permutations of universal and existential quantifiers, are not sound.

It is possible to compute  $SC_R(G)$  by solving  $G$  and using a theorem prover for equational reasoning to compute the equivalence classes of the configurations, but this is very inefficient. To replace this by a computation on the USR, we must be able to recognise whether two fragments of a graph can be permuted in all configurations of the graph. This is not possible in general: If we don't know in advance

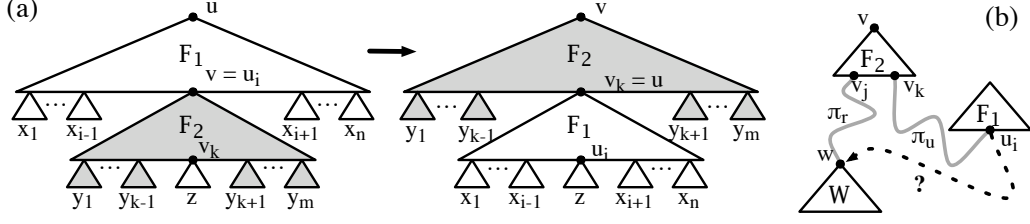


Fig. 4. Diagrams for the proof of Lemma 3.3

which hole of one fragment the other fragment can plug, we can't know whether the two fragments can be permuted. However, in a *hnc* graph, the hole of a fragment which another fragment can plug is determined uniquely (because of Lemma 2.3), and can be recognised without solving the graph.

**Definition 3.2** Let  $R$  be a permutation system. Two fragments  $F_1$  and  $F_2$  with root labels  $f_1$  and  $f_2$  in a graph  $G$  are called  $R$ -permutable iff they are possible dominators of each other and  $((f_1, \text{ch}(F_1, F_2)), (f_2, \text{ch}(F_2, F_1))) \in P(R)$ .

**Lemma 3.3** Let  $R$  be a permutation system, let  $F_1 = u: f_1(u_1, \dots, u_n)$  and  $F_2 = v: f_2(v_1, \dots, v_m)$  be  $R$ -permutable fragments in the *hnc* graph  $G$ , such that  $F_2$  is free, and let  $C_1$  be a configuration of  $G$  in which  $u$  is the father of  $v$ . Then:

- It is possible to apply a  $R$ -rewrite step or an inverse  $R$ -rewrite step to  $C_1$  at  $u$ ; call the resulting tree  $C_2$ .
- $C_2$  is also a configuration of  $G$ .
- $C_2 \approx_R C_1$ .

**Proof.** Let  $i = \text{ch}(F_1, F_2)$  and  $k = \text{ch}(F_2, F_1)$ ; we know that  $((f_1, i), (f_2, k)) \in P(R)$ .

(a)  $F_1$  is a possible dominator of  $F_2$ , so  $u_i$  is plugged with  $v$  in  $C_1$  (Lemma 2.3). Thus the (possibly inverse) rule which justified the tuple  $((f_1, i), (f_2, k))$  is applicable at  $u$ .

(b) We must verify that every dominance edge in  $G$  is realised by  $C_2$ . As Fig. 4a shows, all dominance edges that do not go out of a hole of  $F_1$  are still trivially realised by  $C_2$ . Now let's consider dominances out of the holes of  $F_1$ .

- Dominance edges out of any  $u_j$  with  $j \neq i$  are still satisfied (see the figure).
- Dominance edges from  $u_i$  to a node in  $z$  are still satisfied (see the figure).
- Dominance edges from  $u_i$  to  $v$ : Such edges cannot exist in  $G$  as  $F_2$  is free.
- Dominance edges from  $u_i$  to a node  $w$  in some  $y_j$  with  $j \neq k$ : Such edges cannot exist either.  $F_2$  is a possible dominator of the fragment  $W$  whose root  $w$  is, so there is a simple hypernormal path  $\pi_w$  from  $\text{ch}(F_2, W)$  to  $w$  which doesn't use  $v$ ;  $\text{ch}(F_2, W) = v_j$  because  $v_j$  dominates  $w$  in  $C_1$  (Lemma 2.3). On the other hand,  $F_2$  is a possible dominator of  $F_1$ , so there is a simple hypernormal path  $\pi_u$  from  $v_k$  to  $u_i$  which doesn't use  $v$ . Now if there were a dominance edge from  $u_i$  to  $w$  in  $G$ , then  $v_j$  and  $v_k$  would be in the same biconnected component (they would be connected via  $\pi_u \circ (u_i, w) \circ \pi_w^{-1}$  if  $v$  were removed), which contradicts the freeness of  $F_2$  (see Fig. 4b).  $\square$

## 4 Underspecified redundancy elimination

Now we can finally consider the problem of strengthening an USR in order to remove redundant readings which are equivalent to other readings. We will define an algorithm which gets as its input a graph  $G$ , a chart as computed by COMPUTE-CHART, and a permutability relation  $P(R)$ . It will then remove splits from the chart, to the effect that the chart represents fewer solved forms of the original graph, but at least one representative from each class in  $SC_R(G)$  remains. The subgraph sharing of the original chart will be retained, so the computed chart is still an USR.

The key concept in the redundancy elimination algorithm is that of a *permutable split*. Intuitively, a split of  $G$  is called permutable if its root fragment  $F$  is permutable with all other fragments in  $G$  which could end up above  $F$ . Because of Lemma 3.3, we can then always pull  $F$  to the root by a sequence of rewrite steps. This means that for any configuration of  $G$ , there is an equivalent configuration whose root is  $F$  – i.e., by choosing the split for  $F$ , we lose no equivalence classes.

**Definition 4.1** Let  $R$  be a permutation system. A split  $S$  of a graph  $G$  is called *R-permutable* iff the root fragment  $F$  of  $S$  is  $R$ -permutable with all other fragments in  $G$  which are possible dominators of  $F$  in  $G$ .

In the graph of Fig. 1, all three splits are  $R_{fol}$ -permutable: For each of the upper fragments, the other two upper fragments are possible dominators, but as all three fragments are labelled with existential quantifiers and  $R_{fol}$  contains all permutations of existential quantifiers, the fragments are permutable with each other. And indeed, we can pick any of the three fragments as the root fragment, and the resulting split will describe a representative of the single equivalence class of the graph.

**Proposition 4.2** Let  $G$  be a hnc graph, and let  $S$  be a permutable split of  $G$ . Then  $SC(S) = SC(G)$ .

**Proof.** If  $G$  is unsolvable, the claim is trivially true. Otherwise, let  $C$  be an arbitrary configuration of  $G$ ; we must show that  $S = (F, h_1 \mapsto G_1, \dots, h_n \mapsto G_n)$  has a configuration  $C'$  which is equivalent to  $C$ .

Let's say that the fragments which properly dominate  $F$  in  $C$  are  $F_1, \dots, F_n$  ( $n \geq 0$ ), ordered in such a way that  $F_i$  dominates  $F_j$  in  $C$  for all  $i < j$ . Each  $F_i$  is a possible dominator of  $F$ , by Prop. 2.3. Because  $S$  is permutable, this means that each  $F_i$  is permutable with  $F$  in  $G$ . By applying Lemma 3.3  $n$  times (first to  $F$  and  $F_n$ , then to  $F$  and  $F_{n-1}$ , and so on), we can compute a configuration  $C'$  of  $G$  in which  $F$  is at the root and such that  $C' \approx_R C$ . But  $C$  is a configuration of  $S$ , which proves the theorem.  $\square$

This suggests the following redundancy elimination algorithm:

REDUNDANCY-ELIMINATION( $Ch, G, R$ )

- 1 **for** each subgraph  $G'$  in  $Ch$
- 2     **do if**  $G'$  has an  $R$ -permutable split  $S$
- 3         **then** remove all splits for  $G'$  except for  $S$  from  $Ch$

Because of Prop. 4.2, the algorithm is correct in that for each configuration  $C$  of  $G$ , the reduced chart still has a configuration  $C'$  with  $C \approx_R C'$ . The particular choice of  $S$  doesn't affect the correctness of the algorithm (but may change the number of remaining configurations). However, the algorithm is not complete in the sense that the reduced chart can have no two equivalent configurations. We will illustrate this below. We can further optimize the algorithm by deleting subgraphs (and their splits) that are not referenced anymore by using reference counters. This doesn't change the set of solved forms of the chart, but may further reduce the chart size.

In the running example, we would run REDUNDANCY-ELIMINATION on the chart in Fig. 3. As we have seen, all three splits of the entire graph are permutable, so we can pick any of them e.g., the split with root fragment 2, and delete the splits with root fragments 1 and 3. This reduces the reference count of some subgraphs (e.g.  $\{2, 3, 5, 6, 7\}$ ) to 0, so we can remove these subgraphs too. The resulting chart is shown below, which represents a single solved form (the one shown in Fig. 2).

$$\begin{aligned} \{1, 2, 3, 4, 5, 6, 7\} &: \langle 2, h_2 \mapsto \{1, 4\}, h_4 \mapsto \{3, 6, 7\} \rangle \\ \{1, 4\} &: \langle 1, h_1 \mapsto \{4\} \rangle \\ \{3, 6, 7\} &: \langle 3, h_5 \mapsto \{6\}, h_6 \mapsto \{7\} \rangle \end{aligned}$$

Now consider variations of the graph in Fig. 1 in which the quantifier labels are different; these variant graphs have exactly the same chart, but fewer fragment pairs will be permutable. If all three quantifiers are universal, then the configurations fall into two equivalence classes which are distinguished by the relative scope of the fragments 1 and 2. The algorithm will recognise that the split with root fragment 3 is permutable and delete the splits for 1 and 2. The resulting chart has two solved forms. Thus the algorithm is still complete in this case. If, however, the fragments 1 and 2 are existential quantifiers and the fragment 3 is universal, there are three equivalence classes, but the chart computed by the algorithm will have four solved forms. The problem stems from the fact that neither of the existential quantifiers is permutable as long as the universal quantifier is still in the same subgraph; but the two configurations in which 2 dominates 3 are equivalent.

*Runtime analysis.* Given a graph  $G$  with  $n$  nodes and  $m$  edges, we can compute a table which specifies for each pair  $u, v$  of root nodes whether there is a unique hole of  $u$  from which  $v$  can be reached via a simple hypernormal path which doesn't use  $u$ , and which hole this is. A naive algorithm for doing this iterates over all  $u$  and  $v$  and then performs a depth-first search through  $G$ , which takes time  $O(n^2(n+m))$ , which is a negligible runtime in practice.

Given this table, we can determine the possible dominators of each fragment in time  $O(n)$  (because there are at most  $O(n)$  possible dominators). Thus it takes time  $O(n)$  to decide whether a split is permutable, and time  $O(n \cdot S)$ , where  $S$  is the number of splits in the chart, to run the entire elimination algorithm. The reference counting optimisation adds nothing to this asymptotic runtime, as each split may trigger at most one reference count update for each hole of the split's root fragment.

## 5 Conclusion

We have presented an algorithm for redundancy elimination on underspecified chart representations. It checks for each subgraph in the chart whether it has a *permutable split*; if yes, it removes all other splits for this subgraph. This reduces the set of described readings, while making sure that at least one representative of each original equivalence class remains while maintaining underspecification. Equivalence is defined with respect to a certain class of rewriting systems which approximates semantic equivalence of the described formulas and fits well with the underspecification setting. The algorithm runs in polynomial time in the size of the chart.

The algorithm is useful in practice: it reduces the USRs for (1) and (2) from the introduction to one and two solved forms, respectively. In fact, initial experiments with the Rondane treebank suggest that it reduces the number of readings of a typical sentence by an order of magnitude. It does this efficiently: Even on USRs with billions of readings, for which the enumeration of readings would take about a year, it finishes after a few seconds. However, the algorithm is not complete in the sense that the computed chart has no more equivalent readings. We have some ideas for achieving this kind of completeness, which we will explore in future work. Another line in which the present work could be extended is to allow equivalence with respect to arbitrary rewrite systems.

## References

- [1] Althaus, E., D. Duchier, A. Koller, K. Mehlhorn, J. Niehren and S. Thiel, *An efficient graph algorithm for dominance constraints*, Journal of Algorithms **48** (2003), pp. 194–219.
- [2] Blackburn, P. and J. Bos, “Representation and Inference for Natural Language. A First Course in Computational Semantics,” CSLI Publications, 2005.
- [3] Bodirsky, M., D. Duchier, J. Niehren and S. Miele, *An efficient algorithm for weakly normal dominance constraints*, in: *ACM-SIAM Symposium on Discrete Algorithms* (2004).
- [4] Chaves, R. P., *Non-redundant scope disambiguation in underspecified semantics*, in: *Proceedings of the 8th ESSLLI Student Session*, Vienna, 2003, pp. 47–58.
- [5] Copestake, A. and D. Flickinger, *An open-source grammar development environment and broad-coverage english grammar using HPSG*, in: *Proc. of LREC*, 2000.
- [6] Copestake, A., D. Flickinger, C. Pollard and I. Sag, *Minimal recursion semantics: An introduction.*, Journal of Language and Computation (2004), to appear.
- [7] Egg, M., A. Koller and J. Niehren, *The Constraint Language for Lambda Structures*, Logic, Language, and Information **10** (2001), pp. 457–485.
- [8] Fuchss, R., A. Koller, J. Niehren and S. Thater, *Minimal recursion semantics as dominance constraints: Translation, evaluation, and analysis*, in: *Proc. of ACL*, Barcelona, 2004.
- [9] Koller, A., J. Niehren and S. Thater, *Bridging the gap between underspecification formalisms: Hole semantics as dominance constraints*, in: *Proc. of EACL-03*, 2003.
- [10] Koller, A. and S. Thater, *The evolution of dominance constraint solvers*, in: *Proc. of ACL-05 Workshop on Software*, Ann Arbor, 2005.
- [11] Niehren, J. and S. Thater, *Bridging the gap between underspecification formalisms: Minimal recursion semantics as dominance constraints*, in: *Proc. of ACL-03*, 2003.
- [12] van Deemter, K. and S. Peters, “Semantic Ambiguity and Underspecification,” CSLI, 1996.
- [13] Vestre, E., *An algorithm for generating non-redundant quantifier scopings*, in: *Proc. of EACL*, Berlin, 1991, pp. 251–256.

# Quantifiers in Dependency Tree Semantics

Leonardo Lesmo, Livio Robaldo, Jelle Gerbrandy

*Dipartimento di Informatica - Università di Torino*  
*{lesmo,robaldo,gerbrand}@di.unito.it*

---

## Abstract

Dependency Tree Semantics (DTS) is an underspecified formalism for representing quantifier scope ambiguities in natural language. DTS features a direct interface with a Dependency grammar and an incremental, constraint-based disambiguation mechanism. In this paper, we discuss the meaning of quantifier dependency in DTS by translating its well formed structures into formulae of a Second Order Logic augmented with Mostowskian generalized quantifiers.

---

## 1 Introduction

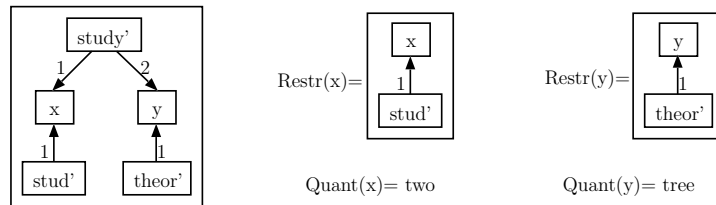
Dependency Tree Semantics (DTS) is an underspecified formalism for dealing with quantifier scope ambiguity. DTS tries to keep the advantages of most common underspecification techniques: it has a straightforward syntax-semantics interface with a Dependency Grammar, just as QLF has [1], and it allows for monotonically adding constraints to take partial disambiguations into account, just as in UDRT [12], MRS [3] or CLLS [4]. These features have been presented in [7] and [8], whereas in [9] DTS is proposed as a possible underspecified semantic structure of Meaning $\Leftrightarrow$ Text Theory [10]. This paper discusses a third property of DTS in further depth: the possibility to represent branching quantifier (BQ) readings. Branching quantification in DTS has partially been discussed in [7] and [8], in which we compared DTS with First Order Logic (FOL). However, FOL is limited in that it allows to represent only standard quantifiers ( $\exists$  and  $\forall$ ); in this paper we compare DTS with the logic developed in [13] and [14], which is a fragment of Second Order Logic which allows for a representation of branching quantification with Generalized Quantifiers.

### 1.1 *Intuitions behind Dependency Tree Semantics*

The key idea of DTS is to specify quantifier scope by explicitly showing the dependencies between involved (quantified) groups of entities, i.e. by implementing a sort of "Skolemization" in the underspecified representation. Well-formed structures in DTS are based on a simple graph  $G$  that represents the

predicate-arguments relations, without any quantification. The nodes of  $G$  are either predicates or discourse referents; each arc connects a predicate with a discourse referent and is labelled with the number of the predicate argument position. With each discourse referent we associate a *quantifier* (given by a function  $QUANT$  from discourse referents to quantifiers) and its *restriction*, which is given by a function  $RESTR$  that associates a subgraph of  $G$  to each discourse referent. In (1), we show a first simple example

(1) Two students study three theorems



The representation in (1) is still ambiguous; to disambiguate, we need to specify how the quantifiers depend on each other. This is done by inserting dotted arcs between discourse referents, named *semdep* arc. In figure 1.a and fig 1.b two fully-specified representations of sentence (1) are given. Fig.1.a shows the reading in which the quantifier ‘three’ depends on (has scope inside) the quantifier ‘two’. In figure 1.b, the arc linking  $x$  to  $y$  specifies that the two students depend on the theorems. In both interpretations, the wide-scope quantifier is linked to a new node called *Ctx* – the context.

But DTS allows for very natural representation of a third reading of sentence (1): in figure 1.c, both discourse referents are linked to the context. This is the branching quantifier (BQ) reading. As we will see, the BQ reading is true only in those models in which we can find a set of two students and a set of three theorems, for which it holds that each student in the first set studies each theorem in the second one. In NL, there are many cases in which the correct

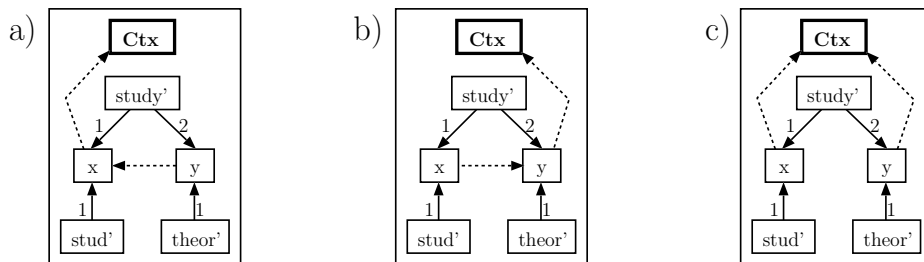


Fig. 1. The three readings of sentence (1)

truth conditions can be captured only via a BQ reading; in fact, it is easy to add some context elements in the sentence in order to force the two involved sets to be constant; for instance, in (2.i), the involved students and theorems are explicitly mentioned in two appositions, while in (2.ii) the prepositional modifier *with my sister* favours an interpretation in which three persons, two friends of mine and my sister, went together to three same concerts.



Finally, even if there are not explicit syntactic elements forcing a BQ reading, in many cases this is done by world knowledge; for example, in (2.iii), world knowledge seems to render the reading in which two students have seen the same three drug dealers the most salient; in fact, the presence of drug-dealers in front of a school is (fortunately) a rare event and this induces to prefer the reading minimizing the number of involved drug dealers.

- (2) (i) Two students, *John and Jack*, study three theorems: *the first three of the book*.  
(ii) Two friends of mine went to three concerts *with my sister*.  
(iii) Two students of mine have seen three drug dealers in front of the school.

Not all possible configurations of *semdep* arcs are allowed. For instance, a well-formed DTS cannot contain cycling paths, which would correspond to a reading in which two sets of entities depend on each other, which is clearly absurd. Furthermore, there are constraints to reduce the available readings to those admitted in NL. In this paper, we will focus on the expressivity of the general formalism, and provide a precise definition of the meaning of all configurations that respect a minimal set of syntactic constraints, and abstract from the question whether they correspond to an actual reading in NL. In other words, in DTS the set of logical admitted readings is kept separate from the subset of readings admitted in NL, and this paper focus on the former.

## 1.2 Formalisation: Syntax of DTS

A well-formed structure (wfs) in DTS is a Scoped Dependency Graph (SDG) as defined below. We take as given a set of predicates **pred** and a set of discourse referents *D*.

### Definition 1.1 [Flat Dependency Graphs (FDG)]

A Flat Dependency Graph is a tuple  $\langle N, L, A, Dom, f \rangle$  s.t.:

- *N* is a set of nodes  $\{n_1, n_2, \dots, n_k\}$ .
- *L* is a set of labels  $\{l_1, l_2, \dots, l_m\}$ ; in fig.1,  $L \equiv \{1, 2\}$ .
- $Dom \equiv \mathbf{pred} \cup D$  is a set *domain objects*: predicates and discourse referents
- *f* is a function  $f : N \mapsto Dom$ , specifying the node referent, i.e. the domain object with which the node is associated. In the following, whenever  $f(n) \in X$ , we will say that node *n* is of type *X*.
- *A* is a set of *arcs*. An arc is a triple  $(n_s, n_d, l)$ , where  $n_s, n_d \in N$ ,  $n_s$  is of type **pred**,  $n_d$  is of type *D* and  $l \in L$ .

Without going into further details, we stipulate that  $G_f$  is a connected acyclic graph such that each node of type **pred** has one node of type *D* for each of its places. Note that there can be two different nodes *u* and *v* s.t.  $f(u)=f(v)$ , i.e. the nodes in *N* can be seen as *occurrences* of symbols from *Dom*.

### Definition 1.2 [Scoped Dependency Graph (SDG)]

A Scoped Dependency Graph is a tuple  $\langle G_f, \text{ctx}, Q, \text{quant}, \text{restr}, \text{SemDep} \rangle$  s.t.:

- $G_f = \langle N, L, A, Dom, f \rangle$  is an FDG.
- $ctx$  is a special element called the context.
- $Q$  is a set of 2-place Mostowskian quantifiers  $\{every, most, two, \dots\}$ <sup>1</sup>
- $quant$  is a total function  $N_D \mapsto Q$ , where  $N_D \subseteq N$  are the nodes of type  $D$
- $restr$  is a function assigning to each  $d \in N_D$  its *restriction*, which is a subgraph of  $G_f$ .
- $SemDep$  is a relation  $N_D \times (N_D \cup \{\{ctx\}\})$ .

When  $SemDep(d, d')$ , we say that  $d$  *depends on*  $d'$ . Note that a discourse referent can depend on more than one other discourse referent. The dependence relation needs to satisfy the following constraints:

- The transitive closure of  $SemDep$  is a partial order on all discourse referents and  $ctx$ , with  $ctx$  as its maximal element.
- Let  $d$  be a discourse referent, and let  $R(d)$  be the smallest set that contains  $d$ , and for which it holds that if  $d'$  is in  $R(d)$  and  $d''$  occurs in the restriction of  $d'$ , then also  $d'' \in R(d)$ . It must hold that:
  - If  $d_1 \in R(d)$ ,  $d_2 \notin R(d)$ , and  $d_1$  depends on  $d_2$ , then also  $d$  depends on  $d_2$
  - If  $d_1 \in R(d)$ ,  $d_2 \notin R(d)$ , and  $d_2$  depends on  $d_1$ , then also  $d$  depends on  $d_1$

These last two constraints serve to exclude certain dependency relations that are ‘logically impossible’, and make sure that, for example, a sentence like “Most representatives of a company took every sample” does not get a reading in which ‘a’ depends on (only) ‘every’ and ‘every’ depends (only) on ‘most’.

## 2 Branching quantification

Branching quantification was introduced by Henkin [5] in the context of FOL; Hintikka [6] showed that it can occur also in NL. A great step toward the definition of a model-theoretic schema for BQ was made by Barwise [2] who merged Hintikka’s BQ account with the theory of Generalized Quantifiers. Barwise’s idea was that the truth-conditions of BQ readings are connected with the monotonicity of the involved quantifiers. He claimed that there is no uniform schema for BQ: the formulae associated to sentences featuring all monotone increasing ( $M\uparrow$ ) quantifiers are different from those associated to sentences featuring all monotone decreasing ( $M\downarrow$ ) quantifiers. According to Barwise, sentences with mixed quantifiers (some  $M\uparrow$  and some  $M\downarrow$ ) make no

---

<sup>1</sup> A 2-place Mostowskian Quantifier [11] (see also [13]) is a symbol  $Q$  such that, if  $x$  is an individual variable and  $\Psi, \Phi$  are formulae then  $Q_x(\Psi, \Phi)$  is also a formula. Semantically,  $Q$  denotes, in every model  $M$  with universe  $A$ , a function  $q$  which takes in input two subsets  $B$  and  $C$  of  $A$  and returns a truth-value. Mostowskian Quantifiers are cardinality quantifiers, in the sense that  $q(B, C)$  depends only on the cardinalities of the sets  $(B \cap C)$ ,  $(B \setminus C)$ ,  $(C \setminus B)$  and  $(A \setminus (B \cup C))$ . Some examples are

- $\|All_x(P_1(x), P_2(x))\|^M = true$  iff  $|( \|P_1(x) \wedge \neg P_2(x)\|^M )| = 0$
- $\|Few_x(P_1(x), P_2(x))\|^M = true$  iff  $|( \|P_1(x) \wedge P_2(x)\|^M )| > \eta$

sense from a linguistic point of view.

On the other hand, Sher [13], [14] observed that since the semantics of linearly ordered quantification is provided regardless to monotonicity, there seems to be no methodological reason for imposing further constraints in case of partially ordered quantification. In other words, even if readings from NL are not available, this should not exclude their logical interpretation.

Sher specified the semantics of BQ on the basis of a precise definition of the involved groups, according to so-called *maximality conditions*; roughly, her claim is that the interpretation of a BQ reading with quantifiers of any type corresponds to the one of Barwise for  $M\uparrow$  quantifiers augmented with a maximality condition requiring that the involved sets are maximal with respect to the body of the formula. Consider the two following sentences:

- (3) (i) Most of the dots and most of the stars are all connected by lines.  
(ii) Few of the dots and few of the stars are all connected by lines.

In Sher's logic (let us name it  $L_0$ ) sentences in (3) are associated with formulas of the following form:

$$(4) \quad \exists P_1, P_2 [ C1 : Q1_x(dot(x), P_1(x)) \wedge \\
C2 : Q2_y(star(y), P_2(y)) \wedge \\
IN : \forall_{xy} [(P_1(x) \wedge P_2(y)) \rightarrow conn(x, y)] \wedge \\
Max(\langle P_1, P_2 \rangle, IN) ]$$

where  $Q1$  and  $Q2$  are the Mostowskian quantifiers corresponding to the determiners in our example:  $Q1=Q2=Most$  for (3.i); and  $Q1=Q2=Few$  for (3.ii). The symbols  $C1$ ,  $C2$ ,  $IN$  are labels on the subformulae and  $Max(\langle P_1, P_2 \rangle, IN)$  is an abbreviation for a maximality condition that states that two sets  $P_1$  and  $P_2$  are maximal with respect to the formula with label  $IN$ , in the sense that there are no strict supersets of  $P_1$  and  $P_2$  that satisfy  $IN$ . Formally, the maximality condition in (4) is the following formula:

$$Max(\langle P_1, P_2 \rangle, IN) \Leftrightarrow \\
\forall P'_1, P'_2 [ \forall_{xy} [ (P_1(x) \wedge P_2(y)) \rightarrow (P'_1(x) \wedge P'_2(y)) \wedge \\
(P'_1(x) \wedge P'_2(y)) \rightarrow conn(x, y) ] \rightarrow \\
\forall_{xy} [ (P'_1(x) \wedge P'_2(y)) \rightarrow (P_1(x) \wedge P_2(y)) ] ]$$

Sher generalizes the schema of (4), so that it applies to any partially ordered set of arbitrary quantifiers. To achieve this, it is necessary to existentially quantify  $n$ -ary *generalized Skolem functions*  $H_i$  rather than simple sets  $P_i$ , and to assert maximality conditions also on the subformulae with label  $C_i$ . Here, an  $n$ -ary Skolem function is just an  $n + 1$ -ary relation  $H$  – we will write  $H(x_1, \dots, x_{n+1})$  if  $x_1 \dots x_{n+1}$  stand in the relation  $H$ , but also write  $H(x_1 \dots x_n)$  for the set of objects  $x_{n+1}$  s.t.  $H(x_1, \dots, x_{n+1})$ . Consider now a branching reading such as in the following sentence:

- (5) Few men inserted a coin in three coffee machines.

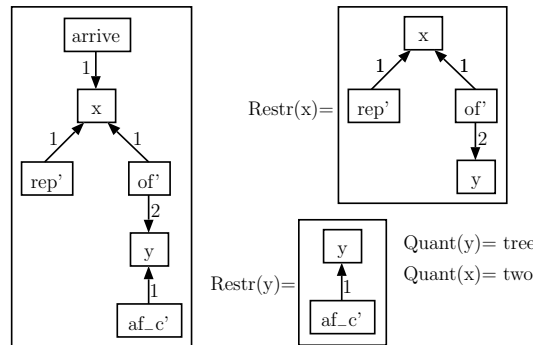
$$\begin{aligned}
& \left. \begin{array}{l} Few_x(\text{man}'(x)) \\ Three_y(\text{CoffeeMach}'(y)) \end{array} \right\} A_z(\text{Coin}'(z)) \text{ --- } \text{Inserted}'(x, z, y) \\
& =_{df} \exists H_x, H_y, H_z [ C_x: Few_x(\text{man}'(x), H_x(x)) \ \& \\
& \quad C_y: Three_y(\text{CoffeeMach}'(y), H_y(y)) \ \& \\
& \quad C_z: \forall_{xy} [(H_x(x) \wedge H_y(y)) \rightarrow A_z(\text{coin}'(z), H_z(x, y))] \ \& \\
& \quad \text{IN}: \forall_{xyz} [H_z(x, y, z) \rightarrow \text{inserted}'(x, y, z)] \ \& \\
& \quad \text{Max}(\langle H_x, H_y \rangle, C_z) \ \& \ \text{Max}(\langle H_z \rangle, \text{IN}) ]
\end{aligned}$$

In this reading, the quantifier  $A$  depends on both  $Three$  and  $Few$ : there can be a different coin for every pair of a man and a coffee machine. This is reflected by the fact that  $H_z$ , the Skolem function associated with the quantifier  $A$ , is a 2-ary function, while  $H_x, H_y$  are 0-ary Skolem functions (that is, predicates). The formula states that we have to find witnesses  $H_x, H_y$  and  $H_z$  such that  $H_z$  corresponds to the extension of  $inserted'$ , and  $H_x$  and  $H_y$  are maximal sets of individuals  $x$  and  $y$  such that the set of objects  $z$  inserted by  $x$  in  $y$ ,  $H_z(x, y, z)$ , includes at least one coin;  $H_x$  is a set of a "few men" and  $H_y$  contains "three coffee machines". See [14] for the formal details.

### 3 Nested Quantification

A limitation of Sher's logic is that it does not handle the case in which one quantifier occurs in the syntactical restriction of another quantifier. Consider:

- (6) Two representatives of three African countries arrive.



In this example, the quantifier  $Three$  occurs in the syntactic restriction of  $Two$ . This corresponds to the fact that the discourse referent  $y$  occurs in the graph  $RESTR(x)$ . This type of reading cannot be directly represented in Sher's logic. Therefore, we propose to extend her definitions to accommodate for these cases as well. Lack of space does not permit us to state the precise definitions; we will give two examples instead which should illustrate how the definitions work. Before discussing the three possible disambiguations of (6),

we introduce a new abbreviation to increase readability.


If  $\Phi$  is a well formed formula,  $x_1 \dots x_n$  a sequence of discourse referents, and  $S_1, \dots, S_n$  a sequence of predicates, we define:

$$\langle S_1, \dots, S_n \rangle \underset{\max}{\subseteq} \Phi[x_1 \dots x_n] \Leftrightarrow \\ \text{Max}(\langle S_1, \dots, S_n \rangle, \forall x_1 \dots x_n [(S_1(x_1) \wedge \dots \wedge S_n(x_n)) \rightarrow \Phi])$$

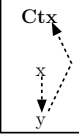
We will omit the reference to the variables  $x_1 \dots x_n$  in the notation when this does not lead to confusion. By using  $\underset{\max}{\subseteq}$ , the formula in (5) can be replaced by the following equivalent

$$\exists H_x, H_y, H_z [ \text{Few}_x(\text{man}'(x), H_x(x)) \ \& \ \text{Every}_y(\text{CoffeeMach}'(y), H_y(y)) \ \& \\ \langle H_x, H_y \rangle \underset{\max}{\subseteq} [ \text{A}_z(\text{coin}'(z), H_z(x, y, z)) \ \& \\ \langle H_z(x, y) \rangle \underset{\max}{\subseteq} \text{inserted}'(x, y, z) ] ]$$


For representing the restriction of quantifiers in the logic, in addition to the Skolem functions  $H_x$  that represent the body of the quantifiers, we introduce *restriction sets*  $\Psi_x$ . The three readings of (6) can now be represented as:



$$\exists H_x, H_y, \Psi_x, \Psi_y [ \text{Two}_x(\Psi_x(x), H_x(x)) \ \& \ \langle H_x \rangle \underset{\max}{\subseteq} (\text{arrive}'(x)) \ \& \\ \langle \Psi_x \rangle \underset{\max}{\subseteq} [ \text{Three}_y(\Psi_y(x, y), H_y(x, y)) \ \& \\ \langle \Psi_y(x) \rangle \underset{\max}{\subseteq} (\text{af\_c}'(y)) \ \& \\ \langle H_y(x) \rangle \underset{\max}{\subseteq} (\text{repr\_of}'(x, y)) ] ]$$



$$\exists H_x, H_y, \Psi_x, \Psi_y [ \text{Three}_y(\Psi_y(y), H_y(y)) \ \& \ \langle \Psi_y \rangle \underset{\max}{\subseteq} (\text{af\_c}'(y)) \ \& \\ \langle H_y \rangle \underset{\max}{\subseteq} [ \text{Two}_x(\Psi_x(y, x), H_x(y, x)) \ \& \\ \langle \Psi_x(y) \rangle \underset{\max}{\subseteq} (\text{repr\_of}'(x, y)) \ \& \\ \langle H_x(y) \rangle \underset{\max}{\subseteq} (\text{arrive}'(x)) ] ]$$



$$\exists H_x, H_y, \Psi_x, \Psi_y [ \text{Two}_x(\Psi_x(x), H_x(x)) \ \& \ \text{Three}_y(\Psi_y(y), H_y(y)) \ \& \\ \langle \Psi_x, H_y \rangle \underset{\max}{\subseteq} (\text{repr\_of}'(x, y)) \ \& \ \langle \Psi_y \rangle \underset{\max}{\subseteq} (\text{af\_c}'(y)) \ \& \\ \langle H_x \rangle \underset{\max}{\subseteq} (\text{arrive}'(x)) ]$$

Let us shortly discuss each of these readings.

In the first reading,  $y$  depends on  $x$ , which is reflected in the fact that  $\Psi_y$  and

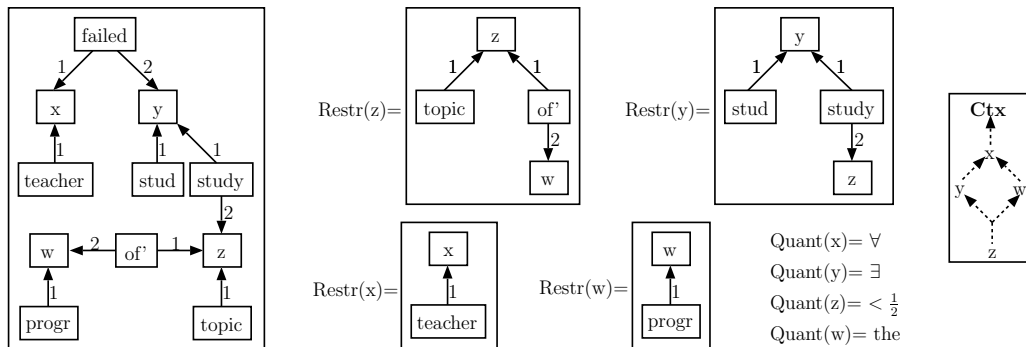
$H_y$  are unary Skolem functions whose values depend on the value for  $x$ . The restriction set of 'three',  $\Psi_y(x)$ , is (for each  $x$ ) the set of all African countries, while  $H_y(x)$  is the set of objects represented by  $x$ . Therefore, the restriction set of 'two',  $\Psi_x$ , is a maximal set of individuals  $x$  that represent three African countries. Two of these individuals must be in  $H_x$  – the set of those that arrive.

In the second reading,  $x$  depends on  $y$ . The set  $\Psi_y$  consists of all African countries. The set  $H_y$  must contain three of these, and it is required that for each element  $y$  in  $H_y$  there are two individuals in the set of all its representatives  $\Psi_x(y)$  that are in  $H_x(y)$ , which consists of all individuals that arrive.

The third formula represents the branching reading of the sentence, in which the two discourse referents do not depend on each other. This formula states that there are sets  $\Psi_x$  and  $H_y$  such that each individual in  $\Psi_x$  represents all elements from  $H_y$  (this is expressed by the maximality condition on the pair  $(\Psi_x, H_y)$ ), and for which it holds that  $H_y$  contains three African countries, and that two of the representatives from  $\Psi_x$  must arrive. In the following, we report a last complex example:

- (7) Every <sub>$x$</sub>  teacher failed two <sub>$y$</sub>  students that studied less than half <sub>$z$</sub>  of the topics in the <sub>$w$</sub>  program.

The following DTS represents a reading of (7) in which the discourse referent  $w$  depends on both  $y$  and  $z$ , and  $y$  and  $z$  depend on  $x$ .



This DTS gets the translation reported below; in this interpretation, the two students and the program depend on a teacher, while the set of topics depends both on a program and on a student. In the formula, the pair of students associated to a teacher  $x \in H_x$  has to belong to the set  $\Psi_y$ , i.e. the set of students  $y$  such that the set of things studied by  $y$ , i.e.  $H_z(x, y, w)$ , contains less than half elements of  $\Psi_z$ , i.e. the set of topic in  $H_w(x)$ , i.e. the program of  $x$ .

$$\begin{aligned}
& \exists H_x, H_y, H_z, H_w, \Psi_x, \Psi_y, \Psi_z, \Psi_w [ \\
& \quad \text{Every}_x(\Psi_x(x), H_x(x)) \ \& \ \{\Psi_x\}_{\max}^{\subseteq} (\text{teacher}'(x)) \ \& \\
& \quad \{H_x\}_{\max}^{\subseteq} [ \text{The}_w(\Psi_w(x, w), H_w(x, w)) \ \& \ \{\Psi_w(x)\}_{\max}^{\subseteq} (\text{progr}'(w)) \ \& \\
& \quad \quad \text{Two}_y(\Psi_y(x, y), H_y(x, y))] \ \& \ \{H_y(x)\}_{\max}^{\subseteq} (\text{failed}'(x, y)) \ \& \\
& \quad \quad \{\Psi_y(x), H_w(x)\}_{\max}^{\subseteq} [ \text{Lth}_z(\Psi_z(x, y, w, z), H_z(x, y, w, z)) \ \& \\
& \quad \quad \quad \{\Psi_z(x, y, w)\}_{\max}^{\subseteq} (\text{topic}'(z) \wedge \text{of}'(z, w)) \ \& \\
& \quad \quad \quad \{H_z(x, y, w)\}_{\max}^{\subseteq} (\text{stud}'(y) \wedge \text{study}'(y, z))] ] ] ]
\end{aligned}$$

## 4 Conclusions and further works

In this paper, a comparison between Dependency Tree Semantics and Sher's work on Branching Quantification and Generalized Quantifiers has been presented. In particular, we have shown how disambiguated DTS structures can be related to formulae of an extension of the formalism from [14] to represent branching quantification. This provides a way to model-theoretically interpret disambiguated DTS structures. Concerning further work, one of the next steps in research on DTS will be extending its expressivity in order to deal with cumulativity, which is a topic that has received very little attention in recent studies on underspecification. Cumulative readings arise from a different kind of branching quantification, as argued in [13], so the step for including them is more natural in DTS than in other underspecified logics that do not take BQ into account.

## References

- [1] Alshawi, H., editor, "The Core Language Engine," Mit Press, Cambridge, MA, 1992.
- [2] Barwise, J., *On branching quantifiers in english*, The Journal of Philosophical Logic (1979), pp. 47–80.
- [3] Copestake, A., D. Flickinger and I. Sag, *Minimal recursion semantics. an introduction*, Technical report, Manuscript, Stanford University (1999).
- [4] Egg, M., A. Koller and J. Niehren, *The constraint language for lambda structures*, J. of Logic, Language and Information **10** (2001), pp. 457–485.
- [5] Henkin, L., *Some remarks on infinitely long formulas*, in: *Finitistic methods, Proc. Symposium of Foundations Math*, Warsaw, 1961, pp. 167–183.

- [6] Hintikka, J., *Quantifiers vs quantification theory*, *Dialectica* (1973), pp. 329–358.
- [7] Lesmo, L. and L. Robaldo, *Dependency tree semantics and underspecification*, in: *Proc. Int. Conf. On Natural language processing (ICON2004)*, Hyderabad, India, 2004.
- [8] Lesmo, L. and L. Robaldo, *From dependency tree semantics to fol*, in: *Proc. 6th Workshop on Computational Semantics (IWCS-6)*, Tilburg, 2005, pp. 384–386.
- [9] Lesmo, L. and L. Robaldo, *Underspecification of quantifier scope in mtt*, in: *Proc. 2th Int.Conf. on Meaning Text Theory*, Moscow, 2005.
- [10] Melcuk, I., *Semantics and the lexicon in modern linguistics.*, in: A. Gelbukh, editor, *In Proc. of the 1st International Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*, 2000, pp. 6–18.  
URL [www.CICLing.com](http://www.CICLing.com)
- [11] Mostowski, A., *On a generalization of quantifiers.*, *Fundamenta Mathematicae* 44 (1957), pp. 12–36.
- [12] Reyle, U., *Dealing with ambiguities by underspecification: Construction, representation and deduction*, *Journal of Semantics* (1993), pp. 123–179.
- [13] Sher, G., *Ways of branching quantifiers*, *Linguistics and Philosophy* (1990), pp. 393–422.
- [14] Sher, G., *Partially-ordered (branching) generalized quantifiers: a general definition*, *The Journal of Philosophical Logic* (1997), pp. 1–43.



# Controlled Language for Geographical Information System Queries

Sela Mador-Haim, Yoad Winter, and Anthony Braun

*Technion I.I.T*

*selam@cs.technion.ac.il, winter@cs.technion.ac.il,  
tonyb@geofocus.co.il*

---

## Abstract

Natural language interfaces to spatial databases have not received a lot of attention in computational linguistics, in spite of the potential value of such systems for users of Geographical Information Systems (GISs). This paper presents a controlled language for GIS queries, solves some of the semantic problems for spatial inference in this language, and introduces a system that implements this controlled language as a novel interface for GIS.

---

## 1 Introduction

Geographical Information Systems (GISs) are information systems for processing of data that pertain to spatial or geographic coordinates [14]. Even though GISs are enjoying a rapidly growing users community, the current systems are often difficult to use or require a long learning process [13]. In the GIS literature [15,16,5,8], it has been well-acknowledged that natural language interfaces (NLIs) would significantly enhance the exploitation of the more complex features of GISs, yet despite the potential value of NLIs for GISs, the work on this subject has so far been rather limited [16]. To the best of our knowledge, existing NLIs for GISs are limited in scope and expressive power and lack the ability to express complex relationships over spatial entities. Some works ([9,17,12]) have demonstrated NLIs using databases that contain geographically related data. Those databases, however, lack any actual spatial information (e.g. geometric polygons representing buildings), and therefore do not deal with the problem of inferring spatial relations from such representations.

In general, the design of NLIs to databases is regarded as a difficult problem since human interaction is often vague, ambiguous or highly contextualized [15,1]. The approach we take in this paper is to avoid many of these problems by designing a system that uses a controlled language for GIS queries. Such controlled languages [10,11], which are based on fragments of English, can be

designed in a way that minimizes the use of vague, ambiguous and context-dependent expressions, while maintaining the ability to express very complex queries in a language that is a subset of English. We benefit from the fact that GISs are a closed, well-defined domain, which enables us to focus on data independent parts of the language. We show that the addition of data dependent portions can be done semi-automatically and requires very low effort.

Our implementation of an NLI for GISs involves four major tasks: first, defining the data independent lexicon, which was done using simple applicative categorial grammar (Ajdukiewicz-Bar-Hillel calculus). Second, we develop a suitable semantic representation for GIS queries, which we call  $\lambda SQL$ , and a method to translate natural-language queries via  $\lambda SQL$  into spatially-enabled SQL. The third task is defining of the semantics of spatial relations (esp. prepositions) in the lexicon in accordance with the intuitive understanding of such relations, which involves tackling certain aspects of spatial prepositions that were never dealt with before. The fourth task is the development of methods to add the data dependent portion of the lexicon with minimal effort, including an automatic tool that generates lexical entries from the actual geographical database in use.

The paper is organized as follows: Section 2 introduces  $\lambda SQL$  and describes the translation scheme from natural language into SQL queries. Section 3 reviews the architecture of the lexicon. Section 4 discusses semantic issues concerning spatial relations in natural languages. Section 5 presents our implementation, and section 6 concludes.

## 2 A compositional approach for building SQL queries

SQL is a recursive language in the sense that it allows using one query as part of an expression within another query. However, due to its complex syntax, the construction of an SQL query in a compositional way from a query in natural language is far from being a straightforward task. One way to tackle this problem is by using an intermediate representation [4,10]. While such an intermediate language avoids the complications of composing SQL queries directly, its downsides are the additional translation phase it requires and the fact that such intermediate languages are usually not as expressive as the target language.

We introduce an intermediate representation language, which we call  $\lambda SQL$ . This language only adds the necessary “compositional glue” to SQL. As a result, only a simple translation process is necessary to convert  $\lambda SQL$  queries into normal SQL syntax.  $\lambda SQL$  expressions are basically expressions in the simply typed  $\lambda$  Calculus with the addition of syntactic sugar for SQL-like syntax.

The typical syntax of a *select* SQL-command for querying a database is:

SELECT < *selectlist* > FROM < *tablelist* > WHERE < *whereclause* >;

The *selectlist* parameter is usually a list of fields to be displayed, but it also allows other expressions such as aggregate functions (e.g. field summation). The *tablelist* parameter is a list of tables to query and *whereclause* is a boolean expression that restricts the rows in the query.

The syntax of  $\lambda SQL$  is very close to that of an SQL *whereclause*, with the addition of  $\lambda$  operators. The atoms of  $\lambda SQL$  are real numbers, strings and typed identifiers. The base types in  $\lambda SQL$  are: *t* - Boolean, *r* - real numbers, *str* - strings, *g* - spatial data and *e* - entries in the database. These base types correspond to the base types that are found in GIS databases, with the addition of one additional type, *e*, for database entries (tuples). Complex expressions are built from atomic ones using function application  $exp1(exp2)$ , infix operators  $exp1\ op\ exp2$ , and the operators  $\lambda v.exp$  and  $\exists v.exp$ . The infix operators in  $\lambda SQL$  correspond to SQL operators, and include Boolean AND/OR, arithmetic operators (+, -, \*, /) and comparators (>, <, =, <=, >=, !=). One additional important operator in  $\lambda SQL$  is the dot operator, as in  $var.fieldname$ , where *var* is of type *e* and *fieldname* is a function from entries in the database to entities of a basic type (i.e. it is of type *et*, *er* or *es*). A dot expression is equivalent to  $fieldname(var)$ , a function that returns the value of a field of a given entry.

In general, the only two syntactic elements in  $\lambda SQL$  that do not correspond directly to SQL syntax are the  $\lambda$  and  $\exists$  operators. Translation from  $\lambda SQL$  expressions to SQL queries is done by recursive traversal over the expression. During traversal, whenever certain patterns are recognized, these patterns are replaced by a corresponding SQL *select* statement. Each  $\lambda$  operator corresponds to a *select* statement, which can be nested inside another *select*. In addition to  $\lambda$  operators, three different synthetic elements may affect the translation pattern:

- P1** A function over a  $\lambda$  expressions, as in  $f(\lambda v.exp)$ , is treated as an aggregate function.
- P2** In the simplest pattern, the type of the variable *x* in  $\lambda x$  is *e*, and it corresponds to a query that returns a set of entries. When the variable that the  $\lambda$  operator binds is of any other base type, the pattern:  $\lambda x.\exists y.(x = exp1\ AND\ y.layer = "layer1"\ AND\ exp2)$  is expected, which is translated into *SELECT exp1 FROM layer1 WHERE exp2*.
- P3** Any additional  $\exists$  operator which is not part of the pattern above is translated as a table join (where *tablelist* parameter contains more than one query). For example, the expression  $\lambda x_e.\exists y_e.(x.layer = "layer1"\ AND\ y.layer = "layer2"\ AND\ exp)$  is translated into: *SELECT x.\* FROM layer1 AS x, layer2 AS y WHERE exp*. Each additional  $\exists$  adds an additional table to the list.

The translation process is guaranteed to be successful due to constraints over the  $\lambda SQL$  expressions in the lexicon that enforce conformity to the above patterns. As an example for  $\lambda SQL$ , consider the following fragment from our lexicon:

| Word      | Category            | Semantics                                                                                                                                      |
|-----------|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------|
| buildings | $N$                 | $\lambda x_e.(x.layer_{es} = "building")$                                                                                                      |
| with      | $N \setminus N / N$ | $\lambda n1_{et}.\lambda n2_{et}.\lambda x_e.(n1(x) \text{ AND } n2(x))$                                                                       |
| more than | $Rs / R$            | $\lambda n_r.\lambda x_r.(x > n)$                                                                                                              |
| two       | $R$                 | 2                                                                                                                                              |
| floors    | $N \setminus Rs$    | $\lambda p_{rt}.\lambda x_e.p(x.floors_{er})$                                                                                                  |
| highest   | $N / N$             | $\lambda n_{et}.\lambda x_e.(n(x) \text{ AND } (x.height_{er} = \max_{(rt)r}(\lambda r_r.\exists y_e.(n(y) \text{ AND } r = y.height_{er}))))$ |

Category  $R$  in the above table corresponds to type  $r$  and  $Rs$  corresponds to the type  $(rt)$ .

The natural language expression “buildings with more than two floors” will be parsed into the  $\lambda SQL$  expression:  $\lambda x_e.(x.layer_{es} = "building" \text{ AND } x.floors_{er} > 5)$ . Note that while functional applications during parsing eliminated most  $\lambda$  operators, the  $\lambda$  operator that is introduced by the lexical entry for *buildings* is not eliminated. This remaining  $\lambda x_e$  is used to describe a query over a variable  $x$ . In order to generate an SQL query, however, one additional piece of information is required: the name of a table to query. This information is provided via the *layer* keyword (layers, or feature sets in GIS terminology, are equivalent to tables in general databases). While usually the *fieldname* following the dot operator is a name for an actual field in the database (such as *floors* in the above example), *layer* is a virtual attribute in  $\lambda SQL$ , used to associate a layer with a variable. Whenever an expression such as  $x.layer_{es} = "building"$  is found, the parser associates  $x$  with the table “building”, and hence the above expression is translated into the SQL query:

```
SELECT x.* FROM building AS x WHERE x.floors>5;
```

A bit more complex example is the query “highest buildings”, which is translated into:  $\lambda x_e.(x.layer_{es} = "building" \text{ AND } x.height_{er} = \max_{(rt)r}(\lambda r_r.\exists y_e.(y.layer_{es} = "building" \text{ AND } r = y.height_{er})))$ . This expression demonstrates several features of  $\lambda SQL$ . Note that *max* is a free identifier, which is expected to be a name of an SQL function. The function *max* receives a  $\lambda$  expression, and is therefore interpreted as an aggregate function. Finally, the expression in the argument of *max* fits pattern P2 above, and the result is:

```
SELECT x.* FROM building AS x WHERE x.floors=(SELECT max(y.floors)
FROM building);
```

### 3 Lexicon architecture

The data independent part of the lexicon is the core of our controlled language. This is the part of the lexicon that involves general logical and spatial operators that do not depend on the actual GIS. By carefully selecting the data-independent lexical items, we are able to express very complex queries while avoiding vagueness and ambiguity problems that often undermine the usability of NLI. An important part of our work is the ability to express spatial relations between GIS objects. However, non-spatial lexical items are an important part of the lexicon as well. In the first part of this section we describe the non-spatial items in the lexicon. In the following part we review the spatially-related lexical items. Finally we present classes of data-dependent lexical items.

#### 3.1 Non-spatial lexical items

Non-spatial lexical items can be partitioned into the following groups:

- Measure units, such as *meters*, *kilometers*, *miles*, *acres*. The lexical definition for these items converts any unit into standard units (e.g. metric units).
- Numerical predicates, such as *less than n*, *at least n*, *between n and m*. Numerical predicates represent sets of real numbers.
- Superlatives: *biggest*, *smallest*, *most*, *least*. The words *most* and *least* can be used to refer to the maximal or minimal value of any numerical field in the database. Other words such as *largest* and *longest* are used as abbreviation for “most area” and “most length”.
- Boolean connectives: *and*, *or*, *not*.
- Other lexical entries: *that*, *which*, *is*, *are*, *with*, *without*, *have*.

#### 3.2 Spatial lexical items

As mentioned before, we aim to design a controlled language that would avoid the pitfalls of vagueness and context-dependent ambiguity. In order to satisfy this requirement, we need to avoid vague qualitative relations such as *near*, *far* and *almost*. Another type of relations that need to be avoided are *projective* relations such as *in front of*, *behind*, *left* and *right*. The meaning of these prepositions involves context-dependent[6] elements that are hard to handle within a controlled language.

The following spatial relations are included in the lexicon:

- Intersectional relations, following Egenhofer’s 9-intersection model [3]: *in*, *outside of*, *borders*, *overlaps*, *crosses*, *contains* and *intersects*. Note that only the first two expressions are prepositions, while the others are verbs.
- Distance: the word *from* is used to specify exact distance, as in “200m from a lake”.
- Constructors: *intersection of*, *border of* and *center of*. These words are used to refer to spatial entities that do not exist in the database, but can be derived from existing objects. For example, assuming “42nd Street” and “Broadway” are objects in the database, “the intersection of 42nd street and Broadway” can be constructed by intersecting the geometrical representations of the two streets.
- Relative orientation: *north of*, *south east of* and the 3-place relation *between* are all used to describe the orientation of one object relative to another object (or objects, as in the case of *between*).
- Superlatives: *closest* and *furthest* are spatially-related superlatives.

### 3.3 Data-dependent lexical items

Data dependent lexical items are lexical items that refer to specific data inside the database and may therefore change from one data set to another. GIS data are divided into separate thematic feature classes or layers, whereby each layer consists of one type of geometrical entity such as a building, street or utility pole. For each layer there is usually an associated set of attributes that represent non-spatial data attached to real world geometric objects. These may be boolean data, numeric data or strings. Examples for such attributes are the number of floors in a building or the name of a street. String values such as street names should be part of the lexicon as well.

Data-dependent items are represented in the lexicon in the form of templates, which are lexical items with parametrized values for layer name, attribute name and attribute value. An example for such a template is:

”#strval”  $N/N\{l = \#layer\} \lambda n.\lambda x.(n(x) \text{ AND } (x.\#attr \text{ like } \#strval))$

The ”#strval” template defines lexical items that refer to strings inside the database. The lexical analyzer searches the database for strings that match lexical tokens that are not present in the lexicon. For each such string the above template is instantiated with the relevant layer name, attribute name and string value. Similar templates are used for layer names and attributes of various types. In case the lexical entries need to be different than the actual names in the database, a definition file is used to add those lexical items and instantiate the relevant templates for those items. No knowledge in  $\lambda SQL$  is required in order to edit the definition file.

## 4 Semantics of spatial prepositions

While some progress was made in semantic theories of prepositional phrases in recent years [18,7], certain aspects of spatial linguistic phenomena have not been extensively treated in the semantic literature, but are nevertheless crucial for interfaces to spatial databases. Two such aspects that are treated in our system and are discussed below.

### 4.1 Eigenspace vs. Existential semantics

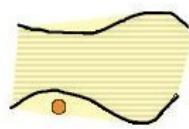
While previous work on prepositional semantics mainly dealt with relationships between two distinct objects, GIS queries often correspond to relationships between sets of objects. Consider the query “buildings that are up to 200m from a lake”. In case there is more than one lake, we expect the system to return any building such that there is at least one lake up to 200m from it. In other words, it appears like the query existentially quantifies over lakes. On the other hand, if we change the query to “buildings that are at least 200m from a lake”, we would expect the system to return buildings that are over 200m away from *all* the lakes. The query “buildings that are between 200m and 500m from a lake” has a yet more complex semantics, and should result in any building such that there is at least one lake less than 500m from it and there is *no* lake less than 200m from it.

The semantics of the above three queries becomes much clearer, however, when instead of interpreting the indefinite “a lake” as a quantifier (existential, universal or other) over the lakes in the database, “a lake” is interpreted as the set of all lakes, and distance is measured with respect to the space taken by the union of all lakes. We refer to this kind of interpretation for indefinites as *eigenspace semantics*. In SQL, the eigenspace of a set of objects is evaluated by using the aggregate function *GeomUnion* over a set of objects, as in:

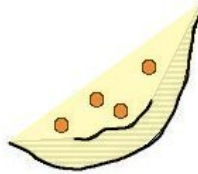
```
SELECT geomunion(x.the_geom) FROM lake AS x;
```

In our framework, eigenspace semantics is treated by enabling a type-shifting from an indefinite noun-phrase into a special category  $G$  used for representing the eigenspace. The  $\lambda SQL$  expression for  $G/N$  type-shifting is:  $\lambda n. geomunion(\lambda g. \exists x. (n(x) \text{ AND } g = x.the\_geom))$  where *the\_geom* is the attribute for the geometrical data of an object in GIS database. The  $\lambda SQL$  expression for the preposition *from*, of category  $((N \setminus N) \setminus RS) / G$ , is then defined by:  $\lambda g. \lambda p. \lambda n. \lambda x. (n(x) \text{ AND } p(distance(x.the\_geom, g)))$ .

It is important to note that while eigenspace semantics are used for spatial prepositions, in the case of other spatial relations that are not expressed using prepositions, such as the verbs *contains* and *intersects*, an indefinite is treated in the usual way, as an existential quantifier. For example, if we ask about “towns that contain a building with more than 10 floors”, the eigenspace semantics would mean finding a town than contains *all* buildings with more than one floor, whereas we expect to get any town that contains at least one building



**A**



**B**

Fig 1. Example for between



Fig 2. Query result in QGIS

with more than 10 floors. We achieve the correct semantics in this case by providing a  $\lambda SQL$  expression for verbs such as *contains* that existentially quantifies over the set of contained objects:  $\lambda n1.\lambda n2.\lambda x.\exists y.(n1(y) \text{ AND } n2(y) \text{ AND } \textit{contains}(x.the\_geom, y.the\_geom))$ .

#### 4.2 Semantics of between

An additional aspect of spatial relations that has so far been ignored in the semantic literature concerns the relations between non-convex objects. A fundamental spatial relation which is quite problematic in the context of non-convex objects is the 3-place relation *between*.

Zwarts and Winter [18] suggest the following definition for *between*:  $X$  is between  $Y$  and  $Z$  if  $X \subseteq \textit{convexHull}(Y \cup Z) \setminus Y \setminus Z$ , for convex objects in  $X$ ,  $Y$  and  $Z$ . The problem is that many objects we deal with in the context of GISs are not convex. For example, it could be quite handy to talk about objects between two streets. However, streets are often non-convex shapes. As can be seen in figure 1, the convex hull for two streets represented by the solid lines includes areas that do not agree with our understanding of the expression *between the two streets*. In order to overcome this problem, we suggest the following definition:

**Definition 4.1** Let  $X$ ,  $Y$  and  $Z$  be sets of points. We say that  $X$  is between  $Y$  and  $Z$  iff either there is a point  $x$  on the border of  $Y$  such that the shortest line connecting  $x$  to  $Z$  crosses  $X$ , but does not cross  $Y$ , or there is a point  $y$  on the border of  $Z$  such that the shortest line connecting  $y$  to  $Y$  crosses  $X$ , but does not cross  $Z$ .

The areas between the streets according to Definition 1 are marked by stripes. As can be seen from the illustration, the new definition is more in agreement with our intuitive understanding of *between*. Note that while the above is a strictly spatial definition of *between*, in some contexts people may use *between* in sloppier ways (e.g., Buxton is between Manchester and Sheffield). In our system, however, we wish to avoid the vagueness of such



sloppy usages.

## 5 Implementation

The NLI presented in this paper was implemented in C++. The parser reads the lexicon from a text file that includes the syntactic categories, and the semantics is represented using  $\lambda SQL$  expressions for all data-independent lexical items. Data-dependent items are represented using templates, as explained in section 3.3. When the user enters a natural-language query, the query is parsed using a bottom-up right-to-left tabular Combinatorial Categorical Grammar (CCG) parser that was developed as part of the NLI prototype. The resulting  $\lambda SQL$  expression is then converted into an SQL query as explained in section 2, which is sent to a spatially enabled database engine.

The system presented here uses PostGIS (<http://postgis.refrations.net/>) as a back-end. PostGIS is an open-source GIS extension to the PostgreSQL database engine, which implements the OpenGIS “Simple features specification for SQL” standard [2]. PostGIS basically supplies a set of functions that operate on vector representations, such as a function that calculates distance between polygons. The SQL queries are sent to PostGIS, which generates the result in a form of a table which is loaded into a GIS front-end that supports PostGIS, such as QGIS (<http://www.qgis.org>).

For example, the query “Buildings that are up to 500m from the intersection of Elm street and Oak street” are converted into the SQL query, which generates the result in figure 2:

```
(SELECT x.* FROM building AS x WHERE distance(x.the_geom, intersection((
SELECT GeomUnion(x2.the_geom) FROM street AS x2 WHERE x2.street_name
LIKE 'elm'),(SELECT GeomUnion(x3.the_geom) FROM street AS x3 WHERE
x3.street_name LIKE 'oak'))))<=500)
```

## 6 Conclusions and future work

This work has presented an interface to GISs that is based on a controlled fragment of English. We believe to have demonstrated that it is possible to build such usable interfaces and express quite complex queries using a simple fragment of English. Future work on this subject can be done at several different levels: expanding the lexicon further by adding quantifiers, comparison between attributes of different objects and possibly anaphoric expressions. More thorough theoretical study is required regarding semantic issues such as eigenspace and *between* presented here, and finally, an empirical study is necessary to evaluate how usable such interfaces are for actual GIS users of varying skills and needs. We believe, however, that the general architecture and prototype demo interface that we suggest can be developed into a useful tool for planners and other professional users of GISs.

## References

- [1] I. Androutsopoulos and G. Ritchie. Database interfaces. In R. Dale, H. Moisl, and H. Somers, editors, *Handbook of Natural Language Processing*, chapter 9, pages 209–240. Marcel Dekker Inc., 2000.
- [2] Open Geospatial Consortium. *Simple Features Specification for SQL*. <http://www.opengis.org/docs/99-049.pdf>.
- [3] M. Egenhofer and J. Herring. Categorizing binary topological relations between regions, lines and points in geographic databases. Technical report, Department of Surveying Engineering, University of Maine, Orono, ME, 1991.
- [4] P.P. Filipe and N.J. Mamede. Databases and natural language interfaces. In *JISBD 2000*, pages 321–332, 2000.
- [5] A.U. Frank and D.M. Mark. Language issues for GIS. In D. MacGuire, M.F. Goodchild, and D. Rhind, editors, *Geographical Information Systems: Principles and Applications*, pages 147–163. Wiley, New York, 1991.
- [6] A. HersHKovits. *Language and Spatial Cognition: an interdisciplinary study of the prepositions in English*. Cambridge University Press, Cambridge, 1986.
- [7] M. Kracht. On the semantics of locatives. *Linguistics and Philosophy*, 25:157–232, 2002.
- [8] D.M Mark, S. Svorou, and D. Zubin. Spatial terms and spatial concepts: Geographic, cognitive and linguistic perspectives. In *International Geographic Information Systems (IGIS)*, pages 101–112, Arlington, VA, 1987.
- [9] M. Minock. A phrasal approach to natural language interfaces over databases. In *NLDB-2005*, Alicante, Spain, June 2005.
- [10] R. Nelken and N. Francez. Querying temporal databases using controlled natural language. In *COLING 2000 - Volume 2*, pages 1076–1080, 2000.
- [11] I. Pratt. Temporal prepositions and their logic. *Artificial Intelligence*, 166(1–2):1–36, 2005.
- [12] Mukesh Kumar Rohil. Natural language processing to query a geographic information system(india) knowledgebase. In *Map India*, India, 2000.
- [13] I. Schlaisich and M. Egenhofer. Multimodal spatial querying: What people sketch and talk about. In C. Stephanidis, editor, *1st International Conference on Universal Access in Human-Computer Interaction*, pages 732–736, New Orleans, LA, August 2001.
- [14] J. Star and J. Estes. *Geographic Information System, An Introduction*. Prentice Hall, Englewood Cliffs, NJ, 1990.
- [15] Fangju Wang. Handling grammatical errors, ambiguity and impreciseness in GIS natural language queries. *Transactions in GIS*, 7(1):103–121, 2003.
- [16] H. Wang, A.M MacEachren, and G. Cai. Design of human-GIS dialogue for communication of vague spatial concepts based on human communication framework. In *GIScience 2004*, Adelphi, MD, 2004.
- [17] J.M. Zelle and R.J. Mooney. Learning to parse database queries using inductive logic programming. In *Thirteenth National Conference on Artificial Intelligence*, pages 1050–1055, Portland, OR, August 1996.
- [18] J. Zwarts and Y. Winter. Vector space semantics: a modeltheoretic analysis of locative prepositions. *Journal of Logic, Language and Information*, 9:171–213, 2000.

# Computing relative polarity for textual inference

Rowan Nairn, Cleo Condoravdi, Lauri Karttunen

*Palo Alto Research Center*

*rnairn@gmail.com, condorav@parc.com, Lauri.Karttunen@parc.com*

---

## Abstract

Semantic relations between main and complement sentences are of great significance in any system of automatic data processing that depends on natural language. In this paper we present a strategy for detecting author commitment to the truth/falsity of complement clauses based on their syntactic type and on the meaning of their embedding predicate. We show that the implications of a predicate at an arbitrary depth of embedding about its complement clause depend on a globally determined notion of relative polarity. We, moreover, observe that different classes of complement-taking verbs have a different effect on the polarity of their complement clauses and that this effect depends recursively on their own embedding. A polarity propagation algorithm is presented as part of a general strategy of canonicalization of linguistically-based representations, with a view to minimizing the demands on the entailment and contradiction detection process.

---

## 1 Introduction

In a 1971 article titled “The Logic of English Predicate Complement Constructions” [9] Lauri Karttunen, 29, wrote:

It is evident that logical relations between main sentences and their complements are of great significance in any system of automatic data processing that depends on natural language. For this reason, the systematic study of such relations, of which this paper is an example, will certainly have a great practical value, in addition to what it may contribute to the theory of the semantics of natural languages.

It is only now that this 35-year old prediction is becoming a reality in the context of automated question answering and reasoning initiatives such as the PASCAL Textual Entailment Challenge (see [7]) and the ARDA-sponsored AQUAINT project (see [10], [12], [4]).

Recognizing whether a given piece of text can be strictly or plausibly inferred from, or is contradicted by, another piece of text is, arguably, a minimal

criterion for Natural Language Understanding (see [2]). We call this task LOCAL TEXTUAL INFERENCE. Textual inferences may be based on purely linguistic knowledge, assumptions about language use by collaborative rational agents, knowledge about the world, or any combination thereof. The semantics of complement constructions is an important part of local textual inference. It has the added advantage of carving out a well-circumscribed domain of inferences based primarily on linguistic knowledge.

A system that computes textual inferences should be able to deduce, for example, that (1b) and (1c) follow from (1a).

- (1) a. Ed forgot to close the door.
- b. Ed intended to close the door.
- c. Ed did not close the door.

There is a clear difference between the two embedding predicates *forget to* and *intend to*. (1c) does not follow from (1b). A speaker or author of (1b) may well BELIEVE in the truth of (1c) but he is not COMMITTED to it by virtue of having said (1b). In the following we focus on cases where the author’s commitment to the truth of a complement clause arises solely from the larger sentence it belongs to, leaving aside other sources of information about the beliefs of the author. The author of (1a) is committed to both (1b) and (1c) but due to different aspects of the meaning of *forget to*, as we will show shortly.

The fact that forgetting to do something entails not doing it does not arise solely from the meaning of the verb *forget* but depends also on the type of its complement. Consider the difference between *forget to* and *forget that*.

- (2) a. Ed forgot that the door was closed.
- b. The door was closed.

(2a) commits the author to the view that the complement (2b) is true rather than false. Furthermore, with *forget that* this commitment is preserved under negation and in questions.

- (3) a. Ed did not forget that the door was closed.
- b. Did Ed forget that the door was closed?

(2a), (3a) and (3b) are alike in committing the speaker to (2b). The difference between *forget that* and *forget to* is striking.

- (4) a. Ed did not forget to close the door.
- b. Did Ed forget to close the door?

In contrast to (1a), in a narrative text (4a) commits the author to the view that Ed closed the door, the opposite of (1b).<sup>1</sup> (4b) is noncommittal either way.

The different semantic behaviors of *forget that* and *forget to* have been known for a long time. There is a large body of linguistic literature, start-

---

<sup>1</sup> In a spoken dialogue it is of course possible, typically with a special intonation, to use (4a) to contradict (1a): *Ed didn’t “forget” to close the door. He never intended to do it.*

ing with Kiparsky & Kiparsky 1971 [11] and Karttunen 1971 [8], about FAC-TIVE constructions such as *forget/remember/know/... that* and IMPLICATIVE constructions such as *forget/remember/manage/bother/... to*. A common view is that factive constructions PRESUPPOSE rather than ENTAIL that the complement sentence is true.<sup>2</sup> Implicative constructions have entailments and some of them also carry presuppositions. For example, (1a) entails (1c) and presupposes (1b). (4a) carries the same presupposition as (1a) but the opposite entailment. While the entailments of implicative constructions are generally quite clear, it is often difficult to pin down exactly what is being presupposed. It may be argued, for example, that (1b) is too specific. Maybe the presupposition is more vague: *Ed ought to have closed the door or Ed was expected to close the door*. All the examples in (5) entail that Ed did not open the door but presuppose a different reason for this fact.

(5) Ed didn't manage/bother/dare/happen to open the door.

In this paper we focus on building a partial computational semantics for implicative constructions ignoring for the time being the presuppositional aspects of their meaning. However, we handle simple factive constructions and the interaction between implicative and factive verbs. The work was carried out in the context of the AQUAINT project using the XLE engine for parsing and semantic analysis.<sup>3</sup> The AQUAINT project conducted a PASCAL-like experiment on local textual inferences based on a more nuanced task. Given a sentence *A*, we may conclude either that *B* is TRUE or that *B* is FALSE or that the answer is UNKNOWN, that is, *B* or its negation cannot be inferred from *A* alone. In contrast, the PASCAL test collapses FALSE and UNKNOWN into FALSE.<sup>4</sup>

We faced two initial challenges. The first is that there are several types of implicative verbs. Some yield an entailment in both affirmative and negative environments but there are others, “one-way implicatives”, that yield entailments only in one or the other environment. Furthermore, the entailment may be either positive or negative depending on the polarity of the environment. For example, *forget to* yields a negative entailment in a positive environment, (1a), and a positive entailment in a negative environment, (4a). But *manage to* works in the opposite way. This type of semantic information is not available in or deducible from any public lexical database such as WordNet, VerbNet or FrameNet. We had to compile ourselves a table of “implication signatures” for a large class of complement-taking constructions.

The second challenge is that implicative and factive constructions may be stacked together. The polarity of the environment of an embedding predicate is determined relatively to the chain of predicates or sentential operators it is in the scope of. Although it may not be obvious at the first glance, (6)

<sup>2</sup> This is not to say that there is a common view on how the notion of presupposition should be construed theoretically.

<sup>3</sup> <http://www2.parc.com/istl/groups/nltt/xle/>

<sup>4</sup> For a critical look at the PASCAL task, see Zaenen, Karttunen and Crouch [12].

commits the author to the view that Ed did not open the door.

(6) Ed didn't manage to remember to open the door.

In 6 *remember* is in a positive clause but the relative polarity of that clause is negative. The computation of relative polarity must be a recursive process.

## 2 Implication signatures

We focused on complement-taking verbs, especially those that take infinitival or *that* complements. Taking the verbs in order of decreasing frequency in the British National Corpus (BNC),<sup>5</sup> we determined their natural implications (if any). Judgments were based on agreement by multiple annotators using resources such as Google search and the Linguist's Search Engine to sample the relevant constructions in the wild. In particular cases it can be difficult to decide between ENTAILMENTS, that is, what the author is actually committed to, and CONVERSATIONAL IMPLICATURES, that is, what a reader/hearer may feel entitled to infer. For example, *Ed did not refuse to participate* might lead the hearer to conclude that Ed participated. But the speaker could continue with *He was not even eligible* indicating the opposite. For this reason we classify *refuse to* as a one-way implicative. Of the 1250 relevant verbs in our lexicon we classified 400 on a first pass. Roughly a third of those carried some kind of implication: a positive or negative entailment, a factive or a counterfactive presupposition. Conversational implicatures were flagged for later attention. Figure 1 shows the classifications of the resulting lookup table.

|                                     | Word in<br>subcat frame                   | Relative Polarity                |                              |
|-------------------------------------|-------------------------------------------|----------------------------------|------------------------------|
|                                     |                                           | (+) positive                     | (-) negative                 |
|                                     |                                           | <b>Entailment</b>                |                              |
| <b>Two-way<br/>implicatives</b>     | <i>manage to</i><br><i>forget to</i>      | (+) positive<br>(-) negative     | (-) negative<br>(+) positive |
| <b>One-way<br/>+implicatives</b>    | <i>force to</i><br><i>refuse to</i>       | (+) positive<br>(-) negative     | none<br>none                 |
| <b>One-way<br/>-implicatives</b>    | <i>attempt to</i><br><i>hesitate to</i>   | none<br>none                     | (-) negative<br>(+) positive |
|                                     |                                           | <b>Presupposition</b>            |                              |
| <b>Factives<br/>Counterfactives</b> | <i>forget that</i><br><i>pretend that</i> | (+) positive<br>(-) negative     | (+) positive<br>(-) negative |
| <b>Neutral</b>                      | <i>want to</i>                            | <b>Entailment/Presupposition</b> |                              |
|                                     |                                           | none                             | none                         |

Fig. 1. Some examples from our verb markup table

<sup>5</sup> <http://www.natcorp.ox.ac.uk/>

### 3 Theoretical and technical prerequisites

Our approach to textual inference relies on parsed text that is further transformed by a process of `CANONICALIZATION`. The mechanism for entailment and contradiction detection (`ECD`) combines structural matching and inference-based techniques. It operates on packed representations, encoding ambiguities, without the need for disambiguation. We will not discuss `ECD` any further here. Instead we will focus on describing in more detail some of the relevant features of the representations on which it operates.

Input text is syntactically analyzed by the `XLE` parser, based on a broad coverage, hand-coded grammar of English. Linguistic semantic representations are constructed from the parse output, using `SKOLEMIZATION` and flattening embedded structures to clausal form. These logical forms are in turn canonicalized to more uniform representations via packed term rewriting as described in Crouch [3]. The implication projection algorithm to be described in the next section forms part of this component of canonicalization and is implemented as a set of recursive rewrite rules that operate on packed representations.<sup>6</sup>

The canonicalized representations that are input to `ECD` are essentially a kind of description logic with contexts.<sup>7</sup> Roughly, each verbal predication corresponds to a constructed concept, an event type with role restrictions. The main concept is provided by a mapping of the verbal predicate to a concept in some background ontology. The role restrictions come from various arguments and modifiers. The constructed concept is named by the skolem introduced by the verbal predicate. Flattening replaces embedded expressions with complex internal structure, such as clausal complements, with atomic first order terms, contexts. The information about the level of embedding of an expression is preserved by associating its content with the corresponding context. Negation and intensional operators also trigger the introduction of new contexts. Contexts thus serve as scope markers since their use enables globally represented information, such as the scope of operators, to be made locally accessible.

The content of the top level context, designated as `t`, represents what the author of the sentence is taken to be committed to. In general, we tie truth of a sentence to the `INSTANTIABILITY` of the skolem corresponding to its head predicate. This, in effect, amounts to the familiar existential closure over events: if the skolem corresponding to a clause’s head predicate denotes an event description, an instantiability declaration for that skolem means that the event description is instantiated. Therefore, an implication that a complement clause is true/false can be construed as an existential/negative existential implication, which in our terms is an implication about the instantiation/non-instantiation of the event type described by the embedded clause.

---

<sup>6</sup> Packing is `XLE`’s mechanism for ambiguity management and operates independently of canonicalization and inference.

<sup>7</sup> For more details see Bobrow *et al.* [1], Crouch [3] and Condoravdi *et al.* [2].

Instantiability is always relative to a context, in the simplest case the context of origin of the skolem. In order to become author commitment, an instantiability declaration has to be associated with the top level context  $\tau$ . When two contexts stand in certain relations to one another, in particular the relations of veridicality and antiveridicality, information can be inherited from one to another. Lifting rules lift assertions from a lower context to a higher context, either as they are, when the two contexts are veridical to one another, or by switching the polarity of instantiability assertions, when the two contexts stand in an antiveridical relation. Negation introduces a context that is antiveridical with respect to the immediately higher context. To illustrate, (7) gives the contextual structure for a negative sentence like *Ed didn't leave Paris* and (8) the corresponding instantiability assertions (*leave\_ev57* is the name for the constructed event type of Ed leaving Paris). One important thing to note is that the assertion `instantiable(leave_ev57)` in `not58` is lifted as `uninstantiable(leave_ev57)` to the top level context  $\tau$ , thus capturing the intuitive meaning that the event type of Ed leaving Paris was not instantiated.

- (7)     `context( $\tau$ )`  
           `context(not58)` *new context triggered by negation*  
           `context_relation(not  $\tau$  not58)`  
           `antiveridical(not58  $\tau$ )` *interpretation of negation*
- (8)     `not58:   instantiable(leave_ev57)`  
           `$\tau$ :   uninstantiable (leave_ev57)` *entailment of negation*

Lexical entailments and presuppositions are similarly overtly spelled out in the representations operated on by ECD. This way the process of canonicalization prepackages some of the local textual inferences. The challenge of course is to figure out which context the relevant instantiability assertions ought to be lifted to, which is what the implication projection algorithm determines.

## 4 The implication projection algorithm

Aside from the onerous task of classifying hundreds of verbs, the complications of this problem stem from the interaction of multiple embedded clauses. As mentioned previously, the entailment yielded by a complement-taking construction is dependent on the polarity of the context it appears in. This polarity in turn is not locally determined but dependent on the embedding structure of contexts. Therefore, a verb in a negative clause is not necessarily in a negative environment since the negativity of a *not* may be neutralized by another negative, as for example in (9).

- (9) Ed refused not to attempt to leave.

Here the normal negative entailment licensed by *not attempt* is neutralized by the negative polarity setting due to the higher-level predicate *refuse*. Notice



that *refuse* does not simply negate the entailment. It cancels it entirely. Embedding within a verb such as *refuse* can also license entailments that were not available previously. Consider (10a), which is compatible with either (10b) or (10c).

- (10) a. Ed attempted to leave.  
 b. Ed left.  
 c. Ed didn't leave.

(11), on the other hand, implies (10c).

- (11) Ed refused to attempt to leave.

Evidently, it is not enough to look at the immediate outer context of a complement construction. The polarity of any context depends on the sequence of potential polarity switches stretching back to the top context. Each complement-taking verb, operating on its parent context's polarity, either switches, preserves or simply sets the polarity for its embedded context, as specified by an entry in the lookup table.

Furthermore, this means that polarity is a relative notion. If the sequence of polarity switches was started at a level below the top context then the final polarity value might turn out different. Thus when we talk about the polarity of a context we mean polarity relative to some ancestor context. Normally, it is the top context which interests us the most, but it may be useful to infer the implications of a clause for other contexts. For example, it is probably useful to infer (12b) from (12a). The algorithm provides for this generality.

- (12) a. John believes that Ed managed to leave.  
 b. John believes that Ed left.

Every context  $C$  then has associated with it a set of ancestor contexts relative to which its polarity is positive (denoted  $\oplus_C$ ) and a set of contexts relative to which its polarity is negative (denoted  $\ominus_C$ ). Every context, including the top one, is positive relative to itself. The polarity sets of a context are computed in terms of its parent's sets ( $\oplus_{p(C)}$  and  $\ominus_{p(C)}$ ) with reference to the verb ( $V_{p(C),C}$ ) which links the two contexts and its signature in the lookup table ( $sig^e(V_{p(C),C})$ ) where the environment superscript  $e$  is either positive  $+$  or negative  $-$ .

$$\oplus_C =_{def} \{C\} \cup \begin{cases} \oplus_{p(C)} & \text{if } sig^+(V_{p(C),C}) = + \\ \ominus_{p(C)} & \text{if } sig^-(V_{p(C),C}) = + \\ \emptyset & \text{otherwise} \end{cases}$$

$$\ominus_C =_{def} \begin{cases} \oplus_{p(C)} & \text{if } sig^+(V_{p(C),C}) = - \\ \ominus_{p(C)} & \text{if } sig^-(V_{p(C),C}) = - \\ \emptyset & \text{otherwise} \end{cases}$$

Figure 2 shows the example sentence *Ed did not forget to force Dave to leave* parsed and with relative polarities assigned to each context. To get to this

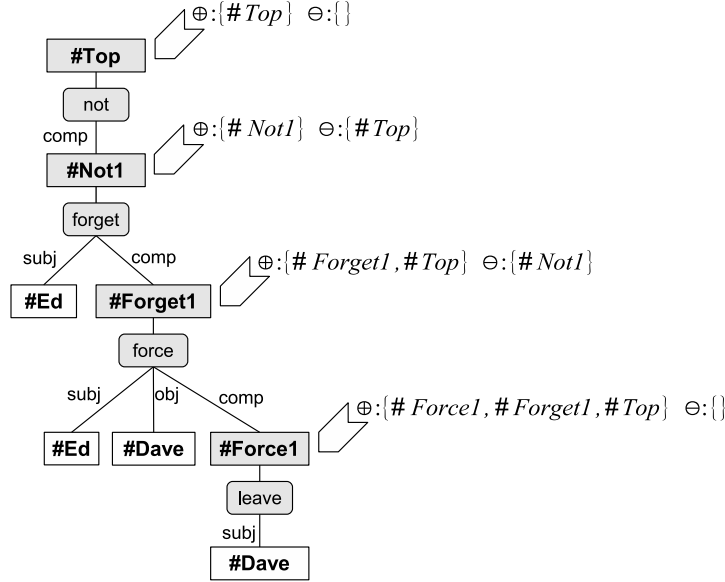


Fig. 2. After the polarity propagation pass

situation the algorithm first assigns the top context the polarity sets  $\{\#Top\}$  and  $\emptyset$ . It then recursively computes the polarity sets for each embedded context using the context-linking verb as an index to the lookup table. *Not* is treated in the same way as *forget to* – they both invert the polarity sets. *Force* is a one-way implicative that disregards the negative polarity set of its parent.

Recall that we needed to work out which concepts should be instantiated in which contexts and, now that we have marked the contexts appropriately with relative polarities, we can extract that information. The head event skolem of a context, and presumably all its role fillers, should be made instantiable not only in the context it arises in but also in all contexts relative to which its originating context has positive polarity. Similarly, an event should be made uninstaniatable in all contexts relative to which its originating context has negative polarity.

$$\begin{aligned} \text{instantiables}(C) &=_{\text{def}} \{\text{head}(C') \mid C \in \oplus_{C'}\} \\ \text{uninstaniatables}(C) &=_{\text{def}} \{\text{head}(C') \mid C \in \ominus_{C'}\} \end{aligned}$$

From the polarity marking in Figure 2 we can conclude that the event concept corresponding to the sentence *Dave left* is in fact instantiable at the top level (as well as in the  $\#Force$  and  $\#Forget$  contexts) and thus we can attribute it as a commitment of the speaker.

## 5 Conclusion and Further Work

The present study is, as far as we know, the first systematic implementation of textual inferences arising from the six types of implicative verbs presented in Figure 1 and their interaction with factive verbs.

In this work we have focused on cases where the judgement of whether the author is committed to the truth or the falsity of a complement clause can be made reliably from the sentence in question. Further work is needed at least in the following three areas.

**Lexicographic gaps.** In our classification we only considered simple verbal and adjectival complements. We have yet to study and determine the semantics of complement constructions associated with nominals in collocations such as *take the trouble to*, *have the foresight to*, *take time to*, for which there is virtually no literature.

**Conversational implicatures.** It is well known that constructions such as *be able to* yield a negative entailment in a negative environment. *Ed was not able to open the door* entails *Ed did not open the door*. There is no entailment in the corresponding affirmative sentence. Yet, if the author writes *Ed was able to open the door* and says nothing to indicate that the door was not opened, the reader is likely to infer, and justifiably so, that Ed opened the door. This kind of CONVERSATIONAL IMPLICATURE is cancelable (Grice [6]). It is not a contradiction to say *Ed was able to open the door but he kept it closed*. If a student asks his professor *Did you have the time to read my paper?* and the professor answers *Yes* but has not read the paper, the answer can be literally true and very misleading at the same time.<sup>8</sup>

**Degrees of “factivity”.** Factive verbs and constructions do not constitute a uniform class. Looking at the pattern of usage of verbs such as *mention that*, *report that*, *say that*, etc. on Google, we observed that in cases such as *He did not mention that Coalition allies now plan to leave* it was virtually always clear from the context that the author believed the complement to be true. The verb *report* is similar to *mention* but there are also cases where *...did not report that X* was meant to suggest that *X* is false. On the other hand, *...did not deny that X* suggests that *X* is true, whereas *...denied that X* is noncommittal with respect to *X*.

## Acknowledgements

This material is based in part on work funded by the U.S. Government, and any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the U.S. Government.

---

<sup>8</sup> For a seminal paper on INVITED INFERENCES, see [5].

## References

- [1] Bobrow, D., C. Condoravdi, R. Crouch, R. Kaplan, L. Karttunen, T. King, V. de Paiva and A. Zaenen, *A basic logic for textual inference*, in: *Proceedings of the AAAI Workshop on Inference for Textual Question Answering*, Pittsburgh, PA, 2005, <http://www2.parc.com/istl/groups/nlitt/papers/textual-inference.pdf>.
- [2] Condoravdi, C., R. Crouch, R. Stolle, V. de Paiva and D. Bobrow, *Entailment, intensionality and text understanding*, in: *Proceedings of the Workshop on Text Meaning, Human Language Technology Conference (HLT-NAACL-2003)*, Edmonton, Canada, 2003, <http://www2.parc.com/spl/members/stolle/Papers/condoravdi-textmeaning.pdf>.
- [3] Crouch, R., *Packed rewriting for mapping semantics to KR*, in: *Proceedings of the Sixth International Workshop on Computational Semantics*, Tilburg, the Netherlands, 2005, [http://www2.parc.com/istl/groups/nlitt/papers/iwcs05\\_crouch.pdf](http://www2.parc.com/istl/groups/nlitt/papers/iwcs05_crouch.pdf).
- [4] Crouch, R., R. Sauri and A. Fowler, *AQUAINT pilot knowledge-based evaluation: Annotation guidelines* (2005), [http://www2.parc.com/istl/groups/nlitt/papers/aquaint\\_kb\\_pilot\\_evaluation\\_guide.pdf](http://www2.parc.com/istl/groups/nlitt/papers/aquaint_kb_pilot_evaluation_guide.pdf).
- [5] Geis, M. and A. Zwicky, *On invited inferences*, *Linguistic Inquiry* **2** (1971), pp. 561–565.
- [6] Grice, H. P., *Logic and conversation*, in: P. Cole and J. L. Morgan, editors, *Speech Acts*, Academic Press, New York, NY, 1989 pp. 41–58.
- [7] Ido Dagan, O. G. and B. Magnini, *The PASCAL recognising textual entailment challenge*, in: *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*, Southampton, U.K., 2005, [http://www.cs.biu.ac.il/~glikmao/rte05/dagan\\_et\\_al.pdf](http://www.cs.biu.ac.il/~glikmao/rte05/dagan_et_al.pdf).
- [8] Karttunen, L., *Implicative verbs*, *Language* **47** (1971), pp. 340–358.
- [9] Karttunen, L., *The logic of English predicate complement constructions* (1971), distributed by the Indiana University Linguistics Club. [http://www2.parc.com/istl/members/karttune/publications/english\\_predicate.pdf](http://www2.parc.com/istl/members/karttune/publications/english_predicate.pdf).
- [10] Karttunen, L. and A. Zaenen, *Veridicity*, in: G. Katz, J. Pustejovsky and F. Schilder, editors, *Annotating, Extracting and Reasoning about Time and Events*, number 05151 in Dagstuhl Seminar Proceedings (2005), <http://drops.dagstuhl.de/opus/volltexte/2005/314>.
- [11] Kiparsky, P. and C. Kiparsky, *Fact*, in: D. Steinberg and L. Jakobovits, editors, *Semantics. An Interdisciplinary Reader*, Cambridge University Press, Cambridge, England, 1971 .
- [12] Zaenen, A., L. Karttunen and R. Crouch, *Local textual inference: can it be defined or circumscribed?*, in: *Workshop on the Empirical Modeling of Semantic Equivalence and Entailment*, Ann Arbor, MI, 2005, <http://www2.parc.com/istl/members/karttune/publications/acl2005workshop.pdf>.

# Using Answer Set Programming in an Inference-Based approach to Natural Language Semantics

Farid Nouioua

LIPN UMR 7030 du C.N.R.S.  
Institut Galilée – Univ. Paris-Nord  
93430 Villetaneuse – FRANCE  
nouiouaf@lipn.univ-paris13.fr

Pascal Nicolas

LERIA  
University of Angers  
2, bd Lavoisier F-49045 Angers cedex  
pascal.nicolas@univ-angers.fr

## 1. Motivation

The traditional tri-partition syntax/semantics/pragmatics is commonly used in most of the computer systems that aim at the simulation of the human understanding of Natural Language (*NL*). This conception does not reflect the flexible and creative manner that humans use in reality to interpret texts. Generally speaking, formal *NL* semantics is referential i.e. it assumes that it is possible to create a static discourse universe and to equate the objects of this universe to the (static) meanings of words. The meaning of a sentence is then built from the meanings of the words in a compositional process and the semantic interpretation of a sentence is reduced to its logical interpretation based on the truth conditions. The very difficult task of adapting the meaning of a sentence to its context is often left to the pragmatic level, and this task requires to use a huge amount of common sense knowledge about the domain. This approach is seriously challenged (see for example [4][14]). It has been showed that the above tri-partition is very artificial because linguistic as well as extra-linguistic knowledge interact in the same global process to provide the necessary elements for understanding. Linguistic phenomena such as polysemy, plurals, metaphors and shifts in meaning create real difficulties to the referential approach of the *NL* semantics discussed above. As an alternative solution to these problems, [4] proposes an inferential approach to the *NL* semantics in which words trigger inferences depending on the context of their apparition. In the same spirit we claim that understanding a *NL* text is a reasoning process based on our knowledge about the norms<sup>1</sup> of its domain i.e. what we generally expect to happen in normal situations. But what kind of reasoning is needed for natural language semantics?

The answer to this question is based on the remark that texts seldom provide normal details that are assumed to be known to the reader. Instead, they focus on abnormal situations or at least on events that cannot be inferred by default from the text by an ordinary reader. A central issue in the human understanding of *NL* is the ability to infer systematically and easily an amount of implicit information necessary to answer indirect questions about the text. The consequences resulting from truth-based entailments are logically valid but they are poor and quite limited. Those obtained by a norm-based approach are defeasible: they are admitted as long as the text does not mention explicit elements that contradict them. However they provide richer information and enable a deeper understanding of the text. That is why the norm-based reasoning must be non-monotonic. In addition to this central question, the representation language must take into account a number of modalities (including the temporal aspect) that are very useful to answer different questions on *NL* texts.

The next section gives a general logical framework to represent in a first order language the necessary knowledge about a domain and allows non-monotonic reasoning. Section 3 shows how to implement our representation language fragment in the formalism of Answer Set Programming by transforming them into extended logic programs. In section 4, we discuss the use of our language in the car crash domain to find automatically the cause of an accident from its textual description. The

---

<sup>1</sup> In A.I., the word norm is commonly used in the « normative » sens. Here, it is rather used in the « normal » sens.

kind of inference rules required in this application is showed through a detailed presentation of the analysis of a text from the corpus we are using. Finally, we conclude and give some perspectives for future work in section 5.

## 2. Knowledge representation language

The explicit information evoked in a given text provides the starting point for the reasoning process that aims to understand it. Thus, the first task to do is to extract from the text this explicit information and to represent it in an adequate language. The richness and flexibility of *NL* constrains the representation language to take into account a number of aspects whose necessity and importance may vary from an application to another. In what follows, we describe a logical language which enhances within the first order framework some aspects that we believe to be useful in an inferential approach to *NL* semantics. Namely, the proposed language allows the representation of time, modalities and non-monotonic inferences (see [7] for more details).

### 2.1 Reification

The first idea that comes to mind when representing knowledge about *NL* statements is to use first order predicates to express properties of objects, agents ...etc. However we need often to treat further aspects. For example, we need to represent modalities on the considered properties and to reason about them i.e. to use the predicate names themselves as variables over which one can quantify in order to avoid the use of ad hoc inference rules, i.e. to factorise the rules at an adequate level of abstraction. To solve this problem within the framework of first order logic, we use the reification technique, commonly used in Artificial Intelligence (*AI*). Instead of writing  $P(X, Y)$  to express the fact that property  $P$  applies to arguments  $X$  and  $Y$ , we write  $Holds(P, X, Y)$ . The property name  $P$  becomes then an argument in the new predicate  $Holds$ . i.e.  $P$  will be a variable over properties and it can be quantified in inference rules.

The use of the reification technique yields to two main drawbacks: first, it forces a fixed arity for the predicate  $Holds$  whereas properties in general may have a different number of arguments. The second problem is the necessity to redefine ad hoc axioms about the properties (negation, conjunction, disjunction... of properties). One possible solution to the first problem is to consider a special binary function *combine* which constructs a new “complex” argument from two other arguments. For example, as the predicate  $Holds$  has three arguments then, the predicate  $Q(X, Y, Z)$  can be reified as :  $Holds(combine(Q, X), Y, Z)$ <sup>2</sup>. In general, this corresponds well to linguistic practice: for example the application of a transitive verb to its complement can be considered as a unique “complex” property comparable to an intransitive verb. Concerning the second problem, it turns out that in practice we often do not need all the axioms but only some particular ones. So we have to represent only those axioms that we really need in the application considered.

### 2.2 Representing time

Generally, narrative texts describe events that take place in a time perceived as continuous. The temporal aspect is crucial in their understanding. Two representation approaches are possible for time: either we represent the continuous time which reflects the physical reality and use the elegant mathematical tools developed for mechanics, or we represent the discrete time which reflects the text structure and which corresponds rather to a naive physics. We chose the second approach, because generally, texts are written by persons who ignore the mathematical details of motion, and they can be understood without having such knowledge. Two approaches are still possible for a discrete model of time. Either we use a linear model in which only the events that happened in reality are represented, or we take into account the unrealized futures as part of the temporal

---

<sup>2</sup> As a concrete example, the ternary predicate  $bump(A, B, T)$  (vehicle  $A$  bumps vehicle  $B$  at time  $T$ ) is written after reification and by using the *combine* function as :  $Holds(combine(bump, B), A, T)$ . The term  $combine(bump, B)$  expresses then the complex property of « bumping the vehicle  $B$  ».

information. In this case, we use a branching time model [5][10]. This last model is richer than the former and can be very useful in some cases. In this paper we are interested only on the linear model. What is important for us in time modelling is to establish an order between the events evoked in the text. Of course, this choice limits the use of our language to applications which do not need deeper structure of time but it remains useful in practice (see section 4 for a possible application). Indeed, the unrealized futures are not completely excluded in our model, as they can be represented implicitly by modalities (see the modality *able* in section 4.2.2).

The semantics used for time in our model is situated somehow between an interval-based and a point-based semantics: the scene of the accident described in the text is decomposed as a succession of ordered time elements. Each time element is denoted by an integer representing its order number. This integer is used as an argument in the predicates. The meaning of the element depends on the nature of the property. If it is a persistent property, the time parameter denotes the entire time interval during which this property remains true (interval based semantics). If the property is not persistent (corresponds to an action or a punctual event) then the temporal argument denotes the starting point of the interval on which the property occurs and causes at least one persistent property to change its truth value.

### 2.3 Modalities

Modalities express properties of the predicates other than their truth value, which can be considered as a null modality. Different types of modal logics have been developed to formalize the reasoning about modalities. Even though the reasoning we want to apply on texts makes use of modalities, it can be carried out without developing new modal logics with ‘complete’ axiomatizations. What we really need is to represent the modalities as first order predicates using the reification technique discussed in section 2.1., and to define only useful axioms as inference rules. For example, to represent the fact that the modality *Mod* is applied to the predicate *P* having  $X_1, \dots, X_n$  as arguments we write :  $Mod(P, X_1, \dots, X_n)$  instead of the classical notation :  $Mod P(X_1, \dots, X_n)$ .

### 2.4 Non-monotonicity

Non-monotonicity is an essential characteristic of the nature of the reasoning used by humans to understand texts. Among the different approaches proposed in the literature to formalise this variant of commonsense reasoning, we have used Reiter’s default logic [11] to represent our inference rules. The fixed point semantics used to compute the default theories extensions seems to be adequate to the nature of the *NL* understanding process. Indeed, as discussed in section 1, the *NL* understanding process cannot be decomposed in a sequence of separate steps but it consists in the simultaneous satisfaction of several linguistic as well extra-linguistic constraints in a manner that can be approached by the search of some fixed point of the meaning of the given text.

Two kinds of inference rules are considered: the strict inferences represented by material implications and the defeasible ones represented by Reiter’s defaults. To facilitate the implementation of our rules on the answer set programming paradigm (see section 3) we limit their forms as follows:

Let  $A_1, \dots, A_n, B, C_1, \dots, C_k$  be first order literals.

The Expression (1) is a material implication. It means that  $B$  is inferred whenever  $A_1, \dots, A_n$  are verified. Two kinds of default rules are considered. The first form (2) corresponds to a “normal” default. It means that if we have  $A_1, \dots, A_n$  then, we can infer  $B$  as long as this is consistent. The second one (3) corresponds to a semi-normal default and its meaning is that in general, when we have  $A_1, \dots, A_n$  then, we can infer  $B$  as long as this is consistent and none of  $\neg C_i$  ( $i=1..k$ ) belongs to the extension<sup>3</sup>. Semi normal defaults are particularly useful to establish a priority order between inference rules which can not be done using only normal defaults[12].

---

3 We use a notation in which  $A : B$  stands for  $\frac{A : B}{B}$  and  $A : B[C]$  stands for  $\frac{A : B, C}{B}$

$$A_1 \wedge \dots \wedge A_n \rightarrow B \quad (1)$$

$$A_1 \wedge \dots \wedge A_n : B \quad (2)$$

$$A_1 \wedge \dots \wedge A_n : B[C_1, \dots, C_k] \quad (3)$$

### 3. Implementation by Answer Set Programming

#### 3.1. Theoretical backgrounds

Answer Set Programming (*ASP*) is a recent paradigm covering different kinds of logic programs, and associated semantics. It allows representing and solving various problems in Artificial Intelligence. On one hand, we can cite combinatorial problems as k-coloring graph, path finding, timetabling, ... On another hand, *ASP* is also concerned by problems arising when available information is incomplete as non-monotonic reasoning, planning, diagnosis, ... The non familiar reader will find additional information about *ASP* on the web site of the working group *WASP* (<http://wasp.unime.it/>).

In the present work we are particularly interested in using *ASP* as a framework for default reasoning. For this we use Extended Logic Programs (*ELP*) to represent knowledge by means of rules containing positive information and strong or default negative information and we interpret them by answer set semantics [3]. Formally, an *ELP* is a set of rules of the form

$$c \leftarrow a_1, \dots, a_n, \text{not } b_1, \dots, \text{not } b_m. \quad n \geq 0 \text{ and } m \geq 0$$

where  $c$ ,  $a_i$  and  $b_j$  are literals.

For a given rule  $r$ , we denote

$$\text{head}(r) = c \quad \text{body}^+(r) = \{a_1, \dots, a_n\} \quad \text{body}^-(r) = \{b_1, \dots, b_m\} \quad r^+ = c \leftarrow a_1, \dots, a_n$$

**Definition** Let  $R$  be a set of rules without default negation ( $\forall r \in R, \text{body}^-(r) = \emptyset$ ),  $R$  is called a Definite Logic Program. A literal set  $X$  is closed wrt  $R$  when  $\forall r \in R, \text{body}^+(r) \subseteq X \Rightarrow \text{head}(r) \in X$ . The set of consequences of  $R$  is  $C_n(R)$  the minimal literal set that is closed wrt  $R$  consistent or equal to the whole set of literals of the language

For a given literal set  $A$  and an *ELP*  $P$ , the reduct of  $P$  by  $A$  is the definite Logic Program

$$P^A = \{r^+ \mid r \in P \text{ and } \text{body}^-(r) \cap A = \emptyset\}$$

**Definition** Let  $P$  be an *ELP* and  $A$  a literal set.  $A$  is an answer set of  $P$  if and only if  $A = C_n(P^A)$

#### Examples

$P1 = \{a \leftarrow \text{not } b., b \leftarrow \text{not } a., \neg c \leftarrow b.\}$  has two answer sets  $\{a\}$  and  $\{b, \neg c\}$

$P2 = \{a \leftarrow \text{not } a.\}$  has no answer set at all.

We have recalled the basic notions of answer set semantics only in the case of propositional rules. But, obviously, for a more flexible knowledge representation, rules may contain variables. In this case, a rule is considered as a global schema for the set of fully instantiated rules that can be obtained by replacing every variable by every constant in the language.

#### Example

$P = \{\text{bird}(1)., \text{bird}(2)., \text{penguin}(2)., \text{fly}(X) \leftarrow \text{bird}(X), \text{not } \text{penguin}(X)., \neg \text{fly}(X) \leftarrow \text{penguin}(X).\}$  is equivalent to the program  $P' = \{\text{bird}(1)., \text{bird}(2)., \text{penguin}(2)., \text{fly}(1) \leftarrow \text{bird}(1), \text{not } \text{penguin}(1)., \neg \text{fly}(1) \leftarrow \text{penguin}(1)., \text{fly}(2) \leftarrow \text{bird}(2), \text{not } \text{penguin}(2)., \neg \text{fly}(2) \leftarrow \text{penguin}(2).\}$

Then,  $P$  (formally  $P'$ ) has one answer set  $\{\text{bird}(1), \text{bird}(2), \text{penguin}(2), \text{fly}(1), \neg \text{fly}(2)\}$ .

Let us mention an important point for our work that is answer set semantics for *ELP* can be viewed as a subcase of default logic [2][3]. By translating every rule  $r = c \leftarrow a_1, \dots, a_n, \text{not } b_1, \dots, \text{not } b_m$  into the default rule :

$$T(r) = a_1 \wedge \dots \wedge a_n : c \text{ [}\neg b_1, \dots, \neg b_m\text{]}$$



By this way :

If  $S$  is an answer set of an *ELP*  $P$ , then  $Th(S)$  is an extension of the default theory  $(\emptyset, T(P))$  every extension of  $(\emptyset, T(P))$  is the deductive closure of one answer set of  $P$ .

Obviously, in whole generality every default theory cannot be translated into an *ELP*. But as we explain it later, it is possible to encode some restricted default theories in an *ELP*. By this way it is possible to envisage realistic applications of default reasoning since several software packages for *ASP* are available today, e.g. the following ones:

*DLV*[8] <http://www.dbai.tuwien.ac.at/proj/dlv>,

*Smodels* [13] <http://www.tcs.hut.fi/Software/smodels>

*Cmodels* [9] <http://www.cs.utexas.edu/users/tag/cmodels.html>

*Nomore++*[1] <http://www.cs.uni-potsdam.de/wv/nomore++>

### 3.2. From Default Logic to ASP

Here, we explain how we have encoded our knowledge base that is originally a default theory, into an extended logic program. A very important point to note is that our original knowledge base does not contain disjunctions. Since a default theory is a pair consisting in a set of classical formulas and a set of default rules, we distinguish two major translations.

| <i>classical formulas</i>                                          | <i>ELP</i>                                                                                                                                                                                          |
|--------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| one fact : $a$                                                     | one rule with an empty body : $a$ .                                                                                                                                                                 |
| a conjunction of $n$ facts : $a_1 \wedge \dots \wedge a_n$         | $n$ rules with empty bodies: $a_1, \dots, a_n$ .                                                                                                                                                    |
| a material implication $a_1 \wedge \dots \wedge a_n \rightarrow b$ | one direct rule $b \leftarrow a_1, \dots, a_n$ .<br>and $n$ contrapositive rules :<br>$\neg a_1 \leftarrow \neg b, a_2, \dots, a_n$ .<br>...<br>$\neg a_n \leftarrow \neg b, a_1, \dots, a_{n-1}$ . |
| <i>default rules</i>                                               | <i>ELP</i>                                                                                                                                                                                          |
| $A_1, \dots, A_n : B$<br>$A_1, \dots, A_n : B[C_1, \dots, C_k]$    | $b \leftarrow a_1, \dots, a_n, \text{not } \neg b$ .<br>$b \leftarrow a_1, \dots, a_n, \text{not } \neg b, \text{not } \neg c_1, \dots, \text{not } \neg c_m$ .                                     |

We have preferred to encode firstly our rules in default logic instead using directly *ASP* because default logic is more compact than *ASP*, which needs more rules, especially for contrapositives. The translation of default logic into *ASP* can be easily automated.

## 4. From the description of an accident to its cause

### 4.1. The corpus

We are working on a sample of 60 representative texts of a larger corpus. These texts are short descriptions of car accident circumstances. They are written (in French) by persons implied in the accidents to be sent to their insurance company<sup>4</sup>. The length of our texts varies between 9 and 167 words. They contain 129 sentences whose length varies between 4 and 55 words; the longest report has 7 sentences and there are 24 reports that contain only one sentence. The total number of word occurrences is 2256. But there are only 500 distinct words corresponding to 391 dictionary entries.

<sup>4</sup> We are grateful to the MAIF insurance company for having given us access to the reports that constitute our corpus.

## 4.2. Our task

### 4.2.1. Finding the cause of the accident

The objective of the system we are developing is to find automatically the cause of an accident from its textual description. Because of the very controversial nature of causality we must define more precisely our objective. We are interested in our study by the interventionist conception of causality in which voluntary actions are privileged as potential causes of events. This is in correspondence with the practical use of causality in *AI*. Moreover, we claim that the most plausible causes for abnormal situations like accidents are those that reflect violation of norms (anomalies)[6]. We consider that the system has understood a text if it finds the same cause as the one given by an ordinary human reader. We have then determined manually the cause of each text and we have used this information to validate the results of the system.

Two essential steps are considered in the overall architecture of the system. The first one “the linguistic step” applies a tagger and syntactical analyser to extract a set of surface relations between words. These relations are then progressively transformed by an adequate “linguistic reasoning” into the so-called “semantic predicates” which express the explicit information provided by the text. The semantic predicates are represented in a “semantic language” as the one discussed in section 2. This part of the system, which is under construction, tries to adapt existing methods to deal with the problems of anaphora resolution and time ordering of the events described in a text. We will not discuss the details of the linguistic step in this paper. The second step: “the semantic step” applies a set of strict and default inference rules based on norms of the road domain to enrich the semantic predicates initially extracted from the text by further semantic predicates enhancing implicit information. The inference rules are designed manually and reflect rudimentary reasoning that any reader of the text makes systematically. This semantic reasoning process stops as soon as the system infers the necessary information that characterizes an anomaly. Section 5 gives further details about the semantic reasoning through an example taken from the corpus.

### 4.2.2. Some specificities

The majority of the semantic predicates used in our system have the form:  $Holds(P, A, T)$  where  $P$  is a simple or a complex property (expressed by the binary function combine),  $A$  is an agent (generally a vehicle involved in the accident) and  $T$  is the order number of a time interval during which (or at the beginning of which) the property  $P$  holds (to simplify, we will say henceforth that property  $P$  holds at time  $T$ ). For example  $Holds(stop, ag, 3)$  means that the agent ‘ $ag$ ’ is stopped at time 3 and  $Holds(combine(follows, ag_1), ag_2, 2)$  means that at time 2, agent ‘ $ag_2$ ’ follows agent ‘ $ag_1$ ’ (in a file of vehicles). When needed a function  $neg$  is applied to a property to have its negation. We introduce the rule (4)

$$Holds(neg(P), A, T) \leftrightarrow \neg Holds(P, A, T) \quad (4)$$

The main modalities that we use in our system cope respectively with duties and capacities :

$must(P, A, T)$  means that at time  $T$ , agent  $A$  has the duty to achieve the property  $P$ .

$able(P, A, T)$  means that at time  $T$ , agent  $A$  is able to achieve the property  $P$ . In terms of branching time, this means that there is some possible future in which  $P$  holds.

The semantic reasoning is designed so that it converges to a “kernel” containing a limited number of semantic predicates<sup>5</sup> in terms of which all possible anomalies can be expressed. In a given text, it is possible that several anomalies coexist. In this case, the system distinguishes between the primary anomaly which can be considered as the most plausible cause of the accident and the other anomalies called “derived anomalies”. A primary anomaly has two forms: either an agent  $A$  has the duty and the capacity to achieve a property  $P$  at a time  $T$  and at time  $T+1$  a property  $P'$  incompatible

---

<sup>5</sup> The predicates of the kernel are :  $Holds(control, A, T)$  [ $A$  has the control of his/her vehicle],  $Holds(moves\_back, A, T)$  [ $A$  moves back],  $Holds(starts, A, T)$  [ $A$  moves off],  $Holds(drives\_slowly, A, T)$  [ $A$  drives fairly slowly],  $Holds(stops, A, T)$  [ $A$  is stopped],  $Holds(comb(disruptive\_factor, X), A, T)$  [ $X$  is a disruptive factor for  $A$ ]

with  $P$  holds (5) or some disruptive and inevitable factor occurs and causes the accident (6). The form of a derived anomaly (7) differs from that of a primary one only on the agent's capacity.

$$\text{primary\_an}(P, A, T) \leftarrow \text{property}(P), \text{vehicle}(A), \text{time}(T), \text{must}(P, A, T), \text{able}(P, A, T), \\ \text{holds}(P', A, T+1), \text{incompatible}(P, P') \quad (5)$$

$$\text{primary\_an}(\text{combine}(\text{disruptive\_factor}, X), A, T) \leftarrow \text{object}(X), \text{vehicle}(A), \text{time}(T), \\ \text{holds}(\text{combine}(\text{disruptive\_factor}, X), A, T) \quad (6)$$

$$\text{derived\_an}(P, A, T) \leftarrow \text{property}(P), \text{vehicle}(A), \text{time}(T), \text{must}(P, A, T), \neg \text{able}(P, A, T), \text{holds}(P', \\ A, T+1), \text{incompatible}(P, P') \quad (7)$$

### 4.3. An example

To illustrate our methodology, let us consider the following text of the corpus (translated into english) and explain the inference rules involved in its analysis :

« *Whereas vehicle B was overtaking me, the driver lost the control of its vehicle. It bumped on the central guardrail , and crossed the ways. It then cut my way. My vehicle A initially bumped on vehicle B on its right side, before being crushed on the guardrail.* »

The set of the semantic predicates extracted from the text are :

$$\text{holds}(\text{overtake}, \text{veh\_b}, 1), \neg \text{holds}(\text{control}, \text{veh\_b}, 2), \\ \text{holds}(\text{combine}(\text{bump}, \text{guardrail}), \text{veh\_b}, 3), \neg \text{holds}(\text{stop}, \text{veh\_b}, 4), \\ \text{holds}(\text{combine}(\text{bump}, \text{veh\_b}), \text{veh\_a}, 5), \text{holds}(\text{combine}(\text{bump}, \text{guardrail}), \text{veh\_a}, 6) \\ \text{vehicle}(\text{veh\_a}), \text{vehicle}(\text{veh\_b}), \text{object}(\text{veh\_a}), \text{object}(\text{veh\_b}), \text{object}(\text{guardrail}).$$

In what follows, we show how the application of inference rules leads to the determination of the primary and the derived anomalies:

Rule(8) states that “*at the starting state 0, each vehicle has the control*”.

$$\text{holds}(\text{control}, A, 0) \leftarrow \text{agent}(A), \text{vehicle}(A) \quad (8)$$

It allows to infer :  $\text{holds}(\text{control}, \text{veh\_a}, 0), \text{holds}(\text{control}, \text{veh\_b}, 0)$

Rule(9) states that “*if B is a vehicle that bumps on A at time T, then B is not stopped at this time*”.

$$\neg \text{holds}(\text{stop}, A, T) \leftarrow \text{vehicle}(A), \text{object}(B), \text{time}(T), \text{holds}(\text{combine}(\text{bump}, B), A, T) \quad (9)$$

It allows to infer:  $\neg \text{holds}(\text{stop}, \text{veh\_b}, 3), \neg \text{holds}(\text{stop}, \text{veh\_a}, 5), \neg \text{holds}(\text{stop}, \text{veh\_a}, 6)$

Rules(10) and (11) state that “*if A is a vehicle that bumps on B at time T, then there is at this time a shock (symmetric) between A and B*”.

$$\text{holds}(\text{combine}(\text{shock}, B), A, T) \leftarrow \text{vehicle}(A), \text{object}(B), \text{time}(T), \text{holds}(\text{combine}(\text{bump}, B), A, T) \quad (10)$$

$$\text{holds}(\text{combine}(\text{shock}, A), B, T) \leftarrow \text{object}(A), \text{object}(B), \text{time}(T), \text{holds}(\text{combine}(\text{shock}, B), A, T) \quad (11)$$

The set of predicates inferred by these rules are :

$$\text{holds}(\text{combine}(\text{shock}, \text{guardrail}), \text{veh\_b}, 3), \text{holds}(\text{combine}(\text{shock}, \text{veh\_b}), \text{guardrail}, 3), \\ \text{holds}(\text{combine}(\text{shock}, \text{veh\_b}), \text{veh\_a}, T), \text{holds}(\text{combine}(\text{shock}, \text{veh\_a}), \text{veh\_b}, T), \\ \text{holds}(\text{combine}(\text{shock}, \text{veh\_a}), \text{guardrail}, T), \text{holds}(\text{combine}(\text{shock}, \text{guardrail}), \text{veh\_a}, T)$$

Rule(12) states that “*if A is implied in two successive shocks at times T and T+1, then we deduce that it lost the control after the first shock (during the time interval T)*”.

$$\neg \text{holds}(\text{control}, A, T) \leftarrow \text{agent}(A), \text{object}(B), \text{object}(C), \text{time}(T), \text{holds}(\text{combine}(\text{shock}, A), B, T), \\ \text{holds}(\text{combine}(\text{shock}, A), C, T+1) \quad (12)$$

It allows to infer:  $\neg \text{holds}(\text{control}, \text{veh\_a}, 5)$

The remainder of information about the control of vehicles  $A$  and  $B$  during the other time intervals are deduced using appropriate rules that handle the persistence of some particular properties. The complete set of

conclusions concerning control is as follows :

$$\begin{aligned} & \text{holds}(\text{control}, \text{veh\_b}, T) \text{ (for } 0 \leq T \leq 1), \neg \text{holds}(\text{control}, \text{veh\_b}, T) \text{ (for } 2 \leq T \leq 6), \\ & \text{holds}(\text{control}, \text{veh\_a}, T) \text{ (for } 0 \leq T \leq 4), \neg \text{holds}(\text{control}, \text{veh\_a}, T) \text{ (for } 5 \leq T \leq 6) \end{aligned}$$

Rule(13) states that “in general if there is a collision between a vehicle A and an object B at time T, then B represents an obstacle for A at time T-1”.

$$\begin{aligned} & \text{holds}(\text{combine}(\text{obstacle}, A), B, T-1) \leftarrow \text{object}(A), \text{vehicle}(B), \text{time}(T), \\ & \text{holds}(\text{combine}(\text{shock}, A), B, T), \text{not } \neg \text{holds}(\text{combine}(\text{obstacle}, A), B, T-1) \end{aligned} \quad (13)$$

We obtain from this rule :

$$\begin{aligned} & \text{holds}(\text{combine}(\text{obstacle}, \text{guardrail}), \text{veh\_b}, 1), \text{holds}(\text{combine}(\text{obstacle}, \text{veh\_a}), \text{veh\_b}, 4), \\ & \text{holds}(\text{combine}(\text{obstacle}, \text{veh\_b}), \text{veh\_a}, 4), \text{holds}(\text{combine}(\text{obstacle}, \text{guardrail}), \text{veh\_a}, 5) \end{aligned}$$

Rules (14) and (15) allows to infer that some obstacles are not predictable. The rule (14) states that “if a vehicle B not controlled represents at time T an obstacle to vehicle A, then this obstacle is not predictable for A at this time T”. Whereas rule (15) states that “in general, if a vehicle B bumps a vehicle A at time T, then B is considered as an unpredictable obstacle for A at time T”.

$$\begin{aligned} & \neg \text{predictable}(\text{combine}(\text{obstacle}, B), A, T) \leftarrow \text{vehicle}(B), \text{vehicle}(A), \text{time}(T), \\ & \text{holds}(\text{combine}(\text{obstacle}, B), A, T), \neg \text{holds}(\text{control}, B, T) \end{aligned} \quad (14)$$

$$\begin{aligned} & \neg \text{predictable}(\text{combine}(\text{obstacle}, B), A, T) \leftarrow \text{vehicle}(A), \text{vehicle}(B), \text{instant}(T), \\ & \text{vrai}(\text{combine}(\text{bump}, A), B, T), \text{not } \text{predictable}(\text{combine}(\text{obstacle}, B), A, T) \end{aligned} \quad (15)$$

By these two rules we can infer :  $\neg \text{predictable}(\text{combine}(\text{obstacle}, \text{veh\_a}), \text{veh\_b}, 4)$ ,  
 $\neg \text{predictable}(\text{combine}(\text{obstacle}, \text{veh\_b}), \text{veh\_a}, 4)$

Rule(16) states that “in general, one must keep the control of one's vehicle ”

$$\begin{aligned} & \text{must}(\text{control}, A, T) \leftarrow \text{vehicle}(A), \text{time}(T), \text{not } \neg \text{must}(\text{control}, A, T), \\ & \text{not } \neg \text{holds}(\text{control}, A, T) \end{aligned} \quad (16)$$

This rule infers :  $\text{must}(\text{control}, \text{veh\_b}, 1)$ ,  $\text{must}(\text{control}, \text{veh\_a}, 4)$

The meaning of rule(17) is that “one must avoid any obstacle”.

$$\begin{aligned} & \text{must}(\text{combine}(\text{avoid}, X), A, T) \leftarrow \text{vehicle}(A), \text{object}(X), \text{time}(T), \\ & \text{holds}(\text{combine}(\text{obstacle}, X), A, T) \end{aligned} \quad (17)$$

This rule infers :  $\text{must}(\text{combine}(\text{avoid}, \text{guardrail}), \text{veh\_b}, 1)$ ,  $\text{must}(\text{combine}(\text{avoid}, \text{veh\_a}), \text{veh\_b}, 4)$   
 $\text{must}(\text{combine}(\text{avoid}, \text{veh\_b}), \text{veh\_a}, 4)$ ,  $\text{must}(\text{combine}(\text{avoid}, \text{guardrail}), \text{veh\_a}, 5)$

Rule(18) states that “in general the duty to avoid an obstacle turns out to the duty to stop (this default is inhibited by a number of situations illustrated in the rule)”

$$\begin{aligned} & \text{must}(\text{stop}, A, T) \leftarrow \text{vehicle}(A), \text{object}(B), \text{time}(T), \text{must}(\text{combine}(\text{avoid}, B), A, T), \\ & \text{holds}(\text{combine}(\text{shock}, B), A, T+1), \text{not } \neg \text{must}(\text{stop}, A, T), \text{not } \text{must}(\text{drive\_slowly}, A, T), \\ & \text{not } \text{holds}(\text{stop}, A, T), \text{not } \text{holds}(\text{combine}(\text{follow}, A), B, T), \text{not } \text{must}(\text{not}(\text{backwards}), A, T-1), \\ & \text{not } \text{must}(\text{not}(\text{move\_off}), A, T-1), \text{not } \neg \text{predictable}(\text{combine}(\text{obstacle}, B), A, T) \end{aligned} \quad (18)$$

We can infer from this rule :  $\text{must}(\text{stop}, \text{veh\_b}, 1)$ ,  $\text{must}(\text{stop}, \text{veh\_a}, 5)$

Rules (19) and (20) are the main rules that allow to infer agent's capacities :

$$\begin{aligned} & \text{able}(P, A, T) \leftarrow \text{vehicle}(A), \text{object}(B), \text{time}(T), \text{action}(\text{Act}), \text{property}(P), \text{pcb}(\text{Act}, P), \\ & \text{available}(\text{Act}, P, A, T) \end{aligned} \quad (19)$$

$$\begin{aligned} & \neg \text{able}(P, A, T) \leftarrow \text{vehicle}(A), \text{object}(B), \text{time}(T), \text{action}(\text{Act}), \text{property}(P), \text{pcb}(\text{Act}, P), \\ & \neg \text{available}(\text{Act}, P, A, T) \end{aligned} \quad (20)$$

they mean that “vehicle A is able to reach property P at time T<sub>n</sub> if and only if there is some action Act which is a “potential cause” for P and which is available for A to reach P at time T (the *contrapositives are omitted*)”.

The occurrences of the relation *pcb* (which abbreviates: potentially caused by) are statically determined and stored in a static database. In our case we have : *pcb(brake, stop)*, *pcb(combine(keep\_state, control)<sup>6</sup>, control)*.

By default, actions are available for agents to reach the corresponding properties. This default inference is inhibited by a number of strict rules. In our case, we obtain :

*available(combine(keep\_state, control), control, veh\_b, 1) (the default is applied)*  
*–available(combine(keep\_state, control), control, veh\_a, 4)<sup>7</sup>*  
*–available(brake, stop, veh\_a, 5)<sup>8</sup>*

From these results it follows :

*able(control, veh\_b, 1), –able(stop, veh\_a, 4), –able(stop, veh\_a, 5).*

The application of rules (5) and (7) we can detect the primary and the derived anomalies :

*primary\_an(control, veh\_b, 1), derived\_an(control, veh\_a, 4), derived\_an(stop, veh\_a, 5)*

Finally, the cause of the accident is expressed by: "the loss of control of vehicle B at time 1"

## 5. Conclusion and perspectives

This paper defends the idea that inferences are at the heart of the problematic of NL semantics. We have showed that the inferences we need to understand natural language are based on our knowledge about the norms of the domain and are non-monotonic since the conclusions of this kind of reasoning are in general defeasible. We proposed a general representation language which takes into account within a first order framework modalities, time and non-monotonicity that are essential aspects in an inferential approach of NL understanding. We presented also how to transform our inference rules into extended logic programs. To illustrate our approach in a practical domain we have used a corpus of 60 short texts describing the circumstances of road accidents. We have used *Smodels* to implement our reasoning system. With about 200 inference rules, the system succeeds to find for each text only one stable model containing the necessary literals which express the primary and the derived anomalies. We have determined manually for each text the answer that we hope to obtain. Thus, the validation criterion is that the system gives for each text the same answer as the predetermined one. The running time varies from a text to another but it does not exceed 30 seconds which is rather encouraging. Many other perspectives of future work are open, among them:

- Analyzing more texts of the same domain in order to verify :
  - The validity of our hypotheses, especially those concerning the relationship between norms and causes and the sufficiency of a linear model of time;
  - that the inference rules have a sufficient degree of generality to be adapted easily to new situations by giving the expected answers for new reports.
  - the adequacy of the proposed representation language to deal with new texts.
- Generalizing the approach to other domains

---

<sup>6</sup> we consider as action the fact of keeping holded some persistent property.

<sup>7</sup> the lost of control because of a shock at time T makes unavailable the action of keeping the control at time T-1.

<sup>8</sup> if a vehicle is not under control, then, any action is unavailable for its driver.

**Acknowledgment.** The authors are indebted to Daniel Kayser for very helpful remarks on previous versions of this text.

## References

- [1] C. Anger, M. Gebser, T. Linke, A. Neumann and T. Schaub. *The nomore++ system*. In C. Baral, G. Greco, N. Leone, and G. Terracina, editors, 8<sup>th</sup> International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'05), 3662 : 422-426. LNAI, Springer-Verlag, Diamante, Cosenza, Italy. 2005.
- [2] N. Bidoit and C. Froidevaux. *General logical databases and programs: Default logic, semantics and stratification*. Information and Computation, 91(1):1554. 1991.
- [3] M. Gelfond and V. Lifschitz. *Classical negation in logic programs and disjunctive databases*. New Generation Computing, 9(3-4):363385. 1991.
- [4] D. Kayser. *Abstraction and natural language semantics*. Philosophical Transactions. R. Soc. Lond. B 358 : 1261-1268. 2003.
- [5] D. Kayser, A. Mokhtari. *Time in a Causal Theory*. Annals of Mathematics and Artificial Intelligence. 22(1-2): 117-138. 1998.
- [6] D. Kayser, F. Nouioua. *About Norms and Causes*. International Journal on Artificial Intelligence Tools. Special Issue on FLAIRS 2004, 14(1-2): 7-23. 2005.
- [7] D. Kayser, F. Nouioua. *Representing Knowledge about Norms*. Proc of the 16<sup>th</sup> European Conference on Artificial Intelligence (ECAI'04), pp. 363-367, Valencia, Spain. 2004.
- [8] N. Leone, G. Pfeifer, W. Faber, T. Eiter, G. Gottlob, S. Perri, and F. Scarcello. *The dlv system for knowledge representation and reasoning*. ACM Transactions on Computational Logic, (to appear). 2006.
- [9] Y. Lierler and M. Maratea. *Cmodels-2: Sat-based answer set solver enhanced to non-tight programs*. 7<sup>th</sup> International Conference on Logic Programming and NonMonotonic Reasoning (LPNMR'04), 2923: 346-350. LNCS, Springer-Verlag, Florida, USA. 2004.
- [10] D.V. McDermott. *A Temporal Logic for Reasoning about Processes and Plans*. Cognitive Science 6: 101-155. 1982.
- [11] R. Reiter. *A Logic for Default Reasoning*, Artificial Intelligence, Special Issue on Nonmonotonic Logic, 13(1-2): 81-132. 1980.
- [12] R. Reiter, G. Crisculo : *On Interacting Defaults*. Proc. of the 7<sup>th</sup> International Joint Conference on Artificial Intelligence. pp. 270-276, Vancouver, Canada. 1981
- [13] T. Syrjänen and I. Niemelä. *The Smodels systems*. Proc. of the 6<sup>th</sup> International Conference on Logic Programming and NonMonotonic Reasoning (LPNMR'01), pp 434-438, Springer-Verlag, Vienna, Austria. 2001.
- [14] t.a.l . Special issue “Compositionnalité”. *Traiteent automatique des langues* 39(1). 1998.

# A Bootstrapping Algorithm for Automatically Harvesting Semantic Relations

**Marco Pennacchiotti**

Department of Computer Science  
University of Rome “Tor Vergata”  
Viale del Politecnico 1  
Rome, Italy  
pennacchiotti@info.uniroma2.it

**Patrick Pantel**

Information Sciences Institute  
University of Southern California  
4676 Admiralty Way  
Marina del Rey, CA 90292  
pantel@isi.edu

---

## Abstract

In this paper, we present Espresso, a weakly-supervised iterative algorithm combined with a web-based knowledge expansion technique, for extracting binary semantic relations. Given a small set of seed instances for a particular relation, the system learns lexical patterns, applies them to extract new instances, and then uses the Web to filter and expand the instances. Preliminary experiments show that Espresso extracts highly precise lists of a wide variety of semantic relations when compared with two state of the art systems.

---

## 1. Introduction

Recent attention to knowledge-rich problems such as question answering [18] and textual entailment [10] has encouraged Natural Language Processing (NLP) researchers to develop algorithms for automatically harvesting shallow semantic resources. With seemingly endless amounts of textual data at our disposal, we have a tremendous opportunity to automatically grow semantic term banks and ontological resources. Methods must be accurate, adaptable and scalable to the varying sizes of domain corpora (e.g., textbooks vs. World Wide Web), and independent or weakly dependent on human supervision.

In this paper we present *Espresso*, a novel bootstrapping algorithm for automatically harvesting semantic relations, aiming at effectively supporting NLP applications, emphasizing two major points that have been partially neglected by previous systems: *generality* and *weak supervision*.

From the one side, *Espresso* is intended as a general-purpose system able to extract a wide variety of binary semantic relations, from the classical *is-a* and *part-of* relations, to more specific and domain oriented ones like *chemical reactants* in a chemistry domain and *position succession* in political texts. The system architecture is designed with generality in mind, avoiding any relation-specific inference technique. Indeed, for each semantic relation, the system builds specific lexical patterns inferred from textual corpora.

From the other side, *Espresso* requires only weak human supervision. In order to start the extraction process, a user provides only a small set of seed instances of a target relation (e.g. *Italy-country* and *Canada-country* for the *is-a* relation.) In our experience, a handful of seed instances, in general, is sufficient for large corpora while for smaller corpora, a slightly larger set is required. To guarantee weakest supervision, *Espresso* combines its bootstrapping approach with a web-based knowledge expansion technique and linguistic analysis, exploiting the seeds as much as possible.

## 2. Relevant Work

To date, most research on lexical relation harvesting has focused on *is-a* and *part-of* relations. Approaches fall into two main categories: pattern- and clustering-based.

Most common are *pattern-based approaches*. Hearst [12] pioneered using patterns to extract hyponym (*is-a*) relations. Manually building three lexico-syntactic patterns, Hearst sketched a bootstrapping algorithm to learn more patterns from instances, which has served as the model for most subsequent pattern-based algorithms.

Berland and Charniak [1] propose a system for *part-of* relation extraction, based on the Hearst approach [12]. Seed instances are used to infer linguistic patterns that, in turn, are used to extract new instances, ranked according to various statistical measures. While this study introduces statistical measures to evaluate instance reliability, it remains vulnerable to data sparseness and has the limitation of taking into consideration only one-word terms.

Improving upon Berland and Charniak [1], Girju et al. [11] employ machine learning algorithms and WordNet [8] to disambiguate *part-of* generic patterns, like [*whole-NP's part-NP*]. This study is the first extensive attempt to solve the problem of *generic relational patterns*, that is, those expressive patterns that have high recall while suffering low precision, as they subsume a large set of instances. In order to discard incorrect instances, Girju et al. learn WordNet-based selectional restrictions, like [*whole-NP(scene#4)'s part-NP(movie#1)*]. While making huge grounds on improving precision/recall, the system requires heavy supervision through manual semantic annotations.

Ravichandran and Hovy [20] focus on efficiency issues for scaling relation extraction to terabytes of data. A simple and effective algorithm is proposed to infer surface patterns from a small set of instance seeds by extracting all substrings relating seeds in corpus sentences. The frequencies of the substrings in the corpus are then used to retain the best patterns. The approach gives good results on specific relations such as *birthdates*, however it has low precision on generic ones like *is-a* and *part-of*. Pantel et al. [17] proposed a similar, highly scalable approach, based on an edit-distance technique, to learn lexico-POS patterns, showing both good performances and efficiency. *Espresso* uses a similar approach to infer patterns, but we then apply refining techniques to deal with various types of relations.

Other pattern-based algorithms include Riloff and Shepherd [21], who used a semi-automatic method for discovering similar words using a few seed examples by using pattern-based techniques and human supervision, KnowItAll [7] that performs large-scale extraction of facts from the Web, Mann [15] and Fleischman et al. [9] who used part of speech patterns to extract a subset of *is-a* relations involving proper nouns, and Downey et al. [6] who formalized the problem of relation extraction in a coherent and effective combinatorial model that is shown to outperform previous probabilistic frameworks.

*Clustering approaches* to relation extraction are less common and have insofar been applied only to *is-a* extraction. These methods employ clustering algorithms to group words according to their meanings in text, label the clusters using its members' lexical or syntactic dependencies, and then extract an *is-a* relation between each cluster member and the cluster label. Caraballo [3] proposed the first attempt, which used conjunction and apposition features to build noun clusters. Recently, Pantel and Ravichandran [16] extended this approach by making use of all syntactic dependency features for each noun. The advantage of clustering approaches is that they permit algorithms to identify *is-a* relations that do not explicitly appear in text, however they generally fail to produce coherent clusters from fewer than 100 million words; hence they are unreliable for small corpora.



### 3. The *Espresso* Algorithm

The *Espresso* algorithm is based on a similar framework to the one adopted in [12]. For a specific semantic binary relation (e.g., *is-a*), the algorithm requires as input a small set of seed instances  $I_s$  and a corpus  $C$ . An instance is a pair of terms  $x$  and  $y$  governed by the relation at hand (e.g., *Pablo Picasso is-a artist*). Starting from these seeds, the algorithm begins a four-phase loop. In the first phase, the algorithm infers a set of patterns  $P$  that captures as many of the seed instances as possible in  $C$ . In the second phase, we define a reliability measure to select the best set of patterns  $P' \subseteq P$ . In phase three, the patterns in  $P'$  are used to extract a set of instances  $I$ . Finally, in phase four, *Espresso* scores each instance and then selects the best instances  $I'$  as input seeds for the next iteration. The algorithm terminates when a predefined stopping condition is met (for our preliminary experiments, the stopping condition is set according to the size of the corpus). For each induced pattern  $p$  and instance  $i$ , the information theoretic scores,  $r_\pi(p)$  and  $r_i(i)$  respectively, aim to express their reliability.

Below, Sections 3.2–3.5 describe in detail these different phases of *Espresso*.

#### 3.1. Term definition

Before one can extract relation instances from a corpus, it is necessary to define a tokenization procedure for extracting terms. Terms are commonly defined as *surface representations of stable and key domain concepts* [19]. Defining regular expressions over POS-tagged corpora is the most commonly used technique to both define and extract terms. We adopt a slightly modified version of the term definition given in [13], as it is one of the most commonly used in the literature:

$$((Adj/Noun)+)/((Adj/Noun)*(NounPrep)?)(Adj/Noun)*Noun$$

We operationally extend the definition of *Adj* to include present and past participles as most noun phrases composed of them are usually intended as terms (e.g., *boiling point*). Thus, unlike many approaches for automatic relation extraction, we allow complex multi-word terms as anchor points. Hence, we can capture relations between complex terms, such as “*record of a criminal conviction*” *part-of* “*FBI report*”.

#### 3.2. Phase 1: Pattern discovery

The pattern discovery phase takes as input a set of instances  $I'$  and produces as output a set of lexical patterns  $P$ . For the first iteration  $I' = I_s$ , the set of initial seeds. In order to induce  $P$ , we apply a slight modification to the approach presented in [20]. For each input instance  $i = \{x, y\}$ , we first retrieve all sentences  $S_{x,y}$  containing the two terms  $x$  and  $y$ . Sentences are then generalized into a set of new sentences  $SG_{x,y}$  by replacing all terminological expressions by a terminological label ( $TR$ ). For example:

“Because/IN HF/NNP is/VBZ a/DT weak/JJ acid/NN and/CC  $x$  is/VBZ a/DT  $y$ ”

is generalized as:

“Because/IN  $TR$  is/VBZ a/DT  $TR$  and/CC  $x$  is/VBZ a/DT  $y$ ”

All substrings linking terms  $x$  and  $y$  are then extracted from the set  $SG_{x,y}$ , and overall frequencies are computed. The most frequent substrings then represent the set of new patterns  $P$ , where the frequency cutoff is experimentally set. Term generalization is particularly useful for small corpora, where generalization is vital to ease the data sparseness. However, the generalized patterns are naturally less precise. Hence, when dealing with bigger corpora, the

system allows the use of  $S_{x,y} \cup SG_{x,y}$  in order to extract substrings. For our experiments, we used the set  $SG_{x,y}$ .

### 3.3. Phase 2: Pattern filtering

In this phase, *Espresso* selects among the patterns  $P$  those that are most reliable. Intuitively, a reliable pattern is one that is both highly precise and one that extracts many instances. The recall of a pattern  $p$  can be approximated by the fraction of input instances in  $I'$  that are extracted by  $p$ . Since it is difficult at run-time to estimate the precision of a pattern, we are weary of keeping patterns that generate many instances (i.e., patterns that generate high recall but potentially disastrous precision). We thus prefer patterns that are highly associated with the input patterns  $I'$ . Pointwise mutual information [4] is a commonly used metric for measuring the strength of association between two events  $x$  and  $y$ :

$$pmi(x, y) = \log \frac{P(x, y)}{P(x)P(y)}$$

We define the reliability of a pattern  $p$ ,  $r_\pi(p)$ , as its average strength of association across each input instance  $i$  in  $I'$ , weighted by the reliability of each instance  $i$ :

$$r_\pi(p) = \frac{\sum_{i \in I'} \left( \frac{pmi(i, p)}{\max_{pmi}} * r_i(i) \right)}{|I'|}$$

where  $r_i(i)$  is the reliability of instance  $i$  (defined in Section 3.5) and  $\max_{pmi}$  is the maximum pointwise mutual information between all patterns and all instances.  $r_\pi(p)$  ranges from  $[0, 1]$ . The reliability of the manually supplied seed instances are  $r_i(i) = 1$ . The pointwise mutual information between instance  $i = \{x, y\}$  and pattern  $p$  is estimated using the following formula:

$$pmi(i, p) = \log \frac{|x, p, y|}{|x, *, y| * |*, p, *|}$$

where  $|x, p, y|$  is the frequency of pattern  $p$  instantiated with terms  $x$  and  $y$  and where the asterisk (\*) represents a wildcard. A well-known problem is that pointwise mutual information is biased towards infrequent events. To address this, we multiply  $pmi(i, p)$  with the discounting factor suggested in [16].

The set of highest  $n$  scoring patterns  $P'$ , according to  $r_\pi(p)$ , are then selected and retained for the next phase, where  $n$  is the number of patterns of the previous iteration incremented by 1. In general, we expect that the set of patterns is formed by those of the previous iteration plus a new one. Yet, new statistical evidence can lead the algorithm to discard a pattern that was previously discovered.

Moreover, to further discourage too generic patterns that might have low precision, a threshold  $t$  is set for the number of instances that a pattern retrieves. Patterns firing more than  $t$  instances are then discarded, no matter what their score is. In this paper, we experimentally set  $t$  to a value dependent on the size of the corpus. In future work, this parameter can be learned using a development corpus.

Our reliability measure ensures that overly generic patterns, which may potentially have very low precision, are discarded. However, we are currently exploring a web-expansion algorithm that could both help detect generic patterns and also filter out their incorrect instances. We estimate the precision of the instance set generated by a new pattern  $p$  by looking at the number of these instances that are instantiated on the Web by previously accepted patterns.

Generic patterns will generate instances with higher Web counts than incorrect patterns. Then, the Web counts can also be used to filter out incorrect instances from the generic patterns’ instantiations. More details are discussed in Section 4.3.

### 3.4. Phase 3: Instance discovery

In this phase, *Espresso* retrieves from the corpus the set of instances  $I$  that match any of the lexical patterns in  $P'$ .

In small corpora, the number of extracted instances can be too low to guarantee sufficient statistical evidence for the pattern discovery phase of the next iteration. In such cases, the system enters a *web expansion* phase, in which new instances for the given patterns are retrieved from the Web, using the Google search engine. Specifically, for each instance  $i \in I$ , the system creates a set of queries, using each pattern in  $P'$  with its  $y$  term instantiated with  $i$ ’s  $y$  term. For example, given the instance “*Italy ; country*” and the pattern  $[Y \text{ such as } X]$ , the resulting Google query will be “*country such as \**”. New instances are then created from the retrieved Web results (e.g. “*Canada ; country*”) and added to  $I$ . We are currently exploring filtering mechanisms to avoid retrieving too much noise.

Moreover, to cope with data sparsity, a *syntactic expansion* phase is also carried out. A set of new instances is created for each instance  $i \in I$  by extracting sub-terminological expressions from  $x$  corresponding to the syntactic head of terms. For example, expanding the relation “*new record of a criminal conviction*” *part-of* “*FBI report*”, the following new instances are obtained: “*new record*” *part-of* “*FBI report*”, and “*record*” *part-of* “*FBI report*”.

### 3.5. Phase 4: Instance filtering

Estimating the reliability of an instance is similar to estimating the reliability of a pattern. Intuitively, a reliable instance is one that is highly associated with as many reliable patterns as possible (i.e., we have more confidence in an instance when multiple reliable patterns instantiate it.) Hence, analogous to our pattern reliability measure in Section 3.3, we define the reliability of an instance  $i$ ,  $r_i(i)$ , as:

$$r_i(i) = \frac{\sum_{p \in P'} \frac{pmi(i, p)}{\max_{pmi}} * r_\pi(p)}{|P'|}$$

where  $r_\pi(p)$  is the reliability of pattern  $p$  (defined in Section 3.3) and  $\max_{pmi}$  is the maximum pointwise mutual information between all patterns and all instances, as in Section 3.3.

*Espresso* finally selects the highest scoring  $m$  instances,  $I'$ , and retains them as input for the subsequent iteration. In this paper, we experimentally set  $m = 200$ .

## 4. Experimental Results

### 4.1. Experimental Setup

In this section, we present a preliminary comparison of *Espresso* with two state of the art systems on the task of extracting various semantic relations.

#### 4.1.1. Datasets

We perform our experiments using the following two datasets:

- **TREC-9:** This dataset consists of a sample of articles from the Aquaint (TREC-9) newswire text collection. The sample consists of 5,951,432 words extracted from the following data files: AP890101 – AP890131, AP890201 – AP890228, and AP890310 – AP890319.
- **CHEM:** This small dataset of 313,590 words consists of a college level textbook of introductory chemistry [2].

We preprocess the corpora using the Alembic Workbench POS-tagger [5].

#### 4.1.2. Systems

We compare the results of *Espresso* with the following two state of the art extraction systems:

- **RH02:** This algorithm by Ravichandran and Hovy [20] learns lexical extraction patterns from a set of seed instances of a particular relation (see Section 2.)
- **PR04:** This *is-a* extraction algorithm from Pantel and Ravichandran [16] first automatically induces concepts (clusters) from a raw corpus, names the concepts, and then extracts an *is-a* relation between each cluster member and its cluster label. For each cluster member, the system may generate multiple possible *is-a* relations, but in this evaluation we only keep the highest scoring one. To apply this algorithm, both datasets were first analyzed using the Minipar parser [14].
- **ESP:** This is the algorithm described in this paper (details in Section 3).

#### 4.1.3. Semantic Relations

*Espresso* is designed to extract various semantic relations exemplified by a given small set of seed instances. For our preliminary evaluation, we consider the standard *is-a* and *part-of* relations as well as three novel relations:

- *succession:* This relation indicates that one proper noun succeeds another in a position or title. For example, *George Bush* succeeded *Bill Clinton* and *Pope Benedict XVI* succeeded *Pope John Paul II*. We evaluate this relation on the TREC-9 corpus.
- *reaction:* This relation occurs between chemical elements/molecules that can be combined in a chemical reaction. For example, *hydrogen gas* reacts-with *oxygen gas* and *zinc* reacts-with *hydrochloric acid*. We evaluate this relation on the CHEM corpus.
- *production:* This relation occurs when a process or element/object produces a result. For example, *ammonia* produces *nitric oxide*. We evaluate this relation on the CHEM corpus.

For each semantic relation, we manually extracted a set of seed examples. The seeds were used for both *Espresso* as well as RH02<sup>1</sup>. Table 1 lists a sample of the seeds as well as sample outputs from *Espresso*.

## 4.2. Precision and Recall

We implemented each of the three systems outlined in Section 4.1.2 and applied them to the TREC and CHEM datasets. For each output set, per relation, we evaluate the precision of the system by extracting a random sample of instances (50 for the TREC corpus and 20 for the

---

<sup>1</sup> PR04 does not require any seeds.

**Table 1.** Sample seeds used for each semantic relation and sample outputs from *Espresso*. The number in the parentheses for each relation denotes the total number of seeds.

|                       | <i>SEEDS</i>           | <i>ESP</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-----------------------|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| T<br>R<br>E<br>C<br>9 | <i>Is-a</i> (12)       | wheat :: crop<br>George Wendt :: star<br>Miami :: city<br>shark :: predator                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|                       | <i>Part-Of</i> (12)    | leader :: panel<br>city :: region<br>plastic :: explosive<br>United States :: alliance                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|                       | <i>Succession</i> (12) | Khrushchev :: Stalin<br>Carla Hills :: Yeutter<br>George Bush :: Ronald Reagan<br>Julio Barbosa de Aquino :: Mendes                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| C<br>H<br>E<br>M      | <i>Is-a</i> (12)       | Na :: element<br>protein :: biopolymer<br>HCl :: strong acid<br>electromagnetic radiation :: energy                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|                       | <i>Part-Of</i> (12)    | ion :: matter<br>oxygen :: water<br>light particle :: gas<br>element :: substance                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|                       | <i>Reaction</i> (13)   | magnesium :: oxygen<br>hydrazine :: water<br>aluminum metal :: oxygen<br>lithium metal :: fluorine gas                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|                       | <i>Production</i> (14) | bright flame :: flares<br>hydrogen :: solid metal hydrides<br>ammonia :: nitric oxide<br>copper :: brown gas                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|                       |                        | Picasso :: artist<br>tax :: charge<br>drug dealers :: felons<br>Italy :: country<br>shield :: nuclear missile<br>biblical quotations :: book<br>trees :: land<br>material :: FBI report<br>Ford :: Nixon<br>Setrakian :: John Griesemer<br>Camero Cardiel :: Camacho<br>Susan Weiss :: editor<br>Na :: element<br>protein :: biopolymer<br>HCl :: strong acid<br>electromagnetic radiation :: energy<br>oxygen :: air<br>powdered zinc metal :: battery<br>atom :: molecule<br>ethylene glycol :: automotive antifreeze<br>hydrogen :: oxygen<br>Ni :: HCl<br>carbon dioxide :: methane<br>boron :: fluorine<br>electron :: ions<br>glycerin :: nitroglycerin<br>kidneys :: kidney stones<br>ions :: charge |

CHEM corpus) and evaluating their quality manually using one human judge<sup>2</sup>. For each instance, the judge may assign a score of 1 for correct, 0 for incorrect, and ½ for partially correct. Example instances that were judged partially correct include “*analyst is-a manager*” and “*pilot is-a teacher*”. The precision for a given set of relation instances is the sum of the judge’s scores divided by the number of instances.

Although knowing the total number of instances of a particular relation in any non-trivial corpus is impossible, it is possible to compute the recall of a system relative to another system’s recall. The recall of a system  $A$ ,  $R_A$ , is given by the following formula:

$$R_A = \frac{C_A}{C}$$

where  $C_A$  is the number of correct instances of a particular relation extracted by  $A$  and  $C$  is the total number of correct instances in the corpus. Following [17], we define the relative recall of system  $A$  given system  $B$ ,  $R_{A|B}$ , as:

$$R_{A|B} = \frac{R_A}{R_B} = \frac{C_A}{C_B} = \frac{P_A \times |A|}{P_B \times |B|}$$

Using the precision estimates,  $P_A$ , from our precision experiments, we can estimate  $C_A \approx P_A \times |A|$ , where  $A$  is the total number of instances of a particular relation discovered by system  $A$ .

<sup>2</sup> In future work, we will perform this evaluation using multiple judges in order to obtain confidence bounds and agreement scores.

**Table 2.** System performance on the *is-a* relation on the TREC-9 dataset.

| SYSTEM | INSTANCES | PRECISION*   | REL RECALL† |
|--------|-----------|--------------|-------------|
| RH02   | 57,525    | 28.0%        | <b>5.31</b> |
| PR04   | 1,504     | 47.0%        | 0.23        |
| ESP    | 4,154     | <b>73.0%</b> | 1.00        |

\* Precision estimated from 50 randomly sampled instances.

† Relative recall is given in relation to ESP.

**Table 4.** System performance on the *part-of* relation on the TREC-9 dataset.

| SYSTEM | INSTANCES | PRECISION*   | REL RECALL†  |
|--------|-----------|--------------|--------------|
| RH02   | 12,828    | 35.0%        | <b>42.52</b> |
| ESP    | 132       | <b>80.0%</b> | 1.00         |

\* Precision estimated from 50 randomly sampled instances.

† Relative recall is given in relation to ESP.

**Table 6.** System performance on the *succession* relation on the TREC-9 dataset.

| SYSTEM | INSTANCES | PRECISION*   | REL RECALL†  |
|--------|-----------|--------------|--------------|
| RH02   | 49,798    | 2.0%         | <b>36.96</b> |
| ESP    | 55        | <b>49.0%</b> | 1.00         |

\* Precision estimated from 50 randomly sampled instances.

† Relative recall is given in relation to ESP.

Tables 2 – 8 reports the total number of instances, precision, and relative recall of each system on the TREC-9 and CHEM corpora. The relative recall is always given in relation to the *Espresso* system. For example, in Table 2, RH02 has a relative recall of 5.31 with *Espresso*, which means that the RH02 system output 5.31 times more correct relations than *Espresso* (at a cost of much lower precision). Similarly, PR04 has a relative recall of 0.23 with *Espresso*, which means that PR04 outputs 4.35 fewer correct relations than *Espresso* (also with a smaller precision).

### 4.3. Discussion

Experimental results, for all relations and the two different corpus sizes, show that *Espresso* greatly outperforms the other two methods on precision. However, *Espresso* fails to match the recall level of RH02 in all but the experiment on the *production* relation. Indeed, the filtering of unreliable patterns and instances during the bootstrapping algorithm not only discards the patterns that are unrelated to the actual relation, but also patterns that are too generic and ambiguous – hence resulting in a loss of recall.

As underlined in Section 3.2, the ambiguity of generic patterns often introduces much noise in the system (e.g, the pattern *[X of Y]* can ambiguously refer to a *part-of*, *is-a* or *possession*

**Table 3.** System performance on the *is-a* relation on the CHEM dataset.

| SYSTEM | INSTANCES | PRECISION*   | REL RECALL† |
|--------|-----------|--------------|-------------|
| RH02   | 2556      | 25.0%        | <b>3.76</b> |
| PR04   | 108       | 40.0%        | 0.25        |
| ESP    | 200       | <b>85.0%</b> | 1.00        |

\* Precision estimated from 20 randomly sampled instances.

† Relative recall is given in relation to ESP.

**Table 5.** System performance on the *part-of* relation on the CHEM dataset.

| SYSTEM | INSTANCES | PRECISION*   | REL RECALL†  |
|--------|-----------|--------------|--------------|
| RH02   | 11,582    | 33.8%        | <b>58.78</b> |
| ESP    | 111       | <b>60.0%</b> | 1.00         |

\* Precision estimated from 20 randomly sampled instances.

† Relative recall is given in relation to ESP.

**Table 7.** System performance on the *reaction* relation on the CHEM dataset.

| SYSTEM | INSTANCES | PRECISION* | REL RECALL†  |
|--------|-----------|------------|--------------|
| RH02   | 6,083     | 30%        | <b>53.67</b> |
| ESP    | 40        | <b>85%</b> | 1.00         |

\* Precision estimated from 20 randomly sampled instances.

† Relative recall is given in relation to ESP.

**Table 8.** System performance on the *production* relation on the CHEM dataset.

| SYSTEM | INSTANCES | PRECISION*   | REL RECALL† |
|--------|-----------|--------------|-------------|
| RH02   | 197       | 57.5%        | 0.80        |
| ESP    | 196       | <b>72.5%</b> | <b>1.00</b> |

\* Precision estimated from 20 randomly sampled instances.

† Relative recall is given in relation to ESP.

relation). However, generic patterns, while having low precision, yield a high recall, as also reported by [11]. We ran an experiment on the *reaction* relation, retaining the generic patterns produced during *Espresso*'s selection process. As expected, we obtained 1923 instances instead of the 40 reported in Table 7, but precision dropped from 85% to 30%.

The challenge, then, is to harness the expressive power of the generic patterns whilst maintaining the precision of *Espresso*. We propose the following solution that helps both in distinguishing generic patterns from incorrect patterns and also in filtering incorrect instances produced by generic patterns. Unlike Girju et al. [11] that propose a highly supervised machine learning approach based on selectional restriction, ours is an unsupervised method based on statistical evidence obtained from the Web. At a given iteration in *Espresso*, the intuition behind our solution is that the Web is large enough that correct instances will be instantiated by many of the currently accepted patterns  $P$ . Hence, we can distinguish between generic patterns and incorrect patterns by inspecting the relative frequency distribution of their instances using the patterns in  $P$ . More formally, given an instance  $i$  produced by a generic or incorrect pattern, we count how many times  $i$  instantiates on the Web with every pattern in  $P$ , using Google. The instance  $i$  is then considered correct if its web count surpasses a given threshold. The pattern in question is accepted as a generic pattern if a sufficient number of its instances are considered correct, otherwise it is rejected as an incorrect pattern.

Although our results in Section 4.2 do not include this algorithm, we performed a small experiment by adding an a-posteriori *generic pattern recovery* phase to *Espresso*. We tested the 7,634 instances extracted by the generic pattern  $[X \text{ of } Y]$  on the CHEM corpus for the *part-of* relation. We randomly sample 200 of these instances and then queried Google for these instances using the pattern  $[X \text{ consists of } Y]$ . Manual evaluation of the 25 instances that occurred at least once on Google showed 50% precision. Adding these instances to the results from Table 5 decreases the system precision from 60% to 51%, but dramatically increases *Espresso*'s recall by a factor of 8.16. Furthermore, it is important to note that there are several other generic patterns, like  $[X's \ Y]$ , from which we expect a similar precision of 50% with a continual increase of recall. This is a very exciting avenue of further investigation.

## 5. Conclusions

We proposed a weakly supervised bootstrapping algorithm, called *Espresso*, for automatically extracting a wide variety of binary semantic relations from raw text. Given a small set of seed instances for a particular relation, the system learns reliable lexical patterns, applies them to extract new instances ranked by an information theoretic definition of reliability, and then uses the Web to filter and expand the instances.

There are many avenues of future work. Preliminary results show that *Espresso* generates highly precise relations, but at the expense of lower recall. As mentioned above in Section 4.3, we are working on improving system recall with a web-based method to identify generic patterns and filter their instances. Early results appear very promising. We also plan to investigate the use of WordNet selectional constraints, as proposed by [11]. We expect here that negative instances will play a key role in determining the selectional restriction on generic patterns.

*Espresso* is the first system, to our knowledge, to emphasize both minimal supervision and generality, both in identification of a wide variety of relations and in extensibility to various corpus sizes. It remains to be seen whether one could enrich existing ontologies with relations harvested by *Espresso*, and if these relations can benefit NLP applications such as QA.

## Acknowledgements

The authors wish to thank the reviewers for their helpful comments and Andrew Philpot for evaluating the outputs of the systems.

## References

- [1] Berland, M. and E. Charniak, 1999. Finding parts in very large corpora. In *Proceedings of ACL-1999*. pp. 57-64. College Park, MD.
- [2] Brown, T.L.; LeMay, H.E.; Bursten, B.E.; and Burdge, J.R. 2003. *Chemistry: The Central Science, Ninth Edition*. Prentice Hall.
- [3] Caraballo, S. 1999. Automatic acquisition of a hypernym-labeled noun hierarchy from text. In *Proceedings of ACL-99*. pp 120-126, Baltimore, MD.
- [4] Cover, T.M. and Thomas, J.A. 1991. *Elements of Information Theory*. John Wiley & Sons.
- [5] Day, D.; Aberdeen, J.; Hirschman, L.; Kozierek, R.; Robinson, P.; and Vilain, M. 1997. Mixed-initiative development of language processing systems. In *Proceedings of ANLP-1997*. Washington D.C.
- [6] Downey, D.; Etzioni, O.; and Soderland, S. 2005. A Probabilistic model of redundancy in information extraction. In *Proceedings of IJCAI-2005*. pp. 1034-1041. Edinburgh, Scotland.
- [7] Etzioni, O.; Cafarella, M.J.; Downey, D.; Popescu, A.-M.; Shaked, T.; Soderland, S.; Weld, D.S.; and Yates, A. 2005. Unsupervised named-entity extraction from the Web: An experimental study. *Artificial Intelligence*, 165(1): 91-134.
- [8] Fellbaum, C. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.
- [9] Fleischman, M.; Hovy, E.; and Echiabi, A. 2003. Offline strategies for online question answering: Answering questions before they are asked. In *Proceedings of ACL-03*. pp. 1-7. Sapporo, Japan.
- [10] Geffet, M. and Dagan, I. 2005. The Distributional Inclusion Hypotheses and Lexical Entailment. In *Proceedings of ACL-2005*. Ann Arbor, MI.
- [11] Girju, R.; Badulescu, A.; and Moldovan, D. 2003. Learning semantic constraints for the automatic discovery of part-whole relations. In *Proceedings of HLT/NAACL-03*. pp. 80-87. Edmonton, Canada.
- [12] Hearst, M. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING-92*. pp. 539-545. Nantes, France.
- [13] Justeson J.S. and Katz S.M. 1995. Technical Terminology: some linguistic properties and algorithms for identification in text. In *Proceedings of ICCL-1995*. pp.539-545. Nantes, France.
- [14] Lin, D. 1994. Principar - an efficient, broad-coverage, principle-based parser. In *Proceedings of COLING-94*. pp. 42-48. Kyoto, Japan.
- [15] Mann, G. S. 2002. Fine-Grained Proper Noun Ontologies for Question Answering. In *Proceedings of SemaNet' 02: Building and Using Semantic Networks*, Taipei, Taiwan.
- [16] Pantel, P. and Ravichandran, D. 2004. Automatically labeling semantic classes. In *Proceedings of HLT/NAACL-04*. pp. 321-328. Boston, MA.
- [17] Pantel, P.; Ravichandran, D.; Hovy, E.H. 2004. Towards terascale knowledge acquisition. In *Proceedings of COLING-04*. pp. 771-777. Geneva, Switzerland.
- [18] Pasca, M. and Harabagiu, S. 2001. The informative role of WordNet in Open-Domain Question Answering. In *Proceedings of NAACL-01 Workshop on WordNet and Other Lexical Resources*. pp. 138-143. Pittsburgh, PA.
- [19] Paziienza M.T. 2000. A domain-specific terminology-extraction system. In *Terminology*, 5:2.
- [20] Ravichandran, D. and Hovy, E.H. 2002. Learning surface text patterns for a question answering system. In *Proceedings of ACL-2002*. pp. 41-47. Philadelphia, PA.
- [21] Riloff, E. and Shepherd, J. 1997. A corpus-based approach for building semantic lexicons. In *Proceedings of EMNLP-1997*.



# Concepts across categories

Hilke Reckman and Crit Cremers

*Leiden University Centre for Linguistics (LUCL)*

*Leiden, Netherlands*

*{h. g. b. reckman, c. l. j. m. cremers}@let.leidenuniv.nl*

---

## Abstract

Verbs or adjectives and their nominalizations and certain adverb adjective pairs can be argued to introduce the same concept. This can be shown through inference patterns, which can be explained if we assume Davidsonian eventualities underlying all predicates. We make a contribution to the underlying state discussion by investigating the advantages and disadvantages of Davidsonian versus Kimian states for statives such as copular predicates. Findings are implemented in our parser Delilah.

---

## 1 Introduction

Several computational semantics systems have by now implemented a form of event analysis for verbs [1,3]. There has been much debate on whether it is desirable to assume underlying states, parallel to underlying events. Katz [9] argues against an underlying state analysis, even for stative verbs, whereas Parsons [12] is ready to accept an underlying state analysis, even for simple nouns. It is clear that states are more problematic than events.

We discuss some cases where words of different categories can be argued to introduce the same concept: verbs and their nominalizations and adjectives and their corresponding abstract nouns. We show that underlying states give us the same advantages as underlying events, with respect to recognizing concepts across categories for the purpose of inference, as they reify the predicates. We then discuss an alternative representation for copular expressions, based on the conviction that the states in these expressions are ontologically different from eventualities, and show that it has unfavorable consequences for inference. We end with a short note on related adjective-adverb pairs.

The present research was carried out in the context of the Narrator project, which aims at the development of a system for storage and retrieval of personal illness relating narratives [13,14]. In this project we use and further develop a semantic parser/generator for Dutch, Delilah [5,4]. Delilah is driven by a Combinatory Categorical Grammar and has a semantic output in first order logic with neo-Davidsonian event structures.

## 2 Verbs and their nominalizations

In this section we use nominalizations of verbs to illustrate our main considerations. Sentence (1a) uses the noun *operatie* ‘operation, surgery’ and (1b) uses the verb *opereren* ‘operate’. The intuition is that (1a) and (1b) are equivalent. They can be inferred from each other.

- (1) a. Marie onderging een operatie.  
Mary underwent an operation  
‘Mary went through/ had surgery.’  
b. Marie werd geopereerd.  
Mary was operated  
‘Mary was operated on.’

The same goes for (2a) and (2b), containing negation.

- (2) a. Marie onderging geen operatie.  
Mary underwent no operation  
‘Mary went through/ had surgery.’  
b. Marie werd niet geopereerd.  
Mary was not operated  
‘Mary was not operated on.’

Since the narratives in Narrator are about experiences of patients (in the prototype being currently developed, on breast cancer), this kind of information is rather relevant and should preferably not be missed or misinterpreted. If one of the search criteria is, for example, that the narrative should tell about a patient who had surgery, then each of these sentences above, if occurring in a narrative, provides the relevant information to determine whether it meets this search criterion or not. And of each pair, both variants provide the same information.

*Opereren* en *operatie* introduce the same concept. Also the relation between *opereren/operatie* and Marie is the same in both (1a) and (1b). Arguably it can also be inferred in both cases that there is yet someone else involved who is not mentioned, a filler for the agent-slot of *opereren/operatie*.

A form of neo-Davidsonian event analysis can be used to give both sentences the same semantic representation. The basic event representation for both (1a) and (1b) is illustrated below. The representation is based on Parsons [11]. (The “concept\_of” relation is comparable to Jurafsky and Martin’s “is-a” [8]). The verb form is taken to name the concept. The verb can be considered as basic in a situation like this, because underived nouns do not usually introduce events. As it does not lie within the scope of this paper to discuss what is the best way to represent time/tense, we keep the representations very simple in that respect.

- (3)  $\exists e.\text{event}(e) \ \& \ \text{concept\_of}(e, \text{operate}) \ \& \ \text{agent\_of}(e, x) \ \& \ \text{theme\_of}(e, \text{Mary}) \ \& \ \text{at-time}(e, \text{past})$

For (1b) this kind of representation is quite standard, and event representations for event-denoting nominalizations have also been suggested before [11,7]. The verb *ondergaan* in (1a) plays a special role. It places the event in time (makes it extensional) and it lets its subject be the theme of the surgery event.

### 3 Adjectives and nouns

In the previous section we have looked at nominalizations of verbs, and seen that event semantics helps us getting the right entailments. Now we will look at adjectives and their nominalizations. The pair below is at least close to equivalent. Who has an illness, is ill. Who is ill, has at least one illness.

- (4) a. Marie had een ziekte.  
       Mary had an illness  
       ‘Mary had an illness.’
- b. Marie was ziek.  
       Mary was ill  
       ‘Mary was ill.’

One could try to treat ‘have an illness’ as a kind of collocation and this way have (4a) interpreted as *ill(Mary)*. This, however leaves no space in the representation for the determiner, which may vary in form and accordingly in interpretation.

For the pair *boos/boosheid*, it is more difficult to come up with two equivalent sentences, for lack of a suitable “support verb”. Still we can observe that (5a) entails (5b).

- (5) a. Jan probeerde zijn boosheid te verbergen.  
       Jan tried his anger to hide  
       ‘Jan tried to hide his anger.’
- b. Jan was boos.  
       Jan was angry  
       ‘Jan was angry’

For Katz, however, stative nominalizations denote either a fact or an extent/degree, but never a state. So (5a) could mean that Jan tried to hide (the fact) that he was angry, or how angry he was, but not the state of his being angry. At least the factive reading seems very intuitive here. It is not clear whether there is also a stative reading. In some other contexts, though, a factive reading is not possible. In (6a) *boosheid* is combined with a durational

predicate. (A fact does not have a duration; once a fact, always a fact.) An extent or degree reading doesn't seem to make a lot of sense either.

- (6) a. Hun boosheid duurt nooit lang.  
           their anger   lasts   never long  
           'Their anger never lasts long.'
- b. Ze   zijn nooit lang boos.  
           they are  never long angry  
           'They never are angry for a long time'

Besides, even if *zijn boosheid* in (5a) does only have a factive reading, how should we represent the content of this fact in such a way that (5b) follows from it and that we faithfully represent the quantifier? (*His anger* is deninite.) We can't choose a representation like *angry(Jan)*, because of the quantifier. But if we represent it as a noun (with a possessive kind of relation to *Jan*), while still using a traditional representation for (5b), then we lose the entailment. So even when embedded in a fact, reification of the predicate still yields better representations.

These considerations lead us to the following type of representation for sentences like (4b) and (5b).

- (7)  $\exists e.state(e) \ \& \ concept\_of(e, \textit{ill/anger}) \ \& \ theme\_of(e, \textit{Marie/Jan}) \ \& \ at-time(e, \textit{past})$

Interestingly, for the adjective-noun pairs it is not always that clear and systematic which is the basic form. For the verb - noun pairs above the verb was always basic and the noun was its nominalization. There are also verbs derived from nouns, but they follow a different pattern. Adjective - noun pairs behave less systematically. In the pair *verdrietig* 'sad' - *verdriet* 'sadness', the adjective seems to be the derived form in Dutch, whereas in English the noun has a nominalizing suffix. And for *boos* 'angry' - *boosheid* 'anger' it is the other way around.

## 4 An alternative representation

We have seen that adjectives and their "nominalizations" display the same kind of inference patterns as verbs and their nominalizations, and that reification of the predicate, through postulating an eventuality argument, makes these patterns follow naturally. This reification seems to be the crucial point, though. And since independent evidence for a Davidsonian analysis for statives is kind of shaky, we should investigate whether we really need the full structure. Maienborn [10] proposes a representation for statives which does involve reification of the predicate, but is different from the Davidsonian event structure representation. In this section we discuss this alternative.

#### 4.1 Kimian states

Maienborn argues for a distinction between Davidsonian states (D-states) and Kimian states (K-states). Examples of verbs introducing D-states are *stand*, *sit* and *sleep*. Examples of verbs introducing K-states are *know*, *hate*, *resemble* and copular expressions. In the latter it is the copula that introduces the K-state.

D-states introduce a normal Davidsonian argument, just like other eventualities. For the K-states Maienborn shows that, like D-states, they are available to anaphoric reference and time modification, and therefore they need a referential argument. This referential argument, she argues though, is of a different ontological kind than Davidsonian eventuality arguments. It is of a more abstract nature, similar to facts and propositions. The main argument is their deviant combinatorial behavior. K-state verbs can not serve as the infinitival complement of a verb of perception (see also examples (12b) and (14a) later in this section), they cannot combine with most adverbials, such as manner adverbs and instrumentals, and neither do they combine with locative modifiers, all of this in contrast with D-states and other eventualities. This brings her to the following (tentative) definition of K-states.

(8) *Kimian states:*

K-states are abstract objects for the exemplification of a property P at a holder x at a time t.

Here are some of Maienborn's (German) examples: (9a), with a D-state, is represented as (9b), and (10a), with a K-state, is represented as (10b). The representations are in a flat DRT notation.

- (9) a. Carol schläft.  
Carol sleeps  
'Carol is sleeping.'

b.  $[s^e, v \mid \text{sleep}(s), \text{theme}(s, v), \text{carol}(v)]$

- (10) a. Carol ist müde.  
Carol is tired  
'Carol is tired.'

b.  $[s^z, v \mid s \approx [\text{tired}(v)], \text{carol}(v)]$

The embedded box in (10b) contains the property that is the K-state, and the discourse referent *s* reifies this property.

#### 4.2 Some modifications

Engelberg [6] proposes a few modifications to this view on K-states. He argues the K-state should not be introduced by the copula, but rather by the post-

copula predicate (e.g. an adjective), because attributively used adjectives also show the relevant behavior, without being accompanied by a copula.

Also, he shows that it is problematic to put individuals introduced by an NP under the copula in the box that is introduced by “ $\approx$ ” and presents the ‘content’ of the state. Because in that case the state in (11a) (being related to Opus) would be a different one than the state in (11b) (being related to George). And while the states in (11b) and (11d) are the same, if Opus is the tuba player of the Deathtöngue, since the subject is in the outer box and therefore extensionalized over, this is not the case for the states in (11a) and (11c).

- (11) a. George is related to Opus.  
 b. Opus is related to George.  
 c. George is related to the tuba player of the Deathtöngue  
 d. The tuba player of the Deathtöngue is related to George.

Identity relations between states get more coherent and intuitive if the content of the box embedded under “ $\approx$ ” is restricted to only the core predicate (e.g. *related(x, y)*).

Now if Engelberg is right that K-states are not more fine grained than events and D-states, and the content of the embedded K-state box is in all cases only a core predicate, one can wonder what the advantage of the Kimian style representation still is. For facts and propositions this kind of representation is useful, exactly because the content of a proposition is more than a single predicate; it is a full-fledged proposition, and it makes sense to assign a referential argument to the proposition as a whole. Individuals introduced by NPs in embedded propositions are not extensionalized over. If George said that he is related to Opus and if Opus is the tuba player of the Deathtöngue, it is not entailed that George said that he is related to the tuba player of the Deathtöngue. The main remaining difference between the D-state and K-state representations seems to be that the K-state predicate directly predicates over its argument(s), whereas in D-states this relation is mediated through theta roles. It is not clear why this should be the case.

#### 4.3 Entailments between K-state and D-state verbs

Representing K-states in a different format than D-states, also causes another complication in the domain of inference. German *liegen* ‘to lie’ is a D-state verb, hence the grammaticality of (12a). *Sein* ‘to be’ and also *sich befinden* ‘to be located’ are K-state verbs, as shown by the ungrammaticality of (12b).

- (12) a. Ich sah das Buch auf dem Tisch liegen.  
 I saw the book on the table lie  
 ‘I saw the book lie on the table.’

- b. \*Ich sah das Buch sich auf dem Tisch befinden.  
 I saw the book refl on the table be-located  
 ‘I saw the book be located on the table’

But (13a) entails (13b).<sup>1</sup> (Not all German speakers seem to like the version with the copula, but with *befinden* (13b) is certainly good.) If these two predicates introduce two very different types of states that require different styles of representation, this entailment is problematic.

- (13) a. Das Buch liegt auf dem Tisch.  
 the book lies on the table  
 ‘The book is lying on the table.’
- b. Das Buch befindet sich/ist auf dem Tisch.  
 the book located refl/ is on the table  
 ‘The book is (located) on the table’

It is of course conceivable that the verb *liegen* actually introduces two substates, one of which is Kimian. Intuitively positional location verbs (with their complements) such as *liegen* refer two different pieces of information. One of these is the location of the subject (expressed by the complement) and the other one is in what kind of position the subject is (upright or lying flat...). The locational information will have to be the K-state that gets us the entailment. That means that the positional information has to constitute the D-state that saves the construction in (12a).

So far the problem seems fixable, be it at the cost of losing the clear-cut distinction between D-state verbs and K-state verbs. (The positional location verbs *stand*, *sit* and *lie* are actually quite a substantial group within the D-state verb class). But it gets worse. The verb *to sleep* is a D-state verb and *to be asleep*, being a copula construction, behaves like a K-state expression, as is illustrated below.

- (14) a. \*Ik zag Carol diep in slaap zijn.  
 I saw Carol deep(ly) in sleep be  
 ‘I saw Carol be fast asleep.’
- b. Ik zag Carol slapen.  
 I saw Carol sleep  
 ‘I saw Carol sleep.’

<sup>1</sup> These examples can be reproduced in Dutch, but there the copula version of (13b) is somewhat marginal.

But we can observe that (15a) entails (15b).

- (15) a. Carol was *diep* in *slaap*.  
Carol was *deep*(ly) in *sleep*  
'Carol was fast asleep.'
- b. Carol *sliep*.  
Carol *slept*  
'Carol was sleeping'

Here it is not plausible that (15a) contains a D-state as well as a K-state, because the presence of this D-state should save (14a).<sup>2</sup>

Although the distinction between two groups of statives with different behavior is very convincing, we conclude that in a semantic representation for inference purposes, it does not seem to be a good idea to treat *to sleep* and *to be asleep* as fundamentally different kinds of entities. We therefore stick to Davidsonian style representations for all states. The differences between the two classes that Maienborn shows are of course real. But as they mainly seem relevant for selectional restrictions, they can probably best be captured as part of the feature structure of the predicates, in a computational system like ours. In Delilah the decision of whether two constituents can combine to form a new one depends on the unifiability of their graphs of features. Here one can include a feature that says for example that a predicate is “abstract”. Verbs of perception, all kinds of adverbials and locative modifiers can then be specified for combining only with concrete predicates. The semantic representation then only needs to contain information that is relevant for inference.

## 5 Adjectives and adverbs

Adjectives and adverbs are closely related categories [2]. (The main group of adverbs that also occur as adjectives are the manner adverbs.) If we assume

---

<sup>2</sup> An anonymous reviewer proposed the representation (1a) for ‘Carol was asleep’. Made consistent with the view that a K-state is the exemplification of a property that would be (1b). (Where the property is ‘being the theme of a sleep event’)

- (1) a.  $[s \mid s \approx [s', v \mid [\text{sleep}(s'), \text{theme}(s', v), \text{carol}(v)]]]$   
b.  $[s^z, v \mid s \approx [s'^e \mid [\text{sleep}(s'), \text{theme}(s', v)], \text{carol}(v)]]]$

With a D-state embedded in a K-state, this looks like an interesting compromise. The main problem with it, is that Maienborn introduces K-states next to D-states in order to derive the different combinatory properties of K-states and D-states from their different ontological status. Now if a K-state embeds a D-state, with the same ontological status as any other D-state, one would expect the embedded D-state to also have the same properties as other D-states, such as being able to have a location. This would make the positing of K-states lose its main advantage.



underlying states for adjectives, we should do so for their adverbial counterparts as well. (This is one of the reasons Katz [9] does not want underlying states for adjectives.) This is not necessarily problematic, because the German *dabei*-construction which Maienborn uses as a diagnostic for whether a predicate has a referential argument, also seems to work for adverbs. In (16) the *da* in *dabei* refers to *schnell*. This means that *schnell* should introduce a referential argument.

- (16) Erstaunlich ist, wie schnell und dabei zuverlässig der neue Mozilla  
amazing is how fast and thereat reliably the new Mozilla  
Firebird Seiten darstellt.  
Firebird web sites displays  
‘Amazing is, how quickly and reliably the new Mozilla Firebird displays web sites.’

This suggests that our representation for these kinds of adverbs can be similar to the one that we have proposed for adjectives.

## 6 Conclusions and further research

We have shown that a nice side effect of (neo-)Davidsonian event representations, is that entailment relations between verbs and their nominalizations and between adjectives and their corresponding nouns follow naturally, without any extra machinery. We have defended the use of a Davidsonian representation for adjectives, by showing that assuming states of different ontological sorts obscures certain inferential relations. Our point of view is that semantic representations should only contain information that is needed for inference. Information that is relevant for selectional restrictions should be accommodated elsewhere, where it does not interfere with inference.

In our parser Delilah we have implemented event structures for verbs and nominalizations of verbs. We will proceed with implementing the proposed structures for adjectives along the same lines. We believe that in general semantic parsers that aim at producing structures that support inference can benefit from such an approach. Further research will have to show how much we need to further refine our event structures, for example by systematically including subevents.

## Acknowledgements

This research was funded by Netherlands Organisation for Scientific Research (NWO). Our participation in the workshop was funded by LUF (Leids Universiteits Fonds) and LUCL.

We also thank the reviewers for their comments.

## References

- [1] Bos, J., S. Clark, M. Steedman, J. R. Curran and J. Hockenmaier, *Wide-coverage semantic representations from a ccg parser*, Proceedings of COLING-04 (2004).
- [2] Broekhuis, H., *Adjectives and adjective phrases*, Working Paper 2, University of Tilburg (1999).
- [3] Copestake, A., D. Flickinger, I. A. Sag and C. Pollard, *Minimal recursion semantics: An introduction* (1999).
- [4] Cremers, C., *Formalizing the syntax* (1999).
- [5] Cremers, C., (*'n*) *betekenis berekend*, *Nederlandse Taalkunde* **7** (2002), pp. 375–395.
- [6] Engelberg, S., *Kimian states and the grammar of predicative adjectives*, *Theoretical Linguistics* **31** (2005), pp. 331–347.
- [7] Higginbotham, J., *On events in linguistic semantics*, in: J. Higginbotham, F. Pianesi and A. Varzi, editors, *Speaking of Events*, Oxford University Press., Oxford, New York, 2000 pp. 49–79.
- [8] Jurafsky, D. and J. H. Martin, “Speech and Language Processing: An Introduction to Natural Language Processing,” *Computational Linguistics and Speech Recognition*, Prentice-Hall, Upper Saddle River, NJ, 2000.
- [9] Katz, G., *Anti neo-davidsonianism: Against a davidsonian semantics for state sentences*, in: C. Tenny and J. Pustejovsky, editors, *Events as Grammatical Objects*, CSLI Publications, Stanford, CA, 2000 pp. 393–416.
- [10] Maienborn, C., *On the limits of the davidsonian approach: The case of copula sentences*, *Theoretical Linguistics* **31** (2005), pp. 275–316.
- [11] Parsons, T., “Events in the semantics of English: a study in subatomic semantics,” MIT press, Massachusetts, 1990.
- [12] Parsons, T., *Underlying states and time travel.*, in: J. Higginbotham, F. Pianesi and A. Varzi, editors, *Speaking of Events*, Oxford University Press, Oxford, New York, 2000 pp. 81–93.
- [13] Toussaint, P. and L. Wolf, *Design of the narrator system: processing, storing and retrieving medical narrative data*, Proceedings of ISoLA-2004 (2004).
- [14] Wolf, L., E. Hoenkamp, R. Overberg, H. Reckman and P. Toussaint, *Design of the narrator system: processing, storing and retrieving medical narrative data*, Society for Design and Process Science (Submitted).

# Multi-dimensional Temporal Logic for Events and States

Satoshi Tojo

*JAIST*

*Asahidai 1-1, Nomi, Ishikawa 923-1292, Japan*

*e-mail: tojo@jaist.ac.jp*

---

## Abstract

The inclusion relation in temporal extents can be regarded as a prime requisite for the temporal expression. We propose a multi-dimensional temporal logic, combining the inclusion relation with the conventional precedence relation. First we define the syntax and the semantics of the fused logic, and then we apply the logic to the classification of occurrences to events and states, based on their upward/downward heredities. Thereafter, we consider the mutual relation between the precedence and the inclusion relations and discuss a proper set of axioms for the realistic time.

---

## 1 Introduction

In this paper, we introduce a multi-dimensional modal logic to represent the temporal structure of events and states. The linear temporal logic is the prime method of representation of time in natural language semantics. Another conventional approach, the interval-based time, has also contributed to the analysis of time in which two intervals are related in such ways that one overlaps the other, one includes the other, and so on [6,14,4]. Among such relations, van Benthem [15] regarded the inclusion relation as important, and defined the modalities  $\Box^\uparrow$  and  $\Box_\downarrow$ , each of which represents ‘all the superintervals’ and the latter ‘all the subintervals,’ respectively.<sup>1</sup> In this paper, we further develop the idea and discuss the logic of the inclusion relation together with the conventional precedence.

Here, we do not consider the internal structure of each interval, and identify a time point with a shorter interval; for fear that the term ‘interval’ might be misunderstood as a sequence of time points, we employ the word *temporal extent*, that is a certain consecutive duration of time, in this study.

The temporal logic is formalized by such modal operators as  $F, G, P,$  and  $H$ , each of which represents ‘some future,’ ‘all the future,’ ‘some past,’ and ‘all

---

<sup>1</sup> In the original literature [15], they are written as  $\Box^{up}$  and  $\Box_{down}$ .

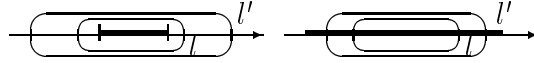


Fig. 1. upwar/downward hereditary

the past,' respectively. We add  $\Box^\uparrow$  and  $\Box_\downarrow$  to these, and propose a polymodal logic with regard to the ordinary temporal order and the inclusion relation. With this logic, we express the distinction of events and states.

In the following Section 2, we explain the intrinsic distinction of events and states in terms of temporal heredity. In Section 3 we show the syntax and semantics of the logic. We apply the logic to events and states in Section 4, and thereafter, we discuss a proper set of axioms for the logic in Section 5. In Section 6 we summarize our contribution.

## 2 Upward/downward heredity

Let us consider an example of a simple detective story. If a murder suspect has an alibi between 2:00<sub>AM</sub> and 4:00<sub>AM</sub>, then (s)he has one also between 2:30<sub>AM</sub> and 3:30<sub>AM</sub>. Because 'have an alibi' is also valid in all the subintervals, this statement is said to be *downward hereditary*. On the contrary, if the presumed time of the death is between 3:00<sub>AM</sub> and 4:00<sub>AM</sub>, then it is also true that the victim died between 2:00<sub>AM</sub> and 5:00<sub>AM</sub>. As 'presumed death time' also holds in all the superintervals, the statement is said to be *upward hereditary* [11].

This distinction can be reduced to the following issue; if an *event* occurs in a point-wise instant, it becomes upward hereditary. Whereas, if a *state* persists with a certain duration of time, it naturally becomes downward hereditary.

The situation is explained in Fig. 1. An event is the perfective<sup>2</sup> view, in which whole the event structure including the beginning point and the culmination point is packed to a sole time point. As in the left-hand side of Fig. 1, if a temporal extent  $l$  includes this occurrence, so does  $l'$  ( $\supseteq l$ ). On the contrary, if some state persists for a given temporal extent  $l'$  as in the right-hand side of the figure, then so does  $l$  ( $\subseteq l'$ ). Hereafter, in case  $l \subseteq l'$ ,  $l'$  is called to be a super-extent of  $l$ , and  $l$  is a sub-extent of  $l'$ . In this paper, we simply call those which are upward hereditary *events*, and those which are downward hereditary *states*, while we call both of them generically *occurrences*. If we claim that an occurrence  $\varphi$  is an event,

$$l \Vdash \varphi \text{ implies } l' \Vdash \varphi \ (l \subseteq l'), \quad (1)$$

and if an occurrence  $\psi$  is a state,

$$l' \Vdash \psi \text{ implies } l \Vdash \psi \ (l \subseteq l'). \quad (2)$$

<sup>2</sup> Note that the perfective view of an occurrence disregards its internal structure and renders the whole as one instant, which is different from the perfect aspect [1].

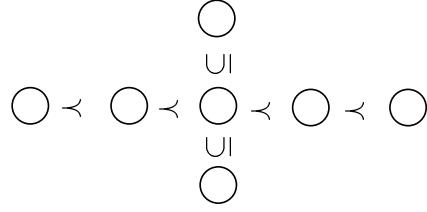


Fig. 2. Two-dimensional time

In the following section, we introduce the two-dimensional time. One dimension is the conventional precedence ( $\prec$ ), and the other is the inclusion relation ( $\subseteq$ ); both of which are given between two temporal extents so that they are arranged in the planar space as in Fig. 2.

### 3 Syntax and semantics of $K_{T\Box}$

In this section, we first give the syntax of the logic, and later, we give its Kripke semantics.

#### 3.1 Syntax

The language consists of propositional variables  $\varphi, \psi, \dots$ , logical connectives  $\neg, \vee, \wedge, \Rightarrow$ , and modal operators  $G, H, \Box^\uparrow, \Box_\downarrow$  where parentheses and punctuation marks are added if necessary.

Modal operators are interpreted in the following way.

- $G\varphi$  at all the future time,  $\varphi$
- $H\varphi$  at all the past time,  $\varphi$
- $\Box^\uparrow\varphi$  in all the super-extents,  $\varphi$
- $\Box_\downarrow\varphi$  in all the sub-extents,  $\varphi$

Modal operators  $F, P, \Diamond^\uparrow$ , and  $\Diamond_\downarrow$  are abbreviations of  $\neg G\neg$ ,  $\neg H\neg$ ,  $\neg\Box^\uparrow\neg$ , and  $\neg\Box_\downarrow\neg$ , respectively.

Note that the two temporal extents in the precedence relation do not share a common time ( $t \not\prec t$ ).  $F\Diamond^\uparrow\varphi$  and  $\Diamond^\uparrow F\varphi$  are differently valuated; the former refers to some future of a super-extent of the current time while the latter does to a super-extent of some future. The former does not include the current time though the latter may include it. Thus, the truth values may be different in the two sentences. Generally speaking, all these operators are not commutative.

A modal logic with the modality  $\Box$  is *normal* if (i) the logic includes all the tautologies, (ii) is closed under Modus Ponens, and (iii) satisfies the following property:

$$(K) \quad \Box(\varphi \Rightarrow \psi) \Rightarrow (\Box\varphi \Rightarrow \Box\psi),$$

and (iv) if  $\varphi$  is a sentence of the logic so is  $\Box\varphi$ . Because all the  $G, H, \Box^\uparrow, \Box_\downarrow$  satisfy the above conditions, the combined logic is *normal*.

First, we introduce the set of axioms for  $G$  and  $H$  and that of  $\Box^\uparrow$  and  $\Box_\downarrow$  independently. The logic  $K_T$ , the minimal tense logic, has the following axioms.

$$(4_{\rightarrow}) \quad G\varphi \Rightarrow GG\varphi \quad (4_{\leftarrow}) \quad H\varphi \Rightarrow HH\varphi$$

$$(C_{\rightarrow}) \quad \varphi \Rightarrow GP\varphi \quad (C_{\leftarrow}) \quad \varphi \Rightarrow HF\varphi$$

while the logic  $K_\square$  includes:

$$(4_\uparrow) \quad \Box^\uparrow\varphi \Rightarrow \Box^\uparrow\Box^\uparrow\varphi \quad (4_\downarrow) \quad \Box_\downarrow\varphi \Rightarrow \Box_\downarrow\Box_\downarrow\varphi$$

$$(C_\uparrow) \quad \varphi \Rightarrow \Box^\uparrow\Diamond_\downarrow\varphi \quad (C_\downarrow) \quad \varphi \Rightarrow \Box_\downarrow\Diamond^\uparrow\varphi$$

$$(T_\uparrow) \quad \Box^\uparrow\varphi \Rightarrow \varphi \quad (T_\downarrow) \quad \Box_\downarrow\varphi \Rightarrow \varphi$$

The logic  $K_T + K_\square$  is the *fusion* of  $K_T$  and  $K_\square$ ,<sup>3</sup> and we denote it as  $K_{T\square}$  hereafter.

### 3.2 Kripke semantics

We introduce Kripke semantics for  $K_{T\square}$ . A Kripke model for the logic is a tuple  $\langle W, \prec, \succ, \subseteq, \supseteq, \Vdash \rangle$ , where  $W$  is a non-empty set of possible worlds, and  $\prec$  and  $\subseteq$  are binary relations on  $W$ . Thus, each temporal extent is regarded as a possible world, and both of the precedence and the inclusion are two different accessibilities between the possible worlds. The semantics, i.e.,  $\Vdash$  is defined inductively as follows.

$$u \Vdash \varphi \wedge \psi \text{ iff } u \Vdash \varphi \text{ and } u \Vdash \psi,$$

$$u \Vdash \varphi \vee \psi \text{ iff } u \Vdash \varphi \text{ or } u \Vdash \psi,$$

$$u \Vdash \varphi \Rightarrow \psi \text{ iff } u \Vdash \varphi \text{ implies } u \Vdash \psi,$$

$$u \Vdash \neg\varphi \text{ iff } u \not\Vdash \varphi,$$

$$u \Vdash G\varphi \text{ iff } \forall v \in W, u \prec v \text{ implies } v \Vdash \varphi,$$

$$u \Vdash H\varphi \text{ iff } \forall v \in W, u \succ v \text{ implies } v \Vdash \varphi,$$

$$u \Vdash \Box^\uparrow\varphi \text{ iff } \forall v \in W, u \subseteq v \text{ implies } v \Vdash \varphi,$$

$$u \Vdash \Box_\downarrow\varphi \text{ iff } \forall v \in W, u \supseteq v \text{ implies } v \Vdash \varphi.$$

In Kripke semantics,  $(4_{\rightleftharpoons})$  and  $(4_{\uparrow\downarrow})$  represent the transitivity,  $(C_{\rightleftharpoons})$  and  $(C_{\uparrow\downarrow})$  the conversion, and  $(T_{\uparrow\downarrow})$  the reflexivity, respectively. A formula  $\varphi$  is *true in model*  $\mathcal{M}$ , denoted by  $\mathcal{M} \models \varphi$ , if  $u \Vdash \varphi$  for every  $u \in W$ . Now, we

<sup>3</sup> Let  $L_1$  and  $L_2$  be two modal logics. If  $L_1$  is axiomatized by the set of axioms  $A_1$  and  $L_2$  is axiomatized by  $A_2$ , then the fusion  $L_1 + L_2$  is axiomatized by the union  $A_1 \cup A_2$  [5,2].

define the veridicality as follows.

$$\begin{aligned}
\mathcal{M} \models G\varphi &\Rightarrow GG\varphi \text{ iff } \forall u, v, w [u \prec v \wedge v \prec w \rightarrow u \prec w], \\
\mathcal{M} \models H\varphi &\Rightarrow HH\varphi \text{ iff } \forall u, v, w [w \succ v \wedge v \succ u \rightarrow w \succ u], \\
\mathcal{M} \models \varphi &\Rightarrow GP\varphi \text{ iff } \forall u, v [u \prec v \rightarrow v \succ u], \\
\mathcal{M} \models \varphi &\Rightarrow HF\varphi \text{ iff } \forall u, v [u \succ v \rightarrow v \prec u], \\
\mathcal{M} \models \Box^\uparrow\varphi &\Rightarrow \Box^\uparrow\Box^\uparrow\varphi \text{ iff } \forall u, v, w [u \subseteq v \wedge v \subseteq w \rightarrow u \subseteq w], \\
\mathcal{M} \models \Box_\downarrow\varphi &\Rightarrow \Box_\downarrow\Box_\downarrow\varphi \text{ iff } \forall u, v, w [u \supseteq v \wedge v \supseteq w \rightarrow u \supseteq w], \\
\mathcal{M} \models \varphi &\Rightarrow \Box^\uparrow\Box_\downarrow\varphi \text{ iff } \forall u, v [u \subseteq v \rightarrow v \supseteq u], \\
\mathcal{M} \models \varphi &\Rightarrow \Box_\downarrow\Box^\uparrow\varphi \text{ iff } \forall u, v [u \supseteq v \rightarrow v \subseteq u], \\
\mathcal{M} \models \Box^\uparrow\varphi &\Rightarrow \varphi \text{ iff } \forall u [u \subseteq u], \\
\mathcal{M} \models \Box_\downarrow\varphi &\Rightarrow \varphi \text{ iff } \forall u [u \supseteq u].
\end{aligned}$$

If  $\prec$  and  $\subseteq$  satisfy all of the above conditions for  $\mathcal{M}$ ,  $\mathcal{M}$  is called to be a  $K_{T\Box}$ -model. Now, we can construct the canonical model [3]; i.e.,  $\forall\varphi, \varphi \notin K_{T\Box}$  iff there exists  $K_{T\Box}$ -model  $\mathcal{M}$  such that  $\mathcal{M} \not\models \varphi$  (completeness).

## 4 Events and states

### 4.1 Downward heredity

A proposition is *gestalt* if it never holds over two temporal extents one of which properly contains the other [11]. That is, the gestalt of an occurrence shows the exact temporal extent where the occurrence takes place on the time axis.

If we directly translate the feature of a state, (2), into a formula,

$$(2)' \quad \varphi \Rightarrow \Box_\downarrow\varphi.$$

Now let us consider the possibility that there exists a super-extent ( $\Box^\uparrow$ ), in all the sub-extents of which  $\varphi$  still holds ( $\Box_\downarrow\varphi$ ). In Fig. 3, if we reside in  $t_1$  and  $\varphi$  is a state ( $\varphi \Rightarrow \Box_\downarrow\varphi$ ), we can assume an enlarged temporal extent  $t_2$  ( $\supseteq t_1$ ) where  $t_2 \Vdash \Box_\downarrow\varphi$ . In case  $\varphi$  does not hold before and after the enlarged extent  $t_2$ , i.e.,

$$t_2 \Vdash H\neg\varphi \wedge \Box_\downarrow\varphi \wedge G\neg\varphi,$$

then the enlarged extent specified by  $t_2$  can be regarded as the *maximal* extent, i.e., the gestalt of the state.

**Example 1** If “Alice was sleeping between  $1_{PM}$  and  $2_{PM}$ ” then we can infer that she has slept in any sub-extent of  $[1_{PM}, 2_{PM}]$ , as:

$$[1:30_{PM}, 2_{PM}] \Vdash \langle\langle A \text{ sleeps} \rangle\rangle,$$

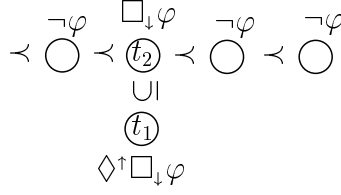


Fig. 3. Maximal duration

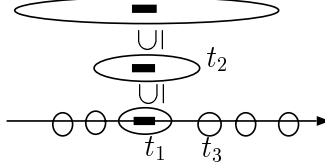


Fig. 4. Upward heredity

where  $A$  stands for ‘Alice.’ In this case, we can also infer that there must be a maximal extent, including  $[1_{PM}, 2_{PM}]$ , in any sub-extents of which she sleeps.

$$[1_{PM}, 2_{PM}] \Vdash \Diamond^{\uparrow}\Box_{\downarrow}\langle\langle A \text{ sleeps} \rangle\rangle.$$

If she actually took a siesta between  $12:30_{PM}$  and  $3_{PM}$ ,

$$[12:30_{PM}, 3_{PM}] \Vdash H\neg\langle\langle A \text{ sleeps} \rangle\rangle \wedge \Box_{\downarrow}\langle\langle A \text{ sleeps} \rangle\rangle \wedge G\neg\langle\langle A \text{ sleeps} \rangle\rangle.$$

Here,  $\langle\langle \rangle\rangle$  is an identical event which happens once and for all, and is not a situation type.

#### 4.2 Upward heredity

On the contrary, if an occurrence of an event is upward hereditary,

$$(1)' \quad \varphi \Rightarrow \Box^{\uparrow}\varphi,$$

and in this case,  $\varphi$  should not appear both in the past and in the future. This situation is depicted in Fig. 4. The thick line is the gestalt of the event and is encircled by its temporal extent. Let  $t_1$  be the original temporal extent of  $\varphi$  and  $t_2 (\supseteq t_1) \Vdash \varphi$ ; if  $t_1 \prec t_3$ , necessarily  $t_3 \not\Vdash \varphi$  even though  $t_3$  may be included in  $t_2$ .

However, in the similar way to the downward heredity, we can assume the minimal extent in which the event occurred, as:

$$\Diamond_{\downarrow}(H\neg\varphi \wedge \Box^{\uparrow}\varphi \wedge G\neg\varphi).$$

**Example 2** If “Betty woke up between  $7:30_{AM}$  and  $8_{AM}$ ” then we can infer that there must be the minimal extent in  $[7:30_{AM}, 8_{AM}]$ , that could be more adequately called an instant, when she got up. Let  $u$  be such a small extent, then:

$$u \Vdash H\neg\langle\langle B \text{ wakes up} \rangle\rangle \wedge \Box^{\uparrow}\langle\langle B \text{ wakes up} \rangle\rangle \wedge G\neg\langle\langle B \text{ wakes up} \rangle\rangle.$$



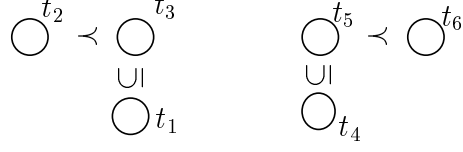


Fig. 5. Triangular constraints

## 5 Triangular constraints

The two accessibilities in Section 3 cannot be independent of each other for a realistic model of time. Here, a model of time means an empirically plausible time, or in other words, all the given temporal extents can be mapped consistently onto the physical time axis, that is linear and unbounded.

As is the left-hand side of Fig. 5, given three temporal extents  $t_1$ ,  $t_2$  and  $t_3$ , if  $t_1$  is included in  $t_3$  and  $t_2$  precedes  $t_3$ , then  $t_2$  should also precede  $t_1$ . The figure suggests the following three constraints. First, seen from  $t_2$ ,

$$F\Diamond_{\downarrow}\varphi \Rightarrow F\varphi, \quad (3)$$

that is, if  $t_2 \Vdash F\Diamond_{\downarrow}\varphi$  then  $t_3 \Vdash \Diamond_{\downarrow}\varphi$ , and thus  $t_1 \Vdash \varphi$ . Next, if we see from  $t_3$ , then:

$$P\varphi \Rightarrow \Box_{\downarrow}P\varphi. \quad (4)$$

Namely, if  $t_3 \Vdash P\varphi$  then  $t_3 \Vdash \Box_{\downarrow}P\varphi$ , i.e., for all  $t_1 \subseteq t_3$ ,  $t_1 \Vdash P\varphi$ . Finally, if we see from  $t_1$ , then:

$$\Diamond^{\uparrow}P\varphi \Rightarrow P\varphi. \quad (5)$$

This means that if  $t_1 \Vdash \Diamond^{\uparrow}P\varphi$  then  $t_3 \Vdash P\varphi$  and thus  $t_2 \Vdash \varphi$ .

Because the axioms (3), (4), and (5) concerns the same arrangement of three temporal extents, the meanings should be equivalent. Actually, we can show that the three axioms are the identical one (see Appendix). We name the axiom  $(\Delta_1)$ . The *duals*<sup>4</sup> of them become:

$$(3)^* G\varphi \Rightarrow G\Box_{\downarrow}\varphi,$$

$$(4)^* \Diamond_{\downarrow}H\varphi \Rightarrow H\varphi,$$

$$(5)^* H\varphi \Rightarrow \Box^{\uparrow}H\varphi.$$

In the very similar way, for the right-hand side of Fig. 5 we can show that the following conditions represent the same axiom. If  $t_4 \Vdash \Diamond^{\uparrow}F\varphi$ , then  $t_5 \Vdash F\varphi$  and  $t_6 \Vdash \varphi$ .

$$\Diamond^{\uparrow}F\varphi \Rightarrow F\varphi. \quad (6)$$

If  $t_5 \Vdash F\varphi$ , then  $t_5 \Vdash \Box_{\downarrow}F\varphi$ .

$$F\varphi \Rightarrow \Box_{\downarrow}F\varphi. \quad (7)$$

<sup>4</sup> The dual is the contraposition of the original formula, the propositional symbols of which are replaced for its negatives, and is denoted by (\*).

If  $t_6 \Vdash P\Diamond_{\downarrow}\varphi$ , then  $t_5 \Vdash \Diamond_{\downarrow}\varphi$  and  $t_4 \Vdash \varphi$ .

$$P\Diamond_{\downarrow}\varphi \Rightarrow P\varphi. \quad (8)$$

The followings are the another family of axioms. We name it  $(\Delta_2)$ . The dual of them become:

$$(6)^* G\varphi \Rightarrow \Box^{\uparrow}G\varphi,$$

$$(7)^* \Diamond_{\downarrow}G\varphi \Rightarrow G\varphi,$$

$$(8)^* H\varphi \Rightarrow H\Box_{\downarrow}\varphi.$$

Combining  $(T_{\uparrow\downarrow})$  with the above formulae, we obtain the following equations of modalities.

$$G \equiv \Box^{\uparrow}G \equiv \Diamond_{\downarrow}G$$

$$H \equiv \Box^{\uparrow}H \equiv \Diamond_{\downarrow}H$$

$$F \equiv \Box_{\downarrow}F \equiv \Diamond^{\uparrow}F$$

$$P \equiv \Box_{\downarrow}P \equiv \Diamond^{\uparrow}P$$

At this stage, the following set of axioms:

$$K_{T\Box\Delta} = K_{T\Box} + \{\Delta_1, \Delta_2\}$$

can be considered as a proper candidate of the two-dimensional temporal structure.

## 6 Discussion

We have proposed a multi-dimensional temporal logic, combining the logic  $K_{\Box}$  of the inclusion relation ( $\subseteq$ ) of temporal extents together with the logic  $K_T$  of the conventional precedence relation ( $\prec$ ), and showed the syntax and the semantics of the fusion of them as  $K_{T\Box}$ . With this logic, we gave explanations for the gestalt of occurrences in terms of temporal extents, as well as the progressive and the perfect aspects. Thereafter, we also added several axioms to constrain the relationship between two different accessibilities,  $\subseteq$  and  $\prec$ , to express the realistic time.

Though we have mainly discussed the distinction of upward/ downward heredity in this paper, we can extend the notion to the classification of aspects. In [15], the progressive and the perfective aspects were represented by  $\Diamond^{\uparrow}$  and  $P$ , respectively. Although these may look rather oversimplified, we can support the idea in that both of the progressive form  $\Diamond^{\uparrow}\varphi$  and the perfect form  $P\varphi$  are downward hereditary ( $\Box_{\downarrow}$ ), as:  $\Diamond^{\uparrow}\varphi \vdash_{(C_{\downarrow})} \Box_{\downarrow}\Diamond^{\uparrow}\Diamond^{\uparrow}\varphi \vdash_{(4_{\uparrow})^*} \Box_{\downarrow}\Diamond^{\uparrow}\varphi$ , and  $P\varphi \Rightarrow \Box_{\downarrow}P\varphi$  (See (4) in Section 5).

In the similar way to the conventional temporal logic, a set of axioms specifies a kind of multi-dimensional temporal logic. Namely, adding or subtracting

some axioms, we can represent different temporal structures. At the current stage,  $K_T + \{(4_{\uparrow\downarrow}), (T_{\uparrow\downarrow})\}$ , i.e.,  $K_{T\Box} - \{(C_{\uparrow\downarrow})\}$  was proved to be *decidable* as in [17]. The sequent rules for  $K_T$  were given in [9] and those for  $\{(4), (T)\}$  with (K) becomes S4. Although  $K_T + S4$  cannot satisfy the cut elimination property [7], we can employ the restricted cut elimination for the subformulae [12,13]. Because the restricted sequent system satisfies the subformula property, we can show that the whole sequent system also satisfies it. If a system has the subformula property, it has a finite model. According to Harrop's theorem [3], if a system has a finite model with a finite set of axioms, it is decidable.

On the contrary, because the system  $\{(4_{\uparrow\downarrow}), (C_{\uparrow\downarrow})\}$  is same as  $K_T$ ,  $K_{T\Box} - \{(T_{\uparrow\downarrow})\}$  is the fusion of two  $K_T$ 's and again becomes decidable.

Although we can claim that  $K_{T\Box\Delta}$  gives the proper relationship of the inclusion and the precedence, the sequent system and the proof method of which would become more complicated. The logical features such as decidability of the extended multi-dimensional modal logic, as well as the proof system, are under investigation.

## References

- [1] B. Comrie. *Aspect*, Cambridge University Press, 1976.
- [2] D. M. Gabbay, A. Kurucz, F. Wolter, and M. Zakharyashev. *Many-dimensional modal logics: theory and applications*, Studies in logic and the foundations of mathematics, vol. 148, Elsevier, 2003.
- [3] R. Goldblatt. *Logics of Time and Computation*, Second Edition, CSLI Lecture Note No.7, Center for the Study of Language and Information, Stanford University, 1992.
- [4] H. Kamp and U. Reyle. *From Discourse to Logic*, Kluwer Academic Publisher's, 1993.
- [5] M. Krachet and F. Wolter. Properties of independently axiomatizable bimodal logics, *Journal of Symbolic Logic* 56, pp. 1469-1485, 1991.
- [6] F. Landman. *Structures for Semantics*, Kluwer Academic Publisher's, 1991.
- [7] A. Maruyama, S. Tojo and H. Ono. Decidability of temporal epistemic logics for multi-agent models, *Proceedings of the ICLP'01 Workshop on Computational Logic in Multi-Agent Systems (CLIMA-01)*, pp.31-40, 2001.
- [8] M. Moens and M. Steedman. Temporal ontology and temporal reference, *Computational Linguistics*, 14(2), pp.15-28, 1988.
- [9] H. Nishimura. *A study of some tense logics by Gentzen's sequential method*, Publications of the Research Institute for Mathematical Sciences, Kyoto University 16, pp.343-353, 1990.
- [10] T. Parsons. : *Events in the Semantics of English*, MIT press, 1990.

- [11] Y. Shoham, *Reasoning about Change*, The MIT Press, 1988.
- [12] M. Takano. Subformula property as a substitute for cut-elimination in modal propositional logics, *Mathematica japonica Vol.37*, pp.1129-1145, 1992.
- [13] M. Takano. A modified subformula property for the modal logics K5 and K5D, bulletin of Section of Logic, Vol. 30, pp.67-70, 2001.
- [14] J. van Benthem. *The Logic of Time - second edition*, Kluwer Academic Press, 1991.
- [15] J. van Benthem. *Epistemic and temporal reasoning*, edited by Dov M. Gabbay, C.J. Hogger and J.A. Robinson, Oxford, Clarendon Press, Handbook of logic in artificial intelligence and logic programming, vol. 4, pp. 292-296, 1995.
- [16] Z. Vendler. *Linguistics in Philosophy*, Cornell University Press, 1967.
- [17] S. Yoshioka and S. Tojo. Many-dimensional Modal Logic of Tense and Temporal Interval and its Decidability, WEC 2005.

## Appendix

We here prove the equality of (3)  $\Leftrightarrow$  (4)  $\Leftrightarrow$  (5).

- (3)  $\vdash$  (4): By  $(C_{\leftarrow})$   $\Diamond_{\downarrow}\varphi \Rightarrow HF(\Diamond_{\downarrow}\varphi)$ , and by (3)  $\Diamond_{\downarrow}\varphi \Rightarrow HF\varphi$ . Its dual becomes  $PG\varphi \Rightarrow \Box_{\downarrow}\varphi$ , and thus,  $PG(P\varphi) \Rightarrow \Box_{\downarrow}P\varphi$ . From  $(C_{\rightarrow})$ ,  $P\varphi \Rightarrow PGP\varphi$ ; hence,  $P\varphi \Rightarrow \Box_{\downarrow}P\varphi$  (4).
- (4)  $\vdash$  (3): By  $(C_{\rightarrow})$ ,  $G\varphi \Rightarrow GP(G\varphi)$ . From (4)  $P(G\varphi) \Rightarrow \Box_{\downarrow}P(G\varphi)$ , and by  $(C_{\leftarrow}^*)$ ,  $PG\varphi \Rightarrow \Box_{\downarrow}PG\varphi \Rightarrow \Box_{\downarrow}\varphi$ . Therefore,  $GPG\varphi \Rightarrow G\Box_{\downarrow}\varphi$ . Combining them, we obtain  $G\varphi \Rightarrow G\Box_{\downarrow}\varphi$ . Its dual becomes  $F\Diamond_{\downarrow}\varphi \Rightarrow F\varphi$ .
- (5)  $\vdash$  (4): By  $(C_{\downarrow})$ ,  $P\varphi \Rightarrow \Box_{\downarrow}\Diamond^{\uparrow}P\varphi$ . Given  $\Diamond^{\uparrow}P\varphi \Rightarrow P\varphi$  (5), the right-hand side of the above is reduced to  $\Box_{\downarrow}P\varphi$ .
- (4)  $\vdash$  (5): By  $(C_{\uparrow})$ ,  $H\varphi \Rightarrow \Box^{\uparrow}\Diamond_{\downarrow}H\varphi$ . Because the dual of (4) becomes  $\Diamond_{\downarrow}H\varphi \Rightarrow H\varphi$ , we obtain  $H\varphi \Rightarrow \Box^{\uparrow}H\varphi$ . Its dual becomes  $\Diamond^{\uparrow}P\varphi \Rightarrow P\varphi$ .

We can show the equality of (6)  $\Leftrightarrow$  (7)  $\Leftrightarrow$  (8) similarly, replacing the modal operators symmetrically.

# Considerations on the nature of metaphorical meaning arising from a computational treatment of metaphor interpretation

A.M.Wallington, R.Agerri, J.A.Barnden, S.R.Glasbey and M.G.Lee

*School of Computer Science,  
University of Birmingham, UK)*  
*A. M. Wallington@cs. bham. ac. uk*

---

## Abstract

This paper argues that there need not be a full correspondence between source and target domains when interpreting metaphors. Instead, inference is performed in the source domain, and conclusions transferred to the target. A description of a computer system, ATT-Meta, that partially implements these ideas is provided.

---

## 1 Introduction

It is now generally accepted, especially since the work of Lakoff and associates (e.g. [14,13,15]) that much of everyday discourse shows evidence of metaphor. Consequently, the question of how metaphor should be interpreted and what the semantic consequences are of using a metaphor is of major importance in determining how discourse should be interpreted.

Like Stern [19], we take the position that much of the interpretation of metaphor is highly context dependent and involves pragmatics. However, we believe that, for metaphor, pragmatics must be informed by theories of Artificial Intelligence and psychology. Thus we have some sympathy for Levinson's ([16] p.161) claim that:

“the interpretation of metaphor must rely on features of our general ability to reason analogically. ... It could be claimed that linguistic pragmatics alone should not be expected to provide such a general theory of analogy, without considerable help from psychological theory<sup>1</sup>.”

We depart from Levinson with respect to analogy, and in this paper, we shall challenge current theories of analogy (e.g. [7,8,11] and what might

---

<sup>1</sup> Levinson makes it clear that psychological theory includes Artificial Intelligence.

be termed correspondence theories of metaphor (e.g, Lakoff’s Conceptual Metaphor Theory) in which a source domain is put in correspondence with a target domain. We shall attempt to show that there is far less parallelism between source and target than is often assumed and that the process of interpreting a metaphor often requires heavy use of inferencing in order to associate source domain entities for which there is no parallel target equivalent, what we term “Map-Transcending Entities” (MTEs), with information that is involved in source to target transfer. Now other AI approaches to metaphor also emphasise the role of source domain inference, Hobbs [10] and Narayanan [18] for example. And, Martin’s MIDAS system includes a process of extending conventional source to target mappings [17]. However, apart from important technical differences between these systems and our own, we would wish to stress the implications extensive inferencing has for source-target parallelism and the repercussions this has for the semantics and pragmatics.

We do not yet have a fully developed semantics and pragmatics of metaphor (although see [9] for some preliminary suggestions based on Stern). However, what we do provide is an extensively developed (though informal) inference-based model of metaphor understanding that employs an event-based formalism similar to that of Hobbs [10], combined with a realization of this approach in a fully implemented system (ATT-Meta) that effects the type of reasoning that we claim is at the heart of much metaphor understanding (See [1,4,21]).

## 2 Correspondence approaches to interpreting metaphor

The work of Lakoff and Johnson e.g. [14,15] not only stressed the ubiquity of metaphor in everyday discourse, but also noted that many metaphorical utterances could be systematically related to each other, all appealing to different aspects of the same source domain and being used to describe the same target domain. In other words, what is involved in metaphor is the mapping of one cognitive domain into another. For example, Lakoff [13] notes that the following metaphors all involve a source domain of journeys being used to describe a target domain of the progress of a love affair: *Look how far we’ve come. It’s been a long, bumpy road. We can’t turn back now. We’re at a crossroads. The relationship isn’t going anywhere. We may have to go our separate ways. We’re spinning our wheels. Our relationship is off the track. The marriage is on the rocks. We may have to bail out of this relationship.* To account for this generalization, Lakoff assumes a “LOVE-AS-JOURNEY” mapping, i.e. “a set of ontological correspondences that characterize epistemic correspondences by mapping knowledge about journeys onto knowledge about love” ([13] p207). Specifically, he assumes the following ontological correspondences:

### THE LOVE-AS-JOURNEY MAPPING

- i. The lovers correspond to travellers.
- ii. The love relationship corresponds to the vehicle.

- iii. The lovers' common goals correspond to their common destination.
- iv. Difficulties in the relationship correspond to impediments to travel.

Lakoff does not spell out in any detail exactly how the epistemic correspondences function, and how inferences made in the source transfer to the target. He does however, claim that mappings “project source domain **inference patterns** onto target domain **inference patterns**” ([13] p245, emphasis added). However, we might turn to more formal work in analogy for a possible mechanism. Gentner (e.g. [7]) argues that complex systematic relations between source domain entities such as causal relations also transfer, whilst non-systematic relations such as attributes do not.

### 3 Map Transcending Entities

Let us return to Lakoff's list above of travel metaphors. Consider the statement that *we're spinning our wheels*. How might we infer from this that the love relationship is not progressing as it might? Plausibly, the following chain of inferences might be entertained. If wheels are referred to, then, defeasibly<sup>2</sup>, a vehicle is involved. The spinning wheels are causing the vehicle not to move as it should. If the vehicle is not moving as it should then it is not moving towards its destination.

What can we make of this pattern of inferences? Both the vehicle and the destination have correspondents in the target domain of the love affair, namely the love relationship and the lovers' common goals. With these correspondences, we might assume that the source domain conclusion can be transferred to become a target domain conclusion. But, this is the transfer of it conclusion. It could never have been reached without the premise that something -the spinning wheels- was causing the vehicle not to proceed. So what is the target correspondent of the spinning wheels whose presence is needed in order to allow the entire inference pattern as opposed to just the conclusion to transfer from source to target? Nothing in the list of four ontological correspondences would seem to be appropriate.

If we look at the other examples listed, we find similar cases where the lack of a target domain correspondent would prevent crucial aspects of the inference pattern mapping from source to target. For example, what is the target domain correspondent of *the rocks* in the utterance *our marriage is on the rocks*? A similar point can be made about the location *off the track* in *our relationship is off the track*. It is not that these statements are uninterpretable. Both would seem to permit the same conclusion that was reached about the spinning wheels, namely that the relationship/vehicle is not progressing towards the destination and hence not towards the lovers' goal.

Note that it does not seem quite right to assume that *the spinning wheels*, *rocks* or *lack of tracks* are “impediments” in the sense of the third of Lakoff's

---

<sup>2</sup> We shall henceforth assume that inferences are defeasible unless otherwise stated.

correspondences. There may be an interpretation under which the *bumps* in the statement *its been a long bumpy road* refer to specific, listable, difficulties in the love relationship, and similarly, *the rocks* may refer, for some, to a particular event, but both could be used more generally. Whatever, if anything, *wheels* might refer to it is a vague and very unspecific target domain entity.

Let us now consider the utterance: *We're at a crossroads*. Stating that we are at some location on a road might be taken to implicate that we are going somewhere along the road and hence have a destination. This would map to a target domain "common goal" However, there is no specific correspondent given for crossroads. The source domain inference that if one is at a crossroads, then there is a choice of possible destinations cannot transfer to the target domain inference that there is a choice of possible relationship goals, if, as is stated, inference patterns are mapped from one domain to another.

Now, a lack of target correspondents for source domain entities is not peculiar to the LOVE AS A JOURNEY conceptual metaphor, but is generally the case. In section 3, we shall give a description of our approach to metaphorical reasoning and our computational system ATT-Meta that performs such reasoning. A sentence that we shall analyse in some detail is the following:

1 In the far reaches of her mind, Mary believed Kyle was having an affair.

We assume that there is a mapping between ideas such as the idea that Kyle was having an affair and physical, manipulable, objects. This reification of ideas has a wide currency. We shall also assume here that the mind is often viewed as a physical space. However, what does *the far reaches* refer to?

So what are we to make of these lacunae which prevent the step by step transfer of inferences about the source becoming inferences about the target? Note that the absence of target domain correspondents of source domain entities is not a mere technical problem in determining how source domain implications transfer and become target domain implications, nor should we assume that Lakoff's claim that entire **inference patterns** as opposed to certain **conclusions** transfer is just an imprecise form of words. If there are no target domain correspondents of source domain entities, then we must assume that there are entities in texts have no reference even indirectly via a target domain equivalent to anything in the real world. There are entities that can only be made sense of in terms of their literal meaning in the source domain; a somewhat similar conclusion to Davidson's [5] well known claim that metaphors have only a literal meaning.

One possibility is that the four correspondences given in the LOVE AS A JOURNEY mapping in section 2 are not intended to be exhaustive and the mapping would if properly specified include correspondences for the entities we have discussed. For example, the *crossroads* example might motivate adding the following correspondence to the mapping.

v. A choice of goals corresponds to a choice of destinations.

However, a number of objections can be made to this view. Many concep-



tual metaphors are open-ended allowing almost any entity associated with the source domain to be used when speaking of the target. Now, not all of these might be conventional. They might make use of what Lakoff and Johnson ([14] p 53) call “the unused part” of the source domain. For example, Lakoff [13] gives an example of a creative use of the conceptual metaphor LOVE IS A JOURNEY, the song lyric *we’re riding in the fast lane on the freeway of love*. There is no correspondent listed for *fast lane* nor for a *freeway of love*. It would seem that no finite list of correspondents would ever be enough. This criticism would also defeat lexically based treatments of metaphor.

The last example contained very little that could plausibly correspond to target-domain entities; perhaps just the inference that a journey and thus a destination is involved. However, it might be conceded that whilst many modifier phrases often do not correspond, overall statement-like constituents of the source will correspond to statements in the target. However, consider the following example from real discourse of a particular metaphorical view running through several consecutive statement-like utterances:

“She was too confused to judge anything. If he’d done nothing else, he’d at least invaded her thoughts and tossed them around until there was only chaos in her head.”

We shall assume that the clauses *he’d at least invaded her thoughts*, [*he had*] *tossed them around* and *there was only chaos in her head* all rely on the metaphorical views of IDEAS AS PHYSICAL OBJECTS and MIND AS PHYSICAL SPACE, and taken together partially convey, in target-domain terms, that the man, “he”, had done something that had resulted in the thoughts of the woman, “her”, **not functioning as they ought**. But is there any need to assume a specific target-domain parallel for *tossing physical objects around*? A similar point can be made about the statement *he’d ... invaded her thoughts*: the invading is just mentioned as a way of emphasizing that he had done something that had had a strong effect on her thoughts.

Now, it is possible that the mention of invading may have been used if the man had introduced thoughts into the woman’s mind by saying things to her; the invading corresponding to introducing. It is then these thoughts that caused the tossing of the woman’s pre-existing thoughts. However, it is possible to imagine the above discourse segment occurring in a context where it is clear that the man had not communicated anything to her.

In short, even a sentence, the traditional unit of analysis of truth, might have no meaning other than in terms of the source domain. In the rest of this paper we shall describe our approach and implemented computational model, ATT-Meta, for reasoning with metaphorical utterances that contain MTEs.

## 4 Inferencing and ATT-Meta

Recall that our proposed fifth correspondence in the previous section had a ‘choice’ in the target corresponding to a ‘choice’ in the source. If we look at the

fourth correspondence, we find something similar. A target domain ‘difficulty’ corresponds to a type of difficulty or ‘impediment’ in the source. Note also that Lakoff argues a-propos the ‘fast lane’ song lyric that driving in the fast lane is exciting and that the excitement transfers to the target. And we could give examples in which other emotional states transfer from the source to the target. In the ‘invasion’ passage, the conclusion that the woman’s mind was not functioning properly was likewise transferred. What all these cases have in common is that they involve essentially ‘invariant’ transfers from the source to the target. These are of a very different nature from the cross-domain correspondences between say journeys and love affairs. We argue that such transfers are affected by what we term “View-Neutral Mapping Adjuncts” and argue that they apply universally, at least by default, regardless of what cross-domain mappings are in play and our system has made a start both at investigating what VNMAs are required and at formalising some of them. (See [2,3,21] for more details.)

In order to sketch our approach let us return to the Anne/Kyle example:

1 In the far reaches of her mind, Anne believed Kyle was having an affair[.]

and compare it to the following variant:

2 Anne had in her mind the belief that Kyle was having an affair.

We assume that both sentences utilize the conceptual metaphors (a term which we replace with the more neutral “metaphorical view” in our work): (A) IDEAS AS PHYSICAL OBJECTS (B) MIND AS PHYSICAL SPACE.

We assume that one correspondence included under view (A) is the following: “Conscious mental processing corresponds to physical manipulation.” (See [12] for motivation). We also assume that with activities such as processing/manipulating, which one can have the ability to perform to a greater or lesser degree, DEGREE is a VNMA and maps over from source to target in an invariant manner. Thus a very low ability to mentally process an idea corresponds to a very low ability to physically manipulate an object.

We assume that accompanying metaphorical view (B) are two ancillary assumptions. Firstly, that the conscious self of the mind’s possessor is metaphorically cast as a person physically located in (a central part of) the mind-region. Secondly, that when a cognitive state (such as believing) is cast as located in a physical sub-region of a mind, then the idea or whatever that is the object of the state is also to be thought of as physically located in that sub-region. As we suggested in the previous section, we assume that there is no known correspondent for the far reaches; it is a map-transcending entity.

So how does the informational contribution of (1) and (2) differ? Plausibly, what (2) principally conveys to the reader is that Anne has the ability to operate in a conscious mental way on the idea that Kyle was having an affair. In brief: Anne is aware of the affair. By contrast, what (1) seems to convey is that the ability to operate holds only to a very low degree. In brief: Anne had very little conscious awareness of the affair.

Thus, the situation described by the ‘far reaches’ utterance is cast as being one where Anne’s conscious self is a person in a central part of Anne’s mind-region, and the idea that Kyle was having an affair is in the far reaches of the mind-region. Now, let us assume that the understander’s common sense knowledge of physical space and physical objects includes the following:

- \* things in the far reaches of a region are usually distant from things in a central part (distance being relative to the scale of the whole region).
- \* if a person is physically distant from a physical object then the person usually has only a very low degree of ability to manipulate that object physically.

Thus, the understander can reason, within the terms of the source domains of the metaphorical views (PHYSICAL SPACE and PHYSICAL OBJECTS), that, probably, Anne’s conscious self has only<sup>3</sup> a very low degree of ability to physically manipulate the idea that Kyle was having an affair.

This conclusion can become the target-domain conclusion that Anne has only a very low degree of ability to operate in a conscious mental way on the idea that Kyle was having an affair, by virtue of the correspondence between physical manipulation and conscious mental processing that was assumed as an aspect of the IDEAS AS PHYSICAL OBJECTS mapping, and by virtue of the VNMA invariantly mapping the very low degree from source to target.

In our approach source-target correspondences are implicit in transfer rules. In the case of the correspondences just mentioned, English glosses of the relevant rules include:

- \* IF in reality X is a person and K is an idea
- \* AND K is being viewed as a physical object
- \* AND person X’s conscious self is being viewed as being able to operate physically on K to at least degree D
- \* THEN presumably in reality X can mentally operate consciously on K to degree at least D.

This rule allows one aspect of the source-domain conclusion to lead to the target-domain conclusion that Anne can mentally operate consciously on the Kyle-affair idea to degree at least “very low”.

In sum, our approach involves the following main types of processing:

- \* Construction of a representation of the direct, source-domain meaning of the utterance, i.e. the meaning it has by taking only the source-domain senses of the metaphorically-used words/phrases in the utterance. This meaning consists of one or more propositions.
- \* In some cases, application of ancillary assumptions associated with the relevant metaphorical views to create further propositions in source-domain terms.
- \* Usually, performance of source-domain reasoning on the basis of the direct

---

<sup>3</sup> A very low degree of ability might implicate that Anne does not have a higher degree, but does not entail it. Hence our addition of ‘only’.

source-domain meaning, the products of ancillary assumptions, and general knowledge relevant to the source domain meaning.

\* Source-to-target transfer acts by application of transfer rules (and VNMA's).

This listing does not imply any particular temporal ordering of the types of processing. Indeed in ATT-Meta the reasoning actually works backwards from reasoning queries posed internally within the system and can involve any intertwining and ordering of instances of the above types of reasoning.

An important feature of our approach that we have not yet mentioned is that it encapsulates the source-domain reasoning based on the literal meaning of the utterance within a special computational context we call a pretence cocoon. Metaphorical transfer acts based on rules such as those above operate between the inside of the pretence cocoon and the reality-context outside. Thus, for the Anne/Kyle example, the understander pretends, within the cocoon, that Anne's mind really is a physical space and that the believing really does occur in the far reaches of this space. Consequences of this are inferred in the pretence cocoon, possibly by substantial amounts of reasoning, using ancillary assumptions and knowledge about physical objects and space. The conclusions reached may then be able to be transmuted, via transfer rules forming part of the relevant metaphorical views, into propositions in the reality environment. However, we ought to stress that many different lines of reasoning will be explored, many ultimately proving unsuccessful.

We should also stress that when a pretence cocoon is created, it is not tagged as having to do with any particular metaphorical view. Only by having the utterance's direct source-domain meaning placed within it, such as the mind having far-reaches, can an inference be made that that the particular metaphorical view MIND AS PHYSICAL SPACE with its associated correspondences is being used. Thus, even the question of the metaphorical views involved in an utterance results from a possibly extensive web of inferences.

Finally note that although Anne's mind is categorized in the pretence as a physical region, this is in addition to its being categorized there as a mind. (Thus, a pretence cocoon is reminiscent of a blend space in Blending Theory: [6].) Given the existence of suitable knowledge rules, such as that a mind is not a physical region, we can get conflicting propositions arising within the pretence, because in general it is wrong to prevent rules about the target domain operating within the pretence. In the present case we would get both strong support for the mind being a physical region and for its not being a physical region. The ATT-Meta system implements conflict-resolution mechanisms that deal with reasoning conflicts in general, and that embody a small number of general principles about conflict resolution in metaphor [1,20]. In the present case, the mechanisms ensure that the proposition that Anne's mind is a physical region wins over the proposition that it is not.

## 5 Conclusion

We have provided a brief outline of some of the ideas in our implemented, inference-based approach to metaphor. Much more detail, including the application to other examples, can be found elsewhere [1,2,4].

The main point has been the use of inference to connect source-domain aspects that are raised by an utterance but not handled by known metaphorical mappings to source-domain aspects in mappings that the understander does know, and particularly to knowledge of what invariant aspects of metaphorical utterances are likely to transfer. By this means, the approach can deal with open-ended extensions of metaphorical beyond what can be readily dealt with by known mappings by themselves, without the need for creating mappings for the unmapped source-domain aspects.

We thus radically downplay source/target parallelism in metaphor in favour of inference, and place great weight on the thesis that metaphors often introduce source-domain aspects that do not need any correspondents in the target domain (let alone already have any): their only purpose is to support useful lines of source-domain inference that connect to known mappings. One of the interesting semantic issues raised is that these unmapped aspects do not by themselves have any meaning in target-domain terms, and it would be a mistake to try to specify such meaning.

## 6 acknowledgements

This work has been supported by current and past grants: EP/C538943/1 and GR/M64208, from the Engineering and Physical Sciences Research Council.

## References

- [1] Barnden, J.A. (2001) Uncertainty and conflict handling in the ATT-Meta context-based system for metaphorical reasoning. In, V. Akman, P. Bouquet, R. Thomason and R.A. Young (Eds), *Procs. Third International Conference on Modeling and Using Context. Lecture Notes in Artificial Intelligence*, Vol. 2116. Berlin: Springer, 15-29.
- [2] Barnden, J.A. and Lee, M.G., (2001). *Understanding open-ended usages of familiar conceptual metaphors: An approach and artificial intelligence system*. Technical Report CSRP-01-05, School of Computer Science, University of Birmingham.
- [3] Barnden, J.A., Glasbey, S.R., Lee M.G. and Wallington, A.M. (2003). Domain-transcending mappings in a system for metaphorical reasoning. In *Proceedings of the Research Note Sessions of the 10th Conference of EACL*.
- [4] Barnden, J.A., Glasbey, S.R., Lee, M.G. and Wallington, A.M. (2004), Varieties and directions of inter-domain influence in metaphor. *Metaphor and Symbol*

- 19(1), 1–30.
- [5] Davidson, D. (1979). What metaphors mean. In, S. Sacks (Ed.), *On Metaphor*. U. Chicago Press, 29-45.
- [6] Fauconnier, G and Turner, M. (2002). *The Way We Think: Conceptual Blending and the Minds Hidden Complexities*. NY: Basic Books.
- [7] Gentner, G. (1983). Structure-mapping: A theoretical framework for analogy. *Cognitive Science*, 7(2), 155–170.
- [8] Gentner, D., Falkenhainer, B. and Skorstad, J. (1988). Viewing metaphor as analogy. In D.H. Helman (Ed.), *Analogical reasoning*. Dordrecht. Kluwer.
- [9] Glasbey, S.R and Barnden, J.A. (submitted). Towards a situation-based discourse semantics for metaphor. Submitted to the journal *Research on Language and Computation*.
- [10] Hobbs, J.R. (1990) *Literature and Cognition* CSLI Lecture Notes, Center for the Study of Language and Information, Stanford University.
- [11] Holyoak, K J. and Thagard, P. (1989). Analogical mapping by constraint satisfaction. *Cognitive Science*, 13(3), 295-355.
- [12] Jaekel, O. (1995). The Metaphorical Concept of Mind, in J.R. Taylor and R.E. MacLaury (eds), *Language and the Cognitive Construal of the World*. Berlin New York, Mouton de Gruyter. 197–229.
- [13] Lakoff, G. (1993). The contemporary theory of metaphor. In A Ortony (Ed.), *Metaphor and Thought*, 2nd ed. Cambridge, UK: Cambridge University Press.
- [14] Lakoff, G. and Johnson, M. (1980). *Metaphors We Live By*. University of Chicago Press.
- [15] Lakoff, G. and Johnson, M. (1999). *Philosophy in the Flesh*. NY: Basic Books.
- [16] Levinson, S. (1983). *Pragmatics*. Cambridge: Cambridge University Press.
- [17] Martin, J. H. (1990). *A Computational Model of Metaphor Interpretation*. NY: Academic Press.
- [18] Narayanan, S. (1999). ‘Moving right along: A computational model of metaphoric reasoning about events,’ *Procs. National Conference on Artificial Intelligence*, pp.121–128. AAAI Press.
- [19] Stern, J. (2000). *Metaphor in Context*. Cambridge, MA and London, UK: Bradford Books, MIT Press.
- [20] Wallington, A.M and Barnden, J.A. (2004). Uncertainty in Metaphorical Reasoning. In *Procs of the Workshop on Computational Models of Natural Argument (CMNA) at ECAI 2004*. August 2004, Valencia, Spain.
- [21] Wallington, A.M., Barnden, J.A. Glasbey S.R. and Lee M. G. (2006). *Metaphorical reasoning with an economical set of mappings*. *Delta*, 22:1.

# Supporting temporal question answering: strategies for offline data collection

David Ahn

*ISLA, University of Amsterdam  
ahn@science.uva.nl*

Steven Schockaert, Martine De Cock, and Etienne Kerre

*Ghent University  
Steven.Schockaert, Martine.DeCock, Etienne.Kerre@UGent.be*

---

## Abstract

We pursue two strategies for offline data collection for a temporal question answering system that uses both quantitative methods and fuzzy methods to reason about time and events. The first strategy extracts event descriptions from the structured year entries in the online encyclopedia Wikipedia, yielding clean quantitative temporal information about a range of events. The second strategy mines the web using patterns indicating temporal relations between events and times and between events. Web mining leverages the volume of data available on the web to find qualitative temporal relations between known events and new, related events and to build fuzzy time spans for events for which we lack crisp metric temporal information.

---

## 1 Introduction

Time structures our world, and the questions we ask reflect that. Not only do we want to know quantitative information—when did some event happen or how long did some state of affairs persist—but we also want qualitative information—what was going on before or during major events, or what happened afterwards. While the amount of information available to answer such questions continues to increase, the temporal information needed is not always fully specified. No information source is obliged to timestamp every referenced event, so while evidence for qualitative temporal relations abounds, there is often no quantitative information to verify it. Furthermore, many events, such as the Cold War or the Great Depression, are inherently vague—with gradual beginnings or endings—or ill-defined aggregations of smaller events.

In order for a temporal QA system to be able to make use of such limited, incomplete temporal information, careful consideration must be given both to

the extraction of temporal information it needs and to the temporal reasoning mechanisms it employs. We are presently at work on a temporal QA system that provides access to events extracted from Wikipedia and satellite events mined from the web and that models the time span of vague events as fuzzy sets and qualitative temporal relations as fuzzy relations. In this paper, we focus on the creation of the knowledge base of events and temporal information, which takes place offline, prior to any user interaction. A separate paper [10] describes the fuzzy reasoning mechanisms the system uses.

In §2 and §3, we introduce temporal questions and sketch the architecture of our temporal QA system. In §4, we describe event extraction from Wikipedia, an online encyclopedia. In §5, we describe web mining for fuzzy and qualitative temporal information. Note that the work described here is still in progress, so while our extraction methods are in use, we are still experimenting with them, and the QA system is not yet complete.

## 2 Temporal questions

There are a variety of question types that fall under the umbrella of *temporal questions*, including questions that ask for times as answers, those that ask for temporal relations, and those that ask for information restricted to a certain time period [6]. The degree of explicitness of temporal reference in a temporal question also varies significantly: some temporal questions refer explicitly to a date or time, while others refer to times only implicitly, by reference to events or states. Here, we focus on *temporally restricted questions*, and in particular, those restricted by events, such as these (from the CLEF 2005 QA track):

- (1) Who played the role of Superman before he was paralyzed?
- (2) What disease did many American soldiers get after the Gulf War?

The data collection we describe, though, can be used to support the answering of other types of temporal questions, as well.

Temporally restricted questions consist of two parts: the main clause, which indicates the information request, and the temporal restriction, which is a subordinate clause or PP headed by a temporal preposition or connective, such as *before*, *after*, *during*, etc., which we refer to as *temporal signals* [7]. The temporal signal that connects the two parts of a temporally restricted question indicates the *temporal relation* that must hold between the time spans of the restricting event and the requested events. Since much of the temporal information we have access to regarding events is vague and incomplete, we explore the use of *fuzzy temporal reasoning* for temporal QA, instead of the standard Allen algebra of temporal interval relations [1]. The model we use is a generalization of Allen's algebra that is suitable for vague events and relations. For crisp events, our reasoning algorithm is equivalent to Allen's path-consistency algorithm. For vague events, fuzzy relations can express that a given qualitative relation is only satisfied to a certain degree [11].



### 3 Architecture of a temporal QA system

Our temporal QA system follows a strategy of extracting information likely to be useful in answering questions—in our case, a knowledge base of events and temporal relations—in a pre-processing stage, before any questions are asked [3,4]. The system consists of several components: a question analysis module, the knowledge base, and an answer selection module.

**Question analysis:** Since we are focusing on temporal questions in which a temporal relation restricts the information being queried, our question analysis module must separate the non-temporal part of the question—the actual information request—from the temporal restriction. Our question analysis module parses the question and extracts phrases headed by temporal signals as potential temporal restrictions. It then uses standard pattern-based techniques to extract keyword queries and the expected answer type.

**Knowledge base:** The knowledge base (KB) consists of two parts: an XML database containing descriptions of individual events and a temporal relation network containing inclusion and before/after relations for events in the KB. Quantitative temporal information about events (i.e., starting and ending dates for crisp events and fuzzy sets for vague ones) is contained in the XML database. The rest of this paper describes the construction of the KB.

**Answer selection:** To answer a temporally restricted question, we must find events that match the non-temporal part of the question and filter out those that do not satisfy the restriction. We treat the problem of finding the events as a retrieval problem, using the keyword queries from question analysis, with event descriptions as target documents. Checking whether an event satisfies the restriction is a matter of inferring whether an appropriate qualitative temporal relation holds between the event and a time or event matching the restriction. We use IR techniques to find events matching the restriction, and we use both quantitative and fuzzy temporal reasoning to make the inference [10]. From the remaining event descriptions, we use standard techniques to extract an answer. Typically, the information request is mapped to a named entity type by the question analysis module, so appropriate named entities are harvested and scored and the top-scoring entity is returned.

### 4 Extracting events from Wikipedia

Wikipedia is a free, open-domain, web-based encyclopedia [12]. In addition to traditional encyclopedia entries, it also has entries for a variety of time periods, which contain lists of historical and/or current events. We extract events from the entries for years. The standardized formatting of year entries in Wikipedia, together with the wiki markup used for this formatting, makes extracting event descriptions straightforward. A typical year entry contains sections delimited by the second-level headings `==Events==`, `==Births==`, and `==Deaths==`. Each of these sections is optionally split into subsections de-

limited by third-level headings indicating months (e.g., `===May===`) or the lack of a date (`===Unknown date===`). Within these subsections are asterisk-delimited lists, each item of which corresponds to an event (or a date with a list of events). Event descriptions begin with a date or a date range, if known, and then continue with one or two sentences describing the event. This text contains phrases marked up as wiki links (pointers to Wikipedia entries):

- (3) `[[March 10]]` - The `[[New Hampshire]]` primary is won by `[[Henry Cabot Lodge]]`, Ambassador to `[[South Vietnam]]`.

Sometimes, an event description begins with a wiki link, set off with a colon, indicating a larger event (which we call a *super-event*) of which it is a part:

- (4) `[[August 8]]` - `[[Watergate scandal]]`: US President `[[Richard Nixon]]` announces his resignation (effective `[[August 9]]`)

Given the structured nature of year entries, simple hand-built patterns can be used to perform what amounts to shallow semantic interpretation, extracting event descriptions from the entries, including temporal location information and limited participant and mereological information (via wiki links and super-events, when present). For each extracted event description, the date(s) and any embedded wiki links are extracted, using simple pattern-matching, and the text of the description is parsed. This information is added to our XML database as an `event` element with the following sub-elements: `date/start_date/end_date` (normalized dates; which one(s) depends on what is given in the entry), `super_event` (wiki link to super event, if present), `description` (text of the description), and `parse` (converted to XML).

From the entries for the years from 1600 to 2005, we have extracted about 33,000 events, somewhat over half (about 19,000) birth and death events.

## 5 Web mining for fuzzy and qualitative information

The basic idea behind web mining is that there is enough information on the web that if there is a significant connection between two events, we should be able to find this connection by searching for patterns that typically express it. We use web mining to build representations of the time span of vague events and to find both additional events related to events already in the KB and for qualitative temporal relations between events in the KB.

While most of the smaller-scale events extracted from Wikipedia come with quantitative temporal information, it is not always fully specified. Furthermore, many of the super-events from Wikipedia, as well as the new events we mine from the web, lack such information. We cope with this by searching the web for beginning and ending dates using a simple pattern-based approach—sending patterns to Google and extracting information from the returned snippets. The patterns we use include, e.g., *⟨event⟩ began on ⟨date⟩* and *⟨event⟩ lasted until ⟨date⟩*. If there is sufficient agreement among different web pages about the beginning and ending date of an event, we represent the

time span of this event as an interval. If not, we use the techniques described in [9] to construct a suitable fuzzy set [13] to represent the time span, which is stored in the XML database as part of the event representation. Of course, for some events, we may fail to find sufficient information about beginning or ending dates, in which case they remain undated, or *ungrounded*, events.

We also use a pattern-based approach to mine the web both for new events and for temporal relations relating ungrounded events to grounded events. Because new events are only usable if they can be temporally connected to events already in the KB, we can use a uniform set of hand-crafted patterns that indicate a temporal relation between events. The patterns we use include, e.g.,  $\langle NP_1 \rangle$  *gave way to*  $\langle NP_2 \rangle$  and  $\langle NP_2 \rangle$  *took place after*  $\langle NP_1 \rangle$ , for before/after relations, and  $\langle NP_1 \rangle$  *and other events during*  $\langle NP_2 \rangle$  and  $\langle NP_1 \rangle$  *took place during*  $\langle NP_2 \rangle$ , for inclusion relations. All our patterns relate NP descriptions of events, which means that they can only be used with known events that have NP descriptions. Fortunately, this includes all super-events and newly mined events, which make up most of the ungrounded events in the KB.

Since we can use the same patterns to mine for new events and for temporal relations for ungrounded events, we combine the tasks. Our basic procedure for mining with these patterns is as follows. Substitute either  $\langle NP_1 \rangle$  or  $\langle NP_2 \rangle$  with the NP description of a known event, send the resulting pattern to Google, and parse and tag the returned snippets with named entities. Extract NPs in the other NP position, discarding those tagged as **person**, **location**, or **organization**. For each remaining NP, if it refers to a known event in the KB, add to the temporal relation network a link between the original known event and the event referred to by the mined NP. Otherwise, add a new event for the mined NP and a link between the original event and the new event.

The hardest step is determining whether a mined NP refers to an event already in the KB. Coreference resolution is clearly necessary to temporal constraints on coreferring event descriptions, but to maintain consistency in the KB, we must be careful in asserting coreference. We do not try to solve the cross-document coreference task completely but instead split mined NPs into two groups. The first group, which includes all indefinite, demonstrative, quantificational, and pronominal NPs, is added to the temporal network but is never considered for coreference. The second group contains NPs that we are confident refer to a unique event and that are thus candidates for coreference.

To find these NPs, we are experimenting with heuristics to determine unique reference. Some heuristics are capitalization-based, while others are based on collocation measures using web hit counts [5,8], such as the ratio between the number of hits for the entire NP and the product of the hits for each of the individual words in the NP (similar to pointwise mutual information [2]). When an NP is determined to be uniquely referring, we use string matching to determine whether it is coreferent with an existing event description. If it is, the temporal relation mined is added to the KB for this existing event. Otherwise, a new event and relation are added to the KB.

## 6 Conclusion

We have described the construction of a knowledge base of events and temporal relations for a temporal QA system. We take advantage of a freely available, structured resource—Wikipedia—to obtain relatively accurate quantitative information about events. We also mine the web to build fuzzy sets for vague events and to find events for which we can only get qualitative information.

**Acknowledgements** The first author was supported by the Netherlands Organization for Scientific Research (NWO), under project number 612.066.302. The second author was supported by a PhD grant from the Research Foundation – Flanders.

## References

- [1] Allen, J., *Maintaining knowledge about temporal intervals*, Communications of the ACM **26** (1983), pp. 832–843.
- [2] Church, K. et al., *Using statistics in lexical analysis*, in: *Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon*, Lawrence Erlbaum, 1991 .
- [3] Fleischman, M., E. Hovy and A. Echihabi, *Offline strategies for online question answering: Answering questions before they are asked*, in: *ACL 2003*, 2003.
- [4] Jijkoun, V., G. Mishne and M. de Rijke, *Preprocessing documents to answer Dutch questions*, in: *BNAIC'03*, 2003.
- [5] Magnini, B., M. Negri, R. Prevete and H. Tanev, *Is it the right answer? Exploiting web redundancy for answer validation*, in: *ACL-02*, 2002.
- [6] Pustejovsky, J. et al., *TERQAS final report*, <http://www.cs.brandeis.edu/~jamesp/arda/time/readings/TERQAS-FINAL-REPORT.pdf> (2002).
- [7] Saurí, R. et al., *TimeML annotation guidelines*, <http://www.cs.brandeis.edu/~jamesp/arda/time/timeMLdocs/annguide12wp.pdf> (2004).
- [8] Schlobach, S., D. Ahn, M. de Rijke and V. Jijkoun, *Data-driven type checking in open domain question answering*, *Journal of Applied Logic* (to appear).
- [9] Schockaert, S., *Construction of membership functions for fuzzy time periods*, in: J. Gervain, editor, *ESSLLI 2005 Student Session*, 2005.
- [10] Schockaert, S., D. Ahn, M. De Cock and E. E. Kerre, *Question answering with imperfect temporal information*, in: *FQAS-2006*, to appear.
- [11] Schockaert, S., M. De Cock and E. E. Kerre, *Imprecise temporal interval relations*, in: *LNCS 3849*, Springer, 2006 .
- [12] Wikipedia, *Wikipedia, the free encyclopedia*, <http://en.wikipedia.org/w/index.php?title=Wikipedia&oldid=35397363>, [Accessed 16-January-2006].
- [13] Zadeh, L. A., *Fuzzy sets*, *Information and Control* **8** (1965), pp. 338–353.

# Formal Semantics Of Verbs For Knowledge Inference

**Igor Boyko, Ph.D.**

Logical Properties Inc., Montreal, Canada  
igor\_m\_boyko@hotmail.com

## Abstract

This short paper is focused on the formal semantic model: Universal Semantic Code (USC), which acquires a semantic lexicon from thesauruses and pairs it with formal meaning representation. The USC model postulates: Knowledge Inference (KI) is effective only on the basis of Semantic Knowledge Representation (SKR). The USC model represents formalized meanings of verbs and phrasal verbs as a main component of its semantic classification. USC algebra defines a formula for the verb, limited set of elements, relations between them, and a natural language interpretation of the formula.

## 1 Introduction

Knowledge Inference applications development depends on natural language processing (NLP) components including lexical classifiers for word sense disambiguation. Word meaning classification and word sense disambiguation techniques facilitate understanding of the terms from different domains.

Numerous approaches of the lexical classification exist. A regular thesaurus defines the meaning of the world but not provides its formal classification what excludes the possibility of KR and KI from such source. Unfortunately in this short paper we are not able to make deep analysis of known methods of knowledge inference in comparison with USC and therefore will talk about main features of the USC model.

Functional classification of verbs based on Universal Semantic Code (Martynov, 1996, 2001) covers the idea of combining the thesaurus and formal representation. In the core the USC model considers verbs as actions and provides inference of the consequences of actions.

## 2 Formalization of the Verb Classes

The USC model has algebraic and logic roots and declares that knowledge can be represented and stored with semantic code, and knowledge inference can be implemented on the basis of the theorems and axioms operating with the semantic code.

Every verb represents an action and every natural language statement comprises an action. Even a statement 'the desk' means the action: 'the desk exists'. Since USC does not make a difference between 'action' and 'verb' we consider 'verb' or 'action' as a main component of the world description. Every action should be surrounded with some elements.

Potentially any action is a reaction on some stimulus where stimulus is an action too. Three-component USC notation  $(X \rightarrow Y) \rightarrow Z$  means stimulus: X by means of Y affects on Z.

The first element of the reaction is always equal to the last element of the stimulus, because some action was implemented with the object Z. An example of the complete USC formula is  $((X \rightarrow Y) \rightarrow Z) \rightarrow ((Z \rightarrow Y) \rightarrow W)$  or shortly  $((XY)Z)((ZY)W)$ . On the abstract level the interpreta-

tion of the formula is: “X by means of Y affects on Z (stimulus) in a result Z by means of Y affects on W (reaction)”.

In USC the reaction part of the formula can be ‘active’ or ‘passive’:

$((XY)Z)((ZY)W)$  – ‘active’ formula

$((XY)Z)(Z(YW))$  – ‘passive’ formula with the interpretation: X by means of Y affects on Z in a result Z holds Y in W.

The difference is in changing the position of the parenthesis in the right part of the formula.

The active reaction represents an active action like: create, destroy, compress, etc. and the passive reaction represents a passive action like: exist, absent, etc.

Each USC formula represents a class of similar actions or similar verbs. The action assigned as a name to the class represents all of those similar actions. The class action (CA) defines a name of the class and has one or minimal number of meanings. For example, the class “fill” comprises a list of actions-analogues in Fig.1. Fig.2 demonstrates actions-analogues for the class “pay”.

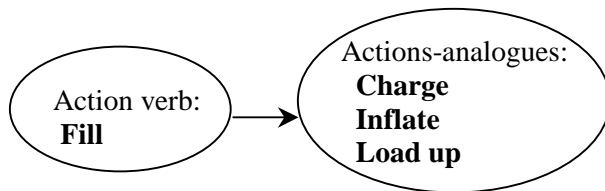


Fig 1. Class “Fill”

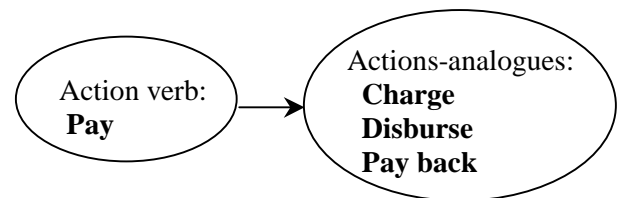


Fig 2. Class “Pay”

We would like to emphasize that the action “charge” is displaced in the both examples according to the meaning.

### 3 Interpretation of the CA

Since each CA represents the class of the actions, we are able to formulate its interpretation for extracting the hidden members of the action.

The action “fill” has the interpretation “X by means of Y fills Z into W”. Then we extract the active members of the action, their roles and substitute them with potential subject and objects of the action. For instance:

- **X** = subject - worker
- **Y** = instrument - loading arm
- **Z** = first object - oil
- **W** = second object - tanker

The complete phrase is: “Worker by means of the loading arm fills oil into the tanker”.

Each action of the class “fill” has the same interpretation. So for the action “charge”, as a member of the class “fill”, the interpretation is: “X by means of Y charges Z in W” and action (phrasal verb) “load up” has the interpretation: “X by means of Y loads up Z in W”.

For the action “pay” the interpretation is: “X by means of Y pays Z”, where:

- **X** = subject - customer
- **Y** = tool - credit card
- **Z** = object - money

The complete phrase is: “Customer by means of credit card pays money”. So the action “charge” as a member of the class “pay” has the interpretation: “X by means of Y charges Z” and the action “pay back” has the interpretation: “X by means of Y pays back Z”.

#### 4 Formal Representation of the CA

So far we have considered two CAs: “fill” and “pay” and determined their sets of variables:

- Fill – XYZW
- Pay – XYZ

Complete formula of the class consists of two parts. The first part of the formula is a stimulus and the second part is a reaction on the stimulus. A procedure of reading of the formula has several steps. For instance, the formula and interpretation for the action “fill”:

$$((X \rightarrow Y) \rightarrow Z) \rightarrow (Z \rightarrow (Y \rightarrow W))$$

“A **worker** by means of the **loading arm** affects **oil** in a result **oil** being kept within the **tanker**” or “A **worker** fills **oil** into **the tanker** by the **loading arm**”.

The operation of implication [ $\rightarrow$ ] demonstrates the direction of the action.

The left part of all USC formula:  $((X \rightarrow Y) \rightarrow Z)$  is identical as a stimulus for all actions, but the right parts are different. The operation of implication between two parts of the formula is a standard logical implication. But implication inside of the parts of the formula is a directed influence of one element onto another.

So for the CA “pay” the formula and interpretation are:

$$((X \rightarrow Y) \rightarrow Z) \rightarrow (Z \rightarrow (Z \rightarrow Y))$$

“A **customer** by means of the **credit card** affects the **money** in a result the **money** being kept out of the **credit card**” or “A **customer** pays the **money** by the **credit card**”.

Those formulas for “fill” and “pay” differ in the right part.

The operation [ $'$ ] is a pointer on the location of one object with respect to another in space and considered as a negation to the location.

USC is a kind of a spatial geometry. All objects in the world can have one of three locations: to be in, to be on the cover, to be out of the cover and notations like: W, W', W'' mean accordingly ‘inside’, ‘not inside’ that is equal to ‘superficially’, ‘not superficially’ that is equal to ‘outside’. For example actions: ‘compress’ is in, ‘join’ is on, ‘disperse’ is out and they are active.

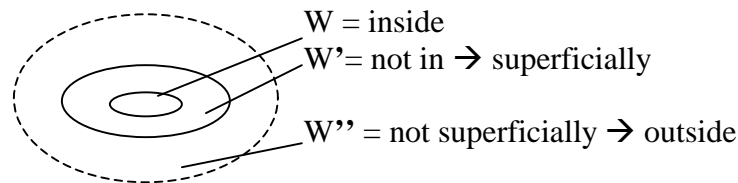


Fig.3 Location of the objects in space

Now we can represent action as four-element structure (Fig.4):

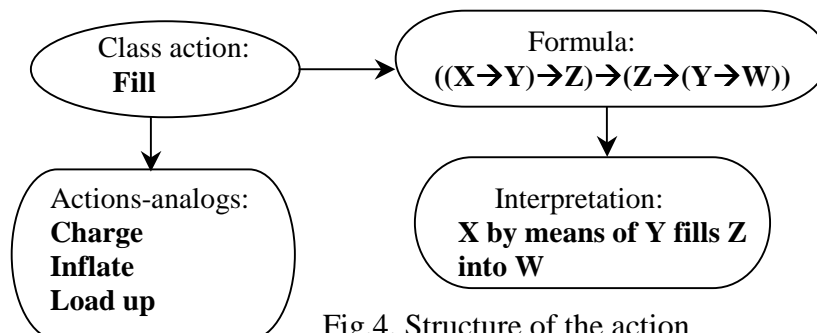


Fig.4. Structure of the action

## 5 USC Axioms

KI with the USC model is based on the axioms of the USC algebra. Relations between USC formulas can be represented as an oriented graph of the axioms. The nodes of the graph are represented by the USC formulas and the arcs are the USC axioms. Since a solution of an intellectual problem is a kind of inference the solution can be obtained as a route of arcs. The algorithm of the problem solving is based on the successive drawing of the route from the target situation to the initial one or vice versa.

The axioms of the USC algebra determine the rules of conversion from one formula into another. For example, the **axiom of transposition** determines changing of parenthesis in the right part of the formula:

$$((XY)Z)((ZW)Z') \rightarrow ((XY)Z)(Z(WZ')) \text{ == if 'create' } \rightarrow \text{ then 'exist'}$$

In the **axiom of diffusion** the right part of the formula can be converted by replacing the variable in the first or second position into the second or third position (Fig.5). With CAs in the positions of the formulas we receive the consequences of the actions in Fig.6.

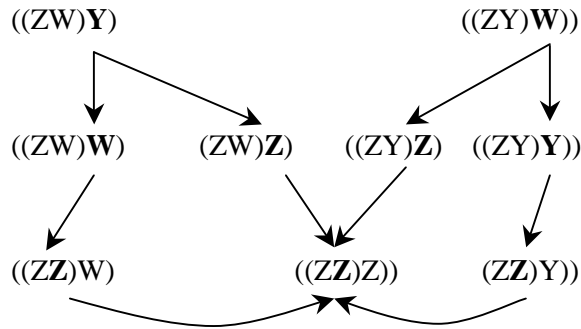


Fig 5. Axiom of diffusion

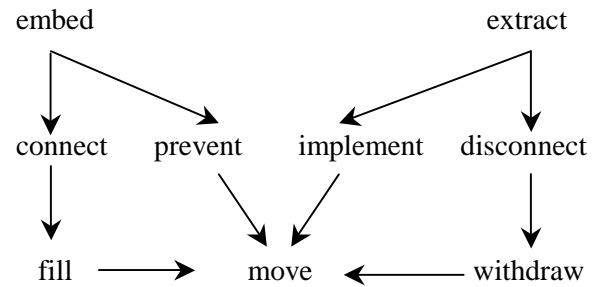


Fig 6. Substitution with CAs

The arrows between formulas determine the direction of the inference from the action to the action. The nodes of the both graphs show antonymic dependence of the class names, like: 'embed' – 'extract' or 'connect' – 'disconnect'.

A set of the USC axioms consists of two parts (Martynov, 2001):

- Four axioms of generation defining sets of variables and their positions in the formula
- Four axioms of transformation defining rules of converting one formula into another

The axioms define the consequence of the actions cannot be arbitrary.

So, the formal part of the USC algebra has been determined as  $\mathbf{A} = \langle \mathbf{M}, \rightarrow, ' \rangle$ , where  $\mathbf{M}$  is a set of elements,  $\rightarrow$  is a binary-non-commutative and non-associative operation on the given set (the operation of implication),  $'$  is a unary operation on the given set (the operation of negation). It strictly corresponds to Lukasiewicz variant of algebra (Lukasiewicz, 1958).

## 6 Semantic Knowledge Inference with USC

To start knowledge inference with USC we should ask: What are we going to infer? Since USC operates with the actions we will calculate the consequences of the actions because each action has a precedent action or a cause and each action is a cause for a consequent action:



(precedent action  $\rightarrow$  current action  $\rightarrow$  consequent action) == (precedent verb  $\rightarrow$  current verb  $\rightarrow$  consequent verb).

As an example we will consider a process of cooking liquid according to the description (Bonnisone, 1985): “The coffee machine’s container comprises cold water and heating elements. The heating elements heat the water in a result the water steam is lifting to the top of the container where grain coffee is displaced. The steam is condensing in the top cold part of the container then percolates through grain coffee and drops into the cap”.

According to the goal the final result is ‘cooked coffee’. Extraction of the actions from the description gives us a consequence of the actions: heat  $\rightarrow$  lift  $\rightarrow$  condense  $\rightarrow$  percolate  $\rightarrow$  drop”. Substitution of the actions with the USC formulas gives a consequence of the formulas:  $(ZY)Y' \rightarrow (ZY)Y'' \rightarrow (ZW)Y' \rightarrow (ZY)W'' \rightarrow Z(YZ'')$ .

Using the graph of the USC axiomatic action relations we are able to verify correctness of the formulas order. We will start the analysis from the last formula  $Z(YZ)$ . According to the axioms this formula cannot be derived from the  $(ZY)W$ . To derive it one intermediate formula should be introduced:  $(ZY)W'' \rightarrow Z(YW'') \rightarrow Z(YZ'')$ . This inference extends the final stage of the process and corresponds to the USC thesaurus: percolate  $\rightarrow$  **cook**  $\rightarrow$  drop. Such inference looks logically correct because cooked coffee is a result of percolation and only then cooked coffee drops down.

On the next step we consider a relation between  $(ZW)Y'$  and  $(ZY)W''$ . According to the axioms the next inference between two formulas should be implemented:

$(ZW)Y' \rightarrow (ZY)W' \rightarrow (ZY)W''$

or condense  $\rightarrow$  **liquefy**  $\rightarrow$  percolate.

If we combine two steps of the inference together then receive the consequence:

$(ZW)Y' \rightarrow (ZY)W' \rightarrow (ZY)W'' \rightarrow Z(YW'') \rightarrow Z(YZ'')$

or condense  $\rightarrow$  liquefy  $\rightarrow$  percolate  $\rightarrow$  cook  $\rightarrow$  drop.

The next step of verification for  $(ZY)Y'' \rightarrow (ZW)Y'$  shows a necessity to introduce an intermediate formula:

$(ZY)Y'' \rightarrow (ZW)Y'' \rightarrow (ZW)Y'$

or lift  $\rightarrow$  **cool**  $\rightarrow$  condense.

The final step of verification shows explicit axiomatic relation  $(ZY)Y' \rightarrow (ZY)Y''$ .

In a result we have the consequence of the actions:

$(ZY)Y' \rightarrow (ZY)Y'' \rightarrow (ZW)Y'' \rightarrow (ZW)Y' \rightarrow (ZY)W' \rightarrow (ZY)W'' \rightarrow Z(YW'') \rightarrow Z(YZ'')$

or heat  $\rightarrow$  lift  $\rightarrow$  cool  $\rightarrow$  condense  $\rightarrow$  liquefy  $\rightarrow$  percolate  $\rightarrow$  cook  $\rightarrow$  drop.

Now we are able to reconstruct the description of the whole process in the extended and corrected form: “The coffee machine’s container comprises cold water and heating elements. The heating elements heat the water in a result the water steam is lifting to the top of the container where grain coffee is displaced. Oh the top the steam is cooling and condensing on the grain coffee. As a result the grain coffee is liquefying and liquid is percolating through. Percolated liquid is a liquid coffee which drops into the cap”.

The example includes the inference with axioms presented and not presented in this short article but all set of rules, axioms and an example of the USC thesaurus could be seen in the book of Martynov V., 2001.

The model was successfully applied for the inventive problems solving (Boyko, 2001) where an inventive solution is a consequence of the actions (technological operations) related through the USC axioms. Besides, the USC inference using the USC thesaurus and axioms can be applied not only for the technical domain but also for SKI in physical, chemical, biological, infor-

mational, and other domains with a condition of having specialized dictionaries coordinated with the USC thesaurus.

## 7 Conclusion

The USC model unites several components including: formal representation of the actions, natural language interpretation, visualization of location of the elements in space, and axioms of inference. The latest published version of the USC action classifier comprises 96 classes divided on two main parts: 48 physical and 48 informational classes (Martynov, 2001). In the article we were able to analyze only the part with physical classes.

Informational classes include actions like ‘forget’, ‘understand’, ‘offend’, ‘order’ etc. Axiomatic relations between them are similar to axiomatic relations for physical actions represented in the article with some restrictions.

All classes relatively paired by the opposite or antonymic principle: create/destroy, lift/low, push/pull, remember/forget, love/hate, etc. “Relatively paired” means the opposite actions can be deduced by axioms and they are located on the same level in the classification table. The whole set of actions comprises 5200 entities. Since 2001 year the number of the classes has not been changed but the names of the classes in some positions has been verified and reconsidered. Axiomatic structure has been changed slightly.

Formal representation of the actions as an intermediate code in “human-computer” interface is the essential property of USC. The USC formulas have been used to represent not only verbs and phrasal verbs, but also to represent deverbal nouns and adjectives for development of the universal principles of machine translation (Boyko, 2002). The USC model can be adjusted to any natural language.

In general the models of formal semantic coding for knowledge inference is a new area of machine learning that has been applied almost exclusively to classification tasks. Most experiments in corpus-based natural language processing present results for some subtasks and there are few results that can be successfully integrated to build a complete NLP system.

## References

- Boyko, I. 2001. *Computer Semantic Search of Inventive Solutions*. TRIZ Journal. USA. March. <http://www.triz-journal.com/archives/2001/03/d/>
- Boyko I. 2002. *Terminological Abstractions for Terminology Classification*. 6<sup>th</sup> International Conference Terminology and Knowledge Engineering. Nancy, France. <http://www.sempl.net>
- Bonnisone P.P., Valavanis K.P., *A Comparative Study of Different Approaches to Qualitative Physics Theories*. Proceedings of the Second Conference Artificial Intelligence Applications (CAIA-85), Miami Beach, 1985.
- Lukasiewicz J. 1958. *Elementy Logiki Matematycznej*. Warszawa.
- Martynov V. 1996. *USC Calculus of Class Word and Class Ideas. Invention Machine Project'96*. Cambridge, MA. <http://www.sempl.net>
- Martynov V. 2001. *Foundations of semantic coding. Summary*. European Humanity University. Minsk. <http://www.sempl.net>

# Ingredients of a first-order account of bridging<sup>1</sup>

Philipp Cimiano

*Institute AIFB, University of Karlsruhe*

## 1 Introduction

The resolution of bridging references is certainly highly interrelated with the computation of discourse relations as well as with the integration of world knowledge into the interpretation of a discourse. In what follows we present examples which clearly corroborate this fact; the following example is taken from [4]:

**Example 1.1** *John entered the room. He saw the chandelier sparkling brightly.*

In this example, the resolution of the bridging reference *the chandelier* depends on the discourse relation by which the second sentence is attached to the first. In fact, assuming that we can not see objects in a room if we are not in it, the chandelier can only be linked to the room in the first sentence if the seeing event follows the entering event, i.e. only if the temporal order imposed by the corresponding discourse relation preserves the surface order. In particular, inferring *Narration* or *Result* (cf. [9]) would be consistent with the resolution, while *Explanation* would not. Now let's consider the following example:

**Example 1.2** *John entered the room. He saw the chandelier through the window.*

Provided that the window is resolved as being part of the room and assuming that we can only see objects through a window which are not in the same room, the definite description *the chandelier* can only be resolved as belonging to the room if the seeing event precedes the entering of the room the chandelier belongs to. In this case thus *Explanation* would be a valid discourse relation, while *Narration* and *Result* would not.

The above examples clearly show how the computation of discourse relations, world knowledge and bridging reference resolution constrain each other.

The following example consisting in a minimal pair also involves temporal aspects and has been discussed in [2] and later in [6]:

**Example 1.3**

- a. *John arrived at the oasis. The camels are standing under the palms.*
- b. *John arrived at the oasis. The camels were standing under the palms.*

The point here is that the camels in the second discourse can not be resolved as being the means of transport by which John arrived as the use of the imperfect shows a preference for interpreting the state in the second sentence

---

<sup>1</sup> The author acknowledges support from the BMBF project SmartWeb.

as temporally overlapping with the arrival (compare [8] and [6]), which would yield an inconsistency as the camels would not be at the oasis as well as be at the oasis at overlapping states. The resolution of the camel as being the means of transport by which John arrived should thus be prohibited by any account of bridging reference resolution.

In general we have to conclude that the resolution of a bridging reference has to be consistent with world knowledge as well as with the consequences introduced by certain discourse relations as well as by tense information. In order to model the information flow between bridging reference resolution, world knowledge, tense information and the computation of discourse relations, in this paper we present a declarative first-order account in which bridging reference resolution is a byproduct of building a minimal model of a discourse as in [5], which thus is also consistent with world knowledge as well as the implications of a certain (inferred) discourse relation. Our approach is in line with the approaches of Gardent and Konrad [5] as well as Hobbs et al. [7] in that minimality is the driving principle of discourse interpretation. In contrast to [2], we rely on FOL in our approach. In our view, there is in fact no obvious reason why discourse interpretation should actually be decidable. For further motivation about why minimal models are interesting as well as for a more detailed overview of related work, the interested reader is referred to [3].

## 2 Ingredients of the logical theory

The logical theory for which we want to find a minimal model consists of the following parts: i) a description of the input discourse, ii) discourse principles, iii) axioms on discourse relations, iv) tense and temporal axioms, and v) world knowledge. We describe each of these components in the following sections.

### 2.1 Input discourse

The input discourse constitutes the variable part of the theory as it varies for each discourse we want to analyze. The input description of the discourse in particular states the surface order of the involved events. Let's for example consider example (1), for which the input description looks as follows, where  $\prec$  denotes the surface order of events:

$$\begin{aligned} &\exists e, e', j, r, c \text{ enter}(e) \wedge \text{agent}(e, j) \wedge \text{patient}(e, r) \wedge \text{event}(e) \wedge \text{past}(e) \wedge \\ &\text{perfect}(e) \wedge \text{room}(r) \wedge \text{see}(e') \wedge \text{agent}(e', j) \wedge \text{patient}(e', c) \wedge \\ &\text{state}(e') \wedge \text{past}(e') \wedge \text{perfect}(e') \wedge \text{chandelier}(c) \wedge e \prec e' \end{aligned}$$

### 2.2 Discourse Principles

It has been argued especially by Asher et al. [2] and furthermore become the main point in SDRT, that discourse segments need to be connected to previous discourse segments by some rhetorical relation. We axiomatize this in our theory as follows:

**Definition 2.1 (Discourse Connectedness)**

$$\forall e \text{ eventuality}(e) \rightarrow \exists e'(e' \prec e \wedge \text{eventuality}(e') \wedge \text{dconnected}(e', e))$$

That means, each event has to be discourse connected to some previously mentioned event (according to the surface order of events). This is in line with the approaches of [7] and [9]. Now we only have to define what *dconnected* means:

**Definition 2.2 (Discourse Relations)**

$$\forall e, e' \text{ dconnected}(e, e') \leftrightarrow \text{drel}(e, e', r_1) \vee \dots \vee \text{drel}(e, e', r_n)$$

where  $r_1 \dots r_n$  are constants representing the discourse relations described in [2] such as *Narration*, *Parallel*, *Result*, *Explanation*, *Elaboration*, *Background*, etc. So, in contrast to the work in [2] we are treating discourse relations as first-order constants instead of relations.

*2.3 Axioms on Discourse Relations*

Further, we need to define axioms specifying the spatio-temporal consequences of a given discourse relation. For the purposes of this paper, we will need temporal consequences on *narration*, *result* and *explanation* (compare [9]) as well as the *spatial consequences on narration* in [2].

*2.4 Tense and Temporal Axioms*

Kamp has argued in [8] that the *simple past* – *Passé Simple*, as he discusses it for French – is typically used to report the successive elements of the main course of action of a story, while the imperfect serves to present the setting in which the action is taking place. In particular, Kamp presents a procedure ([8], p405) describing how a sentence in the perfect or in the imperfect relate to the preceding discourse. The procedure basically states the following: an event reported in the imperfect overlaps with all preceding events in imperfect until the first event reported in the perfect is encountered. The former ones are thus interpreted as describing the circumstances under which the punctual event (reported in the perfect) occurred. This is not the case, for the perfect, for which Kamp claims that a succession of sentences in the perfect convey a similar temporal order of the reported events.

The second principle is probably too strong to be axiomatized (as temporal order does not always correspond to the surface order). Thus, we only axiomatize the first principle on the imperfect. Before, however, we need to introduce the notion of overlap  $\oplus$  between eventualities. In fact, we will introduce a function  $\oplus_f$  denoting the intersection between two eventualities. Further, we will have a special sign  $\perp$  denoting the empty intersection. The corresponding predicate  $\oplus_p$  is then defined in terms of  $\oplus_f$  as follows:

**Definition 2.3 ( $\oplus_p$ )**

$$\begin{aligned} \forall e, e' \text{ e } \oplus_p \text{ e}' &\leftrightarrow \text{e } \oplus_f \text{ e}' \neq \perp (\text{Definition}) \\ \forall e \text{ e } \oplus_p \text{ e} & (\text{Reflexivity}) \\ \forall e, e' \text{ e } \oplus_p \text{ e}' &\rightarrow \text{e}' \oplus_p \text{ e} (\text{Symmetry}) \\ \forall e, e', e'' \text{ e } \supset e \wedge e'' \supset e &\rightarrow \text{e}' \oplus_p \text{ e}'' \wedge \text{e}' \oplus_f \text{ e}'' \supset \text{e} (\text{LeftAbut}) \\ \forall e, e', e'' \text{ e } \supset e' \wedge e'' \oplus_p \text{ e} \wedge e'' \oplus_p \text{ e}' &\rightarrow \\ \forall e''' (\text{e}''' \supset e' \rightarrow \text{e}''' \oplus_p \text{ e} \wedge (\text{e} \oplus_f \text{ e}'') \oplus_f \text{ e}'' \neq \perp) & (\text{LeftAbut\&Overlap}) \end{aligned}$$

**Definition 2.4 (Imperfect)**

$$\begin{aligned} \forall e \text{ eventuality}(e) \wedge \text{past}(e) \wedge \text{prog}(e) &\rightarrow (\forall e' (\text{eventuality}(e') \wedge \text{past}(e') \wedge \\ e' \prec e \wedge \neg \exists (e'') (\text{eventuality}(e'') \wedge \text{past}(e'') \wedge \text{perfect}(e'') \wedge \\ e' \prec e'' \prec e) &\rightarrow e' \oplus_p e) \end{aligned}$$

Further, we will have a homogeneity axiom similar to the one proposed in [1] stating that if a condition  $P$  holds at the eventuality  $e$ , then it also holds for any part of  $e$ . The way in which we express this is by saying that for any overlapping eventuality  $e'$  the conditions of  $e$  hold in particular at the intersection of  $e$  and  $e'$ . The following axiom is actually an axiom schema which needs to be instantiated for all different conditions which can hold at a given eventuality:

**Definition 2.5 (Homogeneity)**

$$\begin{aligned} \forall e, e' e \oplus_p e' \wedge P(e) &\rightarrow P(e \oplus_f e') \\ \forall e, e' e \oplus_p e' \wedge \neg P(e) &\rightarrow \neg P(e \oplus_f e') \end{aligned}$$

Further, for events in general we assume the existence of an *event nucleus* structure as in [10] consisting of a preparatory and a consequent phase.

**Definition 2.6 (Nucleus)**

$$\begin{aligned} \forall e, e' e' \in \text{prep}(e) &\rightarrow e' \in \text{nucleus}(e) \wedge e' \supset_c e (\text{Preparation}) \\ \forall e, e' e' \in \text{conseq}(e) &\rightarrow e' \in \text{nucleus}(e) \wedge e \supset_c e' (\text{Consequent}) \\ \forall e, e', e'' e'' \oplus_p e \wedge e' \in \text{nucleus}(e) &\rightarrow e'' \oplus_p e' (\text{NucleusOverlap}) \end{aligned}$$

**2.5 World Knowledge**

The last ingredient in our logical theory are axioms encoding world knowledge. Besides having axioms encoding a concept hierarchy with the corresponding disjointness axioms, most importantly we will have axioms describing pre-conditions and effects of events. The axioms needed for the purposes of this paper are shown in Figure 1. It is important to note that most of the above axioms should actually be formulated in a non-monotonic fashion, i.e. *it is only normally the case that if we see something through a window, the object in question is in another room*. However, a non-monotonic knowledge representation and reasoning scheme is out of the scope of this paper. We refer the interested reader to [9].

**3 Application to Examples**

Let's start the discussion of example 1.1. We will assume the input description 1 and get the following inferences:

1.  $dconnected(e, e')$  (Event Connectedness)
2.  $drel(e, e', narration) \vee \dots \vee drel(e, e', result)$  (Discourse Relations)
3.  $\exists l \text{ lamp}(l) \wedge \text{in}(l, r)$  (Rooms have lamps) and  $\forall e \text{ loc}(e, l, r)$  (Location)
4.  $\exists s, s' \neg \text{loc}(s, j, r) \wedge s \supset_c e \wedge \text{loc}(s', j, r) \wedge e \supset_c s' \wedge \text{cause}(e, s')$  (Entering)
5.  $\exists l' \text{ location}(l') \wedge \text{loc}(e', j, l') \wedge \text{loc}(e', c, l')$  (Seeing implies same location)
6.  $l=c$  (minimality)

Now interesting is how the computation of discourse relations is affected by the bridging reference resolution: Assume that  $e$  and  $e'$  are connected by *Narration*; then we get by **Temporal Consequences on Narration** as well as

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p><b>Definition 2.7 (Rooms have lamps; chandeliers are some sort of lamps)</b><br/> <math>\forall r (room(r) \rightarrow \exists l lamp(l) \wedge in(l, r))</math><br/> <math>\forall c (chandelier(x) \rightarrow lamp(c))</math></p> <p><b>Definition 2.8 (Entering a room)</b><br/> <math>\forall e, p, r (enter(e) \wedge agent(e, p) \wedge patient(e, p) \wedge person(p) \wedge room(r) \rightarrow</math><br/> <math>\exists e', e'' (e' \supset c e \wedge \neg loc(e', p, r) \wedge loc(e'', p, r) \wedge cause(e, e'') \wedge e \supset c e''))</math></p> <p><b>Definition 2.9 (Seeing implies same location)</b> <math>\forall e, p, o (see(e) \wedge agent(e, p) \wedge</math><br/> <math>patient(e, o) \wedge person(p) \wedge object(o) \rightarrow</math><br/> <math>\exists l (loc(e, p, l) \wedge loc(e, o, l)))</math></p> <p><b>Definition 2.10 (Seeing through a windows implies a different location)</b><br/> <math>\forall e, p, o (seeThroughWindow(e) \wedge agent(e, p) \wedge patient(e, o) \wedge person(p) \wedge object(o) \rightarrow</math><br/> <math>\exists l, l' (loc(e, p, l) \wedge loc(e, o, l') \wedge l \neq l'))</math></p> <p>An arrival always implies a preparatory traveling event as well as a means of transport spatio-temporally correlated with the traveler:</p> <p><b>Definition 2.11 (Arriving implies travelling)</b><br/> <math>\forall e, p, l arrive\_at(e, p, l) \wedge event(e) \wedge person(p) \wedge location(l) \rightarrow</math><br/> <math>\exists e' \wedge travel\_to(e', p, l) \wedge e' \in prep(e)</math></p> <p><b>Definition 2.12 (Travelling implies a mode of transport)</b><br/> <math>\forall e, p, l travel\_to(e, p, l) \wedge event(e) \wedge person(p) \wedge location(l) \rightarrow</math><br/> <math>\exists m modeOfTransport(m) \wedge \exists l' (loc(e, p, l') \wedge loc(e, m, l'))</math></p> <p><b>Definition 2.13 (loc is functional)</b> <math>\forall e, o, l, l' loc(e, o, l) \wedge loc(e, o, l') \rightarrow l = l'</math></p> <p><b>Definition 2.14 (Location)</b><br/> <math>\forall o, l in(o, l) \rightarrow \forall e loc(e, o, l)</math><br/> <math>\forall e, o, o', l under(e, o, o') \wedge loc(e, o', l) \rightarrow loc(e, o, l)</math><br/>         (...)</p> |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Fig. 1. World Knowledge Axioms

**Spatial Consequences on Narration:**  $e < e'$  and  $\exists l loc(s', j, l) \wedge loc(e', j, l)$ . Assume that  $e$  and  $e'$  are connected by *Result*; then we get with **Consequences on Result:**  $e \supset c e' \wedge cause(e, e')$  and thus  $s' \oplus_p e'$  (**Left Abut**), i.e. John's being in the room overlaps with the seeing.

Assume that  $e$  and  $e'$  are connected by *Explanation*; then we get with **Consequences on Explanation:**  $e' \supset c e \wedge cause(e', e)$  and thus  $s \oplus_p e'$  (**Left Abut**), the latter leading to a contradiction as  $s$  and  $e'$  have contradictory conditions. In fact, it holds that  $\neg loc(s \oplus_f e', j, r)$  (from 4 above and **Homogeneity**) as well as  $loc(s \oplus_f e', j, r)$  (from 3, 5 and 6 above, **loc is functional** and **Homogeneity**), which clearly results in a contradiction due to the fact that *loc* is functional. Thus, assuming that the chandelier is interpreted as belonging to the room, *Explanation* can not be inferred as discourse relation while *Result* and *Narration* are consistent with the assumption that the chandelier belongs to the room mentioned in the first sentence. Given these explanations, example 1.2 is easy to explain. In example 1.2, world knowledge implies that neither *Result* nor *Narration* can be inferred because in both cases a state  $s$  in which the condition  $loc(s, j, r)$  holds would yield a contradiction with **seeing through a window implies different locations**. In fact, the discourse relations *Result* or *Narration* can only be predicted in example 1.2 if the chandelier is accommodated with the result that the model would not be minimal anymore. For example 1.3 b, we assume the following input:

$$\begin{aligned} &\exists e, e', j, o, c, p arrive\_at(e) \wedge agent(e, j) \wedge patient(e, o) \wedge \\ &event(e) \wedge past(e) \wedge perfect(e) \wedge oasis(o) \wedge camels(c) \wedge \\ &under(e', c, p) \wedge state(e') \wedge past(e') \wedge prog(e) \wedge palms(p) \end{aligned}$$

Assuming that the palms are resolved as belonging to the oasis, we yield the following inferences:

1.  $loc(e', p, o)$  and thus  $loc(e', c, o)$  (Location)
2.  $dconnected(e, e')$  (Event Connectedness)
3.  $drel(e, e', narration) \vee \dots \vee drel(e, e', result)$  (Discourse Relations)
4.  $\exists e'' travel\_to(e'', j, o) \wedge e'' \in prep(e) \wedge e'' \supset e$  (Arriving implies traveling & Prep)
5.  $e' \oplus_p e$  (Imperfect) and thus  $e' \oplus_p e''$  (Nucleus)
6.  $\exists m modeOfTransport(m) \wedge loc(e'', j, l) \wedge loc(e'', m, l)$  (Mode of Transport)
7.  $\neg loc(s, j, o) \wedge s \supset e \wedge loc(s', j, o) \wedge e \supset s'$  (Arrival)
8.  $m = c$  (Minimality)

Thus we get an inconsistency in every model which identifies the camels  $c$  with the mode of transport  $m$ . This inconsistency is due to the fact that John is spatio-temporally correlated with the mode of transport during  $(e'' \oplus_f s) \oplus_f e'$ , i.e.  $loc((e'' \oplus_f s) \oplus_f e', j, l) \wedge loc((e'' \oplus_f s) \oplus_f e', c, l)$  (5,6 above, **Homogeneity, Left Abut, Left & Overlap**), which yields yield a contradiction with  $\neg loc((e'' \oplus_f s) \oplus_f e', j, o)$  and  $loc((e'' \oplus_f s) \oplus_f e', c, o)$  due to 1,7 above, **Homogeneity, Left Abut, Left Abut & Overlap and Loc is functional**.

## 4 Conclusion

We have presented an approach to bridging reference resolution taking into account the information flow between a certain resolution, the computation of discourse relations as well as linguistic and world knowledge. In our approach this information flow is declarative and emerges as a byproduct of building a minimal model for a logical theory as in [5]. Our contribution lies in spelling out the ingredients of such a logical theory.

## References

- [1] Allen, J. and G. Ferguson, *Actions and events in temporal logic*, Journal of Logic Computation **4** (1994), pp. 531–579.
- [2] Asher, N. and A. Lascarides, *Bridging*, Journal of Semantics **15** (1999).
- [3] Cimiano, P., *A first-order account of the relation between bridges, discourse relations and world knowledge*, Technical report, Institute AIFB, University of Karlsruhe (2006), <http://www.aifb.uni-karlsruhe.de/WBS/pci/bridging.pdf>.
- [4] Clark, H., *Bridging*, in: P. Johnson-Laird and P. Wason, editors, *Thinking, Readings in Cognitive Science*, Cambridge University Press, 1977 pp. 411–420.
- [5] Gardent, C. and K. Konrad, *Interpreting definites using model generation*, Journal of Language and Computation **1** (2000), pp. 193–209.
- [6] Gardent, C. and B. Webber, *Towards the use of automated reasoning in discourse disambiguation*, Journal of Logic, Language and Information **10** (2001), pp. 487–509.
- [7] Hobbs, J., M. Stickel, D. Appelt and P. Martin, *Interpretation as abduction*, Artificial Intelligence **63** (1993), pp. 69–142.
- [8] Kamp, H., *Events, instants and temporal reference*, in: *Semantics from Different Points of View*, Springer, 1979 .
- [9] Lascarides, A. and N. Asher, *Discourse relations and defeasible knowledge*, in: *Meeting of the Association for Computational Linguistics*, 1991, pp. 55–62.
- [10] Moens, M. and M. Steedman, *Temporal ontology and temporal reference*, Computational Linguistics (1988).



# A Computational Theory of Inference for Arithmetic Explanation

Albert Goldfain

*Department of Computer Science and Engineering  
Center for Cognitive Science  
University at Buffalo  
Buffalo, NY 14260  
ag33@cse.buffalo.edu*

---

## Abstract

Mathematical understanding can be measured by a cognitive agent's ability to explain itself, i.e., answer relevant questions about its mathematical activities. Two inference techniques, rule-based inference and path-based inference, are applied to an implemented computational cognitive agent using the SNePS knowledge-representation, reasoning, and acting system.

---

## 1 Introduction

When engaged in classroom mathematical activities, students are often encouraged to “show their work” as they progress towards a solution and to “explain their answers” after a solution has been found. This justification, required well before a student learns how to produce rigorous logical proofs, is a demonstration that the student understands *how* the problem is solved and *why* the obtained result is a solution. Students “show their work” by explicitly providing the intermediate computations needed to reach a solution. When confronted with a problem of averaging, e.g., 2, 3, 15, and 20, students should show the intermediate sum  $2+3+15+20=40$  and the intermediate division  $40/4 = 10$ . Students show that they understand averaging *in terms of* the simpler operations of addition and division.

Unlike human students, computers and calculators are usually expected to produce fast, reliable results *without* an explanation. These results often take the form of a numerical output devoid of intermediate calculations, linguistic content, and problem-specific semantic information. In such a role, the computer is a tool for the human user, a tool that produces the correct answers without justification. However, in the field of artificial intelligence, there are several reasons to design computational agents that can both produce the correct answers and can explain their answers. Such agents could better

communicate and interoperate with human users. Knowing the “how” and “why” behind mathematical routines is required for a computational model of mathematical cognition in a cognitive agent. Extending these abilities can be a great benefit to autonomous embodied agents that must reason in the real world with minimal human interaction.

This paper is a preliminary investigation of the inferential and representational requirements for a computational agent that can produce mathematical explanations at the level of basic arithmetic and counting. Explanation is treated as a justification of procedural action (the kind of solution a student might give), rather than as a formal proof (the kind of solution a mathematician might give).

## 2 Question Answering

To probe a cognitive agent’s mathematical understanding, a series of questions can be posed to the agent after it has completed a mathematical activity. For example, if an agent has just determined that 2 is the greatest common divisor of 8 and 6, a highly idealized explanation dialogue might look like the following:

*Q1: Why is 2 the greatest common divisor of 8 and 6?*

*A1: 2 is the greatest of the common divisors of 8 and 6.*

*Q2: Why is 2 a common divisor of 8 and 6?*

*A2: 2 is a divisor of 8 and 2 is a divisor of 6.*

*Q3: Why is 2 a divisor of 6?*

*A3: There is a number that, when multiplied by 2, gives 6, and that number is 3.*

*Q4: Why is 2 times 3 = 6?*

*A4: Multiplication is repeated addition; 2 plus 2 is 4 and 4 plus 2 is 6*

*Q5: Why is 2 plus 2 = 4?*

*A5: When I count from 2 for two numbers I end up at 4*

*Q6: How do you know that you will end up at 4?*

*A6: I counted two groups of oranges, with 2 oranges in each, ending up with 4 total oranges.*

*Q7: What is 2?*

*A7: It is a number and the greatest common divisor of 8 and 6.*

*Q8: What is a number?*

*A8: Some examples are 2,4,6 and 8 . . . It is something that can be counted, added, multiplied . . . and something that can be the result of finding a greatest common divisor*

A human addressee will likely stop short of such a detailed answer. However, for computational agents, such a dialogue will be a useful Turing-test-style demonstration of mathematical understanding. The ability to produce such answers demands a representational and inferential capacity beyond that of just finding a greatest common divisor. During a dialogue, an agent may need to: (1) call upon linguistic information (e.g., for A2 above), (2) call upon real-world experiences (e.g., for A6 above), (3) infer the relationships between

procedures and their constituent sub-procedures (e.g., for A4 above), and (4) infer relationships between concepts (e.g., for A8 above). The question for a computational agent then becomes: where does this (required) additional knowledge come from?

One answer to this question is to design an agent that accumulates knowledge *during* a mathematical activity. The ability to assimilate newly inferred material with existing knowledge is essential for mathematical understanding [2]. Mathematical understanding is driven by *doing* things, not simply by *thinking* about things. This is consistent with the cognitive foundations of mathematics presented by Lakoff & Núñez [4], in which a metaphoric relationship is established between a human activity (such as object collection) and a formal operation (such as addition).

The activity-driven nature of mathematical understanding can be witnessed even in the developmentally early routine of counting. Children turn a routine that is nothing more than a meaningless recitation of ordered words (much like a nursery rhyme) into a tool for ascribing cardinal size and ordinal position to real-world entities. The semantics of counting routines arises from children performing tasks that require counting, not from the contemplation of number-name meanings. Outside the context of the number line, number-names are meaningless identifiers.

### 3 SNePS

My computational theory is implemented in the SNePS knowledge-representation, reasoning, and acting system [11]. The fundamental data-structure of SNePS is a propositional semantic network. A SNePS network represents the beliefs of Cassie, the SNePS cognitive agent. A semantic network is abstract enough to represent both numeric and linguistic information. This is one of the reasons semantic networks have been used in models of arithmetic word-problem solving [1]. Numeric information is basically syntactic [8], but obtains a conceptual-role semantics [6] through integration with a system such as SNePS.

Cassie's acting system is SNeRE (the **SNePS Rational Engine** [3]). Acts in SNePS are either primitive or complex. Primitive acts are the fundamental repertoire of acts available to Cassie. Complex acts are composed of sets of primitive acts that are structured by control acts (i.e., acts for sequencing, iteration, and conditionals) and are bound to plans for performing them with the `ActPlan` predicate function. SNePS has a uniform representation for both conceptual and procedural knowledge so that all of Cassie's SNeRE plans are stored in the same network as her conceptual knowledge. Most importantly, Cassie can add beliefs to her network *while* she is acting, constituting an episodic memory of the act, which can be accessed during an act to prompt further action or after an act for answering questions.

SNIP, the **SNePS Inference Package**, enables inferential operations over

the beliefs in Cassie’s network. This allows SNePS to serve as a logical rule-based system. Commands for finding and deducing nodes (representing propositions) allow the user to ask Cassie questions and are the foundation for the path-based and rule-based inference techniques described in the next section.

### 3.1 Rule-Based Inference

At the highest level of abstraction, an `ActPlan` for an arithmetic act is set up as follows:

```
all(x,y)({Number(x),Number(y)} => ActPlan(Add(x,y),CountAdd(x,y))).
```

The semantics of such a statement is roughly: If you can infer that both  $x$  and  $y$  are numbers, then a plan for doing this generic arithmetic operation (e.g., `Add`) is this specific arithmetic operation (e.g., `CountAdd`). When Cassie is told to perform one of the generic arithmetic acts on two given inputs, say `perform Add(2,3)`, she produces the following sequence of inferences:<sup>1</sup>

- (i) Try to deduce a plan for the act `Add(2,3)`. This amounts to asking the open question `ActPlan(Add(2,3),?x)?`.
- (ii) Once Cassie finds an asserted node corresponding to `Add(x,y)`, she backchains and wonders whether `Number(2)` and `Number(3)` are asserted beliefs.
- (iii) Once Cassie determines that `Number(2)` and `Number(3)` both hold, she retrieves the plan corresponding to the act `Add(2,3)`. In this case, she finds `CountAdd(2,3)`.
- (iv) Cassie then attempts to `perform CountAdd(2,3)` by finding a plan for it.

This cycle of actions continues until Cassie completes the task. When a primitive action is reached, Cassie does not try to further decompose the action.

During the course of executing a specific arithmetic operation, Cassie needs to store the result using the `CountSum` case frame. We also want to be able to say that each of these specific arithmetic results is a result for the abstract act. Cassie is given the following rule:

```
all(x,y,z)(CountSum(x,y,z) => Sum(x,y,z)).
```

This tells Cassie that a `CountSum` is a `Sum per se`. Thus, the moment Cassie believes the result `CountSum(2,3,5)`, she will infer `Sum(2,3,5)` by forward chaining. This provides a specific and generic access point for arithmetic results.

These arithmetic acts force Cassie to perform successively simpler operations. Rule-based inference establishes a connection between three things: a general act (e.g., addition), a specific plan for performing that act (e.g., count-addition), and a specific performance of that plan (e.g., the count-addition of

<sup>1</sup> Cassie also checks whether an act has any preconditions and any effects. Since the arithmetic operations involve mental acts only, we omit these inferences.

2 and 3). Since Cassie explicitly deduces propositions at the knowledge level using this technique, rule-based inference can be seen as a model of “conscious” inference (i.e., Cassie is attending to the information that triggers the inference).

### 3.2 Path-Based Inference

Paths in SNePS networks are ordered sequences of arc-labels between nodes. Paths are specified using a syntax similar to regular expressions [9]. Path-based techniques can be used as a model of unconscious inference for SNePS agents [10]. A relation between nodes in a given network may be inferred by the presence of a certain path between those nodes. This is an “unconscious” activity, because the newly inferred relation is added to the agent’s belief space without an explicit knowledge level deduction.

For each new number generated by a counting procedure, Cassie builds a **Successor** relation to hold between that number and its immediate predecessor. Using path-based inference, this link can be exploited to generate the **GreaterThan** relation, which holds between each new number generated and *all* preceding numbers.

Path-based inference takes advantage of the fact that Cassie’s knowledge is stored in a semantic network, in which a node’s “meaning” is determined by its position in the network relative to other nodes [5,6]. This type of inference can also be used in SNePS to determine the conceptual-role semantics for vague concepts such as “number”. To define a concept such as “number” in the context of a completed arithmetic activity, a series of path-based inferences can be performed to give the elements of the class number, to give the acts with argument type number, and to give the operations that have thus far resulted in numbers.

This technique can also be used to provide semantic meaning for arithmetic procedures. The possibility of treating an arithmetic operation as either a concrete object (as a node) and a process (as an **ActPlan**) during inference is an important result of using SNePS to model mathematical understanding (see [7]).

## 4 Conclusion and Further Research

Inference is central to explanations of all sorts. We have seen that for a domain such as arithmetic, in which complex procedures can be rigidly built up from simpler procedures, the knowledge inferred during an agent’s action can provide a justification for that action. For a computational agent, the choice of a knowledge-representation system determines the methods of inference that can be applied. SNePS is particularly useful for implementing my theory because it can be used as a logical rule-based system (by applying node-based inference) or as a traditional semantic network (by applying path-based

inference). The agent implementation is still in its early stages and there are several avenues to pursue.<sup>2</sup>

## References

- [1] Greeno, J. G., *Instructional Representations Based on Research about Understanding*, in: *Cognitive Science and Mathematics Education*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1987 pp. 61–88.
- [2] Hiebert, J. and P. Lefevre, *Conceptual and Procedural Knowledge in Mathematics: An Introductory Analysis*, in: *Conceptual and Procedural Knowledge: The Case of Mathematics*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1986 pp. 1–27.
- [3] Kumar, D., *From Beliefs and Goals to Intentions and Actions: An Amalgamated Model of Inference and Acting*, Technical Report 94-04, Department of Computer Science, State University of New York at Buffalo (1994).
- [4] Lakoff, G. and R. Núñez, “Where Mathematics Comes From: How the Embodied Mind Brings Mathematics Into Being,” Basic Books, New York, NY, 2000.
- [5] Quillian, M. R., *Semantic Memory*, in: *Semantic Information Processing*, MIT Press, Cambridge, MA, 1968 pp. 216–270.
- [6] Rapaport, W. J., *Holism, Conceptual-Role Semantics, and Syntactic Semantics*, *Minds and Machines* **12** (2002), pp. 3–59.
- [7] Sfard, A., *On The Dual Nature of Mathematical Conceptions: Reflections on Processes and Objects as Different Sides of the Same Coin*, *Educational Studies in Mathematics* **22** (1991), pp. 1–36.
- [8] Shapiro, S. C., *Representing numbers in semantic networks: prolegomena*, Proceedings of the 5th International Joint Conference on Artificial Intelligence (1977), p. 284.
- [9] Shapiro, S. C., *Path-based and node-based inference in semantic networks*, ACM, New York, NY, 1978 pp. 219–225.
- [10] Shapiro, S. C., *Cables, paths and “subconscious” reasoning in propositional semantic networks*, in: J. F. Sowa, editor, *Principles of Semantic Networks*, Morgan Kaufmann, San Mateo, CA, 1991 pp. 137–156.
- [11] Shapiro, S. C. and W. J. Rapaport, *The SNePS Family*, *Computers and Mathematics with Applications* **23** (1992), pp. 243–275.

---

<sup>2</sup> I would like to thank William J. Rapaport and Stuart C. Shapiro for reading earlier drafts of this paper.

# Towards a Logical Foundation of Semantic Networks – A Typology of Descriptive Means for Semantic Inference

Hermann Helbig, Ingo Glöckner  
Intelligent Information and Communication Systems  
FernUniversität in Hagen, Germany

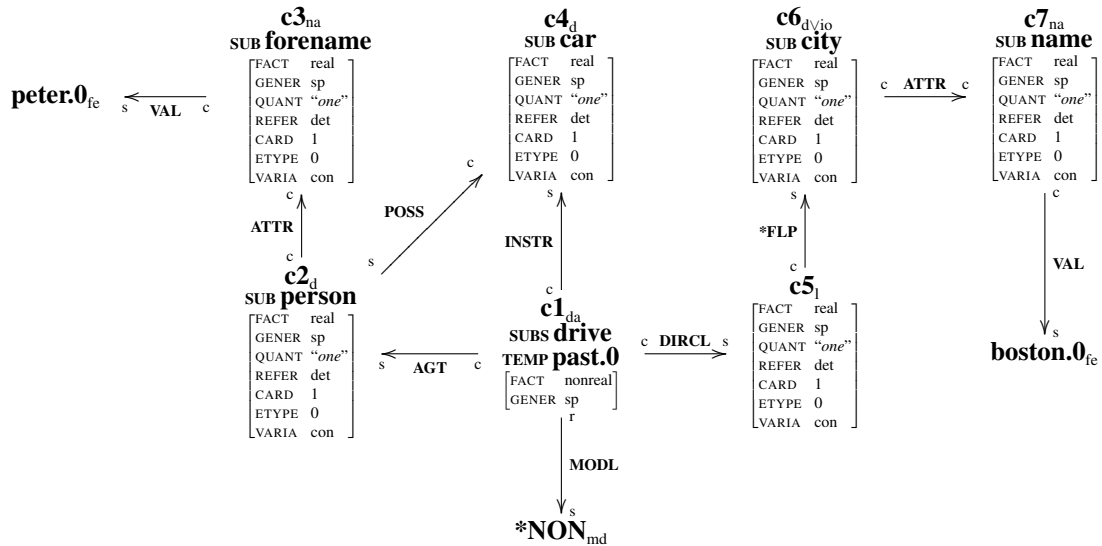
## Abstract

Semantic Networks (SN) are a knowledge representation paradigm especially suited for the meaning representation of natural language expressions. In order to clearly define their basic constructs, the relations and functions used in a semantic network must be given a logical characterization. The paper exemplifies this strategy for Multilayered Extended Semantic Networks (the so-called MultiNet paradigm). In particular, it is shown that the axioms characterizing the logical properties of the expressional means of an SN have to be classified according to different criteria which are connected with specific types of inference.

## 1 Introduction

Semantic Networks have a long tradition as a paradigm for representing cognitive structures, starting with Quillian [9]. As to the logical underpinning of relations and functions used in such networks, we find a logically oriented and a more linguistically oriented approach. The works in the first line, like Shapiro's SNePS [10], and Brachmann's KL-ONE [1], have a clear logical foundation, but none of them give a systematic and complete description of the relations and functions constituting an SN. By contrast, linguistically oriented work normally discusses selected semantic relations (so-called cognitive roles or theta-roles) in greater detail [5], however not on a formal logical level. Moreover the proposed relations and roles are not contrasted with each other to form a balanced system of expressional means. In SNePS [10], for example, the guideline for choosing the appropriate relations is deliberately given the status of a recommendation only. To meet the requirements on a knowledge representation formalism useful for NLP (especially the universality, homogeneity, and interoperability criteria [4, Chap. 1]), one needs a commitment on a clearly defined repertory of expressional means. Multilayered Extended Semantic Networks (the MultiNet paradigm [4]) were designed to fulfill these requirements, by fixing a set of expressional means and formalizing axioms which describe these functions and relations. The formalism is comprehensively documented and successfully used in NLP applications, e.g. for describing large semantically based computational lexica [3], and as the backbone of NL interfaces [7]. Its application in the

Figure 1: “It is not true that Peter didn’t drive to Boston with his car”



InSicht question-answering system [2] has been successfully evaluated in CLEF 2004, with a MultiNet knowledge base generated from 4,9 Mio sentences.

MultiNet extends simple semantic networks by the following features: (1) Every node is labeled by a sort from a predefined ontology of sorts and by bundles of layer attributes. (2) MultiNet admits functions and relations of arbitrary arity. (3) The arcs (relations) are formally characterized by associated axioms. (4) Subnetworks can be encapsulated to form concepts of higher order which can be connected to other concepts by relations and functions. (5) The relationships in the network are assigned a knowledge type and thus marked as categorically valid (*c*), prototypically valid (*p*), modally restricted (*r*), or situationally bounded (*s*) with regard to each argument, as shown in Fig. 1.

## 2 The Expressional Means of MultiNet

To characterize an SN, we need a precise specification of the relations corresponding to the arcs (links). Our formalism provides about 140 relations and functions characterized on the basis of a uniform schema and by logical axioms. Table 1 sketches the relations and functions needed in this paper. MultiNet distinguishes 45 sorts of conceptual entities used to define the signatures of relations and functions [4, Sect. 17.1]. Some of these sorts are explained in Table 1. The sorts are also needed to constrain the applicability of logical rules. For example, the law of double negation holds for semantically total properties (sort [tq]) like *dead* and its negation *alive*, where *not alive* means the same as *dead*. Another sort [gq] is used for gradable properties like *friendly* and *unfriendly*. Although *unfriendly* means *not friendly*, the law of double negation does not hold in this case, i.e. if someone is *not unfriendly* this does not mean that the person is *friendly*.

MultiNet not only classifies conceptual entities by their sorts but also by the values of



| Relation | Signature                              | Short Characteristics                                     |
|----------|----------------------------------------|-----------------------------------------------------------|
| AFF      | $si \times [o \cup si]$                | C-Role – Affected object                                  |
| AGT      | $si \times o$                          | C-Role – Agent                                            |
| ANTE     | $[t \cup si] \times [t \cup si]$       | Temporal successorship                                    |
| ATTR     | $o \times at$                          | Specification of an attribute                             |
| AVRT     | $si \times o$                          | C-Role – Averting/Turning away from an object             |
| CAUS     | $si' \times si'$                       | Relation between cause and effect (Causality)             |
| CIRC     | $si \times si$                         | Relation between situation and circumstance               |
| COMPL    | $p \times p$                           | Complementarity relation                                  |
| DIRCL    | $[si \cup o] \times l$                 | Relation specifying a direction                           |
| FIN      | $si \times [t \cup si]$                | Relation between a situation and its temporal end         |
| LOC      | $[o \cup si] \times l$                 | Relation specifying the location                          |
| MIN      | $qn \times qn$                         | Smaller-than relation                                     |
| MODL     | $si \times md$                         | Relation specifying a restricting modality                |
| OBJ      | $si \times [o \cup si]$                | C-Role – Neutral object of a situation                    |
| ORNT     | $si \times o$                          | C-Role – Orientation of a situation toward something      |
| PARS     | $co \times co$                         | Part-whole relationship                                   |
| PROP     | $o \times p$                           | Relation between object and property                      |
| SUB      | $o \times \bar{o}$                     | Relation of conceptual subordination (for objects)        |
| SUBS     | $si \times \bar{si}$                   | Relation of conceptual subordination (for situations)     |
| TEMP     | $si \times [t \cup si]$                | Relation specifying the temporal embedding of a situation |
| VAL      | $at \times [o \cup qn \cup p \cup fe]$ | Relation between an attribute and its value               |

Table 1: Strongly abbreviated description of relations used in this paper. Explanation of sorts: objects  $o$  include concrete objects  $co$  (*house*) and attributes  $at$  (*height*); situations  $si$  (*write*); locations  $l$  (*here*), times  $t$  (*now*), modal descriptors  $md$  (*impossible*); properties  $p$  (*dead*), quantifiers and measurements  $qn$  (*many, two litres*), formal entities  $fe$  (*figures or names*). The notation  $si'$  demands [FACT = *real*], and the notation  $\bar{si}$  demands [GENER = *ge*].

six so-called ‘layer attributes’. Here we are concerned only with two of these attributes: **Facticity**. We discern three kinds of facticity: [FACT=*real*] for existing entities (*Eiffel tower*), [FACT=*non*] for non-existing entities (*the light ether*), and [FACT=*hypo*] for hypothetical entities (*quarks*). Apart from the *extensional negation* expressed by a non-existing situation with [FACT=*non*], MultiNet supports the *intensional negation* of a situation  $s$ , expressed by the relation ( $s$  MODL \*NON). Both types occur in the example shown in Fig. 1. Facticity must be anchored in the logical language since special inference rules apply to hypothetical and non-existing objects.

**Genericity**. The GENER attribute (degree of generality) divides the world of concepts into generic objects with [GENER=*ge*] (*house*) and specific objects with [GENER=*sp*] (*<my house>*). In this way, assertions about the generic concept can be clearly separated from assertions about instances of that concept. Generic concepts are also needed to model prototypical knowledge. Consider “*Lions feed on antelopes*”. A modeling by a universal quantifier ranging over all lions would be inadequate because the sentence expresses only default knowledge.

### 3 A Typology of Axioms for Inferences over an SN

While a logical expression is either true or false in first-order logic (FOL), a semantic formalism dealing with NL must support different degrees of reliability. Moreover, logical calculi normally do not give a clue how to use the axioms in an effective inference strategy. These considerations suggest the following cross-classification of axioms.

#### 3.1 Conceptually Bound vs. Conceptually Non-bound Axioms

**R-axioms.** From a syntactical point of view, there are two types of expressions describing axiomatic knowledge. The first type contains no lexical constants but only relation and function symbols (apart from logical signs). These expressions are called *conceptually non-bound* or *R-Axioms*. The following R-Axiom connects causality and time, saying that effects never take place before the cause:  $(x \text{ CAUS } y) \rightarrow \neg(y \text{ ANTE } x)$  (1) Other examples are given by axioms (4), (5), (7) below. Axioms which are conceptually not bound have to be treated with care by the reasoner, since an R-axiom for a relation R can be applied in inferences over the SN wherever R is involved (global effect). Inter alia, R-axioms serve to express the symmetry or transitivity of relations, i.e. properties which are difficult to handle efficiently.

**B-axioms.** Axioms containing the representative of at least one concept are called *conceptually bound* or B-axioms. Thus, with every selling act  $s$  there is a buying act  $b$  entailed by  $s$ . The corresponding relationship is given by the following axiom:

$$(s \text{ SUBS } \textit{sell}) \wedge (s \text{ AGT } a) \wedge (s \text{ OBJ } o) \wedge (s \text{ ORNT } d) \rightarrow \\ \exists b(b \text{ SUBS } \textit{buy}) \wedge (b \text{ OBJ } o) \wedge (b \text{ AVRT } a) \wedge (b \text{ AGT } d) \quad (2)$$

Another example of a B-axiom is (6) which contains only one concept. Such B-axioms have only a local effect, i.e. they are applied only in those cases where one concept has to be connected to another during the inference process. Here, we meet the **Frame Problem** in Artificial Intelligence: In a B-axiom like (2), only the change of participant roles (like AGT, AFF, AVRT, and OBJ) is specified, but nothing is said about the local, temporal and circumstantial embedding of the main situation (mainly represented by LOC, TEMP, and CIRC, resp.) The transfer of these specifications must be handled by axiom schemata for classes of concepts: While the temporal specification of a selling act like  $s$  in (2) transfers unchanged to  $b$ , there is no such transfer of the specification  $(s_1 \text{ TEMP } t_1)$  of a sending act  $s_1$  to the corresponding receiving act  $s_2 = sk(s_1)$ . For the latter class we have:

$$(s_1 \text{ SUBS } \langle \textit{send-act} \rangle) \wedge (s_1 \text{ TEMP } t_1) \wedge \\ (sk(s_1) \text{ SUBS } \langle \textit{receive-act} \rangle) \wedge (sk(s_1) \text{ TEMP } t_2) \rightarrow (t_1 \text{ ANTE } t_2) \quad (3)$$

#### 3.2 Categorically vs. Prototypically Valid Axioms

**Categorically Valid Axioms.** It seems to be a contradiction to speak of axioms which are restricted in their validity. But, if we want to formalize natural language semantics, we must also account for *prototypical* regularities.

The following axiom expresses knowledge which is categorically valid:

$$(p_1 \text{ COMPL } p_2) \rightarrow (o \text{ PROP } p_1) \vee (o \text{ PROP } p_2) \quad (4)$$

Axiom (4) states that one from two complementary properties (if applicable at all) must hold. It is obvious that there is no exception from this rule.

**Prototypically Valid Axioms.** By contrast, rule (5), governing the inheritance of the part-whole relationship within the SUB hierarchy, has only the status of default (or prototypically valid) knowledge:

$$(d_1 \text{ SUB } d_2) \wedge (d_3 \text{ PARS } d_2) \rightarrow \exists d_4 [(d_4 \text{ SUB } d_3) \wedge (d_4 \text{ PARS } d_1)] \quad (5)$$

It is a good assumption that a conceptual object subordinated to a generic object inherit known parts from the latter. However, there are exceptions. While ships normally have a keel, there are also ships which have not.

Categorically valid axioms lead to monotonic reasoning, while prototypically valid axioms call for nonmonotonic reasoning. The standard approach to default reasoning based on a truth-maintenance system does not scale up, though. In MultiNet, we warrant that every deduction step involving a default again produces only default knowledge. The newly generated default knowledge has to be checked for *local* contradictions in a neighborhood of the concepts involved. Semantic networks can help defining such neighborhoods as their link structure gives a natural notion of vicinity for concepts.

### 3.3 Deductive Axioms vs. Destructive Axioms

**Deductive Axioms.** Many axioms, like (1) through (5), can be used in a deductive process to derive new knowledge, given by the conclusion, provided that the premise be fulfilled. The important feature of monotonic deduction is that no piece of knowledge in the knowledge base must ever be retracted.

**Destructive Axioms.** There are also axiomatic regularities which not only generate new knowledge but also cancel earlier knowledge. Into this class of ‘destructive’ axioms we number the derivation of the temporal end of a situation  $s$ :

$$(e \text{ SUBS } end) \wedge (e \text{ AFF } s) \wedge (e \text{ TEMP } t) \rightarrow (s \text{ FIN } t) | \text{DEL } (s \text{ TEMP } \_) \quad (6)$$

Thus if an activity  $e$  ends a situation  $s$  at time  $t$ , then a new relation FIN for  $s$  must be established and the earlier specification of  $s$  by the relation TEMP must be deleted. While the first type of axioms can be treated by symbolic derivations, the latter type requires actions on the knowledge base like deleting arcs.

### 3.4 Epistemically Restricted vs. Non-restricted Axioms

**Epistemically Restricted Axioms.** There are axioms which are epistemically restricted in the sense that their validity is only warranted within a certain epistemic or cognitive context. A typical example is the restricted transitivity of CAUS:

$$(k_1 \text{ CAUS } k_2) \wedge (k_2 \text{ CAUS } k_3) \rightarrow (k_1 \text{ CAUS } k_3) \quad (7)$$

This axiom is connected with a fading effect preventing infinite prolongation of causal chains by a presumed (but not strongly valid) transitivity of CAUS. This effect is due to the so-called INUS-conditions [8], i.e. humans asserting a causal relation emphasize

a certain cause and neglect other necessary conditions for this relationship.

**Epistemically Non-restricted Axioms.** For most axioms no epistemically motivated restriction can be observed. In particular, the transitivity of conceptual subordination (8) and of spatial inclusion (9) hold unconditionally:

$$(o_1 \text{ SUB } o_2) \wedge (o_2 \text{ SUB } o_3) \rightarrow (o_1 \text{ SUB } o_3) \quad (8)$$

$$(o \text{ LOC } (*\text{IN } m)) \wedge (m \text{ LOC } (*\text{IN } n)) \rightarrow (o \text{ LOC } (*\text{IN } n)) \quad (9)$$

For *epistemically restricted* axioms, we propose the use of built-in procedures which treat borderlines of epistemic or functional levels by special parameters for controlling the inference process.

## 4 Conclusion

The MultiNet formalism is intended for the semantic representation of unrestricted language and thus supposed to represent the facticity status, degree of generality, modal embedding, and other characteristics of NL concepts. The meaning of the relations and functions on which the formalism is based, can be made precise by axioms which capture their expected behaviour. These axioms differ with respect to the classificatory dimensions of categoricity, conceptual boundedness, and epistemic restriction. We have shown that these dimensions also affect the validity and efficiency of inference.

## References

- [1] R. J. Brachman and J. G. Schmolze. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2):171–216, 1985.
- [2] S. Hartrumpf. Question answering using sentence parsing and semantic network matching. In *Proc. of CLEF 2004*, pages 385–392, Bath, England, Sept. 2004.
- [3] S. Hartrumpf, H. Helbig, and R. Osswald. The semantically based computer lexicon HaGenLex. *Traitement automatique des langues*, 44(2):81–105, 2003.
- [4] H. Helbig. *Knowledge Representation and the Semantics of Natural Language*. Springer, Berlin, 2006.
- [5] R. Jackendoff. *Semantic Structures*. MIT Press, Cambridge, Massachusetts, 1990.
- [6] F. Lehmann, editor. *Semantic Networks in Artificial Intelligence*. Pergamon Press, 1992.
- [7] J. Leveling and H. Helbig. A robust natural language interface for access to bibliographic databases. In *Proceedings of SCI 2002*, pages 133–138, Orlando, Florida, July 2002.
- [8] J. L. Mackie. *The Cement of Universe*. Oxford University Press, Oxford, 1974.
- [9] M. R. Quillian. Semantic memory. In M. Minsky, editor, *Semantic Information Processing*, pages 227–270. MIT Press, Cambridge, Massachusetts, 1968.
- [10] S. C. Shapiro and W. J. Rapaport. The SNePS family. In Lehmann [6], pages 243–275.

# The ALLIGATOR Theorem Prover for Dependent Type Systems: Description and Proof Sample

Paul Piwek

*Centre for Research in Computing  
The Open University, Milton Keynes, UK  
p.piwek@open.ac.uk*

---

## Abstract

This paper introduces the ALLIGATOR theorem prover for Dependent Type Systems (DTS). We start with highlighting a number of properties of DTS that make them specifically suited for computational semantics. We then briefly introduce DTS and our implementation. The paper concludes with an example of a DTS proof that illustrates the suitability of DTS for modelling anaphora resolution.

---

## 1 Introduction

Automated *symbolic* inference requires a formal language as the substratum for reasoning. Blackburn and Bos ([7]) make a good case for the use of First Order Predicate Logic (FOPL) in computational semantics, citing both practical (availability of high performance theorem provers and to a lesser extent model builders) and theoretical reasons (they discuss a range of interesting phenomena which can be dealt with in FOPL).

We agree with the idea that FOPL is a good starting point, but also think that for computational semantics to develop further as a field, extensions going beyond FOPL should be actively explored. In this paper, a research tool is described that takes such explorations in one particular direction. The tool – ALLIGATOR – is a theorem prover for Dependent Type Systems (DTS) [4,5]. The Sicstus Prolog source code of this prover is available, free of charge, for research purposes ([18]). DTS are an attractive option for computational semantics for a number of reasons:

- (i) DYNAMIC POTENTIAL (cf. [15]): The notion of a *context* that is built up *incrementally* is inherent to DTS.
- (ii) FLEXIBILITY: By varying a limited number of parameters, it is possible to switch from, for example, propositional to predicate logic, or first order to higher order logics. Additionally, although the basic underlying logic is constructive, DTS allows for the flexible use of axioms to regain full classical

- logic, or more fine-grained alternatives. For example, it is possible to specify for individual predicates whether they are bivalent.
- (iii) **EXTENSIBILITY:** A DTS-context includes what is known as the signature in FOPL. Consequently, the signature can be extended incrementally, making it possible to model the acquisition of new concepts by language users.
  - (iv) **PROOF-OBJECTS:** In DTS, Gentzen-style natural deduction proofs are first-class citizens. This gives us the following advantages: (a) *Reliability:* It allows us to heed the *de Bruijn criterion* for reliable proof systems: “A proof assistant satisfies the de Bruijn criterion if it generates ‘proof-objects’ (of some form) that can be checked by an easy algorithm.” (cited from [5]) (b) *Naturalness:* DTS proofs correspond with natural deduction proofs. This is of interest if one is concerned with models of human reasoning in natural language understanding. In psychology, some schools of thought argue that natural deduction is a good approximation of human reasoning (see, e.g., [21]). (c) *Relevance:* Proof objects can help to identify proofs which are valid but spurious in the sense that they do not really consume their premises (see [14]). (d) *Justification of behaviour:* Explicit proof objects provide direct access to the justifications that an agent has for the conclusions and the interpretations that it constructs. This is particularly useful for dialogue agents that need to respond to utterances of other agents. Such responses can themselves again be queried, for example, through clarificatory questions (*cf.* [22]) and why questions (A:*p*, B: no,  $\neg p$ , A: Why  $\neg p$ ?). In order to respond appropriately, the agent needs to access its own background knowledge and how it was used to draw conclusions. DTS proof objects provide a compact representation of this information.
  - (v) **APPLICATIONS:** DTS-style analyses exist for a wide range of linguistic phenomena including donkey sentences ([23]), anaphoric expressions and temporal reference ([20]), belief revision ([8]), bridging anaphora ([19]), clarification ellipsis ([10]), metonymy ([9]), inter-agent communication, knowledge and observation ([1]), ontological reasoning for feedback dialogues ([6]), and human-machine dialogue ([2]). Additionally, there is research on relating DTS proof-theoretic natural language semantics to model-theoretic approaches ([12]), and there are studies employing the related formalism of labelled deduction to natural language semantics ([16]). In 2005, the 2<sup>nd</sup> Workshop on Lambda-Calculus, Type Theory, and Natural Language took place at King’s College London ([11]).

We concede that none of the properties we have listed is on its own unique to DTS. However, to the best of our knowledge, no extant logical calculus combines all these properties in a *single system* with well-understood meta-mathematical properties (DTS play a central role in theoretical computer science, see [4]).

## 2 Dependent Type Systems

DTS come in a wide variety of flavours and variations. All these systems share, however, two features: a *typing system* and a notion of *dependency*. Firstly, DTS are

*type systems*. That is, given a set of assumptions  $\Gamma$ , also known as the *context*, they provide rules for determining whether a particular object, say  $a$ , belongs to a given type, say  $t$ . We write  $\Gamma \vdash a : t$ , if, given the context  $\Gamma$ ,  $a$  is of type  $t$ , i.e.,  $a$  *inhabits* type  $t$ . The objects that are classified using type systems are (normalizing) terms of the  $\lambda$ -calculus.  $\Gamma$  is a sequence of statements  $x_1 : t_1, \dots, x_n : t_n$  (with  $n \geq 0$ ).

*Dependency* is the second feature of DTS, and it comes in two forms. First, there is dependency between statements in the context: in order to use a type  $t_k$  to classify an object  $x_k$ , this type  $t_k$  needs to have been introduced in that part of the context that *precedes* it or  $t_k$  has to be a sort. In other words,  $t_k$  can only be used if (1) it itself inhabits a type or can be constructed from other types that are available in the context preceding it, or (2) it belongs to a fixed and usually small set of designated types that are called *sorts*. Because sorts need no preceding context, they make it possible to keep contexts finite.

Second, there is a variety of dependency that occurs *inside* types. Since type systems are used to classify terms of the  $\lambda$ -calculus, they can also deal with functions. A function  $f$  from objects of type  $t_1$  to objects of type  $t_2$  inhabits the function type  $t_1 \rightarrow t_2$ . *Dependent* function types are a generalization of function types: a dependent function type is a function type where the range of the function changes depending on the object to which the function is applied. The notation for dependent function types is  $\Pi x : A. B$  (we also use our own alternative ‘arrow notation’:  $[x : A] \Rightarrow B$ ). If we apply an inhabitant of this function type, say  $f$ , to an object of type  $A$ , then the resulting object  $fa$  ( $f$  applied to  $a$ ) is of type  $B$ , but with all free occurrences of  $x$  in  $B$  substituted with  $a$  (that is, the type of  $fa$  is  $B[x := a]$ ).

One way to make the leap from type systems to logic is as follows. From a logical point of view, we are interested in propositions as the constituents of deductive arguments. In classical logic, one focuses on judgements of the following form: the truth of proposition  $q$  follows/can be derived from the truth of the propositions  $p_1, \dots, p_n$ . We reason from the truth of the premises to the truth of the conclusion. To do logic in a DTS, we move from *truth* to *proof*: we, now, reason from the proofs that we (assume to) have for the premises to a proof for the conclusion. In other words, we are interested in judgements of the following form:  $a$  is proof of proposition  $q$  follows/can be derived assuming that  $a_1$  is a proof of  $p_1$ ,  $a_2$  is a proof of  $p_2, \dots$ , and  $a_n$  is a proof  $p_n$ . Such a judgement can be formalized in a DTS as  $a_1 : p_1, \dots, a_n : p_n \vdash a : p$ . Thus, we read  $a : p$  as ‘ $a$  is a proof for  $p$ ’. Thus, we model proofs as ( $\lambda$ -calculus) terms and propositions as (a certain class of) types in DTS. This is known as the Curry-Howard-de Bruijn embedding.

The embedding is grounded in the Brouwer-Heyting-Kolmogorov interpretation of proofs as *constructions*; e.g., a proof for a conditional  $p \rightarrow q$  is identified with a method that transforms a proof of  $p$  into a proof for  $q$ . In a DTS, this is formalized by modelling the proof  $f$  for a type  $p \rightarrow q$  as a function from objects of type  $p$  to objects of type  $q$ , such that if  $a$  is a proof of  $p$ , then  $f$  applied to  $a$  is a proof of  $q$  (i.e.,  $fa : q$ ). Universal quantification is dealt with along the same lines. In a DTS, the counterpart for universal quantification is the dependent function type. In particular,  $\forall x \in A : P(x)$  becomes  $(\Pi x : A. Px)$ . A proof for this type is a

function  $f$  which, given any object  $a : A$ , returns the proof  $fa$  for  $Pa$ .

PURE TYPE SYSTEMS (PTS; [4]) are of particular interest, because of their generality. With a small number of parameters, PTS can be tailored to match a wide variety of DTS. ALLIGATOR implements an extension of PTS with  $\Sigma$  types.  $\Sigma$  types are also known as dependent product types and can be used to model  $\wedge$  and  $\exists$ .

### 3 System Architecture, Implementation and Proof Sample

There is no room for a detailed description of the system here, for that we refer to the documentation and code available at [18]. What we can offer is, firstly, a list of differences between ALLIGATOR and other DTS provers: (a) ALLIGATOR directly constructs proof objects for natural deduction proofs. Other provers for DTS typically work with internal representations that are only at the end of the reasoning process *translated* to natural deduction proof objects. For example, COCKTAIL ([13]) uses tableaux and translates these, whereas TPS ([3]) is based on the mating method. The handbook chapter by Barendregt and Geuvers on proof assistants for DTS ([5]) lists a number of further automated theorem provers, none of which works directly with proof objects. (b) ALLIGATOR was not developed with mathematical/program specification reasoning in mind, but rather for inferences in language interpretation. As a consequence, it has been streamlined to link up with notation and functionality relevant to computational semantics (specifically, allowing for notation which is close to [15] and omission of inductive types). (3) To the best of our knowledge, ALLIGATOR is the only automated theorem prover which directly conforms to the specification of Pure Type Systems ([4]), the most general and flexible kind of DTS (most DTS can be emulated in PTS; see [17] for an overview of DTS and their counterparts in PTS).

ALLIGATOR 1.0 has been implemented in Sicstus PROLOG and been tested with version 3.12.2 of Sicstus. An overview of the architecture is presented in Figure 1.a. Note that the system applies both forward and backward inferencing. Most of the forward inferencing takes place before backward inferencing (though some backward inferencing rules do also have forward inferencing component). Reduction of terms is also carried out mainly before backward inferencing. Inferencing is done with a flattened representations of DTS terms (the arrow notation). Proofs are checked at the end of the inferencing process for their correctness (the code for proof checking is separate from the theorem proving code).

Currently, ALLIGATOR has the status of an experimental research tool. It is intended for testing computational solutions to theoretically challenging problems in computational semantics. Scalability has, so far, not been given much attention, though it will obviously need to be addressed if the system is to be used in large-scale practical applications. Currently, the system is merely intended as a *baseline* and *starting point* for implementing efficient and effective proof search heuristics. We now conclude with an example of the use of ALLIGATOR.

The discourse ‘The barn contains a chain saw or a power drill. It . . .’ (p. 205 of [15]) poses a problem for the structural approach to anaphora resolution proposed in



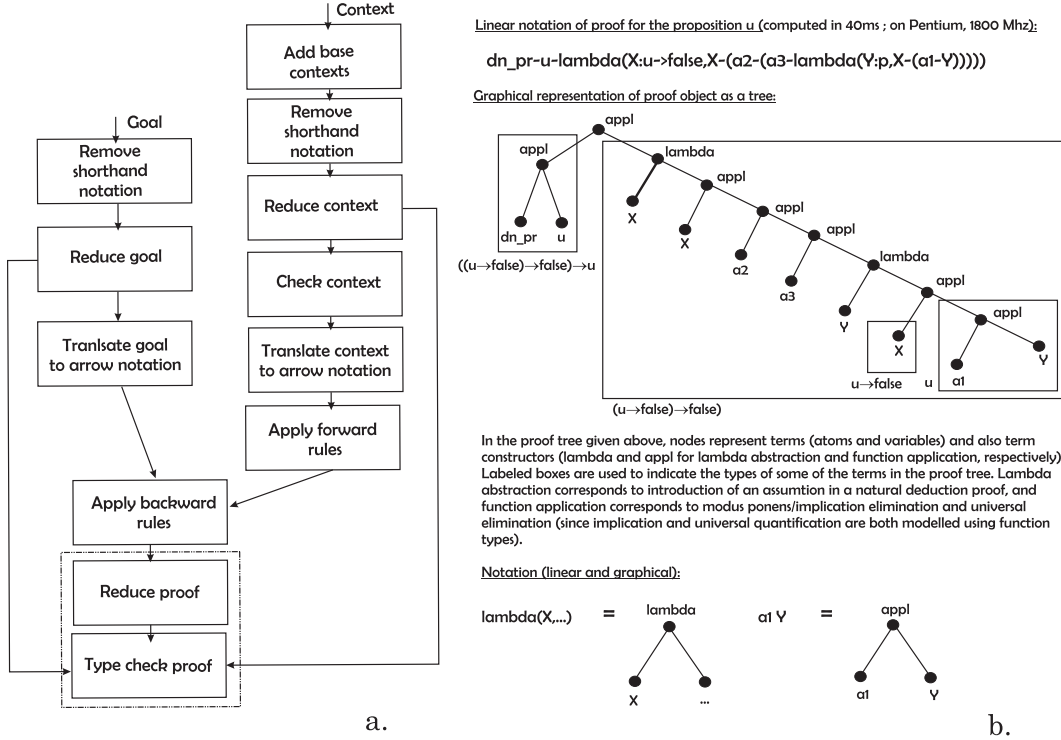


Fig. 1. Overview of ALLIGATOR 1.0 Architecture (a) and graphical representation of a proof-object (b)

[15]: the first sentence does not directly introduce an accessible discourse referent that can bind the pronoun ‘It’. Rather, an object (something that is either a power drill or a chain saw) needs to be *inferred* from a disjunction. An inferential account of anaphora resolution that can deal with such cases is presented in [19]. In a nutshell, the idea is that an anaphoric expression triggers a proof goal that needs to be filled with a proof from the context.

ALLIGATOR can construct an antecedent for ‘It’ from the context (for lack of space, our formalization is in propositional logic). Firstly, the relevant propositions need to be available:  $false:prop$  (*false* is a proposition),  $p:prop$  (*there is a chain saw* is a proposition),  $q:prop$  (*there is a power drill* is a proposition),  $u:prop$  (*there is something* is a proposition). Given these propositions, we can now introduce background information such as  $a1:p \rightarrow u$  (roughly, *chain saws are things*) and  $a2:q \rightarrow u$  (*power drills are things*). Although DTS are constructive, we can engage in classical reasoning by including the double negation rule in our background knowledge:  $dn\_pr:([P:prop] \Rightarrow (((P \rightarrow false) \rightarrow false) \rightarrow P))$ . It involves higher order quantification over all propositions  $P$ . Finally, assume that the context has been updated with the information expressed by the sentence ‘The barn contains a chain saw or a power drill’, with disjunction modelled using implication and negation:  $a3:(p \rightarrow false) \rightarrow q$ . Given this information, ALLIGATOR can generate the proof/antecedent for ‘It’ (modelled here as  $u$ ) given in Figure 1.b.

**Acknowledgements** We would like to thank the anonymous reviewers of ICoS-5 for their helpful comments and suggestions.

## References

- [1] Ahn, R., “Agents, Objects and Events: A computational approach to knowledge, observation and communication,” Ph.D. thesis, Eindhoven University of Technology (2001).
- [2] Ahn, R., R. Beun, T. Borghuis, H. Bunt and C. van Overveld, *The DenK-architecture: a fundamental approach to user-interfaces*, Artificial Intelligence Review Journal **8** (1994), pp. 431–445.
- [3] Andrews, P., M. Bishop, S. Issar, D. Nesmith, F. Pfennig and H. Xi, *TPS: A Theorem-Proving System for Classical Type Theory*, Journal of Automated Reasoning **16** (1996), pp. 321–353.
- [4] Barendregt, H., *Lambda Calculi with Types*, in: *Handbook of Logic in Computer Science*, **2**, Clarendon Press, Oxford, 1992 .
- [5] Barendregt, H. and H. Geuvers, *Proof-assistants using Dependent Type Systems*, in: *Handbook of Automated Reasoning*, Elsevier, 2001.
- [6] Beun, R., R. van Eijk and H. Prüst, *Ontological Feedback in Multiagent Systems*, in: N. Jennings, C. Sierra, L. Sonenberg and M. Tambe, editors, *International Joint Conference on Autonomous Agents and Multiagent Systems* (2004), pp. 110–117.
- [7] Blackburn, P. and J. Bos, *Computational Semantics*, Theoria **18** (2003), pp. 27–45.
- [8] Borghuis, T. and R. Nederpelt, *Belief Revision with Explicit Justifications: An Exploration in Type Theory*, CS-Report 00-17, Eindhoven University of Technology (2000).
- [9] Bunt, H. and L. Kievit, *Agent-dependent metonymy in a context-change model of communication*, in: *Computing Meaning II*, Studies in Linguistics and Philosophy **77**, Kluwer Academic Publishers, Dordrecht, 2001 pp. 75–95.
- [10] Cooper, R. and J. Ginzburg, *Clarification ellipsis in dependent type theory*, in: J. Bos and C. Matheson, editors, *Proceedings of Edilog, the 6th Workshop on the Semantics and Pragmatics of Dialogue*, University of Edinburgh, 2002.
- [11] Fernández, M., S. Lappin and C. Fox, editors, “Procs. of Lambda Calculus, Type Theory and Natural Language Workshop,” King’s College, London, 2005.
- [12] Fernando, T., *A type reduction from proof-conditional to dynamic semantics*, Journal of Philosophical Logic (2001), pp. 121–153.
- [13] Franssen, M. and H. de Swart, *Cocktail: A Tool for Deriving Correct Programs*, Rev. R. Acad. Cien. Serie A. Mat. **98** (2004), pp. 95–111.
- [14] Helman, G., “Restrictions on Lambda Abstraction and the Interpretation of Some Non-Classical Logics,” Ph.D. thesis, University of Pittsburgh (1977).
- [15] Kamp, H. and U. Reyle, “From Discourse to Logic,” Kluwer Academic Publishers, Dordrecht, 1993.
- [16] Kempson, R., W. Meyer-Viol and D. Gabbay, *Syntactic Computation as Labelled Deduction: WH a case study*, in: R. Borsley and I. Roberts, editors, *Syntactic Categories*, Academic Press, 2000.
- [17] Laan, T., “The Evolution of Type Theory in Logic and Mathematics,” Ph.D. thesis, Eindhoven University of Technology (1997).
- [18] Piwek, P., *ALLIGATOR Theorem Prover Home Page*, [mcs.open.ac.uk/pp2464/alligator](http://mcs.open.ac.uk/pp2464/alligator) (2006).
- [19] Piwek, P. and E. Kraemer, *Presuppositions in Context: Constructing Bridges*, in: P. Bonzon, M. Cavalcanti and R. Nossum, editors, *Formal Aspects of Context*, Applied Logic Series **20**, Kluwer Academic Publishers, Dordrecht, 2000.
- [20] Ranta, A., “Type-Theoretical Grammar,” Clarendon Press, 1994.
- [21] Rips, L., “The Psychology of Proof: Deductive Reasoning in Human Thinking,” The MIT Press, Cambridge, Massachusetts, 1994.
- [22] Stone, M., *Specifying Generation of Referring Expressions by Example*, in: *Procs AAAI Spring Symposium on Natural Language Generation in Spoken and Written Dialogue*, Stanford, 2003.
- [23] Sundholm, G., *Proof Theory and Meaning*, in: *Handbook of Philosophical Logic III*, D. Reidel, 1986 pp. 471–506.