Editing OWL through generated CNL

Conference or Workshop Item

Editing OWL through generated CNL

Richard Power[1], Robert Stevens[2], Donia Scott[1], and Alan Rector[2]

[1] Department of Computing
The Open University
Milton Keynes, MK7 6AA, U.K.
r.power@open.ac.uk
[2] School of Computer Science
The University of Manchester
Manchester, M13 9PL, UK

Abstract. Traditionally, Controlled Natural Languages (CNLs) are designed either to avoid ambiguity for human readers, or to facilitate automatic semantic analysis, so that texts can be transcoded to a knowledge representation language. CNLs of the second kind have recently been adapted to the requirements of knowledge formation in OWL for the Semantic Web. We suggest in this paper a variant approach based on automatic *generation* of texts in CNL (as opposed to automatic analysis), and argue that this provides the best of both worlds, allowing us to pursue human readability in addition to a precise mapping from texts to a formal language.

1 Introduction

Several research groups have proposed interfaces for the Semantic Web based on Controlled Languages [12, 13, 2, 8, 1]. These systems are designed for use by domain experts who wish to encode knowledge without having to work directly in formalisms like OWL and RDF, which are designed for computer processing and data exchange rather than for easy comprehension by people. After some initial training, authors type in sentences drawn from a restricted subset of English (or some other natural language); the input is parsed and transformed into statements in OWL or some other Semantic Web formalism.

Our purpose in this paper is to introduce an alternative approach in the same tradition. We agree with the research groups cited above that CNL-based interfaces for editing (or viewing) ontologies and other metadata on the Semantic Web are plausible solutions to an urgent problem. Our distinctive proposal is to avoid any reliance on automatic interpretation of human-authored text. The CNL texts representing the content of a knowledge base are produced not through human authoring but through automatic Natural Language Generation (NLG). This brings several immediate advantages: for instance, it eliminates any possibility of an interpretation error (by the system); it also eliminates any need to train users to adhere to the CNL grammar. However, the idea is at first sight paradoxical: if all texts are generated by the system, how does the human author convey the desired content?

Over the last decade, our group has pioneered a method by which this can be done. The key idea is that the author specifies content by direct manipulation of a generated 'feedback text' which expresses both the current content of the knowledge base *and the options for extending it*. By preserving the link between constituents of the feedback text and the formal representation of its meaning, the interface supports editing at the level of meaning, not text. The user can select spans linked to individual objects or events, and perform operations like classification, or assigning values to properties, presented through menus (again expressed in the CNL) which pop up when a span is selected. This method, which we have called WYSIWYM[3] or more generically Conceptual Authoring, has been employed in several application domains [10, 3, 5, 4] and shown through evaluation studies to be intuitive for subject-matter experts [6].

Two advantages of using NLG have already been mentioned — no interpretation errors, minimal user training. A more subtle advantage is that we can design the CNL with the sole purpose of favouring human understanding: we no longer have to meet requirements resulting from automatic interpretation or human authoring. An NLG system could easily support several CNL dialects, some based on other natural languages, allowing multilingual access to the same knowledge base. Different texts could be generated for editing and viewing, favouring precision and ease of manipulation in the first case, and fluency in the second.

However, to achieve these benefits we need to overcome a limitation of the Conceptual Authoring systems implemented so far. Except for one or two small experimental prototypes, they are all designed for editing ABoxes (asserted facts about individuals) using classes from a fixed TBox (definitions and general statements about classes)[4]. The purpose of this paper is to indicate how the theoretical model underlying Conceptual Authoring can be adapted to cover knowledge bases embracing TBox as well as ABox [11].

2 Editing a knowledge base

Conceptual Authoring was originally developed as a way of defining the input for multilingual NLG, in applications where content could be formally represented by an ABox [10]. The method depends on a model for systematically aligning a generated feedback text to a graph representing the ABox, with nodes denoting individuals and arcs denoting relations among individuals. Events and propositions are treated as individuals, so that for example the sentence 'a doctor arrived' might express an ABox fragment with two individuals, one belonging to the class *PastArrival* and one to the class *MedicalDoctor*, related by the property *hasAgent*:

$$PastArrival(e_1)\ \&\ MedicalDoctor(e_2)\ \&\ hasAgent(e_1, e_2)$$

[3] What You See Is What You Meant

[4] Methods have been proposed for making WYSIWYM open-ended by allowing users to add new terms for atomic classes [7]; these need to be extended in order to support full TBox editing with open properties and complex classes.

Editing begins by offering the user a place-holder text indicating a constraint on the individual that embraces the whole sentence (e.g., it must be an event or situation). When this span is clicked, a menu pops up offering a list of options computed from the specific event/situation classes in the TBox, also expressed in CNL. Typically these options will include further place-holder texts constraining the participants in the event. Here is a typical editing sequence:

User action	**Feedback text**
Choose *New Event*	[Something happened]
Choose from pop-up menu	[Someone] shaved [someone]
Choose from pop-up menu	A barber shaved [someone]
Copy the barber individual	*A barber* shaved [someone]
Paste on to place-holder	A barber shaved himself

The important point to understand here is that all editing operations are defined on the ABox graph, not on the text, which is used only as a means of *presenting* the ABox and the options for updating it. The editing process depends on a model for aligning a text to a graph, in which text spans are mapped to nodes, and span-subspan relations are mapped to arcs.

Can a similar process be devised for editing a knowledge base embracing ABox and TBox? The simplest approach, in our view, is to adopt an alignment model in which sentences denote axioms, and major sentential constituents denote classes; editing can then be performed by selecting any span denoting a class, and replacing it by a span denoting another class, chosen as before from a menu of options generated by the system. ABox assertions can be incorporated into this model by treating individuals as enumerated classes containing a single element [11]. As an illustration, consider a description logic with the following resources for constructing classes and axioms:

Description	**Syntax**	**OWL**
atomic class	$AName$	AName
universal class	$\top$	Thing
enumerated class	$\{a\}$	oneOf(a)
exists restriction	$\exists R.C$	someValuesFrom(R,C)
class inclusion	$C \sqsubseteq D$	subClassOf(C,D)

The author is allowed to add new terms to the knowledge base (individual, class or property names), and must link them to words (drawn from a wide-coverage lexicon) of a syntactic category constrained by the CNL. In a very simple CNL, individuals could be expressed by proper names, classes by count nouns, and properties by transitive verbs. A fixed CNL grammar for expressing the description language takes care of the rest. A knowledge base comprises a list of axioms; when the user invokes *New axiom*, a trivial axiom $\top \sqsubseteq \top$ is added, and can be edited by substitution operations on the classes. Here is an example from the People+Pets domain [9]:

Logical form	**Feedback text**
$\top \sqsubseteq \top$	Everything is a thing
$P \sqsubseteq \top$	Every pet is a thing
$P \sqsubseteq A$	Every pet is an animal

Obviously P and A here are arbitrary labels that would be replaced in a Semantic Web application by URIs meeting the W3C namespace conventions. Here is a more complex sequence using both individuals and properties:

Logical form	**Feedback text**
$\top \sqsubseteq \top$	Everything is a thing
$\{m\} \sqsubseteq \top$	Mary is a thing
$\{m\} \sqsubseteq \exists R.\top$	Mary owns one or more things
$\{m\} \sqsubseteq \exists R.P$	Mary owns one or more pets

In these sequences, transitions from one line to the next are made by clicking on a word/phrase representing a class, and choosing from a menu of permissible substitutions. For instance, starting from 'Everything is a thing', the user might click on 'Everything' and then choose 'Every pet' from the following pop-up menu:

Mary
Every animal
Every dog
Every pet
Everything that likes one or more things
Everything that owns one or more things
...

This method raises an obvious problem of scale: for any non-trivial ontology, classes will have to be selected from thousands of alternatives, and some kind of search mechanism will therefore be needed. One solution already used in WYSIWYM applications [3, 6, 5] is a menu equipped with a text field through which users can narrow the focus by typing in some characters from the desired word or phrase. In an ontology editor this search mechanism could be enhanced by using the ontology itself in order to pick options that are conceptual rather than orthographic neighbours — for instance on typing in 'dog' the user would obtain a focussed list containing 'poodle' and 'pekingese' as well as 'doggerel'.

3 Conclusion

We have described a new approach to editing hybrid knowledge bases using controlled natural language. The novelty of this approach is that it relies entirely on natural language *generation*, thereby avoiding possible pitfalls of interpretation-based methods (e.g., interpretation errors, training effort) as well as bringing potential benefits (e.g., multilinguality, more flexibility in designing the CNL). However, our proposals have not yet been developed in detail or applied and evaluated for large-scale ontologies[5]. It remains to be seen (a) whether the proposed editing method is intuitive for users, (b) whether users are able (and willing) to define the necessary lexical resources during ontology building, (c) whether such an

[5] These tasks are being addressed in SWAT (Semantic Web Authoring Tool), a joint project between the Open University and the University of Manchester.

editing tool will scale up to large ontologies and expressive DLs like OWL Full, and (d) whether the differences between interpretation-based and generation-based approaches are important or merely an implementation detail.

References

1. Raffaella Bernardi, Diego Calvanese, and Camilo Thorne. Lite natural language. In *7th Int. Workshop on Computational Semantics (IWCS-7)*, 2007.
2. A. Bernstein and E. Kaufmann. GINO – a guided input natural language ontology editor. In *Proceedings of the 5th International Semantic Web Conference*, Athens, Georgia, 2006.
3. Nadjet Bouayad-Agha, Richard Power, Donia Scott, and Anja Belz. PILLS: Multilingual generation of medical information documents with overlapping content. In *Proceedings of the Third International Conference on Language Resoures and Evaluation (LREC 2002)*, pages 2111–2114, Las Palmas, 2002.
4. Paolo Dongilli. Discourse Planning Strategies for Complex Concept Descriptions. In *Proceedings of the 7th International Symposium on Natural Language Processing*, Pattaya, Chonburi, Thailand, 2007.
5. R. Evans, P. Piwek, L. Cahill, and N. Tipper. Natural Language Processing in CLIME, a Multilingual Legal Advisory System. *Journal of Natural Language Engineering*, 14(1):101–132, 2008.
6. Catalina Hallett, Donia Scott, and Richard Power. Composing queries through conceptual authoring. *Computational Linguistics*, 33(1):105–133, 2007.
7. F. Hielkema, C. Mellish, and P. Edwards. Using WYSIWYM to create an open-ended interface for the semantic grid. In *Proceedings of the 11th European Workshop on Natural Language Generation*, Schloss Dagstuhl, 2007.
8. K. Kaljurand and N. Fuchs. Verbalizing OWL in Attempto Controlled English. In *Proceedings of OWL: Experiences and Directions*, Innsbruck, Austria, 2007.
9. Qing Lu and V. Haarslev. OntoKBEval : A Support Tool for DL-based Evaluation of OWL Ontologies. In *Proceedings of the 2006 International Workshop on OWL: Experiences and Directions* , Georgia, USA, 2006.
10. R. Power and D. Scott. Multilingual authoring using feedback texts. In *Proceedings of the 17th International Conference on Computational Linguistics and 36th Annual Meeting of the Association for Computational Linguistics*, pages 1053–1059, Montreal, Canada, 1998.
11. Andrea Schaerf. Reasoning with individuals in concept languages. *Data and Knowledge Engineering*, 13:141–176, 1994.
12. R. Schwitter and M. Tilbrook. Controlled natural language meets the semantic web. In *Proceedings of the Australasian Language Technology Workshop*, pages 55–62, Macquarie University, 2004.
13. C. Thompson, P. Pazandak, and H. Tennant. Talk to your semantic web. *IEEE Internet Computing*, 9(6):75–78, 2005.