

Proceedings of the 1st Workshop on Natural Language Processing and Automated Reasoning (NLPAR) 2013

Editors: Chitta Baral¹ and Peter Schüller²

¹ Arizona State University, USA

² Sabanci University, Turkey

Preface

This volume contains the papers presented at NLPAR2013: Workshop on Natural Language Processing and Automated Reasoning 2013 held on September 15, 2013 in A Corunna, Spain.

The NLPAR Workshop received 6 submissions from different international institutions and research communities. Each submission was reviewed by 3 program committee members. The committee decided to accept 5 papers. The program also includes 2 invited talks.

The organizing committee wants to thank the Workshop Chair of LPNMR, Marcello Balduccini, and the Program Chairs of LPNMR, Pedro Cabalar and Tran Cao Son, for their support in embedding this workshop into the LPNMR conference organization.

Peter Schüller acknowledges support from Sabanci University. This workshop was managed using EasyChair.

September 15, 2013
A Corunna, Spain

Chitta Baral
Peter Schüller

Table of Contents

Some Recent Advances in Answer Set Programming (from the Perspective of NLP)	1
<i>Marcello Balduccini</i>	
Three Lessons in Creating a Knowledge Base to Enable Reasoning, Explanation and Dialog	7
<i>Vinay Chaudhri, Nikhil Dinesh and Daniela Inglezan</i>	
Qualitative Analysis of Contemporary Urdu Machine Translation Systems	27
<i>Asad Abdul Malik and Asad Habib</i>	
The NL2KR System	37
<i>Chitta Baral, Juraj Dzifcak, Kanchan Kumbhare and Nguyen Vo</i>	
A Default Inference Rule Operating Internally to the Grammar Devices . .	48
<i>Christophe Onambélé Manga</i>	
BOEMIE: Reasoning-based Information Extraction	60
<i>Georgios Petasis, Ralf Möller and Vangelis Karkaletsis</i>	
Recognizing Implicit Discourse Relations through Abductive Reasoning with Large-scale Lexical Knowledge	76
<i>Jun Sugiura, Naoya Inoue and Kentaro Inui</i>	

Program Committee

Marcello Balduccini	Drexel University
Chitta Baral	Arizona State University
Johan Bos	University of Groningen
Vinay Chaudhri	SRI International
Esra Erdem	Sabanci University
Christian Fermüller	TU Wien
Michael Gelfond	Texas Tech University
Yuliya Lierler	University of Nebraska at Omaha
Peter Schüller	Sabanci University

Some Recent Advances in Answer Set Programming (from the Perspective of NLP)

Marcello Balduccini

College of Computing and Informatics
Drexel University
marcello.balduccini@gmail.com

1 Introduction

Answer Set Programming (ASP) [12, 13, 15, 8] is a logical language for knowledge representation and reasoning that combines a non-monotonic nature, strong theoretical foundations, an intuitive semantics, and substantial expressive power. The language has been successfully used for modeling a number of very diverse domains (e.g. [7, 14]) and for capturing key reasoning tasks such as planning, diagnostics, learning and scheduling (e.g. [10, 5, 1]). All of this makes ASP a prime candidate for use in the sophisticated knowledge representation and reasoning tasks involved in Natural Language Processing (NLP).

In this note I will give an overview of some of my recent work on ASP that I believe may be useful in the context of NLP.

2 Motivation

Reasoning about natural language involves various knowledge-intensive tasks. Particularly challenging from this perspective are the extraction of semantic content from phrases (e.g. anaphora resolution) and the disambiguation of phrases using world knowledge and commonsense knowledge. These two tasks are not only challenging, but also heavily interconnected.

Consider the following collection of passages and proposed corresponding reasoning:

- “John was walking his dog. He said hi.”
To conclude that “he” refers John, we can use knowledge about grammar, stating that “he” normally refers to a human male. Additionally, the fact that saying hi is a capability proper of humans, confirms the correctness of the association.
- “John was walking his dog. He ran away after a rabbit”
This type of sentence is quite common especially in spoken English. Commonsense tells us that “he” here refers to John’s dog. A justification for this is the everyday knowledge that running away after a rabbit is a behavior common of dogs and other animals with hunting habits. Humans typically

do not run away after rabbits (although carefully capturing this last statement appears to be a rather interesting and intricate modeling task in itself). Although according to grammar rules “he” should be associated with John, it is also typical for people, and especially pet owners, to refer to their pets by “he” or “she.”

- “John and Frank entered the room. Frank left right away. He came out two minutes later.”

In this case the difficulty in finding which object “he” refers to derives from the fact that two human males are mentioned in approximately the same locations of the passage. To properly link this occurrence of “he” to John, one needs to follow the evolution of the domain described by the passage. The phrase “came out two minutes later” appears to refer to the room that John and Frank had initially entered. The second sentence states that Frank has already left the room. So, John is the only other person *of interest in the passage* who is left in the room, and thus it is reasonable to assume that “he” refers to him.

- “Andrea and Frank entered the room, but he left empty-handed.”

To reason about this sentence, it is useful to recall that, in English, Andrea is both a male and a female name. Reasoning by cases, one can observe that, if Andrea is a man, then the occurrence of “he” in the sentence is ambiguous. On the other hand, if Andrea is a female, then it can be concluded without ambiguity that “he” refers to Frank. Under the assumption that the speaker or writer crafted the sentence in such a way as to convey the relevant information in an unambiguous way, then it is reasonable to assume that “he” refers to Frank. Moreover, one can conclude that Andrea is a woman. This information can be stored and used later in reasoning about other parts of the passage.

- “Andrea cannot be the one who took the computer from that room. Andrea and Frank did enter the room, but he left empty-handed.”

Let us suppose that this passage is in the context of an investigation aimed at determining who stole a computer from a room. The first sentence focuses the discussion on Andrea and on Andrea’s innocence. Let us reason again by cases on the possible associations of “he” – Andrea and Frank. Under both possible associations, no grammar rules are violated, as long as Andrea is a man. If “he” refers to Frank, however, the first and the second sentences appear to have no logical connection, while their construction suggests that indeed some link exists. On the other hand, if “he” refers to Andrea, then the link is clear: the first sentence claims Andrea’s innocence, and the second sentence offers evidence in support of the claim. Similarly to the previous example, the second case appears to be preferred based on the commonsensical assumption that the speaker or writer crafted the passage in such a way as to convey the relevant information in an unambiguous *and economical* way.

Whereas carefully crafted, written-language passages may require relatively limited reasoning, everyday, colloquial language such as the one exemplified here requires substantial reasoning for a proper understanding. For the success of

practical systems with natural language interfaces, I argue that everyday, colloquial language must be supported.

Overall, it appears that successfully reasoning about the semantic content of sentences such as the ones shown above requires a sophisticated combination of world knowledge, commonsense, and (commonsensical) information about speaker’s/writer’s behavior and intentions. It is my belief that ASP and its extensions can be useful in tackling such a task.

In the rest of this note I describe some extensions of ASP I authored or co-authored, and which may be useful in capturing certain aspects of the reasoning about natural language. For a thorough discussion on ASP and on its use for knowledge representation, the reader is referred to the existing literature (e.g. [8]).

3 CR-Prolog

CR-Prolog [6] is an extension of ASP that adds to the language constructs, called consistency-restoring rules (cr-rules), designed to capture certain advanced aspects of non-monotonic reasoning.

A central, well-known feature of languages for non-monotonic reasoning such as ASP is that the programmer can write “defeasible statements,” which are normally true, but may not apply to certain cases, called exceptions. A well-known example is that of the statement “birds normally fly.” While true for most birds, this statement has exceptions, such as penguins and birds with broken wings, and hence the use of the word “normally.”

In most languages for non-monotonic reasoning the exceptions must be explicitly listed. In the example above, if a new type of bird is discovered that does not fly, suitable statements must be added to the system, saying that that type of bird is an exception. If the exceptions are not added, the systems will apply the default statement and conclude that the birds of the new type fly. From a practical perspective, having to know in advance all the exceptions may be a limiting factor in the development of autonomous systems, since there may not be sufficient understanding of the problem domain for such a complete list. It is worth observing that, *in everyday reasoning, humans are typically capable of postulating exceptions to defaults, especially when they observe phenomena that contradict such defaults.*

Cr-rules are an attempt to capture this capability, allowing a reasoner to postulate exceptions to default statements, but only when strictly necessary. Making such assumptions is considered strictly necessary when the reasoning process is otherwise inconsistent. This is the case, for example, of a system given observations that contradict its knowledge base. For instance, the cr-rule:

$$\text{exception}(H, A) \leftarrow^+ \text{human}(H), \text{animal}(A), \text{small}(A).$$

can be used in combination with a default “normally, humans do not chase small animals” to state that, under exceptional, *unknown*, circumstances, the default

can be violated. When inconsistencies arise in the knowledge base, the system can then use the cr-rule to postulate exceptions to the default.

My co-authors and I demonstrated that cr-rules allow for elegantly capturing types of non-monotonic reasoning that are otherwise difficult or impossible to capture. We also showed that they can be used to formalize concisely diagnostic reasoning and certain types of planning.

4 EZCSP

EZCSP [2] is an extension of ASP aimed at increasing performance and scalability in certain – rather large – application domains.

To see how the ability to reason about numbers is important in reasoning about natural language, consider the following passage: “The train left at 10. A couple of hours later, we were having lunch in Paris.” To determine if “10” refers to 10am or 10pm, one may reason as follows. Normally, “a couple of hours” means two hours. Moreover, let us assume that it is common knowledge that in Paris people have lunch between noon and 2pm. Hence, if we reason by cases, the interpretation that “10” refers to “10am” sets the time of the speaker’s lunch in Paris to noon. The interpretation that “10” refers to “10pm” sets the time of the lunch to midnight. The second interpretation contradicts the custom of having lunch between noon and 2pm, and thus the former interpretation is preferred.

This reasoning process relies on the ability to process effectively numerical information. Although ASP allows in principle for natural and concise formalizations of many kinds of knowledge, in practical applications efficiency often degrades quickly when dealing with numerical information and variables with large domains. To overcome this limitation, I designed EZCSP to allow for the use of constructs from constraint programming within ASP programs. For example, the rule:

$$required(hour(T) \geq 12) \leftarrow lunchtime(T).$$

states that, if timestamp T refers to lunch time, then the hour of T must be greater than or equal to 12.

The new language makes it possible to represent and reason about numerical information efficiently, while at the same time keeping the representations elegant, concise and elaboration tolerant, as usual in ASP. Differently from other languages that combine ASP and constraint programming (e.g. [11]), EZCSP includes support for global constraints (a powerful type of construct from constraint programming), and both language and solver are designed to be independent from the underlying ASP and constraint solvers chosen for the computation.

5 ASP{f}

One shortcoming of EZCSP is that it does not allow one to perform full-fledged non-monotonic reasoning on numerical quantities. For example, one cannot easily

state that “in Paris, people *normally* have lunch between noon and 2pm” and reason with evidence that “John had lunch at 3pm.”

This is due to the monotonic nature of the underlying constraint programming constructs. A further shortcoming of EZCSP is that performance in the presence of variables with large non-numerical domains still tends to be limited, because constraint programming constructs mainly apply to numerical quantities. In fact, these limitations are shared by all the research attempts aimed at hybridizing ASP and constraint programming. To overcome these issues, I have developed a new language, called ASP{f} [3, 4], which adds to ASP the ability to represent, and reason about, arbitrary (non-Herbrand) functions, including but not limited to numerical functions. With ASP{f}, the default about lunch time in Paris can be captured in a simple way by statements such as:

$$hour(T) < 14 \leftarrow lunchtime(T), \text{ not } hour(T) \geq 14.$$

which intuitively states that “lunch ends at 2pm unless otherwise specified.” The observation about John’s lunch time can be encoded by $\{lunchtime(t_1), hour(t_1) = 15\}$. In ASP{f}, this observation is sufficient to defeat the default.

In ASP{f} it is also possible to capture rather complex numerical calculations, such as:

$$\begin{aligned} financially_sound(C) \leftarrow \\ revenue(C) > sum[employee(E, C) = salary(E)] + investments(C). \end{aligned}$$

This rule states that company C is financially sound if its revenue is greater than the sum of the salaries paid to the employees and of the investments made by the company. Another example, demonstrating ASP{f}’s ability to deal with non-numerical information, is the rule:

$$\leftarrow siblings(P1, P2), first_name(P1) = first_name(P2), \text{ not } exception(P1, P2).$$

which captures the commonsensical statement that two siblings shouldn’t have the same first name. Rather than relying on constraint programming, the new language includes “native” support for functions, and, differently from other attempts in this direction (e.g. [9]), is crafted in such a way that state-of-the-art inference engines for ASP can be extended to support ASP{f} with relatively simple modifications.

6 Conclusions

In this note I have described some challenges of the task of reasoning about natural language that are relevant to ASP and to commonsense and non-monotonic reasoning in general. I have also discussed recent extensions of ASP that I have developed, and which I believe may be useful in tackling these tasks. Of course, many other extensions of ASP exist, which can be useful for this endeavour. For the reader’s convenience, a small selection of relevant works was cited in this note.

References

1. Balduccini, M.: Learning Action Descriptions with A-Prolog: Action Language C. In: Amir, E., Lifschitz, V., Miller, R. (eds.) *Procs of Logical Formalizations of Commonsense Reasoning, 2007 AAAI Spring Symposium* (Mar 2007)
2. Balduccini, M.: Representing Constraint Satisfaction Problems in Answer Set Programming. In: *ICLP09 Workshop on Answer Set Programming and Other Computing Paradigms (ASPOCP09)* (Jul 2009)
3. Balduccini, M.: Correct Reasoning: Essays on Logic-Based AI in Honour of Vladimir Lifschitz, chap. 3. A “Conservative” Approach to Extending Answer Set Programming with Non-Herbrand Functions, pp. 23–39. *Lecture Notes in Artificial Intelligence (LNCS)*, Springer Verlag, Berlin (Jun 2012)
4. Balduccini, M.: ASP with non-Herbrand Partial Functions: a Language and System for Practical Use. *Journal of Theory and Practice of Logic Programming (TPLP)* (2013)
5. Balduccini, M., Gelfond, M.: Diagnostic reasoning with A-Prolog. *Journal of Theory and Practice of Logic Programming (TPLP)* 3(4–5), 425–461 (Jul 2003)
6. Balduccini, M., Gelfond, M.: Logic Programs with Consistency-Restoring Rules. In: Doherty, P., McCarthy, J., Williams, M.A. (eds.) *International Symposium on Logical Formalization of Commonsense Reasoning*. pp. 9–18. *AAAI 2003 Spring Symposium Series* (Mar 2003)
7. Balduccini, M., Gelfond, M., Nogueira, M.: Answer Set Based Design of Knowledge Systems. *Annals of Mathematics and Artificial Intelligence* 47(1–2), 183–219 (2006)
8. Baral, C.: *Knowledge Representation, Reasoning, and Declarative Problem Solving*. Cambridge University Press (Jan 2003)
9. Cabalar, P.: Functional Answer Set Programming. *Journal of Theory and Practice of Logic Programming (TPLP)* 11, 203–234 (2011)
10. Erdem, E.: Application of Logic Programming to Planning: Computational Experiments. In: *Proceedings of the 5th International Conference on Logic Programming and Non-monotonic Reasoning (LPNMR-99)*. No. 1730 in *Lecture Notes in Artificial Intelligence (LNCS)*, Springer Verlag, Berlin (1999)
11. Gebser, M., Ostrowski, M., Schaub, T.: Constraint Answer Set Solving. In: *25th International Conference on Logic Programming (ICLP09)*. vol. 5649 (2009)
12. Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In: *Proceedings of ICLP-88*. pp. 1070–1080 (1988)
13. Gelfond, M., Lifschitz, V.: Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing* 9, 365–385 (1991)
14. Grasso, G., Leone, N., Manna, M., Ricca, F.: ASP at Work: Spin-off and Applications of the DLV System. In: Balduccini, M., Son, T.C. (eds.) *Symposium on Constructive Mathematics in Computer Science* (Oct 2010)
15. Marek, V.W., Truszczyński, M.: The Logic Programming Paradigm: a 25-Year Perspective, chap. Stable Models and an Alternative Logic Programming Paradigm, pp. 375–398. Springer Verlag, Berlin (1999)

Three Lessons in Creating a Knowledge Base to Enable Reasoning, Explanation and Dialog

Vinay K. Chaudhri, Nikhil Dinesh, and Daniela Inclezan

Artificial Intelligence Center,
SRI International, Menlo Park, CA, 94025

Abstract. Our work is driven by the hypothesis that for a program to answer questions, explain the answers, and engage in a dialog just like a human does, it must have an explicit representation of knowledge. Such explicit representations occur naturally in many situations such as engineering designs created by engineers, a software requirement created in unified modeling language or a process flow diagram for a manufacturing process. Automated approaches based on natural language processing have progressed on tasks such as named entity recognition, fact extraction and relation learning. Use of automated methods can be problematic in situations where the conceptual distinctions used by humans for reasoning are not directly expressed in natural language or when the representation must be used to drive a high fidelity simulation.

In this paper, we report on our effort to systematically curate a knowledge base for substantial fraction of text in a biology textbook [26]. While this experience and the process is interesting on its own, three aspects can be especially instructive for future development of knowledge bases by both manual and automatic methods: (1) Consider imposing a simplifying abstract structure on natural language sentences so that the surface form is closer to the target logical form to be extracted. (2) Adopt an upper ontology that is strongly motivated and influenced by natural language. (3) Develop a set of guidelines that captures how the conceptual distinctions in the ontology may be realized in natural language. Since the representation created by this process has been quite effective for answering questions and producing explanations, it gives a concrete target for what information should be extracted by the automated methods.

Keywords: knowledge representation, ontologies, automated reasoning, conceptual models, knowledge acquisition from text

1 Introduction

Classical approach to achieving intelligent behavior has been driven by the knowledge representation hypothesis proposed by Smith [27]: Any mechanically embodied intelligent process will be comprised of structural ingredients that (a) we as external observers naturally take to represent a propositional account of the knowledge that the overall process exhibits, and (b) independent of such external semantic attribution, play a formal but causal and essential role in engendering the behavior that manifests that knowledge. In the context of this framework, an intelligent program requires a formal representation of knowledge that can be manipulated by an automated reasoner with the goal that

it will enable a variety of tasks including answering questions, producing explanations and engaging in a dialog.

There are some domains such as engineering, manufacturing, and finance where structured representations are routinely created and are a part and parcel of a routine workflow. Automated methods based on natural language processing (NLP) techniques are quite effective at creating some limited forms of structured representations such as named entity extraction [21] and relation extraction [7].

We have recently completed a substantial knowledge engineering effort that has resulted in a knowledge base called `KB_Bio_101` that represents a significant fraction of an introductory college-level biology textbook [11, 10]. We have used `KB_Bio_101` as part of a prototype of intelligent textbook called *Inquire* that is designed to help students in learning better [8]. *Inquire* answers questions [10], gives explanations and engages in dialog through natural language generation [1].

In this paper, we describe three specific aspects of the knowledge engineering process and discuss the lessons that can be drawn from this effort which can inspire the development of a new breed of manual as well as automated knowledge acquisition methods. These lessons are: (1) re-formulating sentences as universal truths so that the surface form of knowledge is closer to the knowledge to be extracted (2) using a linguistically motivated ontology into which the knowledge is extracted (3) using a set of guidelines that define how various conceptual distinctions are expressed in natural language. These three techniques were instrumental in creating `KB_Bio_101` that enabled *Inquire* to answer students questions and led to learning gains as have been reported in a previous paper [8]. We have organized the paper by first discussing the techniques that we used in creating the knowledge representation followed by a discussion on how these can be instructive for future manual, automated as well as semi-automated knowledge acquisition methods.

2 Reformulating Input Sentences

A textbook is written for pedagogical purposes. Therefore, the authors adopt a style of writing which is varied, interesting, and that tells a story. This invariably involves first introducing concepts at an abstract level, and later adding more details, and in some cases, contradicting and/or overriding the information that has been previously introduced.

In contrast, an automated reasoning system needs to encode knowledge only once, and in a succinct manner, using sentences in a formal language. While the axioms can be arbitrarily complex, in practice, there are frequently occurring axiom patterns, for example, axioms to represent necessary and sufficient properties of a concept, cardinality constrains, subclass and disjointness statements, etc. For the purpose of the current discussion, we will work with one such axiom pattern known as universal truth: a set of facts that are true for all instances of a concept.

To determine what should be represented from a textbook, a knowledge encoder must gather all the sentences that describe that concept. In general, a sentence will mention more than one concept. To determine which concept a sentence actually refers to, the encoder reformulates that sentence as a universal truth. A sentence may result in

more than one universal truth. In our current process, the encoders work at the level of a single chapter. Once the sentences in a chapter have been reformulated as universal truths, they can be sorted on the concept so that we now have available all the sentences that describe a particular concept which can then be used for representation. This process deals with the pedagogical style of the textbook by collecting information about a concept in one place in a similar surface syntax.

Let us now illustrate this process by taking two example sentences (numbered I and II) in Table 1.

Textbook Sentence	Universal Truth	Concept	Plan
I. A chemical signal is detected when the signaling molecule binds to a receptor protein located at the cells surface or inside the cell.	During signal reception, the signaling molecule binds to a receptor protein located at the cells surface or inside the cell.	Signal-Reception	Signal-Reception – subevent → Attach Attach – base → Receptor-Protein Attach – object → Molecule ...
II. The binding of the signaling molecule changes the receptor protein in some way, initiating the process of transduction.	During signal reception, the binding of the signaling molecule changes the receptor protein in some way.	Signal-Reception	Signal-Reception – subevent → Bind Attach – base → Receptor-Protein ₁ Attach – result → Receptor-Protein ₂ Receptor-Protein ₁ – has-state → Receptor-Protein ₂
	During cell signaling, the binding of the signaling molecule initiates the process of transduction.	Cell-Signaling	Cell-Signaling – subevent → Signal-Reception Cell-Signaling – subevent → Signal-Transduction Signal-Reception – next-event → Signal-Transduction

Table 1: Procedure for creating KB content from sentences

2.1 From Sentences to Universal Truths

Syntactically, a universal truth (or a UT) is a statement of the form: (a) Every X Y (b) In X, Y (c) During X, Y. In these statements, X is a noun phrase denoting a concept and Y is a clause or verb phrase denoting information that is true about the concept. The concept (X) may not be directly mentioned in the sentence and it might be inferred from the context and the teacher’s understanding of biology.

The universal truth associated with sentence I has the form – “During X, Y”, where the concept “X” is “signal reception”. The phrase “signal reception” is not directly mentioned in the sentence, but is inferred from the phrase “a chemical signal is detected” based on the context in which the sentence appears in the textbook.

2.2 From Universal Truths to Knowledge Representation Plans

When formalized in logic, each universal truth leads to an existential rule, ie, a rule whose antecedent has one variable that is universally quantified, and whose consequent has one or more variables which are existentially quantified. Each universal truth is converted to a *plan*: which is a set of literals that would appear in the consequent of the existential rule suggested above. The plan for a universal truth is made by taking into account the plans for all its superclasses and dependent concepts. For example, the plan for Cell-Signaling would take into account the plan for Signal-Reception, which is a step of Cell-Signaling.

Consider the first universal truth in Table 1 – “During signal reception, the signaling molecule binds to a receptor protein located at the cell’s surface or inside the cell”. A portion of the plan for this universal truth is shown in the fourth column and this can be understood as follows:

- Signal-Reception – subevent → Attach – One of the steps of signal reception is an “attach” or “bind” event.
- Attach–object → Molecule – The object (ie, the entity that undergoes attachment) of the attach event is a molecule.
- Attach – base → Receptor-Protein – The base (ie, the entity that the object attached to) is a receptor protein.
- We omit the remaining literals, which show the “signaling” role of the molecule and the location of the protein.

Taken together, these literals can be understood as – “one of the steps of signal reception is the attachment of a molecule to a receptor protein”. The event Attach and the relations object and base are provided by the upper ontology called the Component Library (CLIB) which we will discuss in more detail in the next section.

The plans for a knowledge base are similar to design specification or a pseudo code for a program. Writing the plans first helps an encoder to think through the overall design of the representation before entering it into the knowledge base.

2.3 From Plans to Knowledge Representation

The plans are entered into the KB using a graphical interface called *concept maps* [12]. Figure 1 shows the concept map for Signal-Reception; the white color denotes that it is universally quantified, while all other concepts are existentially quantified. The concept map can be read as the following existential rule: “Every signal reception event has a subevent in which a molecule attaches to a receptor protein, resulting in a change in the state of the protein”.

There are several side-benefits of reformulating these sentences as universal truths: (1) The sentence form is closer to the actual logical form that will be represented in the knowledge base, making the task of creating the concept graphs much easier (2) universal truths aid in developing a consensus understanding of the content of the textbook (3) They help the encoder in thinking through which concepts should the knowledge be associated with.

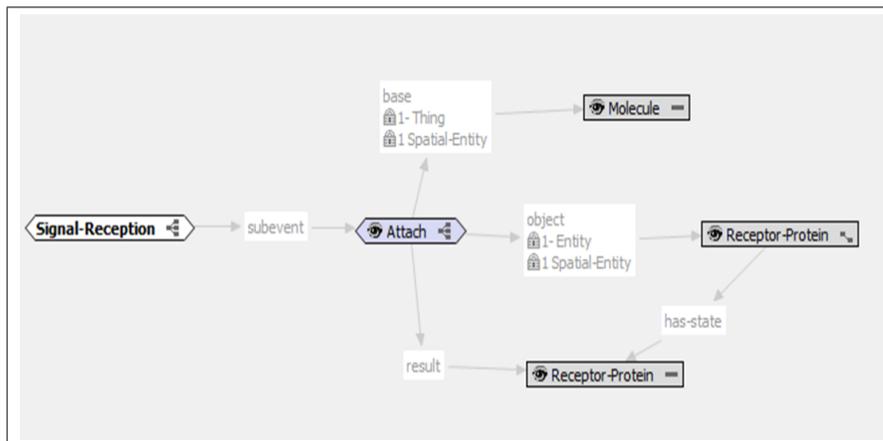


Fig. 1: Concept Map for Signal Reception

3 Linguistically Motivated Upper Ontology

Wordnet is by far the most commonly used resource in natural language processing for reasoning about entailments [22]. One of the reasons for the success of Wordnet is that it is linguistically motivated and it encodes knowledge at the level of words. This ensures good coverage and makes it easy for people to understand what it should or should not contain. Wordnet is, however, not an ontology and has several limitations when it comes to supporting automated reasoning [16].

Component Library (or CLIB) is a linguistically motivated ontology designed to support representation of knowledge for automated reasoning [3]. CLIB adopts four simple upper level distinctions: *entities* (things that are), *events* (things that happen), *relations* (associations between things) and *roles* ways in which entities participate in events.

For the purpose of this discussion, we will focus on the taxonomy of physical actions where action is a subclass of Event. The reason for focusing on actions is to illustrate how the library of actions is grounded in language and helps us assess coverage in a manner similar to assessing coverage for Wordnet, and yet, defines the actions to support automated reasoning, explanation generation and dialog.

In the original version of CLIB [3], the Action has 42 direct subclasses and a total of 147 subclasses in all. Examples of direct subclasses include Attach, Impair, Move, and Store. Other subclasses include Move-Through which is a subclass of Move, and Break which is a subclass of Damage which is a subclass of Impair. These subclasses were developed by consulting lexical resources, such as Wordnet [22], Longman Dictionary of Contemporary English [30] and Roget's thesaurus [20].

We will now discuss how this linguistic grounding of the ontology helped us address the following two problems in our recent effort to represent knowledge from a biology textbook: (a) ensuring that we have an adequate coverage of actions that occur in the

textbook (b) developing guidelines that inform an encoder which action from the library should be used to model a verb appearing in a sentence.

3.1 Ensuring Coverage

To check whether CLIB had adequate coverage to support all the process representations that we will need to create for the textbook, we analyzed the *verbs* appearing in the textbook. We investigated whether and how their meaning could be represented using CLIB actions and determined what new action classes should be added to CLIB when no pre-existing classes matching its meaning was found.

The main body of the biology textbook *Campbell Biology* consists of 30,346 sentences. We extracted all the verbs appearing in these sentences which gave us a list of 2,870 verbs. The actual number of verbs is smaller, as some of the identified verbs are in fact just different forms of the same verb (e.g., *is* and *were*, two forms of the verb *to be*, were counted as different verbs). Next, we stemmed verbs based on their frequency, which ranged from 1 to 18,407. The sixteen verbs with a frequency higher than 400 can be seen in Table 2. There were 800 verbs with a frequency greater or equal to ten.

Verb	Frequency	Verb	Frequency	Verb	Frequency	Verb	Frequency
18,407	to be	860	to produce	629	to make	460	to increase
3,805	to have	708	to include	528	to cause	451	to grow
1,433	to call	658	to form	499	to develop	429	to become
936	to use	646	to occur	488	to do	413	to help

Table 2: Textbook Verbs with a Frequency Higher than 400

We analyzed all the verbs with frequency greater than 10 to check whether their meaning was adequately represented using some action in CLIB. As a result of this exercise, we identified whether a new action class should be added or we should extend the meaning of an existing action class.

We identified 21 new action classes that should be added to CLIB. While adding these classes, we used the principle of correspondence, ie, in many cases pairs of actions go together and both should be present in the action library. For example, the initial version of CLIB contained a class called Attach referring to an *asymmetric* attachment of one entity to another, but there was no class for a *symmetric* attachment between two entities. We remedied this problem by introducing the class Bind, which corresponds to Attach. We introduced the class Expel as a counterpart of Take-In, where Expel and Take-In are the subclasses of Move-Out-Of and Move-Into, respectively. Other newly introduced classes (e.g., Kill) refine the range of one of the relations in their superclasses (e.g., Kill is a subclass of Destroying a *living* entity).

The remaining proposed action classes specify the manner in which an action is performed. For instance, Fly, Run, Swim, Crawl, Hop, and Climb were added as new subclasses of Locomotion. Alternatively, manner could be described via one or more relations defined on action classes. This second option would avoid possible problems related to an increased size of the CLIB action hierarchy and the need to re-organize it.

Finally, one example of an existing action class whose meaning should be extended is Support. Initially, this action class was defined as “*to prevent from falling,*” whereas

for use in the domain of biology it is useful to extend its meaning by adding the expression “*or provides some other kind of structural support.*”

The discussion in this section illustrates how grounding the ontology in natural language text helped assess its coverage in relation to the knowledge that needs to be modeled, and informed us how the library should be extended.

3.2 Choosing an Action Class

When a knowledge encoder is representing a sentence that describes some process knowledge, a choice needs to be made on which action class to use. This choice needs to be systematic so that it is consistent across the representation of different processes across the book as well as consistent across multiple encoders. We approached this problem by systematically analyzing how different verbs should be mapped to actions in CLIB.

For the purpose of this analysis, we limited ourselves to the 800 verbs that had a frequency greater than or equal to ten. We analyzed these verbs based on their usage in the textbook, starting with the most frequent ones. For each verb, we selected a maximum of 30 sentences that contained it drawn from different parts of the textbook to ensure that we were considering representative usage. Two challenges we faced in this exercise are as follows.

1. A large number of verbs have (obviously) multiple meanings, depending on the context in which they were used. So, we must deal with different senses when choosing an appropriate CLIB action.
2. The specification of CLIB actions contains definitions and examples related to *common sense* domains, which are not always helpful when dealing with *specialized* knowledge from the domain of biology. For instance, the CLIB action Support is defined as “to put an object in a state that prevents it from falling;” the use of this CLIB event is illustrated by the sentence:

- (1) Tom supported the roof with a heavy beam.

However, the use of the verb *support* in biological descriptions can also refer to a state that prevents something from changing its shape:

- (2) Intermediate filaments support cell shape.

To address the above challenges we first developed a procedure for identifying an action class by considering one fourth of the selected verbs, and then tested the procedure on the remaining verbs. We expressed this procedure as a set of guidelines for encoding verbs using CLIB actions. In this process, we realized that frequently-occurring verbs, especially those with a frequency greater than 400, tended *not* to describe an actual action taking place and therefore did not require an event to capture their meaning. This was generally not the case with lower frequency verbs. We have extensive set of guidelines to handle verbs with frequency greater than 10. For the present discussion, we illustrate the procedure by considering several examples.

Example 1. Textbook Sentence: The groove is the part of the protein that recognizes and binds to the target molecules on bacterial walls.

Corresponding Universal Truth(s): The protein binds at the groove with the target molecules, which are situated on the bacterial walls.

Encoding: The encoder needs to choose a CLIB action class to represent the verb *binds*. CLIB contains an action class, *Attach*, for asymmetrical attachments. We check that the sentence describes an asymmetrical attachment by verifying that the reverse sentence – “*The target molecules on the bacterial walls attach to the protein*” – does not make sense. To represent this process, we will use the action class *Attach* and assign values to the participant relations for it as follows: object = *protein*, site = *groove*, and base = *target molecules on bacterial walls*. We will discuss the procedure for choosing the relations in the next section.

Example 2 (Guidelines for the Verb to cross). When analyzing sentences containing the verb *to cross*, we first determined that such sentences normally translate into UTs of one of the following two types:

- (a) *Entity X is crossed (interbred) with entity Y.*
- (b) *Entity X crossed entity Y.*

For UTs of type (a), whether the usage is in the context of an experiment in which an action class corresponding to that experiment should be used. In this case, conducting a cross breeding experiment is a domain-specific class to be created and maintained by the domain experts.

For UTs of type (b), the relevant CLIB class is *Move-Through* with participant relations having the values: object = X, base = Y.

We have developed systematic guidelines to help the encoders in identifying a suitable action class from CLIB. Normally, the CLIB action selected to encode a biological process is designated as its superclass. However, there are two exceptions: sometimes the identified CLIB action describes a *subevent* of the biological process, not its superclass; other times, there is a more specific action in the KB that should be made the superclass. We illustrate this using examples.

- (3) Most often these existing proteins are modified by **phosphorylation**, the **addition** of a phosphate group onto the protein.

In the above sentence, should *Add* be one of the subevents of *Phosphorylation*, or the superclass of *Phosphorylation*, or neither?

We address the subevent possibility first. Let us assume that we have a biological process *P* and we have identified a CLIB action *A* that could be used to model it. We use the following test to determine whether *A* should be a step of *P* or its superclass: If it is appropriate to say “*During P, A happens*”, and *P* is already known to have other substeps of *P*, then *A* should be a sub-step. If we apply these guidelines to (3), we notice that it is appropriate to say that “*during phosphorylation, addition happens*,” but the textbook does not describe any other subevent of phosphorylation. So, *Add* should not be modeled as a substep of *Phosphorylation*.

Next, we consider the superclass possibility. If P is a *complex* biological process and A describes just the overall outcome of P but does not capture its intricacies, then A should not be the superclass of P ; this is especially valid if P has multiple steps. In this situation, a more specific biological process from the KB should be selected as the superclass of P . The reason behind this approach is that, in such cases, the CLIB actions tends to abstract away too many of the relevant details of the biological process. The CLIB action is useful, though, in expressing the common sense definition of the process. For instance, although Phosphorylation is described as an addition of a phosphate group to a protein in (3), encoding this process as a specialization of the CLIB action Add is not a good choice as it would result in an overly simplified model. We prefer to make Phosphorylation a subclass of Synthesis-Reaction, which is a subclass of Chemical-Reaction and is better suited for capturing the complexity of this process.

The discussion above illustrates the kind of procedures we needed to develop to identify suitable actions classes that should be used when modeling a process verb in a textbook sentence.

4 Guidelines for Choosing Semantic Relations

CLIB provides two types of relations between events and entities, motivated by “case roles” in linguistics [c.f. 2] :

- Participant relations – agent, base, instrument, raw-material, result, object
- Spatial relations – destination, origin, path, site.

CLIB provides a semantic definition of each relation, together with common sense examples as shown in Table 3. In the examples, the event in boldface is related to the entity in italics.

Relation	Definition	Example
agent	The entity that initiates, performs, or causes an event.	<i>John</i> swatted the fly
base	Event references something as a major or relatively fixed thing	Vlad attached the sign to <i>the post</i>
site	The specific place of some effect of an event, as opposed to the locale of the event itself	The nurse stabbed the needle in <i>my arm</i> at the hospital

Table 3: Definition of relations in CLIB with examples

After a CLIB action is selected for modeling some biological process described by a sentence, the next step is to identify the semantic relationships between the action class and its various participants. It is well known that semantic distinctions are not always directly expressed in language [19] making it difficult to apply the definitions of the relations as shown above. The following pairs of relations are especially difficult to distinguish.

- agent and instrument;
- raw-material and instrument;
- base and path.

If the choice between these relationships is not made consistently and correctly, it significantly interferes with the system's ability to generate good natural language sentences to support explanation generation. To further make this point, we consider two specific problems caused by lack of proper usage.

1. The same entity is assigned to two or more semantic relations of the same event. With such encoding, the translation into English of events is unnatural, as shown by the following automatically produced sentence:

(4) A gated channel is closed **by a stimulus with a stimulus**.

The above sentence results from an action Close with object = *gated channel* and agent = instrument = *stimulus*.

2. A required relation is assigned an overly general entity such as *Physical-Object* or *Tangible-Entity*. Such process models are only partially useful in answering questions. Furthermore, their translations into natural language are difficult for end-users to understand.

(5) A gene is moved **into an object**.

The above sentence resulted from an action Move-Into with object = *gene* and base = *a tangible entity*.

To address this issue, we developed a more detailed characterization of how the semantic relations might be expressed in language and how an encoder could be better supported in choosing the most appropriate relation. Such characterization involves specifying *syntactic clues* and *examples from the domain of biology*. Syntactic definitions are usually easier to follow, as they are more precise. There is however one semantic relationship, base, that has an irregular syntactic definition, which varies across CLIB events. Additionally, there are some prepositions that are associated with more than one semantic relationship (e.g., *from* may indicate either a donor or an origin). For these reasons, a combined approach based on both semantic *and* syntactic definitions, as summarized in Table 4, works the best. Such an approach benefits from the advantages of both methods while diminishing their disadvantages.

For the pairs of relations that were particularly difficult to distinguish, we performed a deeper comparative analysis and provided additional guidelines, as described in Subsection 4.1.

We tested these guidelines and our definitions by asking the domain experts to convert sample encodings created into English sentences and then assessing whether the resulting sentences were of good quality. We consider a few representative examples of this evaluation in Subsection 4.2, together with suggestions for correcting them.

4.1 Distinguishing between Problematic Pairs of Relations

In this section, we discuss examples of relations that were too difficult to distinguish for encoders as originally defined in CLIB, and our approach for developing a procedure to better distinguish them.

Distinguishing between agent and instrument. In natural language, entities denoting the agent or the instrument of an event can both be realized as the grammatical subject of a sentence, which makes it difficult to distinguish between the two:

- (6) *Birds* **eat** small seeds.
- (7) *Intermediate filaments* **support** cell shape.

The subjects of sentences (6) and (7) are mapped into the agent and instrument relations, respectively, based on the original semantic definitions of these relations, which requires the agent to be *sentient*, but the instrument need not be sentient:

- An agent is active, while an instrument is passive, being used by the agent if there is one.
- An agent is *typically* considered sentient, *if only metaphorically*, while an instrument need not be.

Applying these definitions and distinctions is not always straightforward because different people have different understandings of what *sentient* means. This is illustrated by the following example sentence:

- (8) A biomembrane blocks hydrophilic compounds.

A biomembrane is part of a living thing, so it is not clear whether by itself, it is sentient or not. To solve this problem, we complemented the specifications of the two slots by adding some syntactic tests for disambiguation:

- Transform a sentence written in the active voice into an equivalent sentence in the passive voice. The agent is the entity preceded by the preposition *by*, if such an entity exists. (e.g., By transforming (6) into an equivalent sentence in the passive voice, we obtain: “Small seeds **are eaten** *by birds*.” The noun *birds* is preceded by the preposition *by*, hence it must indicate the agent.)
- If the subject of a sentence can be replaced by a phrase containing the preposition *with* or *using* when the sentence is transformed into its passive voice equivalent, then that entity is an instrument. (e.g., The sentence “Cell shape **is supported** *using intermediate filaments*” sounds natural, so *the intermediate filaments* are the instrument in sentence (7).)

By performing these syntactic tests on sentence (8), and using the semantic definitions above, we can determine that *the biomembrane* should be the agent of the described event.

Distinguishing between raw-material and instrument. Consider the following sentences:

- (9) A planarian **detects** light *using a pair of eyespots*.
- (10) The Calvin cycle **produces** sugar *using ATP and NADPH*.

Here, the preposition *using*, normally associated with the instrument relation, appears in both of the sentences. However, only (9) specifies an instrument; (10) specifies a raw-material.

To determine what sets the two cases apart, we analyzed several sentences which contained verbs such as *to use*, *to produce*, *to form*, *to consume*, etc. We determined that the following distinctions capture how these two relations are expressed in language:

- A raw-material is an entity that is used up in an event and does not come out of it the same way it entered the process.
- An instrument is an entity that facilitates the occurrence of the event, but it is not consumed by the process.

This new definition clarifies why (10) is an example of a raw-material: ATP and NADPH are used up by the Calvin cycle.

Distinguishing between base and path. Consider the sentence:

(11) A molecule moves through *the cell membrane*.

which describes a Move-Through action. According to the original CLIB guidelines for Move-Through, *the cell membrane* should be mapped into the base relation. This conflicts with the syntactic guidelines in Table 4, which indicate that *the cell membrane* should be the path, because it is preceded by the preposition *through*. However, opting for either of the two relations seems to cause problems as we discuss below.

Let us assume that we opt for using the slot base in (11), and let us consider the sentence:

(12) A molecule moves into *the cell*.

According to the CLIB guidelines for action Move-Into, *the cell* in (12) should be the base of a Move-Into event. This leads to conflicting definitions for the slot base: in the parent class Move-Through it must be the Barrier that is crossed; in the subclass Move-Into it must be a Container into which an object is moved.

If we opt for using the slot path in (11), then we run into a different problem. In the sentence:

(13) A molecule moves through *a pore* of the cell membrane.

there would be no relation to assign to *the pore*, given that the slot path—the most natural choice—is already assigned the value *the cell membrane*. This is an even bigger issue than the first option.

To remedy this problem, we decided to allow the slot base to have different definitions for different action classes, even if these action classes are connected by subclass relationships in the CLIB ontology. The new general definition of base says that it must be “a major or relatively fixed thing that the event references” and that cannot be associated with other slots. More specific definitions are given in relation to each action class for which this relation is relevant.

4.2 Testing Our Definitions and Guidelines

To test the guidelines that we have described above, we asked the encoders to apply them to encode a few representative actions, and then manually convert them into English. Such a task is in direct support of our goals to enable explanation and dialog.

In most cases the guidelines were effective, ie, when they were followed, the resulting representations led to good natural language sentences. In this section, we will discuss only those cases where the guidelines were not effective and suggest solutions for improving them.

- (14) Liquid is transported by a eukaryotic cell to cytoplasm **inside a vesicle** through a plasma membrane using an organic molecule. (Pinocytosis)

In (14), *the vesicle* is mapped into the instrument slot. From a syntactic point of view, the preposition *inside* normally indicates association with the base slot. However, in the process of pinocytosis, the vesicle functions more like a carrier that transports the liquid. Thus semantically it is closer to an instrument. Note that instruments are indicated by the expression *using*, which is also associated with raw-material. We believe that the encoder used the preposition *inside* for the instrument because the *using* relationship had already been used to capture the raw-material in this sentence. One suggestion would be to use the expression *consuming* for the raw-material, and the preposition *using* for the instrument, resulting in a new sentence:

Liquid is transported by a eukaryotic cell to cytoplasm **using** a vesicle through a plasma membrane **consuming** an organic molecule.

Next, consider the following sentence:

- (15) An image is produced **using a radioactive tracer by a PET scanner**.

In (15), the *radioactive tracer* is assigned to slot agent and the *PET scanner* to the slot instrument, but the prepositions associated with the two expressions indicate a reversed assignment to slots. What happens in reality is that the image is produced by the PET device based on the computer analysis of concentrations of the tracer. Therefore, both syntactically and semantically the *tracer* should be the instrument and the *PET scanner* should be the agent.

- (16) A cell recognizes another cell (a target cell) **at a plasma membrane**.

In (16), the *plasma membrane* is assigned the role base, while the preposition *at* is normally related to the slot site. Semantically, what this means is that Cell-Cell-Recognition is a function of the plasma membrane. According to the guidelines for modeling of *Functions* [9], this information would be modeled by making the has-function slot of the plasma membrane point to Cell-Cell-Recognition. Then, the plasma membrane can be assigned the role of site in this event, as it specifies a particular place on the agent cell where the effect of recognition occurs.

- (17) Transferring **by an electron** from a chemical (a reducing agent) to another chemical (an electron recipient). (Reduction)

In (17), the *electron* is assigned the role of donor, although it is preceded by the preposition *by* usually associated to agent. Reduction is defined as “a reaction in which the atoms in an element accept electrons.” Hence, semantically, electrons are not a donor (nor an agent), but rather the object of this transfer. To fix this case, we replace the preposition *by* with the preposition *of* as in:

Transferring **of** an electron from a chemical (a reducing agent) to another chemical (an electron recipient).

(18) A cell receives a signal **at a receptor protein carried by a chemical**.

In (18), the *receptor protein* is assigned to slot instrument, and the *chemical* to slot object. Syntactically, the preposition *at* is used to denote the site. If we look at the definition of this process, we see that it uses a different verb than *receives*: “The target cell’s *detection* of a signaling molecule coming from outside the cell.” Moreover, in the encoding of this process, the chemical *plays the role* of a signal. Hence, this sentence could be reformulated as

A chemical entity playing the role of a signal is detected by a cell using a receptor protein.

As a result, the following assignment of values to slots would be appropriate, according to the information in Table 4: object = *chemical* with plays = *signal*, base = *cell*, instrument = *receptor protein*.

5 Discussion and Lessons Learned

Let us now step back and draw some higher level conclusions from the techniques we have presented here.

Reformulating a sentence as a UT can be more generally viewed as a way to arrive at a surface structure of a sentence which is more closely aligned with the ultimate logical form that needs to be created. Of course, the idea of UT needs to be generalized to a broader set of axiom templates to support sufficient properties, constraints, disjointness etc. A closely related notion was first introduced under the name of abstract syntax trees (ASTs) [15]. UTs can be viewed as a specific instance of an AST. The use of ASTs is more broadly applicable to manual knowledge curation efforts in which the acquisition process starts from text, and an AST generation provides a graceful migration from the informal textual knowledge to a more formal logical form. In the context of automated knowledge acquisition using natural language processing methods, availability of ASTs can make the task of logical form generation substantially more tractable. The sentences in the textbook are so complex that unless one uses some form of AST, the task of getting a reasonable logical form is almost impossible. Therefore, the use of ASTs as a technique to add knowledge capture is the first major lesson or take away from the process described here.

CLIB was originally created to be a linguistically motivated upper ontology. The action names are grounded in language and the semantic relationships based on research

in linguistics. As we saw, the linguistic grounding of CLIB was quite effective in achieving coverage of core concepts that were needed for modeling knowledge in the biology textbook. Even though CLIB defines semantic relationships and a few key axioms for each of the action in the library, it is far from clear how to argue the completeness of those axioms. There are several concepts in CLIB that capture distinctions that are not usually expressed in language. One such example is the concept of Tangible-Entity. As we saw during the discussion, such concepts were problematic for natural language generation, because if such concepts appear in the output, the end-users will fail to naturally understand their meaning. Ideally speaking, the usage of such concept names in an ontology should be minimized, and preferably, avoided. We expect CLIB to have special strength for natural language processing application because of its linguistically motivated concepts and semantic relationships. While we cannot claim that CLIB has yet proven its value in being an inferentially valuable knowledge resource in the same way that Wordnet is a lexical resource, continuing to develop CLIB in that direction is still a sensible direction for future work. Accordingly, we encourage and advocate other researchers to make their ontologies as linguistically grounded as possible.

Use of a combination of syntactic and semantic guidelines was essential in ensuring a systematic encoding of knowledge. We developed guidelines that helped encoders determine which semantic relationship is most appropriate for use in a process description. The linguistically motivated semantic relationships have the strength of being general across multiple domains. But, as the complexity of the guidelines indicates, they can also be difficult for humans to use and apply in a consistent manner. We hope that developing the guidelines that we presented in this paper will provide a foundation for automated and semi-automated tools that could either acquire such relationships from text automatically, or provide much better support to encoders as they make their choices. The basic idea of using a combination of syntactic and semantic guidelines is quite general and can be adopted by a broad range of applications.

6 Related Work

Several well-known upper ontologies exist today that have been used to create knowledge bases and overlap in their goals and coverage with CLIB. One of them is DOLCE [6], which is a higher-level ontology than CLIB. It contains approximately 100 concepts in total, whereas CLIB contains more than 1000, 147 of which are action classes. In DOLCE, events are called *occurents*. Entity-event relations are denoted by the expression *participation*. DOLCE distinguishes between *temporary* and *constant* participation (and other types of participation as well), distinctions that are not present in CLIB. Similarly to CLIB, DOLCE was used in domain-specific applications. Borgo and Leitão, for instance, used DOLCE to model a manufacturing domain [5].

Other commonly used upper ontologies are: Basic Formal Ontology (BFO) [17, 29] containing 36 classes in total; General Formal Ontology (GFO) [18] containing 79 classes; or Suggested Upper Merged Ontology (SUMO) [23] containing 20,000 terms. As far as we know, there is no published research on guidelines for encoding knowledge described by natural language sentences, for any of these ontologies. However, we

believe that the method we describe in this paper is general enough to be applicable to these upper ontologies as well.

There are several specialized biological or biomedical ontologies currently in use. They generally tend to have a large number of concepts. Systems Biology Ontology (SBO) [14] is an ontology dedicated to a specific branch of biology. It incorporates the concept of *interaction*, which roughly corresponds to events in CLIB. The Gene Ontology (GO) [13] is designed to facilitate the description of gene products. The Systematized Nomenclature of Medicine Clinical Terms (SNOMED-CT) [31] is a much larger biomedical ontology, containing over 400,000 concepts. It is currently in use in different countries. There has been substantial research in revising and auditing this large ontology [25, 32, 28]. In contrast with the issues we discussed in relation to CLIB, the problems identified by this body of work concerned the *taxonomy* of SNOMED-CT. Some similarities with our approach are present however, such as a close collaboration between knowledge engineers and domain experts, and a need to address the mismatch between a common sense meaning of words and their usage in the ontology.

A different type of research with converging goals to ours is Proposition Bank (PropBank) [24] — “a corpus of text annotated with information about basic semantic propositions.” The goal of PropBank is to define a methodology for mapping nouns in a sentence into *arguments* of the verb in that sentence. PropBank arguments correspond loosely to relations of CLIB, but a PropBank argument may reflect the meaning of one or more CLIB relations (e.g., Arg0 denotes both agents and experiencers). As a result, the task we address is much more difficult than the one of PropBank.

One of the resources used by annotators of PropBank texts is a database describing the arguments associated to each verb in a selected vocabulary. For instance, the arguments specified for the verb *to move* are: (a) Arg0: mover (b) Arg1: moved (c) Arg2: destination. If the same noun (entity) plays more than one role in a sentence, only the argument with the highest rank is assigned. This solution could be used in our application as well, in order to prevent awkward translations into natural language when the same entity appears several times in a sentence.

A second resource used by annotators is a detailed set of guidelines provided by [4] for the mapping of nouns into arguments, with specific instructions for sentences with different syntactic structures (e.g., declarative sentences, questions, etc.). Our work also focuses on developing guidelines for a consistent assignment of entities to participant relations of events, but we operate at a higher level of abstraction. We do not look at sentences expressed in natural language directly; rather we assume that sentences are transformed into Universal Truths first.

7 Summary and Conclusions

The work reported in this paper has been driven by the assumption that an explicit representation of knowledge is critical for a system to support reasoning, explanation and dialog. We described some key aspects of creating a knowledge base from a biology textbook. Even though we used specific examples from our project, there are three broad lessons that are of interest to other projects using both manual and automated techniques for knowledge acquisition. These lessons are: (1) reformulating the

sentences so that their abstract structure is closer to the logical form to be acquired (2) use of a linguistically motivated upper ontology (3) use of a combination of syntactic and semantic guidelines to specify how ontological distinctions are expressed in language. We further hope that the three lessons at a general level, and the specifics of the guidelines that we presented, will inspire a new breed of manual, semi-automatic and fully automatic tools for creating knowledge representations that are well-suited for reasoning, explanation and dialog.

8 Acknowledgment

This work has been funded by Vulcan Inc. and SRI International.

Relation	Semantic Definition	Syntactic Definition	Biology Examples
agent	The entity that initiates, performs, or causes an event.	<ul style="list-style-type: none"> the grammatical subject of a sentence in active voice preposition: <i>by</i> (sentence in passive voice) (Assume that biological entities like protein, bacteria, etc., can be agents too.) 	<p>A <i>virus</i> enters a cell.</p> <p>A cell is penetrated <i>by a virus</i>.</p>
object	The entity that is acted upon by an event; the main passive participant in the event.	<ul style="list-style-type: none"> the grammatical object of a sentence in active voice preposition: <i>of</i> 	<p>A <i>virus</i> enters a cell.</p> <p>A cell is penetrated by a virus.</p> <p>... the penetration of a cell by a virus.</p>
instrument	The entity that is used (by the agent if there is one) to perform an event.	<ul style="list-style-type: none"> preposition: <i>with</i> / preceded by: <i>using</i> 	An animal walks <i>using its legs</i> .
raw-material	The entity/ material used as input for an event.	<ul style="list-style-type: none"> the grammatical object of verbs like <i>to use, to consume</i>, etc. preceded by: <i>using</i> 	<p>The Calvin cycle uses <i>the ATP and NADPH</i> to produce sugar.</p> <p>Water is converted to hydrogen.</p> <p>Chemicals are transported, <i>using energy</i>.</p>
result	The entity that comes into existence as a result of an event.	<ul style="list-style-type: none"> the grammatical object of verbs like <i>to produce, to create</i>, etc. preposition: <i>to</i> / preceded by: <i>producing</i> 	<p>Plants produce <i>their own sugars</i> by photosynthesis.</p> <p>Water is converted <i>to hydrogen</i>.</p>
donor	The entity that releases the object of an event (possibly unintentionally).	<ul style="list-style-type: none"> preposition: <i>from</i> 	Heat is transferred <i>from the warmer body</i> to the cooler body.
recipient	The entity that receives (takes possession of) the object of an event.	<ul style="list-style-type: none"> preposition: <i>to</i> 	Heat is transferred from the warmer body <i>to the cooler body</i> .
base	An entity that the event references as something major or relatively fixed.	<i>Irregular – depends on the verb.</i>	<p>Water moves <i>into a cell</i>.</p> <p>Water moves <i>out of a cell</i>.</p> <p>A signal molecule attaches <i>to a receptor protein</i>.</p>
beneficiary	The entity that benefits from an event.	<ul style="list-style-type: none"> preposition: <i>for</i> 	
experiencer	The entity that experiences an event.	<p>For a sentence containing a verb describing an emotional or psychological action:</p> <ul style="list-style-type: none"> the sentence subject (sentence in active voice) preposition: <i>by</i> (sentence in passive voice) 	<p>Plants sense gravity and the direction of light.</p> <p>Gravity and the direction of light are sensed <i>by plants</i>.</p>
origin	The place where an event (typically a movement) begins.	<ul style="list-style-type: none"> preposition: <i>from</i> 	Water moves <i>from a hypotonic solution</i> to a hypertonic solution.
destination	The place where an event (typically a movement) ends.	<ul style="list-style-type: none"> preposition: <i>to</i> 	Water moves from a hypotonic solution <i>to a hypertonic solution</i> .
away-from	The place away from which an event transpires, but not necessarily where the event starts.	<ul style="list-style-type: none"> preposition: <i>away from</i> 	The plasma membrane pulls <i>away from the wall</i> .
toward	The place toward which an event transpires, but not necessarily where the event ends.	<ul style="list-style-type: none"> preposition: <i>toward</i> 	Daughter chromosomes move <i>toward opposite ends of the cell</i> .
path	The place (or other entity) along or through which an entity moves.	<ul style="list-style-type: none"> preposition: <i>across, along, through</i> 	A protein moves into a cell <i>through a pore</i> .
site	The specific place of some effect of an event, as opposed to the locale of the event itself.	<ul style="list-style-type: none"> preposition: <i>at</i> 	The protein binds <i>at the groove</i> with the target molecules of bacterial walls.

Table 4: Summary of guidelines for mapping entities into slots.

Bibliography

- [1] Eva Banik, Eric Kow, and Vinay K. Chaudhri. User-controlled, robust natural language generation from an evolving knowledge base. In *ENLG 2013 : 14th European Workshop on Natural Language Generation*, 2013.
- [2] K. Barker, T. Copeck, S. Delisle, and S. Szpakowicz. Systematic construction of a versatile case system. *Journal of Natural Language Engineering*, 3(4):279–315, 1997.
- [3] K. Barker, B. Porter, and P. Clark. A library of generic concepts for composing knowledge bases. In *First International Conference on Knowledge Capture*, 2001.
- [4] Claire Bonial, Jena Hwang, Julia Bonn, Kathryn Conger, Olga Babko-Malaya, and Martha Palmer. English PropBank annotation guidelines, 2012.
- [5] Stefano Borgo and Paulo Leitão. The role of foundational ontologies in manufacturing domain applications. *LNCS*, 2004.
- [6] Stefano Borgo and Claudio Masolo. Foundational choices in DOLCE. In Steffen Staab and Ruder Studer, editors, *Handbook on Ontologies*. Springer, second edition, 2009.
- [7] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. Toward an architecture for never-ending language learning. In *AAAI*, 2010.
- [8] Vinay K Chaudhri, Britte Cheng, Adam Overholtzer, Jeremy Roschelle, Aaron Spaulding, Peter Clark, Mark Greaves, and Dave Gunning. Inquire Biology: A textbook that answers questions. *AI Magazine*, 34(3), September 2013.
- [9] Vinay K. Chaudhri, Nikhil Dinesh, and Craig Heller. Conceptual models of structure and function. Technical report, SRI International, 2013.
- [10] Vinay K. Chaudhri, Stijn Heymans, Michael Wessel, and Son Cao Tran. Query answering in object oriented knowledge bases in logic programming. In *Workshop on ASP and Other Computing Paradigms*, 2013.
- [11] Vinay K. Chaudhri, Michael A. Wessel, and Stijn Heymans. *KB_Bio_101*: A challenge for OWL Reasoners. In *The OWL Reasoner Evaluation Workshop*, 2013.
- [12] P. Clark, J. Thompson, K. Barker, B. Porter, V. Chaudhri, A. Rodriguez, J. Thomere, S. Mishra, Y. Gil, P. Hayes, and T. Reichherzer. Knowledge entry as the graphical assembly of components. In *First International Conference on Knowledge Capture*, 2001.
- [13] The Gene Ontology Consortium. Gene ontology: tool for the unification of biology. *Nat Genet*, 25:25–29, 2000.
- [14] M. Courtot, N. Juty, C. Knpfer, D. Waltemath, A. Zhukova, A. Drger, M. Dumontier, A. Finney, M. Golebiewski, J. Hastings, S. Hoops, S. Keating, D.B. Kell, S. Kerrien, J. Lawson, A. Lister, J. Lu, R. Machne, P. Mendes, M. Pocock, N. Rodriguez, A. Villeger, D.J. Wilkinson, S. Wimalaratne, C. Laibe, M. Hucka, and N. Le Novre. Model storage, exchange and integration. *Molecular Systems Biology*, 7(543), 2011.

- [15] Nikhil Dinesh, Aravind K. Joshi, Insup Lee, and Oleg Sokolsky. Logic-based regulatory conformance checking. In *Monterey Workshop*, pages 147–160, 2007.
- [16] Aldo Gangemi, Nicola Guarino, Claudio Masolo, and Alessandro Oltramari. Sweetening Wordnet with DOLCE. *AI Magazine*, 24(3):13–24, 2003.
- [17] P. Grenon, B. Smith, and L. Goldberg. Applying BFO in the biomedical domain. *Health Technology and Informatics*, 102:20–38, 2004.
- [18] Heinrich Herre, Barbara Heller, Patryk Burek, Robert Hoehndorf, Frank Loebe, and Hannes Michalek. General formal ontology (GFO): A foundational ontology integrating objects and processes. <http://www.onto-med.de/ontologies/gfo/>, 2013.
- [19] Graeme Hirst. Ontology and the lexicon. In *Handbook on ontologies*, pages 269–292. Springer, 2009.
- [20] S. M. Lloyd, editor. *Roget's Thesaurus*. Longman, 1982.
- [21] Andrew McCallum and Wei Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 188–191. Association for Computational Linguistics, 2003.
- [22] George A. Miller and Christiane Fellbaum. Wordnet then and now. *Language Resources and Evaluation*, 41(2):209–214, 2007.
- [23] Ian Niles and Adam Pease. Towards a standard upper ontology. In *Proceedings of the international conference on Formal Ontology in Information Systems - Volume 2001*, FOIS '01, pages 2–9, New York, NY, USA, 2001. ACM.
- [24] Martha Palmer, Dan Gildea, and Paul Kingsbury. The Proposition Bank: A corpus annotated with semantic roles. *Computational Linguistics Journal*, 31(1), 2005.
- [25] Alan Rector, Luigi Iannone, and Robert Stevens. Quality assurance of the content of a large DL-based terminology using mixed lexical and semantic criteria: experience with SNOMED CT. In *Proceedings of the sixth international conference on Knowledge capture, K-CAP '11*, pages 57–64, New York, NY, USA, 2011. ACM.
- [26] Jane B. Reece, Lisa A. Urry, Michael L. Cain, Steven A. Wasserman, Peter V. Minorsky, and Robert B. Jackson. *Campbell biology*. Benjamin Cummings imprint of Pearson, Boston, 2011.
- [27] Brian C. Smith. *Reflection and Semantics in a Procedural Language*. PhD thesis, Massachusetts Institute of Technology, 1982.
- [28] Kent A. Spackman and Guillermo Reynoso. Examining SNOMED from the perspective of formal ontological principles: Some preliminary analysis and observations. In *KR-MED*, pages 72–80, 2004.
- [29] A. D. Spear. Ontology for the twenty first century: An introduction with recommendations. <http://www.ifomis.org/bfo/documents/manual.pdf>, 2006.
- [30] D. Summers, editor. *Longman Dictionary of Contemporary English*. Longman, 1987.
- [31] Amy Y Wang, Jeremiah H Sable, and Kent A Spackman. The SNOMED clinical terms development process: refinement and analysis of content. *Proc AMIA Symp*, pages 845–9, 2002.
- [32] Yue Wang, Michael Halper, Hua Min, Yehoshua Perl, Yan Chen, Kent A. Spackman, All Communications To, and Yehoshua Perl. Structural methodologies for auditing SNOMED.

Qualitative Analysis of Cotemporary Urdu Machine Translation Systems

Asad Abdul Malik, Asad Habib

Kohat University of Science and Technology, Kohat, Pakistan

asad_12204@yahoo.com, asadhabib@kust.edu.pk

Abstract. The diversity in source and target languages coupled with source language ambiguity makes Machine Translation (MT) an exceptionally hard problem. The highly information intensive corpus based MT leads the MT research field today, with Example Based MT and Statistical MT representing two dissimilar frameworks in the data-driven paradigm. Example Based MT is another approach that involves matching of examples from large amount of training data followed by adaptation and re-combination.

Urdu MT is still in its infancy due to nominal availability of required data and computational resources. This paper provides a detailed survey of the aforementioned contemporary MT techniques and reports findings based on qualitative analysis with some quantitative BLEU metric quantitative results. Strengths and weaknesses of each technique have been brought to surface through special focus and discussion on examples from Urdu language. The paper concludes with proposal of future directions for research in Urdu machine translation.

Keywords: Urdu Machine Translation, Qualitative Comparison, Rule Based MT, Statistical MT, Example Based MT

1 Introduction

Representing text in one natural language, the source language (SL) into another, the target language (TL) is as old as the written literature [1]. At present, the need of translation is continuously growing in business, economy, medical and many other fields. The growth in science and technology in general and computer based solutions in particular have paved the way to the concept of automatic translation called the Machine Translation (MT) [2].

1.1 Urdu

Urdu ranks 19th among the 7,105 languages spoken in the world¹. It is one the most-spoken languages in South Asia [3]. It is also spreading in the West due to the large

¹ <http://www.ethnologue.com/statistics/size>

Diaspora of Indo-Pak Subcontinent citizens. Urdu is the national language of Pakistan and it is used i) as medium of teaching in most of the public schools ii) for junior to mid level administration and iii) in the mass print and electronic media. It is not only spoken in Pakistan but also in India, Bangladesh, Afghanistan and Nepal. Also it has become the culture language and lingua franca of the South Asian Muslim Diaspora outside the Indo-Pak subcontinent, mainly in the Middle East, Europe, Canada and the United States [4].

1.2 Urdu Machine Translation (UMT)

In spite of the large number of speakers around the world, there are very few computational natural language tools available for Urdu. It is a morphologically rich language having many other distinct linguistic characteristics. On the contrary it is still an under-resourced language from the point of view of computational research. We could not find any public domain machine translation tool(s) developed specifically for Urdu. However some trace of basic MT techniques has been discovered [5-9]. In the current work we presented a detailed survey on the contemporary research in UMT. We identified the weaknesses and strengths of each technique and proposed the guidelines for future directions in UMT research.

2 Literature Survey

Some traces of basic UMT research are presented in this section. Naila et al [5] presented a Rule Based English to Urdu Machine Translation (RBMT) technique primarily based on the transfer approach that tries to handle the case phrases and verb postpositions using Paninian grammar. Statistical Machine Translation (SMT) between languages with word order differences was discussed by Bushra et al [6]. Example Based Machine Translation (EBMT) approach was introduced by Maryam and Asif that translates text form English to Urdu that supports idioms and homographs [7]. Parallel corpus for statistical machine translation for English to Urdu text was presented by Aasim et al [8]. Word-Order Issues in English-to-Urdu have been investigated by Bushra and Zeman [9]. In addition, SMT systems such as Google² and Bing³ are already available online. However these systems offer poor translation quality and limited accuracy due to issues related to Urdu syntax and other intrinsic linguistic features.

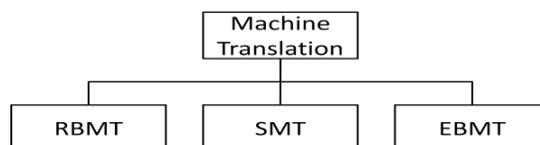


Fig. 1. Paradigms for Machine Translation

² [http:// translate.google.com](http://translate.google.com)

³ <http://www.bing.com/translator>

Contemporary Machine Translation techniques can be broadly categorized into three paradigms as shown in Figure 1.

2.1 Rule Based Machine Translation (RBMT)

To provide suitable rules for translation, the RBMT needs linguistic knowledge of source as well as the target language. Translation depends on formalized linguistic knowledge represented in lexicon along with grammars [10]. RBMT is described by several characteristics; it has firm set of well fashioned rules, several rules rely on present linguistic theories and the grammatical errors are prohibited. The major advantage of RBMT is that if the required knowledge is not found in available literature then ad-hoc heuristic rules are applied [5]. This system contains input sentence analyzer (morphological, syntactic and semantic analysis) and procedures for producing output (structural transfers and inherent Inter-lingua structures).

2.2 Statistical Machine Translation (SMT)

Two models are built in SMT; i) Translation model and ii) Language model. A translation model gives probability of a target sentence given source sentence $P(T/S)$ whereas the language model determines the probability $P(S)$ of the string of target language actually occurring in that language. By using the language model and conditional probabilities of translation model, $P(S/T)$ is calculated using the following formula:

$$P\left(\frac{S}{T}\right) = \frac{P(S)P\left(\frac{T}{S}\right)}{P(T)}$$

Probability based analysis of MT is part of SMT. It has numerous diverse applications such as those in word sense disambiguation or structural disambiguation etc. [11]. The SMT techniques do not need explicit encoding of the linguistic information. It highly depends upon availability of fine and very large amount of bilingual data that presently does not exist for Urdu and other languages spoken in the Indo-Pak Subcontinent region.

2.3 Example Based Machine Translation (EBMT)

Somers referred to EBMT as a hybrid approach of RBMT and SMT [12]. Like SMT, it is depended upon a corpus of available translations. That is why it is similar to (often confused with) translator's aid known as Translation Memory (TM). EBMT and TM both involve comparison of input text with the database of real examples and then find out the nearest match. In TM, a translator selects the candidate target text whereas EBMT makes use of automated procedures that identify the translation fragments. Recombination of these fragments produces the target text [10].

Thus the process is split into three phases [10]. i) "Matching" fragments against the available database of real examples (that are common between EBMT and TM), ii)

“Alignment” identifying corresponding translation fragments and finally iii) “Recombination” that gives the target text. EBMT needs a database of parallel translations that are searched for source language phrases or sentences and their nearest matching target language components are generated as output [11].

EBMT saves the translation examples in different manners. In simple case, examples are saved as pairs of strings with no extra information related to them.

3 Methodology

In this section we discuss the methodologies of three major Machine Translation techniques. English is considered as source language and Urdu as the target language. We compare the strengths and weaknesses of these techniques in Section 4.

3.1 Rule Based Machine Translation

There are three stages in RBMT; i) Analysis, ii) Transfer and iii) Synthesis

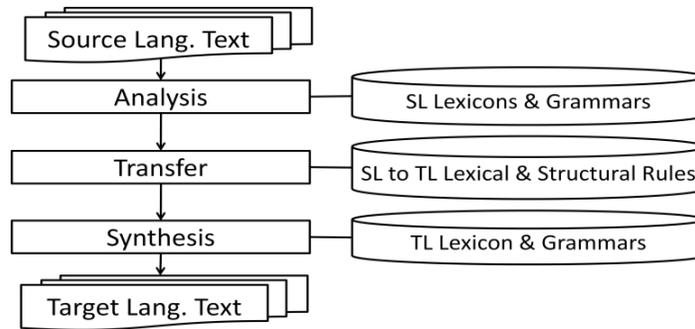


Fig. 2. RBMT Model

Analysis.

The source text is analyzed based upon lexicon and grammar rules of source language. Word segmentation is done and each word is annotated by appropriate POS tag and parse tree of input text is created. A parse tree for the input text “I called you several times” is created as shown in figure 3.

Transfer.

In this stage, parse tree of source language text is ‘transferred’ into parse tree of desired target language according to the lexicon and structural rules of the target language. English is SVO (Subject, Verb, Object) language whereas Urdu is SOV language. Re-ordering of words is inevitable in order to generate the output parse tree as shown in Figure 4.

English to Urdu Translation Rules.

Some coarse grained rules for translation from English to Urdu are mentioned in the following.

1. NP in both languages follows the same rule. So swapping in not required.
2. If NP is having NP and PP, then transform it as in Urdu PP comes before NP.
 English $NP \rightarrow NP + PP$
 Urdu $NP \rightarrow PP + NP$
3. If adverb phrase (AP) appears before verb then swapping is not needed. AP in English can appear in different order depending on the type of AP, however Urdu prefers AP before verb.
 Urdu $AP + V$
4. In Urdu, Verb phrase (VP) is inflected according to gender, number and person (GNP) of the head noun while NP depends upon tense, aspect and modality of the verb phrase (VP). Urdu adjectives are also modified by GNP of the head noun.

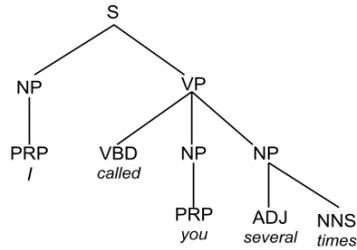


Fig. 3. English Parse Tree

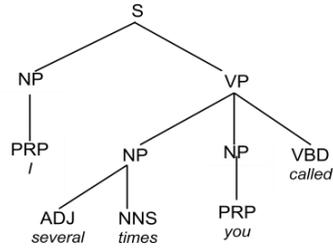


Fig. 4. Parse Tree (transferred in SOV)

Synthesis.

Finally, the target language lexicon and grammar is used to convert the parse tree of target language to the target language surface form. It requires two independent monolingual dictionaries so that appropriate surface form of target language can be generated.

As shown in figure 5 the source text "I called you several times" is translated into "میں کئی مرتبہ آپ کو بلايا" using RBMT.

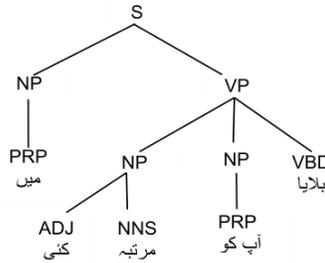


Fig. 5. Urdu parse tree

3.2 Statistical Machine Translation (SMT)

SMT makes use of i) Translation Model, ii) Language Model and iii) Decoder Algorithm.

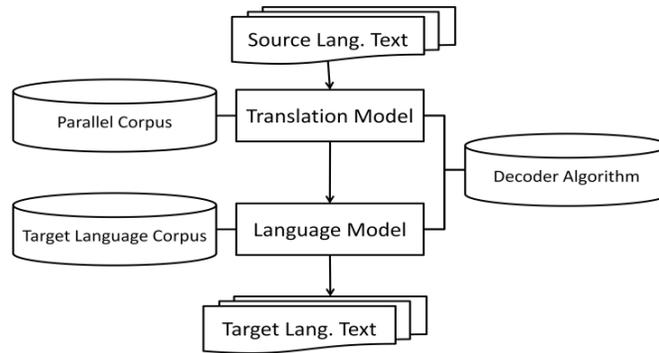


Fig. 6. SMT Model

Translation Model.

Words and phrases in the source text are matched against the target language strings. If the strings are matched the model assigns a probability value $P(T/S)$ to it. This probability shows that what are the chances that the input text string is present in the output or target language. These probability values are pre-assigned in a parallel corpus through human translation. Subsequently machine learning techniques are used to improve the system depending upon the human translated text.

Language Model.

Language model determines the probability $P(S)$ of output text string. It does not require a parallel corpus. It requires text in only one language. We can calculate the value by using N-gram model. In this the probability of occurrence of sentence of length N is the product of probability of each k^{th} word given the occurrence of previous words $k-1$ and $k-2$.

Decoder Algorithm.

After finding the product of translation and language model the decoder algorithm selects the string of output text language with the highest probability value based on the stochastic formula mentioned in Section 2.2.

3.3 Example Based Machine Translation (EBMT)

English to Urdu EBMT is divided into four phases; i) Sentence Fragmentation, ii) Search in Corpus, iii) N-ary Product based Retrieval and iv) Ordering of Translated Text.

Sentence Fragmentation.

For better handling of input sentence by translator, it is better to break the sentence into phrases. On the other hand same results are achieved by storing sentence in the corpus and by gaining a broad coverage by fragmenting and combining using genetic algorithm at run time for obtain new sentences. Fragmentation of a sentence into phrases is handled by using concept of idioms, cutter points and connecting words.

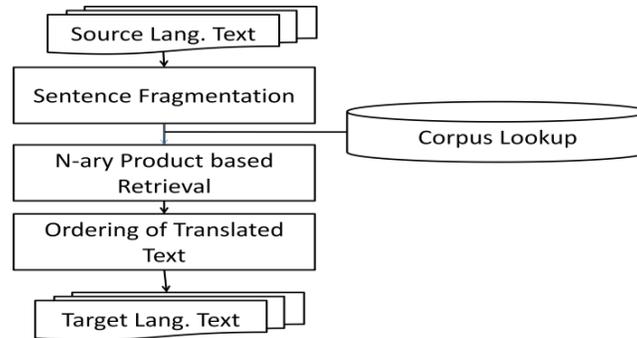


Fig. 7. EBMT Model

Searching in Corpus.

Bilingual corpus is searched for finding whether the input phrase is accessible or not. If the system is unable to locate exact match, then in that situation it will look for the nearest match. Closeness is calculated by threshold at two stages; i) for exact match and ii) for nearest match. This is done by two algorithms “Levenshtein Algorithm” and “Semantic Distance Algorithm”.

N-ary Product Based Retrieval.

The translation for an input sentence is extracted in this stage. And there is possibility that input can have many translations. So the possibilities are collected and the idea of n-ary product is used to record all the feasible sentences.

Ordering of Translated Phrases.

If a single input sentence is divided into pieces and translated into output language phrase, then ordering of these translated phrases are done in this phase.

4 Comparison

4.1 Rule Based Machine Translation

The quality of translation in Rule Based Machine Translation (RBMT) depends upon large number of rules. Therefore its computational cost is very high. Rules are based on both source and target languages, their respective morphological, syntactical and

semantic structures. With a large set of large and fine grained linguistic rules, RBMT generates translation with acceptable quality, but developing system like this needs more time and man hours because this type of linguistic recourses should be hand crafted (Knowledge Acquisition Problem). As RBMT works with exact matches, it is unable to translate text when system does not have enough knowledge about the input. It is also difficult to add more rules for generating high quality output.

4.2 Statistical Machine Translation

The knowledge about translation is acquired automatically from the example data. This is the main reason why SMT is developed fast as compared to RBMT. In a situation where large corpus is available but linguistic knowledge is not readily available then SMT is a preferred method. When input and output languages are not complex morphologically then SMT techniques generate better results. SMT based approaches do not need Bilingual dictionaries. They depend upon the quality of bilingual corpus.

4.3 Example Based Machine Translation

It requires Bilingual dictionary. It translates text by adapting to examples. The computational cost is less than RBMT. By storing proper examples in the DB the system can be upgraded. It works on best matching reasoning, so therefore when the corresponding example is not available in corpus, the translation process becomes complicated. It translates in fail-safe way. Quality of translation depends upon the difference between input text and lookup results for similar examples. EBMT can also notify us that when its translation is improper.

Table 1. Comparison of RBMT, SMT and EBMT

	Advantages	Disadvantages
Rule Based Machine Translation	<ul style="list-style-type: none"> - Effective for core phenomena - Based on linguistic theories - Easy to build an initial system 	<ul style="list-style-type: none"> - Rules are formulated by experts - Sometimes the experts do not agree hence the system remain unreliable. - Difficult to maintain and extend - Ineffective for marginal phenomena
Statistical Machine Translation	<ul style="list-style-type: none"> - Numerical knowledge - Extracts knowledge from corpus - Reduces the human cost - Model is mathematically grounded 	<ul style="list-style-type: none"> - Less linguistic background - Overall lookup cost is high - Hard to capture long distance phenomena - Authenticity of results can be questionable. - Not suitable for free word order languages
Example Based Machine Translation	<ul style="list-style-type: none"> - Extracts knowledge from corpus - Based on translation patterns in corpus - Reduces the human cost 	<ul style="list-style-type: none"> - Similarity measure is sensitive to system - Lookup cost can be high - Knowledge acquisition is problematic - Trade off is required between corpus size and performance.

5 Findings

The qualitative findings are tabulated in table 1, and the quantitative findings are mentioned in table 2. The BLEU metric is used for the evaluation of the machine translated text, five reference sentences were used for calculating the BLEU value. From the value of the BLEU it is clearly shown that EBMT performs better than the rest of the three systems. RBMT was found to be better than both the SMT systems. Out of the two SMT (Google and Bing), Bing translator gave better results than the Google translator.

Table 2. BLEU value of RBMT, EBMT and SMT

	RBMT	EBMT	SMT	
			Google	Bing
BLEU Value	0.8	0.8421	0.6268	0.709

6 Discussion

After detailed literature study and investigation of the above mentioned three MT systems, we can conclude that for languages with similar lexical and syntactic structure e.g. Urdu and Hindi, the Rule based MT technique gives better results. The SMT systems perform better if necessary resources such as annotated corpora etc. are available. At present, most of the systems translate text from source to target language on the basis of single sentence whereas in real life text for translation is much larger than one sentence. Nonetheless, the continuous process of repetitive translation and improvements by human annotators contribute significantly to any MT system.

7 Conclusion and Future Directions

In this paper we explained three main techniques of machine translation; Rule Based Machine Translation, Statistical Machine Translation and Example Based Machine Translation. We explained the methodology of each of these systems and found their comparison based on their respective outputs using carefully selected text. Our current work is preliminary in nature. However it reports significant results based on qualitative analysis.

In order to contribute a significant role to UMT research, at present we are in the process of building the required corpora. We intend to use our corpora to conduct larger scale automated experiments and report quantitative results that are comparable to human translators. Based on our qualitative and quantitative results, we aim at proposing a new model that minimizes flaws in the existing Urdu MT systems. Ideally, we would like to implement our proposed system with fewer requirements of computational and human resources.

8 REFERENCES

1. Abdullah, H., Homiedan.: Machine translation. J. King Saud Uni. Lang. & Trans. 10, 1-21 (1998)
2. Hutchins, J.: Latest Development in Machine Translation Technology: Beginning a New Era in MT RESEARCH. MT Summit IV, 11-34, Kobe, Japan (1993)
3. Lewis, Paul, M., Simons, G.F., Fennig, C.D.: Ethnologue: Language of the World. Seventeenth edition. Dallas, Texas: SIL International (2013)
4. Schmidt, R.L.: Urdu An Essential Grammar. Rutledge Taylor & Francis Group. London and New York (2004)
5. Ata, N., Jawaid, B., Kamran, A.: Rule Based English to Urdu Machine Translation. Proceedings of Conference on Language and Technology (CLT'07). (2007)
6. Jawaid, B., Zeman, D., Bojar, O.: Statistical Machine Translation between Languages with Significant Word Order Difference. Prague (2010)
7. Zafar, M., Masood, A.: Interactive English to Urdu Machine Translation using Example-Based Approach. IJCSE 1 (3), 276-283 (2009)
8. Ali, A., Siddiq, S., Malik, M.K.: Development of Parallel Corpus and English to Urdu Statistical Machine Translation. IJET-IJENS 10(5), 31-33 (2010)
9. Jawaid, B., Zeman, D.: Word-Order Issues in English-to-Urdu Statistical Machine Translation. Prague Bull. Math. Linguistics. 87-106 (2011)
10. Survey of Machine Translation Evaluation. EuroMatrix. (2007)
11. Samantaray, S.D.: Example based machine translation approach for Indian languages. ICCS. 1-10 (2004)
12. Somers, H. :Machine translation and Welsh: The way forward. A Report for the Welsh Language Board, Centre for Computational Linguistics, UMIST, Manchester (2004)

The NL2KR system

Chitta Baral, Juraj Dzifcak, Kanchan Kumbhare, and Nguyen H. Vo

School of Computing, Informatics, and Decision Systems Engineering,
Arizona State University, Tempe, Arizona, USA
chitta@asu.edu, Juraj.Dzifcak@asu.edu, krkumbha@asu.edu,
Nguyen.H.Vo@asu.edu

Abstract. In this paper we will describe the NL2KR system that translates natural language sentences to a targeted knowledge representation formalism. The system starts with an initial lexicon and learns meaning of new words from a given set of examples of sentences and their translations. We will describe the first release of our system with several examples.

Keywords: Natural Language Understanding, Lambda Calculus, Knowledge Representation

1 Introduction and Motivation

Our approach to understanding natural language involves translating natural language text to formal statements in an appropriate knowledge representation language so that a reasoning engine can reason with the translated knowledge and give a response, be it an answer, a clarifying question or an action. To translate natural language text to a formal statement we propose to use the compositional method of Montague [1] where the translation (or meaning) of words are given as lambda calculus formulas and the meaning of phrases and sentences are obtained by composing the meaning of the constituent words. The challenge in doing this is in coming up with appropriate lambda calculus expressions for each word. The challenging aspects in this are: (a) the number of words may be huge, (b) the lambda calculus expression (or meaning) of some words are too complicated for humans to come up with it, and (c) the lambda calculus expressions for the words are target language specific; so it is not a one time affair like compiling traditional dictionaries. To address these challenges we use an inverse lambda algorithm [2] that computes the meaning of a word/phrase G when the meaning of the word/phrase H and the phrase GH (or HG) is known.

The NL2KR system uses an initial lexicon containing some words and their meanings and a set of training corpus containing sentences in natural language and their translations to learn new meanings of words. The system then uses the new learned lexicon to translate new sentences. In this paper, we would like to give an overview of the NL2KR system and examples of using it.

2 Overview

Shown below in Fig. 1 is the architecture of the NL2KR system. It has two sub-parts which depend on each other (1) NL2KR-L for learning and (2) NL2KR-T for translating.

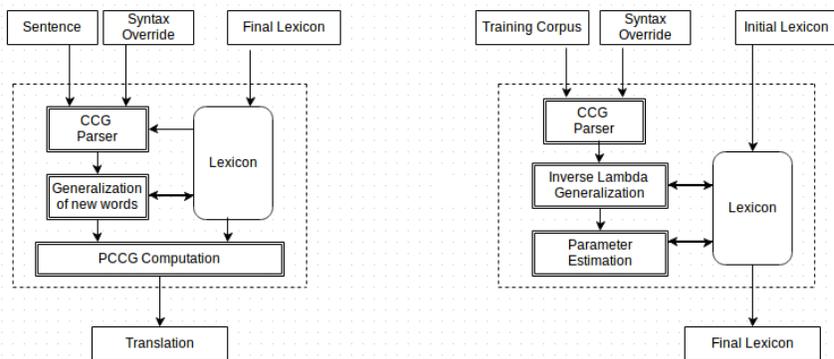


Fig. 1. Architecture of the NL2KR system: NL2KR-T on the left and NL2KR-L on the right

The NL2KR-L sub-part takes an initial lexicon consisting of some words and their meanings in terms of λ -calculus expressions & a set of training sentences and their target formal representations as input. It then uses a Combinatorial Categorical Grammar (CCG) parser to construct the parse trees. Next, the learning sub-part of the system uses Inverse- λ and Generalization algorithms to learn meanings of newly encountered words, which are not present in the initial lexicon, and adds them to the lexicon. A parameter learning method is then used to estimate a weight for each lexicon entry (word, its syntactic category and meaning) such that the joint probability of the sentences in the training set getting translated to their given formal representation is maximized. The result of NL2KR-L is the final lexicon, which contains a larger set of words, their meanings and their weights.

Once the training component finishes its job, the translation sub-part (NL2KR-T) uses this updated lexicon and translates sentences using the CCG parser. Since words can have multiple meanings and their associated λ -calculus expressions, weights assigned to each lexical entry in the lexicon helps in deciding the more likely meaning of a word in the context of a sentence.

3 Using NL2KR

The latest version of NL2KR system can be downloaded from <http://nl2kr.engineering.asu.edu>. It can be run on Linux (64 bit) and OS-X.

3.1 Third Party Software Used by NL2KR

The current version of NL2KR uses the following libraries and tools developed by others:

- Scripting language Python (version ≥ 2.6)
- AspCcgTk version 0.3 ¹
- Stanford Log-linear Part-Of-Speech Tagger ² (version 3.1.5)
- Oracle Java (version ≥ 1.6)
- ASP grounder and solver: gringo (version 3.x), clasp (version 2.x) and clingo (version 3.x) ³

3.2 Installation guide

The NL2KR package contains a readme file and a zipped file which contains

- AspCcgTk
- Jar file and models of Stanford Log-linear Part-Of-Speech Tagger
- Jar file and configurations for NL2KR
- Gringo, clasp and clingo

After unzipping the package, the instruction in the readme file directs how to install NL2KR.

3.3 Files/Folders in the package

Following is a brief list of important files/folders in the package:

- **README**: Installation instruction and examples of how to use NL2KR.
- **NL2KR.jar**: NL2KR's classes packed in a jar file.
- **config.properties**: Default configuration of the NL2KR system.
- **install.sh**: Script to install NL2KR.
- **ccgParser.sh**: Script to get a CCG parse tree of a given sentence.
- **Generalization.sh**: Script that gives generalized meanings of a word.
- **Inverse.sh**: Script to compute the inverse using the inverse lambda algorithms.
- **Lambda.sh**: Script to do application operation, given a function and an argument in λ -calculus.
- **NL2KR-L.sh**: Script to run the NL2KR-L sub-part.
- **NL2KR-T.sh**: Script to run the NL2KR-T sub-part.
- **RunConfiguration**: Example inputs of the preceding scripts.
- **resources**: Folder containing AspCcgTk package, gringo, clasp and clingo.
- **examples**: Folder containing examples in various domains for NL2KR-L and NL2KR-T.

¹ http://www.kr.tuwien.ac.at/staff/former_staff/ps/aspcggtk

² <http://nlp.stanford.edu/software/tagger.shtml>

³ <http://potassco.sourceforge.net/>

3.4 Executing the scripts

To execute any of the scripts one needs to go to the NL2KR's root directory and run

```
./<script name> <RunConfiguration file> [Optional params]
```

where *script name* is the name of one of the six scripts (e.g. ./Lambda.sh), RunConfiguration file contains corresponding parameters for the script, and optional parameters are for the Java Virtual Machine (JVM) to execute the module that corresponds to the script. For learning and testing large dataset with NL2KR-L or NL2KR-T, it is recommended to provide more memory for the JVM and enable garbage collection if needed (i.e. Use -Xmx and -XX:-UseGCOverheadLimit).

3.5 Lambda application

To use the lambda application script, the function and the argument of the lambda application need to be provided. For example, the following snippet in the RunConfiguration file specifies that we need to apply the argument $\#x.x@mia$ to the function $\#y.\#x.loves(x,y)$ (note: $\#$ is for λ).

```
function=#y.#x.loves(x,y)
argument=#x.x@mia
```

The output of the lambda application is $function@argument$. According to lambda application, the y variable is replaced by $\#x.x@mia$ and the result is $\#x.loves(x, \#x0.x0@mia)$, where x in the argument is renamed to $x0$. Running the lambda application script with the preceding configuration yields the output:

```
Function=#y.#x.loves(x,y)
Argument=#x.x@mia
Result= #x.loves(x,#x0.x0 @ mia)
```

However, in the following configuration, the argument cannot be applied to the function since there is no free variable in the function.

```
function = loves(x,y)
argument = #x.x @ mia
```

The output thus would be

```
Function = loves(x,y)
Argument = #x.x @ mia
Cannot apply the argument#x.x @ mia to the function loves(x,y)
```

3.6 Inverse application

Given two lambda expressions g and h , the lambda application gives us $f = g@h$ or $f = h@g$. But sometime, we have only f and g and we need to find h . The inverse application allow us to calculate the lambda expression h so that $f = g@h$ or $f = h@g$. More details about the inverse application can be found in [2]. For inverse application, we need to provide the parent (lambda expression f), the right child (r) or the left child (l). Given one child, the module will calculate the other child so that $f = l@r$.

In the first example below, since there does not exist a lambda expression h (right child) so that $mia@h = \#x.loves(x, mia)$, the inverse algorithm returns right child expression = null. However, when mia is the right child, inverse lambda returns $Leftchild = \#x1.\#x.loves(x, x1)$ because $\#x1.\#x.loves(x, x1)@mia = \#x.loves(x, mia)$. In case the CCG parse tree specifies that the meaning of “Mia” must be in the left child, using left child as $\#x.x@mia$ instead of mia will do the trick and let us have the same right child: $\#x1.\#x.loves(x, x1)$.

Example 1. Input:

```
parent=#x.loves(x,mia)
left_child=mia
```

Output:

```
Parent = #x.loves(x,mia)
Left child =mia
Right child = null
```

Example 2. Input:

```
parent=#x.loves(x,mia)
right_child=mia
```

Output:

```
Parent = #x.loves(x,mia)
Right child = mia
Left child = #x1.#x.loves(x,x1)
```

Example 3. Input:

```
parent=#x.loves(x,mia)
left_child=#x.x @ mia
```

Output:

```
Parent = #x.loves(x,mia)
Left child =#x.x @ mia
Right child = #x1.#x.loves(x,x1)
```

3.7 Generalization

The inverse lambda module is not always adequate to learn new meanings of words when we lack meaning of words that will allow us to use the inverse lambda module. To address that we have developed a generalization module in NL2KR system, where the generalization technique described in [2] is implemented. For example, if we want to find the meaning of the word *plays* with the category $(S\backslash NP)/NP$ using generalization, we can use the lexical entry $(eats, (S\backslash NP)/NP, \lambda y.\lambda x.eats(x, y))$ in the lexicon, where the category of the word *eats* is same as that of the word *plays*. The generalization module will add a new lexical entry $(plays, (S\backslash NP)/NP, \lambda y.\lambda x.plays(x, y))$ to the lexicon. The input of the generalization module is the file path for lexicon (existing dictionary) and a new word, which we want to generalize. Following is an illustration of the use of generalization with the RunConfiguration file having the following:

```
lexicon=./examples/sample/dictionary.txt
word=Mia
cgg=N
```

where **dictionary.txt** contains

```
Vincent N vincent
Vincent N #x.vincent(x)
takes (S\NP)/NP #w.#z.(w@ #x.takes(z,x))
plane N #x.plane(x)
5 boxer N #x.boxer(x)
fights S\NP #x.fight(x)
```

In this case, the word **Mia** can be obtained by generalization from the words Vincent, plane and boxers, each of category N (meaning noun); and the output is

```
New lexical items learned through Generalization:
Mia [N] #x.mia(x)
Mia [N] mia
```

We can restrict generalization by modifying the `config.properties` file. For example, adding the following snippet to `config.properties` will skip the generalization process for NP and N categories, and generalization for the words: *on*, *of*, *by* and *in*.

```
GENERALIZATION_D_EXCLIST=[NP],[N]
GENERALIZATION_D_PREPOSITIONLIST=on,of,by,in
```

3.8 CCG parser

The input of the CCG parser module is the sentence we want to parse and an optional path of the file containing the words and their additional categories we want to use. The parser will parse the input sentence and output its CCG parse tree. Our CCG parser is based on *ASPcggTk* [3] with some modifications such as our use of the Stanford Part-Of-Speech tagger [4] instead of the C&C Part-Of-Speech tagger [5] to improve accuracy. Following is an example snippet of the `RunConfiguration` file.

```
sentence='Every boxer walks'
syntaxFile=./examples/sample/syntax.txt
```

where **syntax.txt** contains

```
takes (S\NP)/NP
Every (S/(S\NP))/NP
Some (S/(S\NP))/NP
walks S\NP
5 fights S\NP
loves (S\NP)/NP
```

The output of the CCG parser is a parse tree of the sentence “Every boxer walks” in ASP format as follows

```
n12kr_token(t1, "Every", "(S/(S\NP))/NP", 1).
n12kr_token(t2, "boxer", "NP", 2).
n12kr_token(t3, "walks", "S\NP", 3).
n12kr_token(t4, "Every boxer", "S/(S\NP)", 1).
5 n12kr_token(t5, "Every boxer walks", "S", 1).
n12kr_child_left(t4, t1).
n12kr_child_right(t4, t2).
n12kr_child_left(t5, t4).
n12kr_child_right(t5, t3).
10 n12kr_valid_rootNode(t5).
```

The predicate *nl2kr_valid_rootNode* is used to specify the root node of the parse tree (corresponds to the whole sentence). *nl2kr_token* is used to specify nodes in the tree and *nl2kr_child_left*, *nl2kr_child_right* are for the left child and the right child of a node. The four atoms of *nl2kr_token* are respectively the node ID, the corresponding phrase, its CCG category and its starting position in the sentence.

3.9 NL2KR-L: the learning module of NL2KR

To run the learning module NL2KR-L we need to set the initial lexicon file path, the override file for syntax categories (optional), the training data file and the output dictionary file path (optional) in the RunConfiguration file of the NL2KR-L. Following is an example snippet of the RunConfiguration file.

```
Ldata=./examples/sample/train.txt
Ldictionary=./examples/sample/dictionary.txt
Lsyntax=./examples/sample/syntax.txt
Loutput=./examples/sample/dictionary_train.out
```

For example, the preceding snippet specifies that: the training data is in **./examples/train.txt**, the initial lexicon is in **./examples/sample/dictionary.txt**, and the override syntax categories are in **./examples/sample/syntax.txt**. The override syntax categories will be used in CCG parsing step as showed in the previous subsection. If it is not specified, the output dictionary is saved as **dictionary_train.out** in **./output** folder.

The training data file contains the training sentences and their formal representation such as:

```
Some boxer walks EX. (boxer(X) ^ walk(X))
John takes a plane EX. (plane(X) ^ takes(john, X))
John walks walk(john)
```

In the above, *EX* denotes $\exists X$.

The initial lexicon contains words and the their meanings that we already know:

```
John N john
takes (S\NP)/NP #w. #z. (w@ #x. takes(z,x) )
plane N #x. plane(x)
boxer N #x. boxer(x)
5 fights S\NP #x. fight(x)
```

The NL2KR-L sub-part learns the new meanings of words in multiple iterations. It stops when it cannot learn any new word. Below we give the output of the script with example inputs.

We start with some snippets of the running output in the following. From line 5 to 9, NL2KR-L was checking if it has the meaning of *Some boxer* and *walks*. It then learned the meaning of *walks* by generalization.

From line 15 to 19, NL2KR-L was trying to learn the meaning of *Some boxer* given the meaning of *walks* and the meaning of the whole sentence *Some boxer walks* from the training data. Using inverse lambda, it figured out that the meaning of *Some boxer* is $\#x1.EX.boxer(X) \wedge x1@X$.

NL2KR-L did not go further to the meaning of “some” because the meaning “boxer” of *boxer* was not helpful.

However, using the second parse tree where the meaning $\#x.\text{boxer}(X)$ of *boxer* is used, NL2KR-L can get the meaning of *some* (line 42-49) : $\#x2.\#x1.EX.x2@X \wedge x1@X$.

```

*****Learning lexicon ...
...
Processing sentence number 1
Processing Parse Tree 1
5 Word : Some boxer walks sem::null
Both children do not have current lambda expression: Some boxer,walks
Generalizing for leafs with no expected lambda: walks
New lexical item Learned by Expansion: walks////[S\NP]////#x.walk(x)
...
10 Processing sentence number 1

Processing Parse Tree 1

Word : Some boxer walks sem::null
15 Applying inverse : EX.boxer(X) ^ walk(X) #x.walk(x)
INVERSE_L Tried:
Some boxer walks(H) = EX.boxer(X) ^ walk(X)
walks(G) = #x.walk(x)
Some boxer(F) = #x1.EX.boxer(X) ^ x1 @ X
20 Word : walks sem::#x.walk(x)
Word : Some boxer sem::null
Applying inverse : #x1.EX.boxer(X) ^ x1 @ X boxer
INVERSE_L Tried:
Some boxer(H) = #x1.EX.boxer(X) ^ x1 @ X
25 boxer(G) = boxer
Some(F) = null
Generalizing for leafs with no expected lambda: Some
Generalizing for leafs with no expected lambda: boxer
Word : boxer sem::boxer
30 Word : Some sem::null

Processing Parse Tree 2

Word : Some boxer walks sem::null
35 Applying inverse : EX.boxer(X) ^ walk(X) #x.walk(x)
INVERSE_L Tried:
Some boxer walks(H) = EX.boxer(X) ^ walk(X)
walks(G) = #x.walk(x)
Some boxer(F) = #x1.EX.boxer(X) ^ x1 @ X
40 Word : walks sem::#x.walk(x)
Word : Some boxer sem::null
Applying inverse : #x1.EX.boxer(X) ^ x1 @ X #x.boxer(x)
INVERSE_L Tried:
Some boxer(H) = #x1.EX.boxer(X) ^ x1 @ X
45 boxer(G) = #x.boxer(x)
Some(F) = #x2.#x1.EX.x2 @ X ^ x1 @ X
Word : boxer sem::#x.boxer(x)
Word : Some sem::null
New lexicon Learned: Some////[(S/(S\NP))/NP]////#x2.#x1.EX.x2 @ X ^ x1 @ X

```

At the end of the learning phase, parameter estimation is run to assign the weights for each meaning of words. NL2KR-L then uses those meanings to check if they are enough to translate the training sentences correctly.

```

*****Evaluation on training set ...

Processing training sentence: Some boxer walks
Predicted Result: EX.boxer(X) ^ walk(X)
5 Correct Prediction

Processing training sentence: John takes a plane
Predicted Result: EX.plane(X) ^ takes(john,X)
Correct Prediction

```

10

...

Following is the output lexicon learned with the example inputs mentioned earlier. Each row contains a word, its CCG category, its meaning and the associated weight. Compared to the initial dictionary, we can see that NL2KR-L learned 14 more word meanings. Note that some words such as *likes* and *eats* are in the “syntax.txt”.

```

Some [(S/(S\NP))/NP] #x2.#x1.EX.x2 @ X ^ x1 @ X 0.0074364278
fight [S\NP] #x.fight(x) 0.01
boxer [N] #x.boxer(x) 0.060000002
boxer [N] boxer -0.041248113
5 a [NP/N] #x4.#x2.EX.x4 @ X ^ x2 @ X 0.009887816
John [NP] john 0.0073314905
John [N] john 0.01
eats [(S\NP)/NP] #w.#z.w @ #x.eats(z,x) 0.01
10 fights [S\NP] #x.fights(x) 0.01
fights [S\NP] #x.fight(x) 0.01
takes [(S\NP)/NP] #w.#z.w @ #x.takes(z,x) 0.01
walks [S\NP] #x.walks(x) -0.08722576
walks [S\NP] #x.walk(x) 0.10615976
plane [N] #x.plane(x) 0.059950046
15 plane [N] plane -0.03995005
likes [(S\NP)/NP] #w.#z.w @ #x.likes(z,x) 0.01
flies [S\NP] #x.fly(x) 0.01
flies [S\NP] #x.flies(x) 0.01
loves [(S\NP)/NP] #w.#z.w @ #x.loves(z,x) 0.01

```

3.10 NL2KR-L in the Geoquery domain

In this subsection, we present an example of using NL2KR-L for the GEOQUERY⁴ domain. GEOQUERY uses a Prolog based language to query a database with geographical information about the U.S. The input of NL2KR-L is specified in the RunConfiguration file as:

```

Ldata=./examples/geoquery/train.txt
Ldictionary=./examples/geoquery/dictionary.txt
Lsyntax=./examples/geoquery/syntax.txt
Loutput=

```

where ./examples/geoquery/train.txt contains

```

How large is texas answer(X) ^ size(B,X) ^ const(B,sid,texas)
How high is mountmckinley answer(X) ^ elevation(B, X) ^ const(B,
pid,mountmckinley)
How big is massachusetts answer(X) ^ size(B, X) ^ const(B,
sid,massachusetts)
How long is riogrande answer(X) ^ len(B, X) ^ const(B, rid,riogrande)
5 How tall is mountmckinley answer(X) ^ elevation(B, X) ^ const(B,
pid,mountmckinley)

```

./examples/geoquery/dictionary.txt contains

```

How S/S #x.answer(X) ^ x@X
texas NP #x.const(x,sid,texas)
mountmckinley NP #x.const(x,pid,mountmckinley)
massachusetts NP #x.const(x,sid,massachusetts)
5 riogrande NP #x.const(x,rid,riogrande)
is (S\NP)/NP #y.#x.x @ y

```

and ./examples/geoquery/syntax.txt contains

⁴ <http://www.cs.utexas.edu/users/ml/geo.html>

```

How S/S
texas NP
mountmckinley NP
massachusetts NP
5 riogrande NP
is (S\NP)/NP
rivers NP
large NP
is (S\NP)/NP
10 high NP
big NP
long NP
the NP/NP
How S/(S\NP)
15 long S\NP
tall NP
colorado NP
arizona NP

```

After the learning module is executed, 15 more word meanings were learned by NL2KR-L and the result is:

```

is [(S\NP)/NP] #y.#x.x @ y -0.0015125279
texas [NP] #x.const(x,sid,texas) 0.07666666
texas [NP] #x.const(x,rid,texas) -0.023256822
texas [NP] #x.const(x,pid,texas) -0.023256822
5 riogrande [NP] #x.const(x,pid,riogrande) -0.02315781
riogrande [NP] #x.const(x,rid,riogrande) 0.07646726
riogrande [NP] #x.const(x,sid,riogrande) -0.02315781
mountmckinley [NP] #x.const(x,sid,mountmckinley) -0.055226557
mountmckinley [NP] #x.const(x,pid,mountmckinley) 0.14075616
10 mountmckinley [NP] #x.const(x,rid,mountmckinley) -0.055226557
massachusetts [NP] #x.const(x,rid,massachusetts) -0.023190754
massachusetts [NP] #x.const(x,sid,massachusetts) 0.0765336
massachusetts [NP] #x.const(x,pid,massachusetts) -0.023190754
long [NP] #x3.#x1.len(B,x1) ^ x3 @ B 0.010111484
15 How [S/S] #x.answer(X) ^ x @ X 0.009999999
high [NP] #x3.#x1.elevation(B,x1) ^ x3 @ B 0.010112147
big [NP] #x3.#x1.size(B,x1) ^ x3 @ B 0.010111821
tall [NP] #x.const(x,rid,tall) -0.014923776
tall [NP] #x.const(x,pid,tall) -0.014923776
20 tall [NP] #x3.#x1.elevation(B,x1) ^ x3 @ B 0.084677815
tall [NP] #x.const(x,sid,tall) -0.014923776
large [NP] #x3.#x1.size(B,x1) ^ x3 @ B 0.010112498

```

3.11 NL2KR-T: the translation sub-part of NL2KR

Similar to NL2KR-L, in the RunConfiguration file of NL2KR-T, we need to set the lexicon file path, the override file for syntax categories(optional), and the testing data file as given below:

```

Tdata=./examples/sample/test.txt
Tdictionary=./output/dictionary_train.out
Tsyntax=./examples/sample/syntax.txt

```

For example, the preceding snippet specifies that: the testing data is in `./examples/sample/test.txt`, the lexicon is in `./output/dictionary_train.out`, and the override syntax categories are in `./examples/sample/syntax.txt`. The lexicon should be the lexicon learned by NL2KR-L.

The content of `./examples/sample/test.txt` is

```

Mia sleeps sleep(mia)
John catches a bus EX. (bus(X) ^ catches(john, X))

```

Running the NL2KR-T script with the inputs specified above, we have the following output where *John catches a bus* is translated to *EX. (bus(X) ^ catches(john, X))* as expected.

```

*****Parsing Sentences ...
...
Parsing test sentence: John catches a bus
Expected Representation: EX. (bus(X) ^ catches(john, X))
5 Generalizing bus = [bus : [N] : #x.bus(x), bus : [N] : bus]
Generalizing catches = [catches : [(S\NP)/NP] : #w.#z.w @ #x.catches(z,x)]
Predicted Result: EX.bus(X) ^ catches(john,X)
Correct Prediction
...

```

Note that the expected translation in “test.txt” is optional. Without it, the evaluation is not correct but NL2KR-T still gives its results.

4 Conclusion and Future Work

In this work, we presented the NL2KR system, which is used for translating natural language to a formal representation. The input of the NL2KR system are training sentences and their formal representation; and an initial lexicon of some known meanings of words. NL2KR system will try to learn the meaning of others words from the training data. We presented six scripts to execute several modules of NL2KR and show how to use them through examples.

In the future, we plan to make NL2KR more scalable and add more features to the NL2KR system such as (1) automatically constructing the initial lexicon and (2) using more knowledge such as word sense to select the correct meaning of words.

References

1. Montague, R.: English as a Formal Language. In Thomason, R.H., ed.: Formal Philosophy: Selected Papers of Richard Montague. Yale University Press, New Haven, London (1974) 188–222
2. Baral, C., Dzifcak, J., Gonzalez, M.A., Zhou, J.: Using Inverse lambda and Generalization to Translate English to Formal Languages. CoRR **abs/1108.3843** (2011)
3. Lierler, Y., Schüller, P.: Parsing Combinatory Categorical Grammar via Planning in Answer Set Programming. In Erdem, E., Lee, J., Lierler, Y., Pearce, D., eds.: Correct Reasoning. Volume 7265 of Lecture Notes in Computer Science., Springer (2012) 436–453
4. Toutanova, K., Klein, D., Manning, C.D., Singer, Y.: Feature-rich part-of-speech tagging with a cyclic dependency network. In: NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, Morristown, NJ, USA, Association for Computational Linguistics (2003) 173–180
5. Clark, S., Curran, J.R.: Parsing the WSJ using CCG and log-linear models. In: ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics, Morristown, NJ, USA, Association for Computational Linguistics (2004) 103

A Default Inference Rule Operating Internally to the Grammar Devices

Christophe Onambélé Manga

UMR SFL, CNRS/Université Paris 8, France
onambelemanga@yahoo.fr

Abstract. Minimalist Grammars (MG) are viewed as a resource consuming system where syntactic operations are triggered when a positive form of a feature matches with its negative form. But a problem arises when a feature lacks a positive/negative value. For the latter case, we introduce a default inference rule in order to account for the underspecification of the feature in a lexical entry.

Keywords: minimalist grammars, feature underspecification, default logic.

1 Introduction

In Bantu syntax, the computational system only needs a noun class formal feature to proceed analysis. Noun classes are sets of words that trigger the same agreement schema. Ewondo (Bantu, A72a)¹ has 14 noun classes². The choice

Table 1. Agreement Marker & Gender in Ewondo

Gender	Classes	Class morpheme	Agr marker on verb	Agr marker on adj
Gender I	1,2	̀̀- / bə-	á-(Á) / b́-	á-(Á) / b́-
Gender II	3,4	̀̀- / m̀̀-	ó- / ḿ-	ó- / ḿ-
Gender III	5,6	à- / m̀̀-	á- *ĺ- *d́- / ḿ-	á- *ĺ- *d́- / ḿ-
Gender IV	7,8	è- / b̀̀-	é- / b́-	é- / b́-
Gender V	9,10	n- / n-	é-(Á) / é-	é-(Á) / é-
Gender VI	9,2	n- / b̀̀-	é-(Á) / b́-	é-(Á) / b́-
Gender VII	9,6	n- / m̀̀-	é-(Á) / ḿ-	é-(Á) / ḿ-
Gender VIII	11,5	ò- / à-	ó- / á- *ĺ- *d́-	ó- / á- *ĺ- *d́-
Gender IX	11,6	ò- / m̀̀-	ó- / ḿ-	ó- / ḿ-
Gender X	19	v̀̀-	ó-	ó-

¹ [Gu1] alphanumeric coding (of bantu languages) is mainly geographic. Nevertheless, the distribution (of languages) is done in zones A, B, ... whether the language has kept a tone model closed to Proto-bantu.

² More information on classes pairing can be found in [On1]. Two locatives noun classes are to be added to Table 1, namely cl16 (v-, à-) and cl17 (ò-).

of a noun class prefix indicates whether the noun is viewed as a unit or a set of units. Except for locatives (cl16, cl17), even-numbered noun classes indicate **augmented** (AUG) and odd-numbered are for **minimal** (MIN)³. As it can be seen in Table 1, each class has a different class morpheme that triggers a different agreement morpheme feature; except for nouns of classes 9, 10 that share the same class feature.

- (1) 1. *mɔ́ngɔ́* *áku* *ámboo*
m-ɔ́ngɔ́ á-a-ku á-mboo
1MIN-child AGR1-past1-fall down AGR1-lay flat
‘the child fell down and laid flat’
2. *bɔ́ngɔ́* *bɛ́ku* *bɛ́mboo*
b-ɔ́ngɔ́ bɛ́-a-ku bɛ́-mboo
2AUG-child AGR2-past1-fall down AGR2-lay flat
‘the children fell down and laid flat’
- (2) 1. *ɲag* *yàdì* *bílòg*
ɲag yà-à-dì bí-lòg
9MIN.cow AGR9-Pres-eat 8AUG-grass
‘The cow grazes’
2. *ɲag* *yâdì* *bílòg*
ɲag yâ-à-dì bí-lòg
10AUG.cow AGR10-Pres-eat 8AUG-grass
‘The cows graze’

In (1), the agreement class feature of the head noun (*mɔ́ngɔ́*, *bɔ́ngɔ́*) spreads on the verbs. In (2) we have the same form of the noun for both the **minimal** and the **augmented**. In fact, when standing alone, one can’t tell whether *ɲag* is **minimal** (i.e class 9) or **augmented** (i.e class 10). It’s rather the agreement it triggers that helps to distinguish one form to another. As already mentioned, Bantu agreement phenomenon is characterized by the spreading of class feature of the head noun all over its dependents including the verb. Structure building rules (**merge**, **move**) in MG are defined in a directional process with a feature checking system that is a mechanism of resource consumption i.e each **selector** feature must match a **selectee** and each **licensor** match a **licensee**. [On1] proposed to formalize bantu multiple agreement in MG by **Head Movement with Copying**, the idea being that a **selector** is not **end-consumed** as the items that select it still exist in the derivations. The aim of this paper is to see how to deal with the balancing of ambiguity versus underspecification in the feature (2) in a resource consumption system like MG. Underspecification has been addressed in type-based grammars [Cr1,Dn1], in Type-Logical Grammars [He1], but never in MG. Here, we propose to associate a **defeasible inference rule** (σ) to lexical items with underspecified class feature. σ is based on **Prototypical**

³ Ewondo grammatical number has been redefined as Minimal (Min) and Augmented (Aug), thus we have one single feature $[\pm\text{aug}]$ [On1]

Reasoning [Re1,An1]. Section 2 proposes three ways that languages can have (or not have) noun classes. Section 3 presents the indeterminacy of class feature of nouns of class 9/10 in Bantu syntax. In Sect. 4 we show how the building of syntactic operations works in MG, Sect. 5 provides a new solution that could help to account for underspecification in MG after showing the limits of the first proposal made in [On1]. The paper ends with a conclusion.

2 Inherent vs Flexible Gender Features

Given examples (3, 4, 5) that show agreement phenomenon encountered in French, English and Ewondo. Imagine one removes **maisons** (**houses**) from (3a), then if a French speaker is asked to give the masculine form of the adjective **belles** (**beautiful_{FEM,PL}**), he would say **beaux** (**beautiful_{MASC,PL}**) because gender is inherent in adjective in French. On the other hand in English (4), the adjective stays unchanged, gender (or number) feature is not inherent in adjective.

- (3) 1. *toutes ces belles maisons*
all_{FEM,PL} this_{FEM,PL} beautiful_{FEM,PL} house_{FEM,PL}
'All these beautiful houses'
2. *tous ces trois jours*
all_{MASC,PL} this_{MASC,PL} three_{MASC,PL} day_{MASC,PL}
'All these three days'
- (4) 1. *all the beautiful houses*
all_∅ the_∅ beautiful_∅ house_{PL}
2. *the desperate housewives*
the_∅ desperate_∅ housewife_{PL}

For a Ewondo⁴ speaker, if he is asked to give the gender class of a determinative⁵, he will be unable to give one. He needs to know the syntactic context in which this determinative appears to tell what its class marker is.

- (5) 1. *mə-mōs mə-tē mə-sə mə-lá*
6AUG-day AGR6-this AGR6-all AGR6-three
'All these three days'

⁴ Unless specified, all the examples that aren't French or English are from Ewondo language

⁵ The class marker allows to distinguish between substantives and determinatives. Substantives are the set of nouns that [Gr1, p. 7] called **inherent gender** because this category triggers agreement. The second one he called **derived gender** is made of words that agree with the first one. In Ewondo (as in most Bantu languages), there are two nominal categories that share the fact to have the same nominal prefix. We term this second one as "determinative"

2. *bi-soá* *bi-t̄* *bi-sə* *bi-lá*
 8AUG-plate AGR8-this AGR8-all AGR8-three
 ‘All these three plates’

The following observations⁶ can be made: (i) adjective in French is an unmarked form that potentially agrees with the noun; (ii) in Ewondo, we can’t indicate the class marker of a determinative except it appears in a construction, that means we need the presence of a substantive that bears a specified noun class marker to tell what are the class markers of the others items. Determinatives don’t have pre-specified class marker, they inherit the class marker of the head noun; (iii) adjective in English is invariable. French and Ewondo speakers differ in whether they are able to produce a particular inflected form of an adjective in isolation. This is an experimental finding, and can be explained in many ways. One possible explanation is simply that speakers of any gendered language, when faced with such a task, think of an appropriate context and report the form the adjective takes in that context. The different behaviour of the French and Ewondo speakers is a result of there being only two genders in French, and thus that it is much easier to think of an appropriate context.

3 The Problem

3.1 Ambiguity in the Feature

In Ewondo, nouns of classes 9, 10 are problematic if one wants to determine their respective noun class. In (6), the DPs subjects aren’t different as can be found (*chicken vs chickens*) in English. It’s rather the agreement class marker the noun triggers (AGR9 *ȳə* and AGR10 *ȳə̄*) that differentiates *kúb* in (6a, b) [Ow1, p. 65] is 9MIN and 10AUG respectively.

- (6) 1. *kúb* *ȳək̄n*.
 kúb *ȳə̄-à-k̄n*
 9MIN.chicken AGR9-Pres-be sick
 ‘The chicken is sick’
 2. *kúb* *ȳək̄n*.
 kúb *ȳə̄-à-k̄n*
 10AUG.chicken AGR10-Pres-be sick
 ‘The chickens are sick’

As in most Bantu languages, it’s assumed their nominal class morphemes are originally homophones *n-* (see Table 1). It’s also difficult to say whether a given noun has a root /NCVC(V)/ or /CVC(V)/ with a class morpheme *n-*. Linguists usually argue by analogy to others noun classes: if most nouns of classes 9, 10 begin with a nasal⁷, and if there are less nouns in others classes with that

⁶ My thanks to Greg Kobele for valuable comments after my aviva.

⁷ and there is a high percentage of initials [nD] and [nT] (where [D] is a voiced occlusives and [T] is a non voiced occlusives).

structure, then people assume that roots can't generally begin with NC; therefore nouns in classes 9, 10 that always have a NC initial must actually have a prefix /n-/. An answer to this argument is the possibility to mark a contrast between roots with NC and C initials in noun classes 9, 10. In Ewondo, we have nouns with [nD] but also words with [T] and [z]; such thematic roots are ambiguous. Two explanations are possible : (i) there is a phonological deletion of /n/ in front of voiceless consonants and fricatives, and (ii) there is a real contrast between NC and C initials in these noun classes. In Ewondo, the prefix n- is deleted when it's followed by another nasal (7), an unvoiced consonant (8) or by a voiced consonant /z/ (9):

- (7) n+ɲàk → ɲag: cow (8) n+tsit → tsid: animal (9) n+zək → zæg: pineapple

In short, nouns of classes 9, 10 are morphologically invariable and neutral for class distinction. One may think there is no difference between **minimal** and **augmented** number.

3.2 Distinction between Class 9, 10 Nouns

As noted, for those nouns that don't change in **minimal/augmented** form, the distinction is made by the agreement they trigger (10, 11).

- (10) 1. *kúb* *é-nḁ* *o-nḁn*
 9MIN.chicken AGR9-PRES.be 11MIN-bird
 'The chicken is a bird'
2. *kúb* *é-nḁ* *a-nḁn*
 10AUG.chicken AGR10-PRES.be 5AUG-bird
 'Chickens are birds'
3. **kúb* *é-nḁ* *a-nḁn*
 10AUG.chicken AGR10-PRES.be 5AUG-bird
 'Chicken are birds'
- (11) 1. *zæg* *é-bədḁ* *á tḁbəlḁ*
 9MIN.pineapple AGR9-PRES.put down on 1MIN.table
 'The pineapple is on the table'
2. *zæg* *é-bədḁ* *á tḁbəlḁ*
 10AUG.pineapple AGR10-PRES.put down on 1MIN.table
 'Pineapples are on the table'

In (10, 11), tone doesn't help to distinguish the two noun classes. Originally, the **augmented** form is obtained by adjoining a suprasegmental High tone |´| to the noun of class 9. This floating High tone⁸ attaches either to the noun or

⁸ Regarding the architecture of tonal representations, floating tones (not associated) are usually represented in a circle : spreading high tone , spreading low tone .

- (17) \mathfrak{H} \mathfrak{H} The floating Low tone blocks the High tone of *kúb* (chicken) so that it can't spread up to the verb root.

And the Low tone goes on this verb root, as the latter already bears a Low tone, nothing changes. We can conclude that tonal distinction on the agreement feature can be useful to distinguish the *covert* class feature of nouns of class 9/10.

4 Minimalist Grammars

MG [St1] attempt to implement the so-called *minimalist* principles introduced by [Ch1]. A MG is a quadruplet $(V, \text{Cat}, \text{Lex}, F)$: $V = \{P \cup I\}$, set of non syntactic features (*vocabulary*) where P represents the phonetic features and I the semantics features; $\text{Cat} = \{\text{base} \cup \text{selector} \cup \text{licensor} \cup \text{licensee}\}$, finite set of non syntactic features (*categories*) which are partitioned into four kinds ($x : \text{base}$ (c, t, v, d, n, \dots), $=x : \text{selector/probe}$, $-x : \text{licensee}$, $+x : \text{licensor}$ (feature that trigger move)); $\text{Lex} =$ finite set of expressions built from V and Cat (*lexicon*); $F = \{\text{merge} \cup \text{move}\}$: set of generating functions. *Merge* and *Move* are built with trees where: (i) internal nodes are labelled with direction arrows ($<$ or $>$) indicating where the head of the structure is, (ii) leaves are pairs $\langle \alpha, \beta \rangle$ with $\alpha =$ vocabulary item and $\beta =$ set of features. *Merge* (or external merge) is a binary operation that takes two trees and puts them together. The tree whose first feature is $=x$ merges with a tree whose category feature is x to built a new tree. Features $=x$ and x are deleted after merging.

$$(18) \text{merge} (t_1^{-x}, t_2^x) = \begin{array}{l} < & \text{if } t_1 \text{ is a lexical item} \\ & \widehat{t_1 t_2} \\ \text{merge} (t_1^{-x}, t_2^x) = & > & \text{if } t_1 \text{ is not a lexical item} \\ & \widehat{t_1 t_2} \end{array}$$

Move (or internal merge) is a unary operation that targets (some part of) an expression to remerge it higher in the structure. *Move* is applied to a subtree with a feature $-x$. Given a subtree with $-x$ written t_2^{-x} that appears in a tree t_1^{+X} , we write $t_1[t_2^{-x}]^{+X}$. t_1^{+X} is the maximal projection of $t_1[t_2^{-x}]^{+X}$ i.e the largest subtree with $-x$ as its head. After extraction, the subtree t_2^{-x} merges as specifier of the head of the tree, features served for *Move* operation are removed from the tree. The *shortest move constraint* (SMC) that applies to *Move* requires there should be exactly one maximal projection $t_1[t_2^{-x}]^{+X}$ displaying a subtree t_2^{-x} . The original place of t_2^{-x} is then filled by an empty tree ϵ i.e a single featureless node.

$$(19) \text{move} (t_1[t_2^{-x}]^{+X}) = \begin{array}{l} > \\ \widehat{t_2 t_1[\epsilon]} \end{array}$$

There are few syntactic operations implemented in MG (Scrambling & Adjunction [FG1], Head Movement [St2], Copying [Ko1], Head Movement with Copying [On1]). In MG, **Merge** and **Move**, need a selecting feature matching a selected feature (both being of the same category) to drive derivations. Now, what's happened if the selecting feature is un(der)specified?

5 On Underspecification in Minimalist Grammars

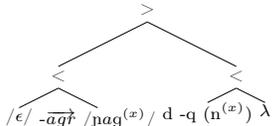
5.1 Unspecified Class Feature

Given the examples below where nouns of classes 9, 10 are in subject position (20) and in object position (21), the analysis developed in [On1] for (20) is based on the theoretical claim that nouns of classes 9, 10 are not lexically specified for their class.

(20) *ɲag yàdì bílɔg*
 ɲag yà-à-dì bí-lòg
 9MIN.cow AGR9-Pres-eat 8AUG-grass
 'The cow grazes'

(21) *ńsɔmɔ áwé ɲag*
 ń-sɔmɔ á-a-wé ɲag
 1MIN-huntsman AGR1-PAST1-kill 9MIN/10AUG.cow
 'The huntsman has killed the cow/cows.'

These nouns enter the derivation with an uninstantiated variable x that will be valued through postsyntactic insertion of the class morpheme of the agreement feature $+\overrightarrow{agf}$ on TP. Variable x instantiation means to copy on the subject DP the value of the agreement feature on T head.

(22)  (22) is built in 3 steps:
 (a) merge($n^{(x)} \leftarrow !c1 -k /ε/, n^{(x)} -\overrightarrow{agf} /ɲag/$),
 (b) merge($=>^+c1 +k d -q /ε/, a$),
 (c) move(b).

Postsyntactic insertion means the agreement class feature on the verb is substituting for the variable x yielding the corresponding noun class feature on the noun. As we said it's the agreement feature on the verb that give the information about the nominal class morpheme of the DP. The substitution process is made in two steps : (i) **covert movement** then (ii) **agree** (for detailed step-by-step justification see [On1]). If (20) is appropriately treated in [On1], the solution provided is still problematic for (21) where noun of classes 9, 10 are in object position. To solve this problem, let's try another approach. Following [Ro1], we distinguish three features: ϕ -features are specified class feature for inherent noun classes; θ -features are underspecified class feature for noun of

Table 2. Syntactic Class Features

Inherent noun classes		Neutral noun classes		Derived noun
cl1	cl2	cl9	cl10	flexible
cl3	cl4			
cl5	cl6			
cl7	cl8			
cl11				
cl16				
cl17				
cl19				

class 9/10; α -features are flexible and inherited class agreement feature found on derived nouns (i.e determinatives) and verbs. If we think of noun class feature as Attribute-Value feature system, we could say, noun of class 9, 10 has an Attribute specification "n" without a Value (i.e without a class number). That means, a word like `ᵛᵃḡ(cow)` is represented with the feature `nθ`. The difference being that a noun with a specified noun class (say `m-ᵛᵃḡᵛ: 1MIN-child`) will be represented with a specified class feature `n1`. α -features' transmission is done through HMC. The question now is how to formalize θ -feature in MG?

5.2 Default Inference Rule

A default rule will be used to model feature underspecification through prototypical reasoning, the latter is used when most instances of a concept have some property¹⁰. Default Logic [Re1,An1] is a nonmonotonic reasoning approach allowing to rely on incomplete information about problem. A default theory **T** is a pair (F, I) where F is a set of FOL sentences representing the background information, I represents the defeasible information (i.e a countable set of defaults rule).

Definition 1. A default rule (say σ) is an inference rule of the form:

$$\frac{\delta : \rho_1, \dots, \rho_n}{\xi} \begin{cases} \delta & = \text{prerequisite, } pre(\sigma) \\ \rho_1, \dots, \rho_n & = \text{justifications, } just(\sigma) \text{ or } simply(\sigma) \\ \xi & = \text{consequent of } \sigma, cons(\sigma). \end{cases} \quad (23)$$

interpreted as: given δ and as there is no information that $\neg\rho_i$, conclude ξ by default. A default rule is called **normal** if and only if it has the form:

$$\frac{\delta : \xi}{\xi} \quad (24)$$

¹⁰ That means for us the case when most instances of noun class feature have Attribute-Value property.

A semantics for Default Logic is provided through the notion of **extension** [Re1,AS1,An1]. An extension for a default theory \mathbf{T} ($\mathbf{T} = (F, \Gamma)$) is a set of FOL sentences \mathbf{E} where: (a) $F \subseteq \mathbf{E}$; (b) $\mathbf{E} = \Delta(\mathbf{E})$ where Δ denotes the deductive closure; (c) \mathbf{E} should be closed under the application of defaults from Γ i.e if $\frac{\delta: \rho_1, \dots, \rho_n}{\xi}$, $\delta \in \mathbf{E}$ and $\neg \rho_1 \notin \mathbf{E}, \dots, \neg \rho_n \notin \mathbf{E}$ then $\xi \in \mathbf{E}$.

Definition 2. For $\mathbf{T} = (F, \Gamma)$, let $\Pi = (\sigma_0, \sigma_1, \dots)$ be a finite or infinite sequence of default rules from Γ without multiple occurrences. Π is viewed as possible order in which default rules from Γ are applied, so a default rule doesn't need to be applied more than once in such a reasoning. The initial segment of Π with length k is denoted $\Pi[k]$. Sets of first-order formulae, $\mathbf{In}(\Pi)$ and $\mathbf{Out}(\Pi)$ are associated to such sequence as Π : (a) $\mathbf{In}(\Pi) = \Delta(F \cup \{\text{cons}(\sigma) \mid \sigma \text{ occurs in } \Pi\})$, $\mathbf{In}(\Pi)$ collects the information gained by the application of the default in Π and represents the **current knowledge base** after the default in Π have been applied; (b) $\mathbf{Out}(\Pi) = \{\neg \rho \mid \rho \in \text{just}(\sigma) \text{ for some } \sigma \text{ occurring in } \Pi\}$, $\mathbf{Out}(\Pi)$ collects formulae that should not turn out to be true i.e that should not become part of the current knowledge base even after subsequent application of the other default rules.

Definition 3. Π is called a **process** of \mathbf{T} iff σ_k is applicable to $\mathbf{In}(\Pi[k])$, for every k such that σ_k occurs in Π .

Definition 4. For a given process Π of \mathbf{T} : (a) Π is a **successful process** iff $\mathbf{In}(\Pi) \cap \mathbf{Out}(\Pi) = \emptyset$, otherwise it is a **failed process**; (b) Π is a **closed process** iff every $\sigma \in \Gamma$ that is applicable to $\mathbf{In}(\Pi)$ already occurs in Π . Closed processes correspond to the desired property of an extension \mathbf{E} being closed under application of default rules from Γ .

Definition 5. For the application of a default rule, a **consistency condition** should be satisfied.

$$\sigma = \frac{\delta: \rho_1, \dots, \rho_n}{\xi} \quad (25)$$

is **applicable** to a deductively closed set of formulae \mathbf{E} iff $\sigma \in \mathbf{E}$ and $\neg \rho_1 \notin \mathbf{E}, \dots, \neg \rho_n \notin \mathbf{E}$.

Proposition 1. The rule of thumb when treating nouns of class 9, 10 is to say these nouns are of class 10 unless stated by the grammarian they are of class 9¹¹. If the latter is done, the default rule (i.e the rule of thumb) already mentioned isn't rejected, it's simply no more applicable as the missing information is now known. In a classical logic setting¹², we need to say what is the Value of Attribute n for classes 9, 10 nouns. As we don't know, no decision could be taken in such a system. But in default reasoning, the previous rule of thumb can be applied.

¹¹ That means, only the **augmented** is grammatically marked, **minimal** will have a zero grammatical marker that is realised as a low tone by default. In fact, that's the strategy generally used in natural language that the **minimal** has a zero morpheme.

¹² As well as in Stablerian MG

Proof. We write Attribute with indices to differentiate between \mathbf{n}_1 (that stands for class 10 nouns) and \mathbf{n}_0 (that stands for class 9 nouns). So, for $\mathbf{T} = (F, \Gamma)$, let $F = \{n_1, n_0\}$ and $\Gamma = \{\frac{n_1:-9}{10}, \frac{n_0:-10}{9}\}$. The default theory \mathbf{T} is represented as : $\mathbf{T} = (\{n_1, n_0\}, \{\frac{n_1:-9}{10}, \frac{n_0:-10}{9}\})$. Let also assume that $\sigma_1 = \frac{n_1:-9}{10}$, $\sigma_0 = \frac{n_0:-10}{9}$ and $\Pi = (\sigma_1, \sigma_0)$. As Γ contains only (a finite number of) two default rules, closedness doesn't matter. We apply default rules as long as they are **applicable**, and then we get a closed process. So, we apply the first default σ_1 and check default σ_0 with respect to the knowledge collected after the application of σ_1 . For $\Pi[\sigma_1]$ we have: $\text{In}(\Pi[\sigma_1]) = \Delta(\{n_1, n_0, 10\})$, $\text{Out}(\Pi[\sigma_1]) = \{9\}$, $\text{In}(\Pi[\sigma_1]) \cap \text{Out}(\Pi[\sigma_1]) = \emptyset$, so, we say $\Pi[\sigma_1]$ is closed and successful process. For $\Pi[\sigma_0]$ we have: $\text{In}(\Pi[\sigma_0]) = \Delta(\{n_1, n_0, 9\})$, $\text{Out}(\Pi[\sigma_0]) = \{10\}$. In fact σ_0 can't be applied as $10 \in \Delta(\{n_1, n_0, 10\})$ which is our current knowledge base before we apply σ_0 . We know $\text{In}(\Pi[k+1]) = \Delta(\Pi[k]) \cup \Delta(\Pi[k+1])$ and $\text{Out}(\Pi[k+1]) = \text{Out}(\Pi[k]) \cup \text{Out}(\Pi[k+1])$ so $\text{In}(\Pi[\sigma_0]) = \Delta(\{n_1, n_0, 10, 9\})$, $\text{Out}(\Pi[\sigma_0]) = \{10, 9\}$, $\text{In}(\Pi[\sigma_0]) \cap \text{Out}(\Pi[\sigma_0]) = \{10, 9\}$, thus, we say $\Pi[\sigma_0]$ is failed process. We could have stopped the proof earlier as application of σ_1 blocks application of σ_0 and vice versa, so there are no more extension of \mathbf{T} . From the application of the first default rule σ_1 , we know Attribute \mathbf{n} has Value 10, so it is not consistent to assume 9. Thus $\Delta(\{n_1, n_0, 10\})$ is the only extension of \mathbf{T} . \square

We associate a defeasible inference rule σ to lexical items with feature ambiguity. A default rule on an underspecified Attribute \mathbf{n} is marked using an arrow \uparrow^σ indicating to map Attribute \mathbf{n} to the result of σ . Once the Value of \mathbf{n} is calculated, then the MG derivation can proceed (and not the inverse). The idea being that the default rule blocks the derivation. So the derivation tree for a word like ɲag (cow) is:

$$(26) \quad \begin{array}{c} < \\ \swarrow \quad \searrow \\ [n \uparrow^\sigma] \leftarrow \text{!cl} \text{-k} / \epsilon / \text{nx} / \text{ɲag} / \end{array} \quad \text{where } \mathbf{x} \text{ is an anonymous variable that match with any value collected after the application of } [n \uparrow^\sigma].$$

MG are by definition encapsulated, which means that they make reference only to their own internalised system, and not to any external formal system, such as a logic for general reasoning. This is intrinsic to the claim of the language faculty being prior, feeding into more general reasoning devices but separate from them. If the current proposal is in line with Stabler's formalization, then we think MG clearly differentiate a Stabler form of minimalism with others. That might mean that the notion of encapsulation may be rather different for a Stabler form of grammar than others.

6 Conclusion

In this paper we attempt to introduce a rule in a Stablerian MG that could help to account for feature underspecification in a resource consuming system.

The proposal rests on default reasoning that allows to deal with incomplete information about a problem.

Acknowledgments

We thank Stan Dubinsky, Ruth Kempson and two anonymous reviewers for their constructive comments, which helped us to improve the manuscript.

References

- [AS1] Antoniou, G., Sperschneider, V.: Operational concepts of nonmonotonic logics. Part 1. Default logic. *Artificial Intelligence Review* **8** (1994) 3–16
- [An1] Antoniou, G.: A tutorial on default logics. *ACM Computing Surveys*. Vol.31. No **3** (1999) 337–359
- [Ch1] Chomsky, N.: *The Minimalist Program*. Cambridge The MIT Press (1995)
- [Cr1] Crysmann, B.: Underspecification and neutrality: a unified approach to syntcretism. In *Proceedings of the Joint Conference on Formal Grammar and Mathematics of Language*. CSLI publications (2009) 1–12
- [Dn1] Daniels, M.: On a type-based analysis of feature neutrality and the coordination of unlikes. In *Proceedings of the 8th International Conference on Head-Driven Phrase Structure Grammar*. CSLI Publications (2001) 137–147
- [FG1] Frey, W., Gärtner, H.-M.: On the treatment of scrambling and adjunction in minimalist grammars. In *Proceedings of the Conference on Formal Grammar* (2002) 41–52
- [Gr1] Gregersen, E., A.: Prefix and pronoun in Bantu. *International journal of American linguistics*. Vol. 33. No **3**. Part II. Indiana University Press (1967)
- [Gu1] Guthrie, M.: *The classification of the Bantu languages*. Oxford University Press for the International African Institute (1948)
- [He1] Heylen, D.: Underspecification in Type-Logical Grammars. In *Logical Aspects of Computational Linguistics*. LNCS **1582** (1999) 180–199
- [Ko1] Kobele, G.: *Generating Copies: An Investigation into Structural Identity in Language and Grammar*. Ph.D. thesis. UCLA (2006)
- [On1] Onambélé, C.: *Vers une grammaire minimaliste de certains aspects syntaxiques de la langue Ewondo*. Ph.D. thesis. Université Paris 8 (2012)
- [Ow1] Owona, A.: *L’orthographe harmonisée de l’ewondo*. Magister Thesis. Université de Yaoundé 1 (2004)
- [Re1] Reiter, R.: A logic for default reasoning. *Artificial Intelligence* **13**. (1980) 81–132
- [Ro1] Rooryck, J.: On two types of underspecification: Towards a feature theory shared by syntax and phonology. *Probus* **6** (1994) 207–233
- [St1] Stabler, E.: Derivational minimalism. In *Logical Aspects of Computational Linguistics*. LNCS **1328** (1997) 68–95
- [St2] Stabler, E.: Recognizing Head Movement. In *Logical Aspects of Computational Linguistics*. LNAI-LNCS **2099** (2001) 245–260

BOEMIE: Reasoning-based Information Extraction

Georgios Petasis¹, Ralf Möller², and Vangelis Karkaletsis¹

¹ Software and Knowledge Engineering Laboratory
Institute of Informatics and Telecommunications
National Centre for Scientific Research (N.C.S.R.) “Demokritos”
GR-153 10, P.O. BOX 60228, Aghia Paraskevi, Athens, Greece
`{petasis,vangelis}@iit.demokritos.gr`

² Institute for Software Systems (STS)
Hamburg University of Technology
Schwarzenbergstr. 95, 21073, Hamburg, Germany
`moeller@tu-harburg.de`

Abstract. This paper presents a novel approach for exploiting an ontology in an ontology-based information extraction system, which substitutes part of the extraction process with reasoning, guided by a set of automatically acquired rules.

1 Introduction

Information extraction (IE) is the task of automatically extracting structured information from unstructured documents, mainly natural language texts. Due to the ambiguity of the term “structured information”, information extraction covers a broad range of research, from simple data extraction from Web pages using patterns and regular grammars, to the semantic analysis of language for extracting meaning, such as the research areas of word sense disambiguation or sentiment analysis. The basic idea behind information extraction (the concentration of important information from a document into a structured format, mainly in the form of a table) is fairly old, with early approaches appearing in the 1950s, where the applicability of information extraction was proposed by the Zellig Harris for sub-languages, with the first practical systems appearing at the end of the 1970s, such as Roger Schank’s systems [27, 28], which exported “scripts” from newspaper articles. The ease of evaluation of information extraction systems in comparison to other natural language processing technologies such as machine translation or summarisation, where evaluation is still an open research issue, made IE systems quite popular and led to the Message Understanding Conferences (MUC) [21] that redefined this research field.

Ontology-Based Information Extraction (OBIE) has recently emerged as a subfield of information extraction. This synergy between IE and *ontologies* aims at alleviating some of the shortcomings of traditional IE systems, such as efficient representation of domain knowledge, portability into new thematic domains, and

interoperability in the era of Semantic Web [14]. Ontologies are a means for sharing and re-using knowledge, a container for capturing semantic information of a particular domain. A widely accepted definition of ontology in information technology and AI community is that of “a formal, explicit specification of a shared conceptualization” [29, 10], where “formal implies that the ontology should be machine-readable and shared that it is accepted by a group or community” [4]. According to [31], an ontology-based information extraction system is a system that “processes unstructured or semi-structured natural language text through a mechanism guided by ontologies to extract certain types of information and presents the output using ontologies”. This definition suggests that the main differences between traditional IE systems and OBIEs are: a) OBIEs present their output using ontologies, and b) OBIEs use an information extraction process that is “guided” by an ontology. In all OBIE systems the extraction process is *guided* or *driven* by the ontology to extract things such as classes, properties and instances [31], in a process known as *ontology population* [23].

However, the way the extraction process is guided by an ontology in all OBIEs has not changed much with respect to traditional information extraction systems. According to a fairly recent survey [31], OBIEs do not employ new extraction methods, but they rather employ existing methods to identify the components of an ontology. Current research on the field investigates the development of “reusable extraction components” that are tied to ontology portions that are able to identify and populate [30, 11]. In this paper we propose an alternative approach that tries to *minimise* the use of traditional information extraction components, and substitute their effect with *reasoning*. The motivation behind the work presented in this paper is to propose a new “kind” of ontology-based information extraction system, which integrates further ontologies and traditional information extraction approaches, through the use of *reasoning* for “guiding” the extraction process, instead of heuristics, rules, or machine learning. The proposed approach splits a traditional OBIE in two parts, the first part of which deals with the gathering of evidence from documents (in the form of ontology property instances and relation instances among them), while the second part employs reasoning to interpret the extracted evidence, driven by plausible explanations for the observed relations. Thus, the innovative aspects of the presented approach include a) the use of an ontology through reasoning as a substitute for the embedded knowledge usually found in the extraction components of OBIEs, b) a proposal of how reasoning can be applied for extracting information from documents, and c) an approach for inferring the required interpretation rules even when the ontology evolves with the addition of new concepts and relations.

The rest of this paper is organised as follows: In section 2 related work is presented in order to place our approach within the current state-of-the-art. In section 3 the proposed approach is presented, detailing both the interpretation process and the automatic reasoning rule acquisition. Finally, section 4 concludes this paper and outlines interesting directions for further research.

2 Related Work

Ontology-based information extraction has recently emerged as a subfield of information extraction that tries to bring together traditional information extraction and ontologies, which provide formal and explicit specifications of conceptualizations, and acquire a crucial role in the information extraction process. A set of recent surveys have been presented that analyse the state-of-art in the research fields of OBIEs [13, 14, 31] and ontology learning/evolution [24, 23], a relevant research field since many OBIE systems also perform ontology evolution/learning. OBIE systems can be classified according to the way they acquire the ontology to be used for information extraction. One approach is to consider the ontology as an input to the system: The OBIE is guided by a manually constructed ontology or from an “off-the-shelf” ontology. Most OBIE systems appear to adopt this approach [31]. Such systems include SOBA [5, 3], KIM [25, 26] the implementation by Li and Bontcheva [18] and PANKOW [7], Artequact [15, 2, 1]. The other approach is to construct an ontology as a part of the information extraction process, either starting from scratch or by evolving an initial, seed ontology. Such systems include Text-To-Onto [19], the implementation by Hwang [12], Kylin [32], the work by Maedche et al. [20], the work of Dung and Kameyama [8]. However, all the aforementioned systems employ traditional information extraction methods to identify elements of the ontology, and none attempts to employ reasoning, as the work presented in this paper suggests.

3 The BOEMIE approach

The work presented in this paper has been developed in the context of the BOEMIE project. It advocates an ontology-driven multimedia content analysis, i.e. semantics extraction from images, video, text, audio/speech, through a novel synergistic method that combines multimedia extraction and ontology evolution in a bootstrapping fashion. This method involves, on one hand, the continuous extraction of knowledge from multimedia content sources in order to populate and enrich the ontologies and, on the other hand, the deployment of these ontologies to enhance the robustness of the multimedia information extraction system. More details about BOEMIE can be found in [6, 23].

As already mentioned, the proposed approach splits a traditional OBIE in two parts, the first part of which deals with the gathering of evidence from documents (in the form of ontology property instances and relation instances among them), while the second part employs reasoning to interpret the extracted evidence, driven by plausible explanations for the observed relations. As a result, the typical extraction process is also split in two phases: “low-level analysis” (where traditional extraction techniques such as machine learning are used) and “semantic interpretation”, where analysis’ results are explained, according to the ontology, through reasoning. Each of the two phases identifies different elements of the ontology, whose elements are also split in two groups, the “mid-level concepts” (MLCs - identified by low-level analysis), and the “high-level concepts” (HLCs), which are identified through semantic interpretation.

The implications of this separation are significant: the low-level analysis cannot assume that a Person/Athlete/Journalist has been found in a multimedia document, just because a name has been identified. Instead the low-level analysis reports that a name, an age, a nationality, a performance, etc. has been found, and reports how all these are related through binary relations, extracted from modality-specific information (i.e. linguistic events for texts, spatial relations for images/videos, etc.). The identification of Person/Athlete/Journalist instances is done through reasoning, using the ontology and the reasoning (interpretation) rules, as low-level analysis cannot know how the Person or Athlete concepts are defined in the ontology (i.e. what their properties/axioms/restrictions are). In essence, BOEMIE proposes a novel approach for constructing an OBIE, by keeping the named-entity extraction phase from traditional IE systems, modifying relation extraction to reflect modality-specific relations at the ontological level, and implementing the remaining phases of traditional IE systems through reasoning. For example, low-level analysis of an image is responsible for reporting only that a few tenths of faces have been detected (i.e. the faces of athletes and the audience – represented as MLC instances), along with a human body (i.e. the body of an athlete – represented as an MLC instance), a pole, a mattress, two vertical bars, a horizontal bar, etc. (all these are instances of MLCs). After MLC instances have been identified, the low-level analysis is expected to identify relational information about these MLC instances. For example, the low-level analysis is expected to identify that a specific face is adjacent to a human body and both are adjacent to the pole and the horizontal bar. The low-level analysis is expected to report the extracted relational information through suitable binary relations between each pair of related MLC instances. On the other hand, the low-level is not expected to interpret its findings and hypothesise instances of HLC instances, such as the existence of athletes and their number. It is up to the second phase, the semantic interpretation, to identify how many athletes are involved (each one represented as instance of the “Athlete” HLC), and to interpret the scene shown in the image as an instance of the “Pole Vault” HLC concept, effectively explaining the image.

3.1 Definitions

The approach presented in this paper organises the ontology into four main ontological modules, the “low-level features”, the “mid-level concepts”, the “high-level concepts”, and the “interpretation rules”, which are employed through reasoning in order to provide one or more “interpretations” of a multimedia document.

Definition 1 (low-level features). *Low-level features are concepts related to the decomposition of a multimedia document (i.e. the description of an HTML page into text, images or other objects), and concepts that describe surface forms on single modality documents, such as segments in text and audio documents, polygons in image and video frames, etc.*

Definition 2 (Mid-Level Concept (MLC)). *Mid-level concepts are concepts that can be materialised (i.e. have surface forms) on documents of a single modality. Anything that can be extracted by an OBIE that has a surface form on a document, is an MLC.*

For example, the names of persons, locations, etc. in texts, the faces, bodies of persons in images and the sound events (i.e. applauses) in audio tracks are all MLC concepts. The BOEMIE OBIE extracts only instances of MLCs (*MLCIs*) and relations (i.e. spatial) among them.

Definition 3 (High-Level Concept (HLC)). *High-level concepts are compound concepts formed from MLCs. HLCs cannot be directly identified in a multimedia document, as they cannot be associated with a single surface form (i.e. segment).*

For example, the concept “Person” is an HLC, that groups several MLCs (properties), such as “PersonName”, “Age”, “Nationality”, “PersonFace”, “PersonBody”, etc. Instances of HLCs (*HLCIs*) in the BOEMIE OBIE are identified through reasoning over MLC instances (MLCIs) in the ontology, guided by a set of rules, in a process known as “interpretation”.

Definition 4 (interpretation). *Interpretation is the identification of one or more HLC instances (HLCIs) in a multimedia document.*

An OBIE can have identified several MLC instances (MLCIs) and relations between them in a multimedia document. If these MLC instances satisfy the axioms of the ontology and the interpretation rules are able to generate one or more HLC instances (HLCIs), then this multimedia document is considered as *interpreted* (or *explained*) by the ontology, with the HLC instances (HLCIs) constituting the *interpretation* of the document. If the same MLC instances (MLCIs) are involved in more than one HLC instances (HLCIs) of the same HLC, then the document is considered to have *multiple interpretations*, usually due to *ambiguity*.

3.2 Semantic Extraction

The extraction engine is responsible for extracting instances of concept descriptions that can be directly identified in corpora of different modalities. These concept descriptions are mid-level concepts (MLCs). For example, in the textual modality the name or the age of a person is an MLC, as instances of these concepts are associated directly with relevant text segments. On the other hand, the concept person is not an MLC, as it is a “compound”, or “aggregate” concept in such a way that instances of this concept are related to instances of name, age, gender or maybe instances of other compound concepts. Compound concepts are referred to as high-level concepts (HLCs), and instances of such concepts cannot be directly identified in a multimedia document, and thus associated with a content segment. Thus, such instances and also relationships between these instances have to be hypothesized. In particular, this engine implements a modular approach [13] that comprises the following three level of abstraction: 1. The

low-level analysis, which includes a set of modality-specific (image, text, video, audio) content analysis tools. 2. A modality-specific semantic interpretation engine. 3. A fusion engine, which combines interpretations from each modality³.

The first two levels implement ontology-driven, modality-specific information extraction, while the last one fuses the information obtained from the previous levels of analysis. The first level involves the identification of “primitive” concepts (MLCs), as well as instances of binary relations amongst them. The second level involves the semantic interpretation engine, responsible for hypothesizing instances of high-level concepts (HLCs) representing the interpretation of (parts of) a document. Semantic interpretation operates on the instances of MLCs and relations between them extracted by the information extraction engine. The goal of semantic interpretation is to explain why certain instances of MLCs are observed in certain relations according to the background knowledge (domain ontology and a set of interpretation rules) [9], by creating instances of high-level concepts and relating these instances. Semantic interpretation is performed through calls to a non-standard reasoning service, known as explanation derivation via abduction. The semantic interpretation is performed on the extracted information (MLC/relation instances) from a single modality in order to form modality-specific HLC instances. The fact that content analysis is separated from semantic interpretation, along with the fact that semantic interpretation is performed through reasoning using rules from the ontology, allows single-modality extraction to be adaptable to changes in the ontology.

Once a multimedia document has been decomposed into single-modality elements and each element has been analysed and semantically interpreted separately, the various interpretations must be fused into one or more alternative interpretations of the multimedia document as a whole. This process is performed at a third level, where the modality-specific HLC instances are fused in order to produce HLC instances that are not modality-specific, and contain information extracted from all involved modalities. Fusion is also formalized as explanation generation via abductive reasoning.

Example: the OBIE for the text modality The low-level analysis system implemented in the context of BOEMIE exploits the infrastructure offered by the Ellogon⁴ platform [22], and the Conditional Random Fields [17] machine learning algorithm, in order to build an adaptable named-entity recognition and classification (NERC) system, able to identify MLC instances (MLCIs) and relations between MLCIs. Both NERC and relation extraction components operate in a supervised manner, using MLC instances that populate the (seed or evolved) ontology as training material (whose surface forms are available through their low-level features). The fact that both components use the populated ontology as training source, allows them to adapt to ontology changes, and improve their extraction performance over time, as the ontology evolves. The performance of

³ The fusion engine will not be described in this paper, as it is similar to the semantic interpretation engine. More information can be found at [6, 23].

⁴ <http://www.ellogon.org>

the NERC and relation extraction components has been measured to about 85% and 70% (F-measure), in the thematic domain of athletics, involving news items and biographies from official sites like IAAF⁵ (International Association of Athletics Federations). More details about the low-level analysis system for the text modality can be found in [13].

The modality specific interpretation engine (not only for text, but for all modalities) is a process for generating instances of HLCs, by combining instances of MLCs, through reasoning over instances. Abduction is used for this task, a type of reasoning where the goal is to derive explanations (causes) for observations (effects). In the framework of this work we regard as explanations the high-level semantics of a document, given the middle-level semantics, that is, we use the extracted MLCs in order to find HLCs [9]. The reasoning process is guided by a set of rules, which belong into two kinds, deductive and abductive. Assuming a knowledge base, $\Sigma = (T, A)$ (i.e. an ontology), and a set of assertions Γ , (i.e. the assertions of the semantic interpretation of a document), abduction tries to derive all sets of assertions (interpretations) Δ such as $\Sigma \cup \Delta \models \Gamma$, while the following conditions must be satisfied: (a) $\Sigma \cup \Delta$ is satisfiable, and (b) Δ is a minimal explanation for Γ , i.e. there exists no other explanation Δ' ($\Delta' \subseteq \Delta$) that $\Sigma \cup \Delta' \models \Gamma$ holds. For example, assuming the following ontology Σ (containing both a “terminological component” – TBox, and a set of rules):

$$\begin{aligned}
& \textit{Jumper} \sqsubseteq \textit{Human} \\
& \textit{Pole} \sqsubseteq \textit{SportsEquipment} \\
& \textit{Bar} \sqsubseteq \textit{SportsEquipment} \\
& \textit{Pole} \sqcap \textit{Bar} \sqsubseteq \perp \\
& \textit{Pole} \sqcap \textit{Jumper} \sqsubseteq \perp \\
& \textit{Jumper} \sqcap \textit{Bar} \sqsubseteq \perp \\
& \textit{JumpingEvent} \sqsubseteq \exists_{\leq 1} \textit{hasParticipant.Jumper} \\
& \textit{PoleVault} \sqsubseteq \textit{JumpingEvent} \sqcap \exists \textit{hasPart.Pole} \sqcap \exists \textit{hasPart.Bar} \\
& \textit{HighJump} \sqsubseteq \textit{JumpingEvent} \sqcap \exists \textit{hasPart.Bar} \\
& \textit{near}(Y, Z) \leftarrow \textit{PoleVault}(X), \textit{hasPart}(X, Y), \textit{Bar}(Y), \\
& \quad \textit{hasPart}(X, W), \textit{Pole}(W), \\
& \quad \textit{hasParticipant}(X, Z), \textit{Jumper}(Z) \\
& \textit{near}(Y, Z) \leftarrow \textit{HighJump}(X), \textit{hasPart}(X, Y), \textit{Bar}(Y), \\
& \quad \textit{hasParticipant}(X, Z), \textit{Jumper}(Z)
\end{aligned}$$

And a document (i.e. an image) describing a pole vault event, whose analysis results Γ contain instances of the MLCs “Pole”, “Human”, “Bar” and a relation that the human is near the bar:

$$\begin{aligned}
& \textit{pole}_1 : \textit{Pole} \\
& \textit{human}_1 : \textit{Human}
\end{aligned}$$

⁵ <http://www.iaaf.org/>

$$\begin{aligned} & \text{bar}_1 : \text{Bar} \\ & (\text{bar}_1, \text{human}_1) : \text{near} \end{aligned}$$

The interpretation process splits the set of analysis assertions Γ into two subsets: (a) Γ_1 (bona fide assertions): $\{\text{pole}_1 : \text{Pole}, \text{human}_1 : \text{Human}, \text{bar}_1 : \text{Bar}\}$, which are assumed to be true by default, and (b) Γ_2 (fiat assertions): $\{(\text{bar}_1, \text{human}_1) : \text{near}\}$, containing the assertions aimed to be explained. Since Γ_1 is always true, $\Sigma \cup \Delta \models \Gamma$ can be expressed as $\Sigma \cup \Gamma_1 \cup \Delta \models \Gamma_2$. Then, a query Q_1 is formed from each fiat assertion (Γ_2), such as $Q_1 := \{()\mid \text{near}(\text{bar}_1, \text{human}_1)\}$. Executing the query, a set of possible explanations (interpretations) is retrieved:

$$\begin{aligned} \Delta_1 &= \{\text{NewInd}_1 : \text{PoleVault}, (\text{NewInd}_1, \text{bar}_1) : \text{hasPart}, \\ & \quad (\text{NewInd}_1, \text{NewInd}_2) : \text{hasPart}, \text{NewInd}_2 : \text{Pole}, \\ & \quad (\text{NewInd}_1, \text{human}_1) : \text{hasParticipant}, \text{human}_1 : \text{Jumper}\} \\ \Delta_2 &= \{\text{NewInd}_1 : \text{PoleVault}, (\text{NewInd}_1, \text{bar}_1) : \text{hasPart}, \\ & \quad (\text{NewInd}_1, \text{pole}_1) : \text{hasPart}, (\text{NewInd}_1, \text{human}_1) : \text{hasParticipant}, \\ & \quad \text{human}_1 : \text{Jumper}\} \\ \Delta_3 &= \{\text{NewInd}_1 : \text{HighJump}, (\text{NewInd}_1, \text{bar}_1) : \text{hasPart}, \\ & \quad (\text{NewInd}_1, \text{human}_1) : \text{hasParticipant}, \text{human}_1 : \text{Jumper}\} \end{aligned}$$

Each interpretation is scored, according to a heuristic based on the number of hypothesized entities and the number of involved Γ_1 assertions used, and the best scoring interpretations are kept. For the example interpretation shown above, Δ_2 is the best scoring explanation, as Δ_1 has an excessive hypothesized entity (NewInd_2), and Δ_3 does not use the “Pole” instance from Γ_1 . More details about interpretation through abduction can be found in [6] and [9]. The language of the rules is *SROIQV*, and they can be written in OWL using *nominal schemas* [16]. For instance the first rule of the TBox shown in the previous section can be written as follows:

$$\begin{aligned} & \text{Bar} \sqcap \{Y\} \sqcap \exists \text{hasPart}^-. (\text{PoleVault} \sqcap \{X\} \sqcap (\exists \text{hasPart}. (\text{Pole} \sqcap \{W\}))) \sqcap \\ & \quad (\exists \text{hasParticipant}. (\text{Jumper} \sqcap \{Z\}))) \sqsubseteq \{Y\} \sqcap \exists \text{near}. \{Z\} \end{aligned}$$

where $\{Z\}$ is a nominal variable [16]. However, we are going to use a more “comfortable” notation for rules through out this paper.

3.3 The Role of Interpretation Rules

Rules are considered part of the ontology TBox and their role is to provide guidance to the interpretation process. Their main responsibility is to provide additional knowledge on how analysis results (specified through MLCIs and relations between MLCIs) can be mapped into HLCIs within a single modality, and how HLCIs from various modalities can be fused. As a result, rules can be split in two categories: rules for semantic interpretation, and rules for fusion. Both categories follow the same design pattern for rules: each rule is built around a

specific instance or a relation between two instances in the “head” of the rule, followed by a set of statements or restrictions in the “body” of the rule. When a rule is applied by the semantic interpretation engine, instances can be created to satisfy the rule, either for concepts/relations of the head (forward rules) or for concepts on the head (backward rules).

Forward rules Forward rules perform an action (usually the addition of a relation between two instances) described in the head of the rule, if the restrictions contained in the body have been satisfied. For example, consider the following ABox fragment:

$$\begin{aligned}
 &(personName_1, \text{“JaroslavRybakov”}) : hasValue \\
 &\quad (ranking_1, \text{“1”}) : hasValue \\
 &(person_1, personName_1) : hasPersonName \\
 &\quad personName_1 : PersonName \\
 &\quad\quad person_1 : Person \\
 &\quad\quad\quad ranking_1 : Ranking \\
 &(personName_1, ranking_1) : personNameToRanking
 \end{aligned}$$

This ABox fragment describes the situation where the semantics extraction engine has identified two MLCIs, a person name (“Jaroslav Rybakov”) and a ranking (“1”), connected with the “personNameToRanking” relation. Also, a “Person” instance exists that relates only to the “PersonName” instance, but not to the “Ranking” instance. Despite the fact that the $personName_1$ MLCI is related to the $ranking_1$ MLCI, the $person_1$ HLCI that aggregates $personName_1$ is not related to the “Ranking” instance. In order for the “Person” instance to be related to the “Ranking” instance, a forward rule like the following one must be present during interpretation:

$$\begin{aligned}
 personToRanking(X, Z) \leftarrow & Person(X), PersonName(Y), \\
 & hasPersonName(X, Y), \\
 & personNameToRanking(Y, Z)
 \end{aligned}$$

This rule can be interpreted as follows: if a “Person” instance X and a “PersonName” instance Y are found connected with a $hasPersonName(X, Y)$ relation, and a relation “personNameToRanking” exists between the “PersonName” Y and any instance Z , then add a relation between the “Person” instance X and the instance Z . The fact that the rule is applied in a forward way, suggests that all restrictions in the body have to be met, for the relation “personToRanking” on the head to be added in an ABox.

Backward rules Backward rules on the other hand assume that the restriction described by the head is already satisfied by the ABox (i.e. instances and

relations exist in the ABox), and that the action involves the addition of (one or more) missing instances or relations to satisfy the body. Consider for example the following ABox fragment from the image modality:

$$\begin{aligned} & personBody_1 : PersonBody \\ & personFace_1 : PersonFace \\ & (personBody_1, personFace_1) : isAdjacent \end{aligned}$$

This ABox fragment describes two MLCIs (a person face and a person body) that are found adjacent inside an image. Also, suppose that the TBox contains a backward rule like the following one:

$$\begin{aligned} isAdjacent(Y, Z) \leftarrow & Person(X), PersonBody(Y), PersonFace(Z), \\ & hasPart(X, Y), hasPart(X, Z) \end{aligned}$$

This rule roughly suggests that if a person face and a person body instances are aggregated by a person instance (and thus both body parts are related to the person instance with the “hasPart” relation), then the two body parts must be adjacent to each other. However, since the relation $isAdjacent(personBody_1, personFace_1)$ already exists in the ABox and the rule is a backward one, it will try to hypothesise a “Person” instance X , and aggregate the two body parts.

3.4 Rules for Semantic Interpretation

One domain of rules application is the semantic interpretation of the results obtained from the low level analysis, performed on multimedia resources. During this interpretation process, the MLCIs and the relations among MLCIs are examined, in order to aggregate the MLCIs into HLCIs. Then, relations that hold between MLCIs are promoted to the HLCIs that aggregate the corresponding MLCIs. Finally, an iterative process starts, which tries to aggregate the HLCIs into other HLCIs and again promote the relations, until no other instances can be added to an ABox. As a result, two types of rules are required during interpretation: rules that aggregate concept instances (either MLCIs or HLCIs) into instances of HLCs, and rules that promote relations. However, not all relations must be promoted: only relations that hold between a property instance of an HLCI and an instance that is not a property of the HLCI should be promoted to the HLCI. The aggregation of instances into HLCIs is performed with the help of *backward rules*⁶, while the promotion of relations from properties to the aggregating HLCI is performed with *forward rules*.

3.5 Acquiring Rules

When the ontology changes (i.e. through the addition of a new concept) the interpretation rules must be modified accordingly. We tried to automate this

⁶ Backward rules imply the use of abduction to hypothesize instances not contained in the original ABox.

task by monitoring ontological changes: the actions performed by an ontology expert to the ontology are monitored and reflected to the interpretation rules, following a transformation based approach. Considering as input what an ABox can contain without the current concept definition available, and as output the instances that can be generated from the concept if defined, the rule generation approach tries to find a set of rules that can transform the input into the desired output. In order to perform this transformation, the transformation is split into a set of more primitive “operations” that can be easily transformed into rules.

Assuming the set of all possible concepts C , the set of all possible relations R , a set of predefined operations O on a single concept $c \in C$, and a modification M over c , where $M = \{m_i(c, c_i, r_i)\}_1^N$, $m_i \in O$, $c_i \in C$, $c_i \neq c$, $r_i \in R$, the target is to calculate a rule set $S = \{r_i\}_1^N$, $r_i = T_{m_i}(c, c_i, r_i)$, that corresponds to the modification M . T_{m_i} is a function that transforms a hypothesized initial state (c', c_i, r'_i) to the desired state (c, c_i, r_i) for modification m_i , $T_{m_i} : (c', c_i, r'_i) \rightarrow (c, c_i, r_i)$, $c' \in C$, $r' \in R$. Each function T_{m_i} depends not only on m_i and the two states, but also on the interpretation engine. Since the objective of rules generation was to eliminate manual supervision, a pattern based approach was selected for representing each T_{m_i} . Each pattern is responsible for generating the required rules from transforming the initial state (c', c_i, r'_i) to a final state (c, c_i, r_i) for each operation in O , possibly biased towards the specific interpretation model.

Operations over a Single Concept A set of predefined operations O has been defined that captures all modifications that can be performed on a concept c within the BOEMIE system. This set contains the following operations:

- Definition of a new MLC c : This operation reflects the addition of a new MLC to the ontology TBox, an action that is not associated with the modification of the rules associated with the TBox. For this operation, $T = \{\}$.
- Definition of a new HLC c that aggregates a single concept c' : This operation describes the action of the definition of an HLC based on the presence of either an MLC or an HLC. Typical usage of this operation is when a new HLC c has been defined that aggregates another concept c' , and c' is enough to define this concept c . In such a case, it is assumed that during interpretation an instance of c should be created for every instance of c' found in an ABox. Thus the set of rules T should create an instance of c for every instance of c' . Example of this operation is the definition of “Person” (c) that aggregates either “PersonName” or “PersonFace” (c').
- Addition of a single concept c' to an existing HLC c : This operation deals with the extension of an existing HLC c with a concept c' , i.e. when adding a new property to an existing HLC. In such a case, T should contain rules that aggregate instances of concept c' with instances of concept c , and promote all relations between the instance of c' and instances not aggregated by the instance of c to the c instance. Examples include the extension of “Person” with properties like “Age”, “Gender”, or “PersonBody” and the “SportsEvent” with “Date”, or “Location”.

- Removal of a single concept c' from an HLC c : This operation handles property removals from HLCs. The rule set T is identical to the operation of adding a property to an HLC, with the difference that each rule in T is located and removed from the TBox rules, instead of extending it.
- Removal of HLC c that aggregates a single concept c' : Again, this operation is the negation of creating a new HLC that aggregates a single concept operation. Thus, the rule set T is identical between the two operations, but this operation causes the removal of all rules in T from the TBox.
- Removal of an MLC c : Similar to the addition of a new MLC operation, this operation has no effect on the TBox rule set, i.e. no rules are removed.

Rule templates for concept definition operations In this subsection the templates for generating rules are described, for the operators that do not have an empty set T , and are not related to removals, which share the same T with the corresponding addition operations.

Definition of a new HLC c that aggregates a single concept c' The rule set T during the definition of a new HLC c from a concept c' should contain rules that create instances of c from instances of c' found in the ABox of a multimedia resource. In the interpretation model used in BOEMIE, this can be accomplished by a single backward rule, which can be described with the following pattern:

$$\langle c' \rangle (X) \leftarrow \langle c \rangle (Y), \text{has} \langle c' \rangle (Y, X)$$

For example, if c is “Person” and c' is “PersonName”, the following rule can be generated from this pattern:

$$\text{PersonName}(X) \leftarrow \text{Person}(Y), \text{hasPersonName}(Y, X)$$

Addition of a single concept c' to an existing HLC c The rule set T during the addition of a property c' to an HLC c should contain rules that relate instances of c with instances of c' found in the ABox of a multimedia resource. In addition, it should contain rules that promote the relations of a c' instance with all instances not aggregated by c onto the c instance. This operation reflects an action performed on the definition of concept c , from which the “final” state (c, c', r) is known. The state (c, c', r) is the part of the concept definition that relates to how c aggregates c' . For example, if “Person” in the image modality is defined as having only a single property ($\text{hasPersonFace} : \text{PersonFace}$), and the operation is to extend it also with “PersonBody” through the role “hasPersonBody”, then $(c, c', r) = (\text{Person}, \text{PersonBody}, \text{hasPersonBody})$. According to the adopted interpretation model, c' can be aggregated with c only if c' is related with any property of c . If c'' , $c'' \neq c'$ is an aggregated by c concept, then an “initial” state (c'', c', r'') is hypothesized, relating c' with c'' through the relation r'' . Continuing the example, since “Person” has a single aggregated concept, only one initial state can be hypothesized, i.e. $(c'', c', r'') = (\text{PersonFace}, \text{PersonBody}, \text{isAdjacent})$. Once both initial and final states have been decided, then a rule pattern can be

defined to transform the initial into the final state. In the interpretation model used within BOEMIE, this can be accomplished by a single backward rule, which can be described with the following pattern:

$$\langle r'' \rangle (Y, Z) \leftarrow \langle c \rangle (X), \text{has} \langle c'' \rangle (X, Y), \langle c'' \rangle (Y), \\ \langle r \rangle (X, Z), \langle c' \rangle (Z)$$

Applied to our example, this pattern will lead to the following rule:

$$\text{isAdjacent} (Y, Z) \leftarrow \text{Person} (X), \text{hasPersonFace} (X, Y), \text{PersonFace} (Y), \\ \text{hasPersonBody} (X, Z), \text{PersonBody} (Z)$$

This rule can relate instances of “PersonBody” to instances of “Person”, already related to instances of “PersonFace”. The same process should be repeated for all possible initial states that can be found for concept c .

However these are not the only rules that should be added in set T . Each relation w defined in the TBox that can have as subject concepts c and c' , must be promoted from c' to c . This can be accomplished with forward rules that can be generated by the following pattern:

$$\langle w \rangle (X, Z) \leftarrow \langle c \rangle (X), \langle r \rangle (X, Y), \langle c' \rangle (Y), \langle w \rangle (Y, Z)$$

Please note that in this pattern no type is specified for variable Z , allowing Z to take as value instances of any concept that is in the range of the relation $\langle w \rangle$. Assuming $w = \text{isNear}$, this pattern can lead to the following rule:

$$\text{isNear} (X, Z) \leftarrow \text{Person} (X), \text{hasPersonBody} (X, Y), \text{PersonBody} (Y), \\ \text{isNear} (Y, Z)$$

The rule set T must be extended with a single rule of the above form for each w that can be found in the ontology TBox.

4 Conclusions

In this paper we have presented a novel approach for exploiting an ontology in an ontology-based information extraction system, which substitutes part of the extraction process with reasoning, guided by a set of automatically acquired rules. Innovative aspects of the presented framework include the use of reasoning in the construction of an ontology-based information extraction system that can adapt to changes in the ontology and the clear distinction between concepts of the low-level analysis (MLCs), and the semantic interpretation (HLCs). An interesting future direction is the investigation of how reasoning can be better applied on modalities involving the dimension of time, such as video. In BOEMIE a simple approach has been followed regarding the handling of time sequences, where extracted real objects or events were grounded to timestamps, and artificial relations like “before” and “after” were added. Nevertheless, an enhancement that maintains the temporal semantics from the perspective of reasoning will be an interesting addition.

Acknowledgments.

This work has been partially funded by the BOEMIE Project, FP6-027538, 6th EU Framework Programme.

References

1. Alani, H., Kim, S., Millard, D.E., Weal, M.J., Hall, W., Lewis, P.H., Shadbolt, N.R.: Automatic ontology-based knowledge extraction from web documents. *IEEE Intelligent Systems* 18(1), 14–21 (Jan 2003), <http://dx.doi.org/10.1109/MIS.2003.1179189>
2. Alani, H., Kim, S., Millard, D.E., Weal, M.J., Lewis, P.H., Hall, W., Shadbolt, N.: Automatic extraction of knowledge from web documents. In: *Workshop on Human Language Technology for the Semantic Web and Web Services, 2 nd Int. Semantic Web Conf. Sanibel Island (2003)*
3. Buitelaar, P., Cimiano, P., Frank, A., Hartung, M., Racioppa, S.: Ontology-based information extraction and integration from heterogeneous data sources. *Int. J. Hum.-Comput. Stud.* 66(11), 759–788 (Nov 2008), <http://dx.doi.org/10.1016/j.ijhcs.2008.07.007>
4. Buitelaar, P., Cimiano, P., Magnini, B.: *Ontology Learning from Text: Methods, Evaluation and Applications, Frontiers in Artificial Intelligence and Applications Series, vol. 123.* IOS Press, Amsterdam (7 2005)
5. Buitelaar, P., Siegel, M.: Ontology-based information extraction with soba. In: *Proc. of the International Conference on Language Resources and Evaluation (LREC)*. pp. 2321–2324 (2006)
6. Castano, S., Peraldi, I.S.E., Ferrara, A., Karkaletsis, V., Kaya, A., Möller, R., Montanelli, S., Petasis, G., Wessel, M.: Multimedia Interpretation for Dynamic Ontology Evolution. *Journal of Logic and Computation* 19(5), 859–897 (2009)
7. Cimiano, P., Handschuh, S., Staab, S.: Towards the self-annotating web. In: *Proceedings of the 13th international conference on World Wide Web*. pp. 462–471. WWW '04, ACM, New York, NY, USA (2004), <http://doi.acm.org/10.1145/988672.988735>
8. Dung, T.Q., Kameyama, W.: Ontology-based information extraction and information retrieval in health care domain. In: *Proceedings of the 9th international conference on Data Warehousing and Knowledge Discovery*. pp. 323–333. DaWaK'07, Springer-Verlag, Berlin, Heidelberg (2007), <http://dl.acm.org/citation.cfm?id=2391952.2391991>
9. Espinosa, S., Kaya, A., Melzer, S., Möller, R.: On ontology based abduction for text interpretation. In: Gelbukh, A. (ed.) *Proc. of 9th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2008)*. pp. 194–205. No. 4919 in LNCS, Springer (2008)
10. Gruber, T.R.: Toward principles for the design of ontologies used for knowledge sharing. *Int. J. Hum.-Comput. Stud.* 43(5-6), 907–928 (Dec 1995), <http://dx.doi.org/10.1006/ijhc.1995.1081>
11. Gutierrez, F., Wimalasuriya, D.C., Dou, D.: Using information extractors with the neural electromagnetic ontologies. In: Meersman, R., Dillon, T.S., Herrero, P. (eds.) *OTM Workshops. Lecture Notes in Computer Science, vol. 7046*, pp. 31–32. Springer (2011)

12. Hwang, C.H.: Incompletely and imprecisely speaking: Using dynamic ontologies for representing and retrieving information. In: Franconi, E., Kifer, M. (eds.) KRDB. CEUR Workshop Proceedings, vol. 21, pp. 14–20. CEUR-WS.org (1999)
13. Iosif, E., Petasis, G., Karkaletsis, V.: Ontology-Based Information Extraction under a Bootstrapping Approach. In: Paziienza, M.T., Stellato, A. (eds.) Semi-Automatic Ontology Development: Processes and Resources, chap. 1, pp. 1–21. IGI Global, Hershey, PA, USA (April 2012)
14. Karkaletsis, V., Fragkou, P., Petasis, G., Iosif, E.: Ontology based information extraction from text. In: Paliouras, G., Spyropoulos, C.D., Tsatsaronis, G. (eds.) Knowledge-Driven Multimedia Information Extraction and Ontology Evolution. Lecture Notes in Computer Science, vol. 6050, pp. 89–109. Springer (2011)
15. Kim, S., Alani, H., Hall, W., Lewis, P., Millard, D., Shadbolt, N., Weal, M.: Artequakt: Generating tailored biographies from automatically annotated fragments from the web. In: Workshop on Semantic Authoring, Annotation & Knowledge Markup (SAAKM'02), the 15th European Conference on Artificial Intelligence, (ECAI'02). vol. -, pp. 1–6 (2002), <http://eprints.soton.ac.uk/256913/>, event Dates: July 21-26
16. Krötzsch, M., Maier, F., Krisnadhi, A.A., Hitzler, P.: Nominal schemas for integrating rules and description logics. In: Rosati, R., Rudolph, S., Zakharyashev, M. (eds.) Description Logics. CEUR Workshop Proceedings, vol. 745. CEUR-WS.org (2011)
17. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of the Eighteenth International Conference on Machine Learning. pp. 282–289. ICML '01, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2001)
18. Li, Y., Bontcheva, K.: Hierarchical, perceptron-like learning for ontology-based information extraction. In: Proceedings of the 16th international conference on World Wide Web. pp. 777–786. WWW '07, ACM, New York, NY, USA (2007), <http://doi.acm.org/10.1145/1242572.1242677>
19. Maedche, A., Maedche, E., Staab, S.: The text-to-onto ontology learning environment. In: Software Demonstration at ICCS-2000 - Eight International Conference on Conceptual Structures (2000)
20. Maedche, A., Neumann, G., Staab, S.: Intelligent exploration of the web. chap. Bootstrapping an ontology-based information extraction system, pp. 345–359. Physica-Verlag GmbH, Heidelberg, Germany, Germany (2003), <http://dl.acm.org/citation.cfm?id=941713.941736>
21. Marsh, E., Perzanowski, D.: Muc-7 evaluation of ie technology: Overview of results. In: Proceedings of the Seventh Message Understanding Conference (MUC-7). http://www.itl.nist.gov/iaui/894.02/related_projects/muc/index.html (1998)
22. Petasis, G., Karkaletsis, V., Paliouras, G., Androutopoulos, I., Spyropoulos, C.D.: Ellogon: A New Text Engineering Platform. In: Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC 2002). pp. 72–78. European Language Resources Association, Las Palmas, Canary Islands, Spain (May 29–31 2002)
23. Petasis, G., Karkaletsis, V., Paliouras, G., Krithara, A., Zavitsanos, E.: Ontology Population and Enrichment: State of the Art. In: Paliouras, G., Spyropoulos, C.D., Tsatsaronis, G. (eds.) Knowledge-Driven Multimedia Information Extraction and Ontology Evolution, Lecture Notes in Computer Science, vol. 6050, pp. 134–166. Springer Berlin / Heidelberg (2011), http://dx.doi.org/10.1007/978-3-642-20795-2_6

24. Petasis, G., Karkaletsis, V., Paliouras, G., Krithara, A., Zavitsanos, E.: Ontology Population and Enrichment: State of the Art. In: Paliouras, G., Spyropoulos, C.D., Tsatsaronis, G. (eds.) *Knowledge-Driven Multimedia Information Extraction and Ontology Evolution*, Lecture Notes in Computer Science, vol. 6050, pp. 134–166. Springer Berlin / Heidelberg (2011)
25. Popov, B., Kiryakov, A., Kirilov, A., Manov, D., Ognyanoff, D., Goranov, M.: Kim - semantic annotation platform. In: Fensel, D., Sycara, K.P., Mylopoulos, J. (eds.) *International Semantic Web Conference*. Lecture Notes in Computer Science, vol. 2870, pp. 834–849. Springer (2003)
26. Popov, B., Kiryakov, A., Ognyanoff, D., Manov, D., Kirilov, A.: Kim - a semantic platform for information extraction and retrieval. *Natural Language Engineering* 10(3-4), 375–392 (2004)
27. Schank, R.C., Abelson, R.P.: *Scripts, Plans, Goals and Understanding: an Inquiry into Human Knowledge Structures*. L. Erlbaum, Hillsdale, NJ (1977)
28. Schank, R.C., Kolodner, J.L., DeJong, G.: Conceptual information retrieval. In: *Proceedings of the 3rd annual ACM conference on Research and development in information retrieval (SIGIR '80)*. pp. 94–116. Cambridge, UK (1980)
29. Studer, R., Benjamins, R., Fensel, D.: Knowledge engineering: Principles and methods. *Data & Knowledge Engineering* 25(1-2), 161–198 (März 1998)
30. Wimalasuriya, D.C., Dou, D.: Components for information extraction: ontology-based information extractors and generic platforms. In: *Proceedings of the 19th ACM international conference on Information and knowledge management*. pp. 9–18. CIKM '10, ACM, New York, NY, USA (2010), <http://doi.acm.org/10.1145/1871437.1871444>
31. Wimalasuriya, D.C., Dou, D.: Ontology-based information extraction: An introduction and a survey of current approaches. *J. Inf. Sci.* 36(3), 306–323 (Jun 2010), <http://dx.doi.org/10.1177/0165551509360123>
32. Wu, F., Weld, D.S.: Automatically refining the wikipedia infobox ontology. In: *Proceedings of the 17th international conference on World Wide Web*. pp. 635–644. WWW '08, ACM, New York, NY, USA (2008), <http://doi.acm.org/10.1145/1367497.1367583>

Recognizing Implicit Discourse Relations through Abductive Reasoning with Large-scale Lexical Knowledge

Jun Sugiura, Naoya Inoue, and Kentaro Inui

Tohoku University, 6-3-09 Aoba, Aramaki, Aoba-ku, Sendai, 980-8579, Japan
{jun-s, naoya-i, inui}@ecei.tohoku.ac.jp

Abstract. Discourse relation recognition is the task of identifying the semantic relationships between textual units. Conventional approaches to discourse relation recognition exploit surface information and syntactic information as machine learning features. However, the performance of these models is severely limited for implicit discourse relation recognition. In this paper, we propose an abductive theorem proving (ATP) approach for implicit discourse relation recognition. The contribution of this paper is that we give a detailed discussion of an ATP-based discourse relation recognition model with open-domain web texts.

Keywords: Discourse Relation, Abductive Reasoning, Lexical Knowledge, Association Information

1 Introduction

Discourse relation recognition is the task of identifying the semantic relationship between textual units. For example, given the sentence *John pushed Paul towards a hole.*_(S1) *Paul didn't get hurt.*_(S2), we identify a contrast relationship between textual units (S1) and (S2). Discourse relation recognition is useful for many NLP tasks such as summarization, question answering, and coreference resolution.

The traditional studies on discourse relation recognition divided discourse relations into two distinct types according to the presence of discourse connectives between textual units: (i) an *explicit* discourse relation, or discourse relation with discourse connectives (e.g. *John hit Tom because he is angry.*). (ii) an *implicit* discourse relation, or discourse relation without discourse connectives (e.g. *John hit Tom. He got angry.*). Identifying an implicit discourse relation is much more difficult than identifying an explicit discourse relation. In this paper, we focus on the task of implicit discourse relation recognition.

Conventional approaches to implicit discourse relation recognition exploit surface information (e.g. bag of words) and syntactic information (e.g. syntactic dependencies between words) to identify discourse relations [1, 2, etc.]. However, the performance of these models is severely limited as mentioned in Sec. 2.2. We believe that the problem of these approaches is two fold: (i) they do not capture causality between the events mentioned in each textual units, and (ii) they

do not capture the factuality of these events. We believe that this information plays a key role for implicit discourse relation recognition. Suppose we want to recognize a contrast relation between S_1 and S_2 in the first example. To recognize the discourse relation, we need to at least know the following information: (i) commonsense knowledge: *pushing into a hole* usually causes *getting hurt*; (ii) factuality: *Paul did not get hurt*. Finally, combining (i) and (ii), we need to recognize the unusualness of discourse: something against our commonsense knowledge happened in S_2 . As described in Sec. 3, our investigation revealed that we have several patterns of reasoning and need to combine several kinds of reasoning to identify a discourse relation.

Motivated by this observation, we propose an abductive theorem proving (ATP) approach for implicit discourse relation recognition. The key advantage of using ATP is that the declarative nature of ATP abstracts the flow of information away in a modeling phase: we do not have to explicitly specify when and where to use a particular reasoning. Once we give several primitive inference rules to an ATP system, the system automatically returns the best answer to a problem, combining the inference rules.

In this paper, we attempt to answer the following open issues of ATP-based discourse relation recognition: (i) does it really work on real-life texts?; (ii) does it work with a large knowledge base which is not customized for solving target texts? The contribution of this paper is as follows. We give a detailed discussion of an ATP-based discourse relation recognition model with open-domain web texts. In addition, we show that our ATP-based model is computationally feasible with a large knowledge base.

The structure of this paper is as follows. In Sec. 2, we describe abduction and give an overview of previous efforts on discourse relation recognition. In Sec. 3, we describe our ATP-based discourse relation recognition model. In Sec. 4, we report the results of pilot evaluation of our model with large lexical knowledge.

2 Background

2.1 Weighted abduction

Abduction is inference to the best explanation. Formally, logical abduction is defined as follows:

- **Given:** Background knowledge B , and observations O , where both B and O are sets of first-order logical formulas.
- **Find:** A *hypothesis* (or *explanation*, *abductive proof*) H such that $H \cup B \models O$, $H \cup B \not\models \perp$, where H is a conjunction of literals. We say that p is *hypothesized* if $H \cup B \models p$, and that p is *explained* if $(\exists q) q \rightarrow p \in B$ and $H \cup B \models q$.

Typically, several hypotheses H explaining O exist. We call each of them a *candidate hypothesis* and each literal in a hypothesis an *elemental hypothesis*. The goal of abduction is to find the best hypothesis among candidate hypotheses

by a specific measure. In the literature, several kinds of evaluation measure have been proposed, including cost-based and probability-based [3, 4, etc.].

In this paper, we adopt the evaluation measure of *weighted abduction*, which is proposed by Hobbs et al. [4]. In principle, the evaluation measure gives a penalty for assuming specific and unreliable information but rewards for inferring the same information from different observations. We summarize the primary feature of this measure as follows (see [4] for more detail):

- Each elemental hypothesis has a positive real-valued cost;
- The cost of each candidate hypothesis is defined as the sum of costs of the elemental hypotheses;
- The best hypothesis is defined as the minimum-cost candidate hypothesis;
- If an elemental hypothesis is explained by other elemental hypothesis, the cost becomes zero.

2.2 Related work

Discourse relation recognition is a prominent research area in NLP. Most researchers have primarily focused on explicit discourse relation recognition, employing statistical machine learning-based models [5, 6, etc.] with superficial and syntactic information. The performance of explicit discourse relation recognition is comparatively high; for instance, Lin et al. [7] achieved an 80.6% F-score.

The performance of implicit discourse relation recognition is, however, relatively low (25.5% F-score). Most existing work on implicit discourse relation recognition [1, 2, etc.] extend the feature set of [5] with richer lexico-syntactic information. For example, Pitler et al. [2] exploit a syntactic parse tree and sentiment polarity information of words contained in textual units to generate a feature set. However, the performance is not as high as a practical level.

An abductive discourse relation recognition model is originally presented in Hobbs et al. [4]. However, Hobbs et al. [4] reported the results in a fairly closed setting: they tested their model on two test texts with manually encoded background knowledge which is required to solve the discourse relation recognition problems that appear in two texts. Therefore, it is an open question whether the abductive discourse relation recognition model works in an open setting where the wider range of real-life texts and large knowledge base are considered.

3 Abductive theorem proving for discourse relation recognition

In this section, we describe our discourse relation recognition model. We employ ATP to recognize a discourse relation. Given target discourse segments, we abductively prove that there exists a coherence relation (i.e. some discourse relation) between the discourse segments using background knowledge. We axiomatize (i) definition of discourse relations and (ii) lexical knowledge (e.g. causal knowledge of events) in the background knowledge, which serve as a proof of the existence of a coherence relation.

The motivation of using abductive theorem proving is that we can assume a proposition with the cost even if we fail to find a complete proof of a coherence relation between discourse segments, as mentioned in Sec. 2. By choosing the minimum-cost abductive proof, we can identify the most likely discourse relation.

We first show how to axiomatize the definition of discourse relations (Sec. 3.1). We then conduct an example-driven investigation of lexical knowledge which is required to solve a few real-life discourse relation recognition problems in order to identify a type of lexical knowledge needed for an ATP-based recognition model (Sec. 3.2). In Sec. 3.2, we make sure that our developed theory works on a general-purpose inference engine as we expected. We use the lifted first-order abductive inference engine *Henry*, which is developed by one of the authors.[8] To perform deeper analysis of the inference results, we also improved the existing visualization module provided by Henry. The inference engine and visualization tool are publicly available at <https://github.com/naoya-i/henry-n700/>.

3.1 Axiomatizing definitions of discourse relations

We follow the definitions of discourse relations provided by Penn Discourse Tree-Bank (PDTB) 2.0 [9], a widely used and large-scale corpus annotated with discourse relations.¹ The PDTB defines four coarse-grained discourse relations, but it is still rather difficult to identify all discourse relations. Therefore, we adopt two-way classification: whether it is adversative (*Comparison* in PDTB) or resultative (*Temporal*, *Contingency*, *Expansion* in PDTB). Because a resultative relation can be regarded as relations other than an adversative relation, we first axiomatize the definition of adversative and then consider the other relation.

According to the PDTB Annotation Manual [12], an adversative relation consists of two subtypes: *Concession* and *Contrast*. These subtypes are defined below, respectively.

Concession One of the arguments describes a situation A which causes C, while the other asserts (or implies) $\neg C$. One argument denotes a fact that triggers a set of potential consequences, while the other denies one or more of them.

Contrast Arg1 and Arg2 share a predicate or a property and the difference is highlighted with respect to the values assigned to this property.

The condition of *Concession* can be described as the following axiom:

$$\begin{aligned} & event(e_1, type1, F, x_1, x_2, x_3, s_1) \wedge event(e_2, type2, NF, y_1, y_2, y_3, s_2) \\ & \wedge cause(e_1, F, e_2, F) \wedge event(e_u, type2, EF, y_1, y_2, y_3, s_u) \Rightarrow Adversative(s_1, s_2). \end{aligned}$$

This axiom says that if event e_1 occurs in segment s_1 (roughly corresponding to Arg1 in PDTB) and that event is expected to cause an event of *type2* while such an event of *type2* does actually not occur in segment s_2 (roughly corresponding to Arg2 in PDTB), then the discourse relation between s_1 and s_2 is Adversative. The examples using this type of axiom to recognize discourse relations will

¹ For other corpora, see [10, 9, 11, etc.].

be mentioned later. On the other hand, a typical pattern where the *Contrast* relation holds can be described, for example, as follows:

$$value(e_1, Pos, s_1) \wedge value(e_2, Neg, s_2) \Rightarrow Adversative(s_1, s_2).$$

This axiom says that when the sentiment polarity of e_1 in segment s_1 is Positive and the sentiment polarity of e_2 in segment s_2 is Negative, the discourse relation between s_1 and s_2 is Adversative. The examples of axioms described here represent formation conditions of Adversative and take some variation due to their value of factuality or sentiment.

Furthermore, these axioms can represent conditions of Resultative. For instance, if the sentiment polarity of e_2 in segment s_2 is the same as that of segment s_1 , then the discourse relation between s_1 and s_2 is Resultative as below:

$$value(e_1, Pos, s_1) \wedge value(e_2, Pos, s_2) \Rightarrow Resultative(s_1, s_2).$$

In total, we created 21 axioms for the definition of discourse relations.

Finally, we add the following axioms to connect the definition of discourse relations with the existence of a coherence relation between discourse segments:

$$Adversative(s_1, s_2) \Rightarrow CoRel(s_1, s_2) \quad (1)$$

$$Resultative(s_1, s_2) \Rightarrow CoRel(s_1, s_2), \quad (2)$$

where $CoRel(s_1, s_2)$ indicates that there exists a coherence relation between segments s_1 and s_2 . Given target discourse segments S_1, S_2 , we prove $CoRel(S_1, S_2)$ using the axioms described above and lexical knowledge which is described in the next section.

We formally describe our meaning representation. First, we use $cause(e_a, f_a, e_c, f_c)$ to represent that event e_a with factuality f_a causes event e_c with factuality f_c . Second, we represent an event by using $event(e, t, f, x_1, x_2, x_3, s)$, where e is the event variable, t is the event type of e , f is the factuality of event e , x_1, x_2, x_3 are arguments of event and s is the segment which event e belongs to. Factuality of event e can take one of the following four values: F (Fact; e occurred), NF (NonFact; e did not occur), EF (Expected-Fact; e is expected to occur), and ENF (Expected-NonFact; e is expected not to occur). In addition, the value (sentiment polarity) of event e is represented as $value(e, v, s)$. v is either Pos (Positive) or Neg (Negative).

3.2 Example-driven investigation of lexical knowledge

Next, we manually analyzed a small number of samples for each discourse relation to investigate what types of knowledge are required to explain those samples and how to axiomatize them. In this paper, we manually convert each sample text into the logical forms, extracting main verbs in its matrix clauses as predicates.²

² In future work, we will exploit the off-the-shelf semantic parser (e.g. Boxer [13]) to automatically get the logical forms. Using automatic semantic parsers brings some challenges to us, e.g. how to represent the verbs in embedded clauses. We do not address these issues in this work, because we want to focus on the investigation of types of world knowledge that are required to identify discourse relations.

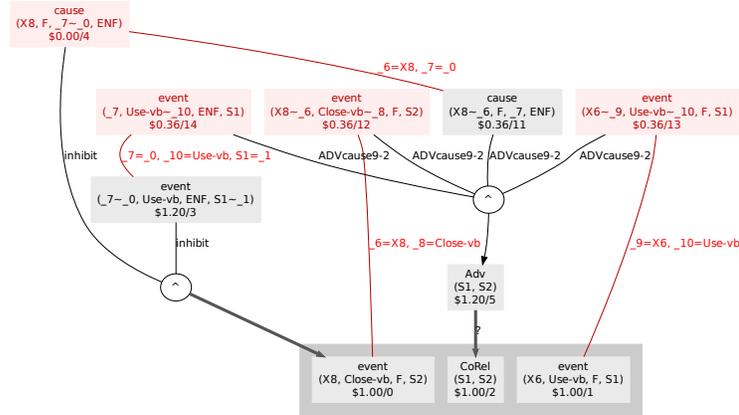


Fig. 1. Example of the abductive proof automatically produced by our system. The black directed edges: backward-chainings. The red undirected edges: unification. The labels of undirected red edges: unifier. The terms starting with a capital letter: constant; otherwise, variable. “ $X \sim Y$ ”: Y is unified with X . The grayed nodes: explained literals. The red nodes: hypothesized literals.

The dataset consists of text which we collect from the Web.³ This website provides English texts for adult English learners as a second language. We collect 16 discourse segment pairs in which half of them can be regarded as Adversative and the others can be regarded as Resultative.

Let us take one of the simplest samples, example (2).

- (2) S1: A lot of traffic once used Folsom Dam Road.
 S2: Right now, the road is closed.
 (Topic=Working, StoryID=174)

In this example, S_1 and S_2 are in the Adversative relation. While the Folsom Dam Road was once used by a lot of traffic, it is not usable now because it is closed. Something was once used but it is now unusable; therefore, the Adversative relation holds. This can be described as the following axiom:

Condition of Adversative

$$\begin{aligned} &event(e_1, type1, F, x_1, x_2, x_3, s_1) \wedge event(e_2, type2, F, y_1, y_2, y_3, s_2) \\ &\wedge cause(e_2, F, e_u, ENF) \wedge event(e_u, type1, ENF, x_1, x_2, x_3, s_1) \Rightarrow Adversative(s_1, s_2) \end{aligned}$$

The causality relation between the “closed” event and the “unusable” event can be described as:

³ <http://www.cdlponline.org/>

Relation between events

$$\begin{aligned} &event(e_1, Use, ENF, x_1, x_2, x_3, s_1) \wedge cause(e_2, F, e_1, ENF) \\ &\Rightarrow event(e_2, Close, F, y_1, x_2, y_3, s_2) \end{aligned}$$

Note that we have $cause(e_2, F, e_1, NF)$ in the left-hand side of the axiom. We use this literal to accumulate the type of reasoning. In an abductive proof, we expect this literal to unify with an elemental hypothesis generated by the axioms of discourse relation.

Fig. 1 shows the result of applying the proposed model to example (2).⁴ In Fig. 1, the observations consists of three literals; occurrence of event whose type is Use in segment S_1 , occurrence of event whose type is Close in segment S_2 , and $CoRel(S_1, S_2)$ which is a symbol of existence of some discourse relation between the segments.

To see how our model combines multiple pieces of knowledge, let us take another example.

- (3) S1: Right now, the road is closed.
 S2: Most of the people who used the road every day are angry.
 (Topic=Working, StoryID=174)

The “closed” event causes the “unusable” event and the “unusable” event than further causes the “angry” event, which can be explained by combining the knowledge that being “unusable” is negative in sentiment polarity and the knowledge that a negative event may cause someone to be angry. These pieces of knowledge can be axiomatized as follows. The proof graph is shown in Fig. 2.

Condition of Resultative

$$\begin{aligned} &event(e_1, type1, F, x_1, x_2, x_3, s_1) \wedge event(e_2, type2, F, y_1, y_2, y_3, s_2) \\ &\wedge cause(e_1, F, e_u, EF) \wedge event(e_u, type2, EF, y_1, y_2, y_3, s_2) \Rightarrow Resultative(s_1, s_2) \end{aligned}$$

Relation between events

$$\begin{aligned} &event(e_1, Use, ENF, x_1, x_2, x_3, s_1) \wedge cause(e_2, F, e_1, ENF) \\ &\Rightarrow event(e_2, Close, F, y_1, x_2, y_3, s_2) \\ &event(e_1, Angry, EF, x_1, x_2, x_3, s_1) \wedge cause(e_2, f, e_1, EF) \\ &\Rightarrow value(e_2, Neg, s_2) \wedge event(e_2, type, f, y_1, y_2, y_3, s_2) \end{aligned}$$

Transitivity

$$cause(e_1, f_1, e_2, f_2) \wedge cause(e_2, f_2, e_3, f_3) \Rightarrow cause(e_1, f_1, e_3, f_3)$$

Polarity

$$value(e_1, Neg, s_1) \Rightarrow event(e_1, Use, ENF, x_1, x_2, x_3, s_1)$$

Through the investigation as illustrated above, we reached the conclusion that the axioms in Table 1 can recognize discourse relations for most examples.

⁴ Throughout this paper, we omit the arguments of events from our representation in a proof graph for readability. Since we do not have a lexical knowledge between nouns and verbs, this simplification does not affect to the result of inference.

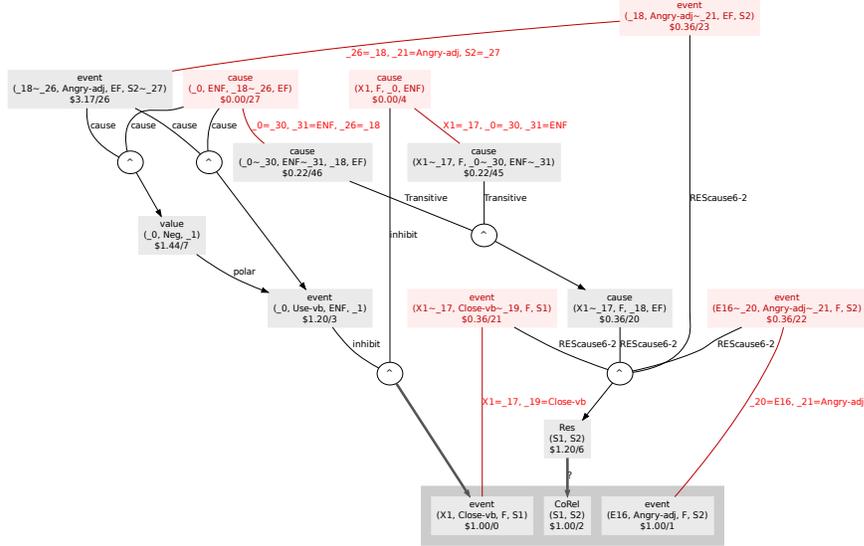


Fig. 2. Example of the abductive proof automatically produced by our system. See the description of Fig. 1.

4 Pilot large-scale evaluation

As mentioned in the previous section, our model assumes that axioms encoding lexical knowledge are automatically extracted from a large lexical resources (see “To be automatically acquired” axioms in Sec. 3.2.) In this section, we extract the axioms of causal relations and synonym/hyperonym relations from WordNet [14] and FrameNet [15], both popular and large lexical resources, and then apply our model to the example texts presented in Sec. 3. Regarding sentiment polarity, we plan to extract the axioms from a large-scale sentiment polarity lexicon such as [16] in future work.

We clarify that our primary focus here is the feasibility of our ATP-based discourse relation recognition model with a large knowledge base. The quantitative evaluation of our model (e.g. the predictive accuracy of discourse relations) is future work. Therefore, we first report how to incorporate WordNet and FrameNet axioms into our knowledge base (Sec. 4.1) and then preliminarily report the computational time of inference required to solve the example problems, showing some interesting output (Sec. 4.2).

Table 1. Set of axiom type.

Scale	Type of knowledge	Examples of axiom
Small-scale (Manually written)	Definitions of discourse relations	$event(e_1, type1, F, x_1, x_2, x_3, s_1)$ $\wedge event(e_2, type2, F, y_1, y_2, y_3, s_2)$ $\wedge cause(e_2, F, e_u, ENF)$ $\wedge event(e_u, type1, ENF, x_1, x_2, x_3, s_1)$ $\Rightarrow Adversative(s_1, s_2);$ $value(e_1, Pos, s_1) \wedge value(e_2, Pos, s_2)$ $\Rightarrow Resultative(s_1, s_2)$
	Transitivity	$cause(e_1, f_1, e_2, f_2) \wedge cause(e_2, f_2, e_3, f_3)$ $\Rightarrow cause(e_1, f_1, e_3, f_3)$
Large-scale (Automatically acquired)	Causal relations	$event(e_1, Use, ENF, x_1, x_2, x_3, s_1) \wedge cause(e_2, F, e_1, ENF)$ $\Rightarrow event(e_2, Close, F, y_1, x_2, y_3, s_2);$ $event(e_1, Angry, EF, x_1, x_2, x_3, s_1) \wedge cause(e_2, f, e_1, EF)$ $\Rightarrow value(e_2, Neg, s_2) \wedge event(e_2, type, f, y_1, y_2, y_3, s_2)$
	Synonym/Hyponym	$event(e_1, Attack, F, x_1, x_2, x_3, s_1)$ $\Rightarrow event(e_1, Destroy, F, x_1, x_2, x_3, s_1)$
	Sentiment polarity	$value(e_1, Neg, s_1) \Rightarrow event(e_1, Die, F, x_1, x_2, x_3, s_1);$ $value(e_1, Pos, s_1) \Rightarrow event(e_1, Die, NF, x_1, x_2, x_3, s_1)$

4.1 Automatic axiom extraction from linguistic resources

We summarize the axioms extracted from WordNet and FrameNet in Table 2. For each resource, we extract two kinds of axioms. First, we generate axioms that map a word to the corresponding WordNet synset or FrameNet frame (Word-Synset, or Word-Frame types). The example WordNet axiom in Table 2 enables us to hypothesize that a “*die*”-typed event can be mapped to WordNet synset 200358431. Second, we also encode a semantic relation between synsets or frames. For example, the causal relation between *Getting* frame and *Giving* frame is encoded as the axiom in the Table.⁵

4.2 Results and discussion

We have tested our large-scale discourse relation recognition model on the randomly selected 7 texts as those presented in Sec. 3. We restricted the maximum

⁵ Note that the mapping axioms have bi-directional implications. By using the bi-directional axioms, we can combine the knowledge from FrameNet and WordNet to perform a robust inference. For instance, we can do an inference like: *pass away* \rightarrow *synsetA* \rightarrow *die* \rightarrow *FNDeath* if we do not have a direct mapping from *pass away* to *FNDeath*. Since the framework is declarative, we do not have to specify when and where to use a particular type of knowledge, which results in a robust reasoning.

Table 2. Axioms automatically extracted from WordNet and FrameNet.

Type	Resources	Example	# axioms
Word-Synset	WordNet	$event(e, Die, f, x_1, x_2, x_3, s)$ $\Leftrightarrow event(e, WNSynset200358431, f, x_1, x_2, x_3, s)$	169,362
Word-Frame	FrameNet	$event(e, Shoot, f, x_1, x_2, x_3, s)$ $\Leftrightarrow event(e, FNUseFireArm, f, x_1, x_2, x_3, s)$	10,358
Causal relations	WordNet relationsA* ¹	$event(e_1, WNSynset20036712, EF, x_1, x_2, x_3, s_1)$ $\wedge cause(e_1, EF, e_2, F)$ $\Rightarrow event(e_2, WNSynset200358431, F, y_1, y_2, y_3, s_2)$ $event(e_1, FNGiving, EF, x_1, x_2, x_3, s_1)$	35,440
	FrameNet relations* ²	$\wedge cause(e_1, EF, e_2, F)$ $\Rightarrow event(e_2, FNGetting, F, y_1, y_2, y_3, s_2)$	6,584
Synonym/ Hyponym	WordNet relationsB* ³	$event(e, WNSynset200060063, f, x_1, x_2, x_3, s)$ $\Rightarrow event(e, WNSynset200358431, f, x_1, x_2, x_3, s)$	177,837

*1: Causality, Entailment, Antonym. *2: IsCausativeOf, InheritsFrom, PerspectiveOn, Precedes, SeeAlso, SubFrameOf, Uses. *3: Meronym, Hyperonym.

number of backward-chaining steps to 2 due to the computational feasibility. For each problem, on average, the number of potential elemental hypotheses was 13,034 and the (typed) number of axioms that were used to generate candidate abductive proofs was 142. The time of inference required to solve each problem was 7.00 seconds on average.

Now let us show one of the proof graphs automatically produced by our system.⁶ In Figure 3, we show the abductive proof graph for the following discourse:

- S_1 : Only 56 people died from the explosion,
 S_2 : but many other problems have been caused because of it.
 (Topic=Activity, StoryID=241)

Although we suffer from the insufficiency of lexical knowledge, the abductive engine gave us the best proof where two segments are tied with a resultative relation. In the proof graph, *Die* and *CauseProblem* events are used to prove “event” literals hypothesized by the axiom of discourse relation. Note that the causal relation between these events is not proven but assumed with \$0.36.

The overall results indicate that we now have a good environment to develop ATP-based discourse processing.

5 Conclusions

We have explored an abductive theorem proving (ATP)-based approach for implicit discourse relation recognition. We have investigated the type of axioms required for an ATP-based discourse relation recognition and identified five types of axioms. Our result is based on real-life Web texts, and no previous work has

⁶ For the simplicity, we used only one axiom for the axiom of discourse relation.

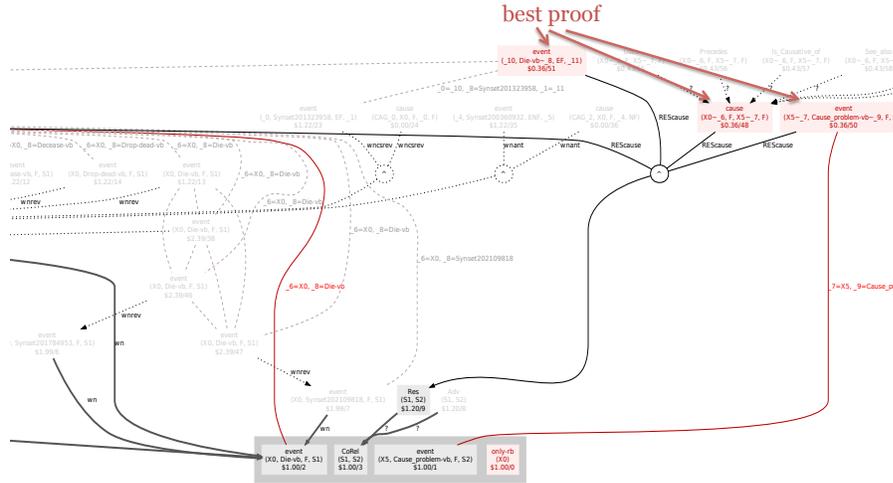


Fig. 3. Abductive proof with potential elemental hypotheses. The grayed-out nodes are those which are *potentially* included in the best proof, but not actually included in the best proof. Similarly, the dotted edges are *potential* explainer-explainee relationships between elemental hypotheses.

done an investigation in the same setting. Also, we have preliminarily evaluated our model with a large knowledge base. We have automatically constructed axioms of lexical knowledge from WordNet and FrameNet, which results in around four hundred thousand inference rules. Our experiments showed the great potential of our ATP-based model and that we are ready to develop ATP-based discourse processing in a real-life setting.

Our future work includes three directions. First, we will create a larger knowledge base, exploiting the linguistic resources which have recently become available. As a first step, we plan to axiomatize Narrative Chain [17], ConceptNet,⁷ and Semantic Orientations of Words [16] to extend our axioms with the large knowledge resources. Second, we plan on applying the technique of automatic parameter tuning for weighted abduction [18] to our model. Third, we plan to create a dataset for abductive discourse processing, where we annotate simple English texts with some discourse phenomena including discourse relations and coreference etc. As the source text, we will use materials for ESL (English as Second Language) learners,⁸ a set of syntactically and lexically simple texts, so that we can trace the detailed behavior of abductive reasoning process. About the semantic representation, we plan to use the Boxer semantic parser [13] to automatically get the event arguments.

⁷ <http://conceptnet5.media.mit.edu/>

⁸ <http://www.cdlponline.org/>

Acknowledgments. This work was supported by JSPS KAKENHI Grant Number 23240018.

References

1. Lin, Z., Kan, M., Ng, H.: Recognizing implicit discourse relations in the penn discourse treebank. In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing. (2009) 343–351
2. Pitler, E., Nenkova, A.: Using syntax to disambiguate explicit discourse connectives in text. In: Proceedings of the ACL-IJCNLP 2009. (2009) 13–16
3. Charniak, E., Goldman, R.: Probabilistic abduction for plan recognition. Brown University, Department of Computer Science (1991)
4. Hobbs, J., Stickel, M., Appelt, D., Martin, P.: Interpretation as Abduction. *Artificial Intelligence* **63**(1-2) (1993) 69–142
5. Marcu, D., Echihabi, A.: An unsupervised approach to recognizing discourse relations. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics. (2002) 368–375
6. Subba, R., Eugenio, B.D.: An effective discourse parser that uses rich linguistic information. In: NAACL. (2009) 566–574
7. Lin, Z., Ng, H., Kan, M.: A PDTB-Styled End-to-End Discourse Parser. Arxiv preprint arXiv:1011.0835 (2010)
8. Inoue, N., Inui, K.: Large-scale cost-based abduction in full-fledged first-order predicate logic with cutting plane inference. In: Proceedings of the 13th European Conference on Logics in Artificial Intelligence. (2012) 281–293
9. Prasad, R., Dinesh, N., Lee, A., Miltsakaki, E., Robaldo, L., Joshi, A., Webber, B.: The Penn Discourse TreeBank 2.0. In: Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC). (2008) 2961–2968
10. Carlson, Lynn and Marcu, Daniel and Okurowski, Mary Ellen: RST Discourse Treebank, LDC2002T07. Number LDC2002T07, Linguistic Data Consortium (2002)
11. Wolf, F., Gibson, E., Fisher, A., Knight, M.: The Discourse GraphBank: A database of texts annotated with coherence relations. LDC (2005)
12. Prasad, R., Miltsakaki, E., Dinesh, N., Lee, A., Joshi, A., Robaldo, L., Webber, B.: The Penn Discourse Treebank 2.0 Annotation Manual. IRCS Technical Report (2007)
13. Bos, J.: Wide-Coverage Semantic Analysis with Boxer. In Bos, J., Delmonte, R., eds.: Proceedings of STEP. Research in Computational Semantics, College Publications (2008) 277–286
14. Fellbaum, C., ed.: WordNet: an electronic lexical database. MIT Press (1998)
15. Ruppenhofer, J., Ellsworth, M., Petruck, M., Johnson, C., Scheffczyk, J.: FrameNet II: Extended Theory and Practice. Technical report, Berkeley, USA (2010)
16. Takamura, H., Inui, T., Okumura, M.: Extracting semantic orientations of words using spin model. In: ACL. (2005) 133–140
17. Chambers, N., Jurafsky, D.: Unsupervised learning of narrative schemas and their participants. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP. (2009) 602–610
18. Yamamoto, K., Inoue, N., Watanabe, Y., Okazaki, N., Inui, K.: Discriminative learning of first-order weighted abduction from partial discourse explanations. In: *CICLing* (1). (2013) 545–558