Generating Natural Language Descriptions of Ontology Concepts

Niels Schütte
Dublin Institute of Technology
Dublin, Ireland
niels.schutte@student.dit.ie

Abstract

This paper gives an overview of ongoing work on a system for the generation of NL descriptions of classes defined in OWL ontologies. We present a general structuring approach for such descriptions. Since OWL ontologies do not by default contain the information necessary for lexicalization, lexical information has to be added to the data via annotations. A rule-based mechanism for automatically deriving these annotations is presented.

1 Overview

There exists a body of works regarding the verbalization of content from RDF data or ontologies like OWL. Some approaches (such as (Galanis and Androutsopoulos, 2007)) rely on rich domain dependent resources, while other approaches try to do away with such resources as much as possible and derive information such as lexicalization data that is not explicitly included in the ontology from the available data.

2 Data Model and Message Definition

The goal of the system is to generate natural language texts from class definitions in an OWL ontology that serve as a description of the class.

To generate textual descriptions, linguistic representations for the contents of the ontology have to be found. Since OWL ontologies do not by default contain the information necessary for lexicalization, lexical information has to be added to the data. In the current system, classes are assumed to represent simple objects of the world and therefore to be realizable as noun phrases that can be lexicalized with the name of the class.

Attributes of classes are described in OWL by defining **restrictions** that apply to so called **properties**. Properties are binary relations among ontology objects. They are realized as syntactic structures that connect objects. During annotation, each property is assigned a certain **relation type** that determines the syntactic structure that is used to realize the property.

2.1 Relation types

The relation types form an abstraction over the possible structural realizations of properties by providing a specification of a surface structure that can be used to realize the property. Depending on the type of the relation, a number of other attributes of the relation may be specified to determine details of the realization, such as lexicalizations for some elements of the structure and a specification about how to fill the parameters of the configuration with the parameters of the property. At the moment there exists a small set of relation types that covers most of the relations in the example ontologies that were considered for the system. This approach corresponds with the results presented in (Hewlett et al, 2005) where the authors affirm to have found a small set of patterns that covers most of the properties in a number of major ontologies.

The relation type of a property also determines whether a property can be expressed as an adjective modifier. This information can be exploited in aggregation operations to create more concise text.

The two most important relation types are the ones called *simple* and *roleplaying*.

simple specifies that the properties should be realized as a simple configuration of two participants that are connected with a verb in the active form. This type fits preferably for properties like "eats" or "produces". The objects in the domain and range position of the property are most often mapped straight to domain and range parameters of the relation. Apart from this, it has to be determined which word to use to lexicalize the verb that

Proceedings of the 12th European Workshop on Natural Language Generation, pages 106–109,
Athens, Greece, 30 – 31 March 2009. ©2009 Association for Computational Linguistics

appears in the realization of the property. A typical sentence formed with a property of this type (in this example the property "eats") would be

A mouse eats only cheese.

roleplaying specifies that the property should be realized as a configuration in which one participant fulfills a certain role for another participants. This relation is typically used to realize properties like "hasColor" or "hasHabitat", since even though the property itself is a binary relation, its name suggests to express it as a configuration that involves, apart from the domain and range objects, a third object whose lexicalization is derived from the name of the property. A sentence for the property "hasParent" of this type would be:

A child has at most 2 humans as parent.

2.2 Automatic Annotation

In this section we describe our approach to automatically generating annotations using rules based on a part of speech analysis of the property name. A rule consists of a pattern and a specification of the relation that is to be used to realize the property. The pattern is a sequence of part of speech elements. A pattern fits a property, if the property name can be split into words whose part of speech are equal to the sequence specified by the pattern [1].

If the pattern fits, the relation is instantiated according to the specification associated in the rule with the pattern. Keywords can be used to assign the objects in the domain or range position to the domain or range slot of the relation. Names of parts of speech detected in the pattern can also be used to assign parts of the property name as lexicalization to elements of the relation. The following rule is currently used in the system:

```
VP -> Simple (SUBJ, OBJ, VP)
```

It assigns properties like "eats" to *simple* relations that use the domain object of the property as domain object and the range subject likewise. The element of the property name "VP" (in the example for "eats", simply "eats") is used to lexicalize the verb of the relation. Detected elements are always reduced to their stem before assigning lexicalizations (e.g. "eat" is actually assigned instead of "eats"). The following rule currently assigns properties like "hasColor" to *roleplaying* relations.

```
VP NP -> RolePlaying(SUBJ, OBJ, VP, NP)
          COND has(VP)
```

The *COND* part specifies an additional condition where certain parts of the pattern have to be filled with special words. The inclusion of special conditions for the rules allows it to create more specific patterns.

At this stage, the automatic assignment is only performed for annotating properties. It is however possible to extend this approach to classnames to create linguistically more complex lexicalizations for classes.

3 Structuring

The description texts generated by our system are structured based on analysis of texts from encyclopedia entries and the possible relations among the available pieces of information. The information available in the definition is dissected into discrete message objects. Before structuring begins, the system attempts to summarize some of the information from the definition.For example it is possible to combine cardinality restrictions without losing information.

The structure of the descriptions consists of an introductory passage, whose main purpose it is to give a quick burst of information about the class, and a sequence of subsequent sections that presents the remaining information about the class structured according to the properties of the class. The description is closed with the presentation of the classes the subject class is disjoint with. In general each element is realized as one complex sentence.

The introduction starts off with information about *what kind of thing* the class is. This is realized by introducing the messages presenting the immediate superclasses of the class. To set the class apart from the superclasses the introduction is enriched with as much additional information as possible and textually sensible. This information is linked as closely as possible to the superclass message. This is realized by adding messages that can be transformed into adjective modifiers to the reference to the subject class in the first sentence, and adding more information as a relative sentence. This results in sentences such as:

A grizzly bear is a large bear that lives only in North America.

[1] We are currently exploring if this approach should be extended to regular expressions instead of sequences.

This phrase consists of three distinct pieces of information from the ontology: the immediate superclass of the class "grizzly bear" and two restrictions for a property named "hasSize" (e.g. ∃ hasSize {Large}) and "livesIn" (e.g. ∀ livesIn NorthAmerica). The first restriction was chosen for this position because it can be expressed as an adjective. Whether and how a message can be transformed into an adjective is determined by the attributes of the relation type of the property of the restriction that is the source of the message. In this case, a manual annotator has decided that the values of the "hasSize" property can be alternatively be directly used as adjectives of the subject of the description instead of using the default realization of the *roleplaying* relation. This decision can just as well be made heuristically in the automatic annotation generation process. The criterion here would be that the word "Size" that specifies the role played by the range object refers to an immediate quality of the class. Other candidates for a class of such words are "Color" or "Gender". However there exists a great number of properties that fit the *roleplaying* pattern for which such a transformation would not be appropriate. Examples include the properties "hasParents" or "hasMaker". In these properties the role refers to an object external to the class rather than to an immediate quality of it.

The rest of the available information is ordered into groups according to the property (**property groups**) that is restricted by the restriction that is contained in the message. This produces groups of messages that all pertain to the same property. Those property groups are the first step towards text sections that deal with one particular attribute of the class that is described through restrictions on each property addressed.

4 Microplanning

In the next step, microplanning is performed to derive complete text specifications. Most of the structuring that is left to be done is performed in the property groups and is linked with microplanning operations such as aggregation and is therefore performed at this stage.

Depending on the types of the restrictions in the messages, rhetorical structures are formed inside each group. Figure 1 gives an overview of possible structures inside a group. The boxes represent complexes of messages based on groups of restrictions. The names refer to the names for restriction types used in the Manchester Syntax for OWL, with CARD summarizing all cardinality restrictions. The labels on the arcs represent rhetorical relations that connect the complexes.

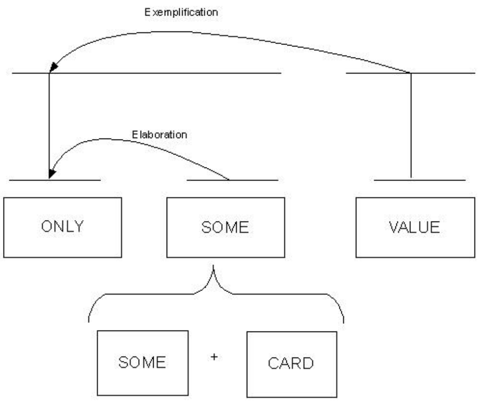

Figure 1: Structure inside groups

The SOME restrictions and CARD restrictions can be combined, since both make statements about the positive existence of objects. This combination is linked to the ONLY restrictions via an elaboration. VALUE restrictions finally can be connected to this complex via an exemplification relation since they make a statement about concrete objects as opposed to the statements about possible objects made by the other restrictions.

An example for a statement generated from a moderately complex structure containing an ONLY restriction and an EXACTLY restriction would be this sentence:

A gizzly bear has only bears as parents and it has exactly two bears as parents.

The semantic content behind this sentence is a group of messages concerning the property "hasParent", that contains messages derived from the restrictions ∀ *parent bear* and = *hasParent 2*. Figure 2 presents the structure that is formed inside the group. The SOME block formed from the cardinality restrictions and the SOME restrictions which are not present in this example. The resulting block is then connected to the ONLY block. It should be noted that the ONLY restriction is exploited to determine the term that is used to lexicalize the range object of the message from the cardinality restriction, since the restrictions given through it are normally more specific than the normal range defined for the property.

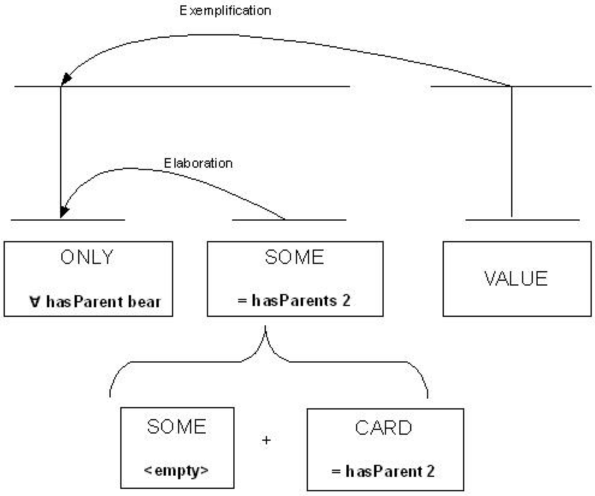

Figure 2: Example of structure inside a group in action

The task of Referring Expression Generation stage in this system currently only makes sure an appropiate pronoun is used in subsequent reference to the subject of the description. In general the neutral pronoun “it” is used, unless a restriction is found that can be interpreted as an information about the gender of the class.

A complete description text for the concept of a grizzly bear taking reference expressions into account may be:

> *A grizzly bear is a large bear that lives only in north america. It has only bears as parents and it has exactly two bears as parents. A grizzly bear can not be an ice bear or a brown bear.*

The first sentence is the introduction of the description. The second sentence is the realization of the property group of the property “hasParent”. The last sentence finally presents the classes the subject class is disjoint with and closes the description.

Surface Generation is performed by the KPML language generation system (Bateman, 1997). The structural relations of the text plan, the linguistic relations inside the messages and the representations of classes are enriched with SPL plan fragments that combine to form a complete specification for a text. The type of a restriction is realized as a modification of the message.

5 Conclusion

The system generates sensible texts for a number of classes in a number of ontologies. The proposed schema for the structure of the text appears to produce natural sounding introductions to the text as well a sensible organization for the remaining bulk of the information. We are not aware of a system that performs the same task to the same degree without relying on more domain specific resources. The system does not and can not cover all imaginable ontologies. Problems especially arise from complex class definitions that contain nested class definitions, since they can require quite complex linguistic structures. For evaluation, testers familiar with the OWL formalism will be asked to judge whether the produced texts accurately represent the specified information, and whether the texts appear natural.

The structure-based annotation mechanism profits from well organized approaches to naming classes and properties, but runs into problems if names cannot be fitted into the expected patterns. In this case, the generated annotations have to be checked manually and need to be corrected. If formal patterns like simple grammars for naming can be agreed upon during the design of the ontology, these patterns can be exploited directly to generate annotations. This might be worth considering as a step in ontology development.

Acknowledgements

The author would like to thank John Bateman for his input to the work and his help with this paper, and John Kelleher for his reviewing and comments.

References

Dimitrios Galanis and Ion Androutsopoulos 2007. *Generating Multilingual Personalized Descriptions from OWL Ontologies on the Semantic Web: the NaturalOWL System*

Xiantang Sun and Chris Mellish 2006. *Domain Independent Sentence Generation from RDF Representations for the Semantic Web*

Daniel Hewlett and Aditya Kalyanpur and Vladimir Kolovski and Christian Halaschek-Wiener 2005. *Effective NL Paraphrasing of Ontologies on the Semantic Web.*

Ehud Reiter and Robert Dale 2000. *Building natural language generation systems.* Cambridge Press.

Bateman, J. A. 1997. *Enabling technology for multilingual natural language generation: the KPML development environment* Journal of Natural Language Engineering 3(1)