Towards Dynamic Wordnet: Time Flow Hydra

Borislav Rizov
Sofia University "St. Kliment Ohridski"
bobyrizov@gmail.com

Tinko Tinchev
Sofia University "St. Kliment Ohridski"
tinko@fmi.uni-sofia.bg

Abstract

Hydra is a Wordnet management system where the Synsets from different languages live in a common relational structure (Kripke frame) with a user-frendly GUI for searching, editing and alignment of the objects from the different languages. The data is retrieved by means of a modal logic query language. Despite its many merits the system stores only the current state of the wordnet data. Wordnet editing and development opens questions for wordnet data, structure and its consistency over time. The new Time Flow Hydra uses a Dynamic wordnet model with a discrete time embeded where all the states of all the objects are stored and accessed simultaneously. This provides the ability to track the changes, to detect the desired and undesired results of the data evolution. For example, we can ask which objects 10 days ago had 2 hyponyms, and 5 days later have 3.

Keywords: wordnet, modal logic language, Kripke frame, Hydra

1 Introduction

The wordnets in the world are evolving and growing in number. A lot of applications for development and visualization of such databases were developed in the last decades and one of them was the system Hydra whose main advantage was the modal logic query language for wordnet. Several years ago we introduced a new web version of the system with a simple, fast and comfortable interface. It allows the visualization and editing of wordnets for several languages simultaneously and concurrently by many users by means of a mobile first web interface. The system also has the ability to clone / replicate data from other languages in the database, which facilitates and accelerates the development of new synonymous sets. This can also be used for linguistic comparisons of language features. One of the challenges in the wordnet world was the alignment of the databases developed for different languages. The concurrent work of different teams in different languages in a single environment could greatly facilitate this task and the overall the wordnet development. In this paper we are presenting a new dynamic model for wordnet, which guarantees the integrity of the data and all intermediate stages of its development. It also implies timely detection of data and structure inconsistency and this saves the very expensive human resources. The user has access to the data states in all the moments of its evolution at the same time. The user is also provided with a powerful modal query language with a new temporal modalities. Hydra prevents the loss of data even in the case of malicious user behaviour. The system has much better database model and the queries are processed much faster than in the previous versions, some of them are in orders of magnitudes faster.

2 Wordnet

Wordnet is a relational model (Koeva et al., 2004) of the language where the language concepts are represented as synonymous sets related to each other with over 20 semantic and lexical binary relations like hyperonymy, meronymy, antonymy and others. The main one is the super-subordinate relation hyperonymy (AKA is-a). It links the more general concepts like animal with its more specific ones like horse and bear. This creates a hierarchical structures of concepts (noun and verb) in the language.

3 Wordnet for many languages

Wordnet development started for English in Princeton (Miller et al., 1990) and then this idea was taken up for more than 40 languages. Most of these are developed or are still in development using the so called synchronous model where the hyperonymy

structure follows this of the Princeton WordNet. Using comon identifier or alignment mappings the synsets encoding similar concepts in the different languages are linked to each other. Such relation or identifier is called ILI - Inter lingual index. These large wordnet databases with this relational model proved to be very useful for many linguistic tasks but experience several important problems. Being developed by different teams using different software platforms, file formats, databases, etc. the Wordnet databases are stored and maintened separately. The alignment (ILI manintanace) is made periodically usually for particular language pairs and particular version of these wordnet databases. Collaborative Interlingual index was developed to help reduce the sparse ILI mapping problem, but it did not succeed much. Some of the main problems in the past are the language database separation and the inconsistent synset identifiers in the central Princeton WordNet database.

4 Static model for wordnet

In a fixed moment of time we have a family of synonymous sets (**synsets**) - the concepts in the language - interconnected by semantic relations like hyperonymy and meronymy. Diving in them we find some associated data like part of speech and the words that they are comprised of. We call a word in a particular synset **literal**. Keep in mind that a single word can be found in several synsets while the literals are unique (can be thought as $\langle synset, word/compound \rangle$ pairs). These literals are connected by lexical relations. In a wordnet database we also have some text data like sample usage, notes about some particular synset or literal features, etc. We call this data **notes**. The notes can be thought as $\langle synset/literal, text \rangle$ pairs.

5 Wordnet as a Kripke frame

Let's consider 3 types of objects - Synset, Literal, Note for the objects in wordnet databases. We define special binary relations to encode the relationships between them. In this way Literal relation connects a particular literal to its parent synset. Usage relation connects a note object with the synset that it is usage example of. We also have the usual relations such as hyperony, meronymy, antonymy, etc. We obtain a Kripke frame $\langle W, R \rangle$, where W is a three-sort universe and R is a set of binary relations between the objects in it. Such a frame naturally introduces a modal logic language, which we'll present in the next sections. Each object in wordnet can be considered as a feature structure with a fixed set ot features depending on its type. Thus the synsets are provided with these features: pos (part of speech), lang (language code), ili (common identifier for the same concept in the different languages). Literals have word and lemma features, while Notes have note (the text they represent). All of the 3 types have some common features like id (unique identifier for the static model), userId - the identifier of the user that made this object, etc.

6 Hydra

The wordnet database management system Hydra (Rizov, 2008) was created in 2006 in order to address the problems found in the development of BulNet - Bulgarian Wordnet. A new model for wordnet as a Kripke frame was introduced where all the linguistic data for the various languages (most importantly Princeton WN and BulNet) live in a single relational structure. Several years later, the system was developed as a modern SPA web application (Rizov and Dimitrova, 2016). The system is in production `http://dcl.bas.bg/bulnet` with wordnets for 22 languages. The data searching is made by means of a modal logic query language.

7 Dynamic model for wordnet

An ordinary static wordnet database is an incomplete instantaneous description of the language. There are synonymous sets in the languages that are not defined, some of the relations are not fully instantiated and the wordnet databases for the different languagages are in different stages of their development. There are also some specific concepts for particular languages. Over time, both the language and its wordnet representation change and evolve. During this evolution, we have a different state of wordnet at any given time. This raises questions about the consistency of the data and its structure defined by means of the binary relations in this time flow.

If we take the snapshots of wordnet in the static model, we get a set of Kripke frames. Let's supply each object in each frame with the timestamp of its frame. Now let's take the union of the resulting set of disjoint frames. We get a single Kripke frame with all the manifestations of all of the objects in the wordnet. Formally it is the set:

$$\{\langle W_t, R_t \rangle\}_{t \in T}$$

where T is the time model. We implement a discrete time model with the assumption that at most one object or relational pair can be changed in a single moment of time. We guarantee this in our implementation and we are making the assumption even stronger. Every change in the data causes the creation of a new moment in this discrete time model. In this way the points of the time are those moments in which a single object or relational instance is changed, created or deleted. Regarding the physical time, the state of the object in a moment in it is the state of this object in the nearest moment in model time preceding the physical moment. In a fixed moment of the model time we can collect all the objects from this and the previous moments taking only the nearest (last version) state of each object. In this way we obtain the static Kripke frame for this particular moment. The collection of all the versions of all the objects we call Dynamic wordnet model.

8 Query language

The construction of wordnet and its editing opens questions about the change of data and their structure (evolution) over time. One may be interested in the availability of certain properties of the data and the relations. He would like to easily detect problems when they occur, to easily correct them without returning to a state with many changes made by many users, as is the case with the use of a backup. For example, object 1 has changed in the past. Meanwhile 2 and 3 have been changed (corrected), 4 has been created. We detect a problem with object 1 and its relations - there is some inconsistency in the structure. With the dynamic model, you can trace the whole process and find out exactly when and why the problem occurred. We do it by means of a modal logic language for wordnet which was created for the early implementations of Hydra and further developed with addition of temporal modalities. The system works with so called model checking - for a given modal formula, the set of the objects in which the formula is true is returned.

8.1 Dynamic wordnet language

We define the modal formulae syntax and the corresponding semantics inductively.

8.1.1 Syntax

In our language we have:

- $\mathbf{N}$ - a set of individual constants (nominals) - in the system we use decimal numbers for them.
- $\mathbf{O}$ - a set of constants for the features in the objects and their values. They use the schema $type('value')$. For instance pos('n') is such constant.
- $\mathbf{R}$ - a set of relation symbols
- $\mathbf{TM}$ - a set of time modifiers

  We have 4 types of temporal modifiers - for a fixed timestamp (real time moment), fixed operation moment (model time moment), relative future and relative past like this:

  - t159737980000;
  - o1235;
  - f5;
  - p3;

Atomic Formulae: $\mathbf{AtomicFor}$

- $\bot$
- $\top$
- $\mathbf{N} \subseteq \mathbf{AtomicFor}$
- $\mathbf{O} \subseteq \mathbf{AtomicFor}$

Formulae: For

- $\mathbf{AtomicFor} \subseteq \mathbf{For}$.

Let q and r be fomulae (queries), R $\in \mathbf{R}$, $\mathbf{t} \in \mathbf{TM}$, then the following are formulae:

- !q
- q & r
- q | r
- q => r
- q <=> r
- <R>q
- [R]q
- $\ll \mathbf{t} \gg$q

We also use some relation modifiers, namely:

- ˜R - the reverse relation of R
- R+ - the transitive closure of R
- R* - the reflexive and transitive closure of R

8.2 Semantics

- A Time structure is $\langle T, t_c, < \rangle$, where $T \neq \phi$ is a finite set, $<$ is a linear ordering, t_c is $max_< T$ (the current moment)
- A Model of time is $\langle \langle T, t_c, < \rangle, m \rangle$, where $m : \mathbf{TM} \times T \to T$
- A static model (Kripke frame for a given moment t) is $\mathfrak{M}_t = \langle W_t, \mathcal{R}_t, V \rangle$, where $W_t \neq \phi$, $\mathcal{R}_t : \mathbf{R} \to \mathcal{P}(\mathbf{W_t} \times \mathbf{W_t})$, $V : \mathbf{N} \cup \mathbf{O} \to \mathcal{P}(\mathcal{W})$ and for $c \in \mathbf{N}$ $V(c)$ has at most 1 element.
- A dynamic model is $\mathcal{D} = \langle \{\mathfrak{M}_t\}_{t \in T}, \mathcal{T} \rangle$, where $\mathcal{T} = \langle \langle T, t_c, < \rangle, m \rangle$ is a model of time.

We define the **truth** of a formula in a object x in the Dynamic model $\mathcal{D}$ by induction on the formula construction:

- $\mathcal{D}, t, x \nVdash \bot$
- $\mathcal{D}, t, x \Vdash \top$
- $\mathcal{D}, t, x \Vdash c$ for $c \in \mathbf{N} \cup \mathbf{O}$ iff $x \in V_t(c)$

  Each object in the database has an identifier and it is a nominal (constant) in our language. A synset identifier is encoded so as to be portable and it depends only on ili (identifier comming from PWN), pos (part of speech code) and the language (code) of the synset. In the implemented system this semantic more concretely is:
  - $\mathcal{D}, t, x \Vdash$ $s iff x is a Synset
  - $\mathcal{D}, t, x \Vdash$ $l iff x is a Literal
  - $\mathcal{D}, t, x \Vdash$ $n iff x is a Note
  - $\mathcal{D}, t, x \Vdash$ type('value') iff x.type = value (for instance x.pos=n, so x is a noun synset)
- $\mathcal{D}, t, x \Vdash$!q iff $\mathcal{D}, t, x \nVdash$ q
- $\mathcal{D}, t, x \Vdash$ q & r iff $\mathcal{D}, t, x \Vdash$ q and $\mathcal{D}, t, x \Vdash$ r
- $\mathcal{D}, t, x \Vdash$ <R>q iff $\exists y (x \mathcal{R}_t(R) y \& \mathcal{D}, t, y \Vdash$ q)
- $\mathcal{D}, t, x \Vdash \ll \mathbf{t} \gg$ q iff $\mathcal{D}, m(\mathbf{t}, t), x \Vdash$ q
- We say that a formula is true in dynamic model at point x, denoted $\mathcal{D}, x \models$ q iff $\mathcal{D}, t_c, x \Vdash$ q

  For the sake of an example we'll use concrete natural numbers in the following:
- $\mathcal{D}, t, x \Vdash \ll$o1235$\gg$q iff $\mathcal{D}, t_0, x \Vdash$ q where $m(o1235, t) = t_0$.

  As mentioned before, every data modification creates a model time moment which is refered as an operation id and t_0.id=1235.
- $\mathcal{D}, t, x \Vdash \ll$t159737980000$\gg$q iff $\mathcal{D}, t_0, x \Vdash$ where t_0 is the nearest previous model moment to this timestamp
- $\mathcal{D}, t, x \Vdash \ll$p3$\gg$q iff $\mathcal{D}, t_0, x \Vdash$ where t_0 is the nearest previous model moment to the moment $t - 3$ days
- $\mathcal{D}, t, x \Vdash \ll$f5$\gg$q iff $\mathcal{D}, t_0, x \Vdash$ where t_0 is the nearest previous model moment to the moment $t + 5$ days

8.3 Query answering

A formula in the defined modal language is a query in Hydra. The result of such query q at a given time moment t is the set of the unique objects with respect to their ids such that their time is the most recent one which is prior to the time t. By default the time t is the current moment t_c when the query is executed. This moment t can be fixed to be some arbitrary moment by means of the GUI, we call this feature *Time Machine*.

9 Example queries

Let's see some useful queries.

- Find the noun synsets that are on top of hyperonymy hierarchy in English:

  ```
  pos('n') & [hypernym]⊥ & lang('en')
  ```
- Find the synsets that are exactly two levels below the top in the hyperonymy hierarchy:

  ```
  [hypernym][hypernym][hypernym] ⊥ &
  <hypernym><hypernym>⊤
  ```
- Find inconsistency between Bulgarian and English:

  ```
  <ili>(lang('en') & pos('n')
  & [hypernym][hypernym]⊥ &
  <hypernym>⊤)

  & lang('bg') & [hypernym]⊥
  ```
- Find the literals that before 3 days were presenting the word 'test' and 2 days later are not:

  ```
  <p3>(word('test') &
  !<f2>word('test'))
  ```

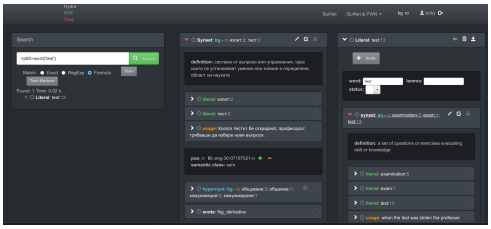

Figure 1: Hydra

10 Graphical user interface and implementation

Hydra is implemented in Javascript. It consists of a modern SPA (single page application) web app and a REST API service. The Wordnet data is stored in a Postgres database and the queries are translated to SQL queries. There is a preprocessing step where for each subformula its model time is determined. While most of the relations are stored in the database, some are implemented directly in the translations of the formula - such are the universal relation U and the transitive closures and reflexive and transitive closures of the other relations. Important feature is that no data can be lost during wordnet development even if there is some hostile user. Every change in the data creates a new copy of the object or relational instance touched with the same id but having the new data. For instance, when an object is deleted, a new record for it is created (operation record) where it is marked as 'deleted'. At any point the user can see the data as it used to be in the past with the so-called *Time Machine* - the user opens a dialog and selects a moment in the past and the data is as it used to be at this given time. The GUI is very simple and powerful. It has a Search Panel that has 3 modes for searching - using word, regular expression and a formula. The first 2 options find the synsets that have literals that match the provided word/regular expression. The latter is using the defined modal query language and it's much more powerful. There are 2 modes to visualize the data found. The first one (called 'Single') is visualizing the object selected from the list of the found items. The second one is aligning a pair of language wordnets that are present in the system. When the user selects a particular item from the list of the found items, the corresponding copies in the aligned languages are visualized. In this way the user can search some spanish word and see the aligned corresponding entries in french and English for example.

The visualization of an object consists of its static data (like pos and language for the synsets) and the relations - all the connected objects by all the relations. The view is recursive and the data for the related objects is visualized on demand. Hydra is also a fully-fledged editor for this wordnet data. A user with sufficient rights can put an object into edit mode, the representational controls are replaced with edit controls and he/she can edit and save the data. Relational pairs are added by means of a wizard. These changes are sent to all the other users by means of notifications.

11 Conclusion and future work

The main achievement of our work is the expantion of Hydra's capabilities with time operators and the most valuable among them is the feature that when something is wrong and damaged in wordnet, we can repair it easily, as well as to understand the cause and user responsible for this error. One weakness is that the use of the system in this form requires competence in logical languages. To overcome this we are developing an GUI assistant to help the linguists with predefined queries and schema queries. For more complex queries some skills would remain required of course.

12 Acknowledgments

We thank the anonymous reviewers for their careful reading and valuable suggestions which made our paper better and more comprehensible.

References

Svetla Koeva, Stoyan Mihov, and Tinko Tinchev. 2004. Bulgarian wordnet – structure and validation. *ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY*, 7(1-2).

George A. Miller, Richard Beckwith, Christiane. Fellbaum, Derek Gross, and Katherine Miller. 1990. Introduction to Wordnet: an on-line lexical database. *International journal of lexicography*, 3(4):235–244.

Borislav Rizov. 2008. Hydra: a modal logic tool for wordnet development, validation and exploration. In *Proceedings of the International Conference on Language Resources and Evaluation, LREC 2008*, Marrakech, Morocco.

Borislav Rizov and Tsvetana Dimitrova. 2016. Hydra for web: A browser for easy access to wordnets. In *Proceedings of the 8th Global WordNet Conference (GWC)*, pages 342–346, Bucharest, Romania. Global Wordnet Association.