Hydra: A Modal Logic Tool for Wordnet Development, Validation and Exploration

Borislav Rizov

Department of Computational Linguistics, Institute for Bulgarian Language, Bulgarian Academy of Science
52 Shipchenski prohod, building 17, Sofia 1113, Bulgaria
bobby@ibl.bas.bg

Abstract

This paper presents a multipurpose system for wordnet (WN) development, named Hydra. Hydra is an application for data editing and validation, as well as for data retrieval and synchronization between wordnets for different languages. The use of modal language for wordnet, the representation of wordnet as a relational database and the concurrent access are among its main advantages (Rizov, 2006).

1. Introduction

Concurrently with wordnets' development for a great variety of languages, several tools for WordNet have been designed such as Princeton viewer(Miller et al., 1990), WordNet viewer/editor VisDic[1], and lately DebVisDic. At the same time, with the development of the Bulgarian wordnet - BulNet [2], the need for a more powerful application for wordnet development enabling specific tasks related to the consistency and completeness of the wordnet database arose. Existing systems have grown inadequate in some very important respects: Application programing interface (API) for WordNet processing that uses abstract language independent of the data representation, multiple-user concurrent access, automatic consistency checks. Beside the above features Hydra supports other user requirements including optimization of visualization of data and relations between certain portions of data, as well as enhancement of editing, undo/redo functions, etc. The system has the following features:

- library for wordnet processing that includes:
  - search engine working with WN modal language. It supports regular expressions.
  - objects that represent entities in the wordnet structure such as synsets and literals.
  - objects representing the relations between the above entities. All these objects have the appropriate interface for the modification of a wordnet structure.
- innovative recursive view/editor
- tree view, which is an enhanced version of the tree view in VisDic
- multiple-user concurrent access for editing and browsing any number of monolingual wordnets
- verifications for data consistency during editing
- parametrized common lookups
- synchronization
- undo/redo of the users operations

Tasks other than editing are reduced to retrieving a set of objects that satisfy a certain property. Provided that a property is definable as a formula in the modal language described below, the system determines all the objects in the WordNet structure validating the formula, and hence the property. Hydra's language is based on the language presented in (Koeva et al., 2004)[3]. Hydra supports a synchronized view of and access to different wordnets through synsets encoding equivalent word senses. This provides exploitation of wordnet as a multilingual multipurpose lexicon, as well as an explanatory dictionary, a dictionary of synonyms, antonyms, hyperonyms, thematic dictionary, etc. In this respect Hydra may be regarded as a browsable and searchable lexicon user interface.

2. WordNet

WordNet is a lexical database that organises lexical information in terms of word meanings (Miller et al., 1990) grouped into sets of strong synonyms, called synsets. Thus, a synset encodes a lexicalized concept and each lexeme denoting the concept (called a literal) is a member of the synset. Each synset is supplied with a gloss (an explanatory definition), examples of the usage with real language sentences, and possibly with various notes on grammatical, semantic, pragmatic characteristics of the synset or of particular literals. The synsets are interlinked through various conceptual-semantic and lexical relations having different properties. Some of the most prominent are hyperonymy (relation between a superordinate and subordinate concept), meronymy (part-whole relation), antonymy (oppositeness relation), etc. Every synset is supplied with a unique identification key that is identical for equivalent synsets in all languages (Miller et al., 1990).

3. Modal Language for WordNet

The principal purpose of the modal WordNet language introduced here is to provide a clean and uniform formalism for expressing complex queries with sufficient expres-

[1] http://nlp.fi.muni.cz/projekty/visdic/
[2] http://dcl.bas.bg/wordnet_en.html
[3] With some modifications.

sive power for the most important tasks and validation procedures required in the development and exploration of a wordnet (such as searching, validation, synchronization, etc.) to be handled. Wordnet structure is represented as a relational database. The information retrieval and management is handled by means of SQL. Although it is a more powerful query language than the one described in this paper (as this modal language is defined in it) it has certain drawbacks as a front-end language. Its use is more complicated, the queries corresponding to the formulae being long and hard to write, even for short ones. The presented language is much easier to learn by common users than standard SQL. The following solution also dispenses with information retrieval procedures involving programming skills required by other types of representation.

Another advantage of the use of abstract language is that it allows unproblematic modification in the wordnet structure such as addition of new Relations or changes in the database architecture (with the corresponding translation of the formulas). It is also possible for another back-end to be used or the current one to be modified. All of the above may be performed without changing the language and the already expressed queries (formulae) and statements (e.g. ones defining wordnet consistency).

Besides, being a modal language, certain modal logic theoretical implications can be applied to it. A similar language was axiomatised and its completeness and soundness was proved(Koeva et al., 2004). First, we present the syntax of the language, and then proceed with a definition of a WordNet structure and the semantics of the language.

3.1. Syntax

Atomic formulae:

- Var – enumerable set of propositional variables.
- $\Sigma^{Literal}, \Sigma^{Note}, \Sigma^{Synset}$ – finite sets of nominals (constants).
- Sets of Boolean constants
  - $\{q^{Literal}, q^{Note}, q^{Synset}\}$
  - $B^{pos} = \left\{q_n^{pos}, q_v^{pos}, q_{adj}^{pos}, q_{adv}^{pos}, ...\right\}$
  - $B^{ili} = \{q^{ili}_{ENG20-06307086-n}, q^{ili}_{BUL-370295703}, \cdots\}$
  - $B^{def} = \left\{q_1^{def}, q_2^{def}, ...\right\}$
  - $B^{lang} = \left\{q_{bg}^{lang}, q_{en}^{lang}, ...\right\}$
  - $B^{bcs} = \left\{q_1^{bcs}, q_2^{bcs}, ...\right\}$
  - $B^{word} = \left\{q_{mouse}^{word}, q_{cat}^{word}, q_{person}^{word}, ...\right\}$
  - $B^{lemma} = \left\{q_{mouse}^{lemma}, q_{cat}^{lemma}, ...\right\}$
  - $B^{sense} = \{q_1^{sense}, q_2^{sense}, ...\}$
  - $B^{note} = \{q_1^{note}, q_2^{note}, ...\}$

Relational symbols:

Let $RS = \{ \equiv, R^{lnote}, R^{snote}, R^{usage}, R^{literal} R^{hypernym}, R^{holo\_part}, R^{holo\_member}, R^{holo\_portion}, R^{near\_antonym}, R^{be\_in\_state}, R^{category\_domain}, R^{similar\_to}, R^{also\_see}, R^{region\_domain}, R^{usage\_domain}, R^{derived}, R^{participle}, R^{eng\_drivative}, R^{subevent}, R^{verb\_group}, R^{causes}, R^{ili}, R^{bg\_derivative} \}$. $Rel = RS \bigcup \left\{R^{-1} \mid R \in RS\right\}$ is the set of relational symbols.

Formulae:

- The atomic formulae are formulae.
- If φ and ψ are formulae, then:

  $(\neg\varphi), (\varphi \vee \psi), (\varphi \wedge \psi), (\varphi \rightarrow \psi), (\varphi \leftrightarrow \psi)$ are formulae.
- If φ is formula and $R \in Rel$ is relational symbol, then:

  $([R]\,\varphi), (\langle R\rangle\,\varphi)$ are formulae.

3.2. Semantics

The semantics of the defined modal language is based on the classical Kripke semantics. **Kripke structure** is a tuple $\langle W, I\rangle$, where:

- I is the interpretation of the nominals, boolean constants and relational symbols, where:
  - $I(c) \in W$ for any nominal c.
  - $I(q) \subseteq W$ for any boolean constant q.
  - $I(R) \subseteq W \times W$ for any relational symbol R.

A valuation over the structure is $V : Var \longrightarrow 2^W$.

A Kripke structure is called **WordNet structure** if:

- $\left\{I\left(q^{Literal}\right), I\left(q^{Synset}\right), I\left(q^{Note}\right)\right\}$ is a partition of W
- $\{I(q) \mid q \in B\}$ is a partition of $I\left(q^{Synset}\right)$, when $B \in \left\{B^{ili}, B^{bcs}, B^{lang}, B^{def}\right\}$
- $\{I(q) \mid q \in B\}$ is a partition of $I\left(q^{Literal}\right)$, when $B \in \left\{B^{word}, B^{lemma}, B^{sense}\right\}$
- $\bigcup \left\{I(q) \mid q \in B^{Note}\right\} = I(q^{Note})$
- $I\left(R^{-1}\right) = I(R)^{-1}$
- $I\left(R^{LNote}\right) \subseteq I\left(q^{Literal}\right) \times I\left(q^{Note}\right)$
- $I\left(R^{SNote}\right) \subseteq I\left(q^{Synset}\right) \times I\left(q^{Note}\right)$
- $I\left(R^{Usage}\right) \subseteq I\left(q^{Synset}\right) \times I\left(q^{Note}\right)$
- $I\left(R^{Literal}\right) \subseteq I\left(q^{Synset}\right) \times I\left(q^{Literal}\right)$
- $I\left(R^{Literal}\right)^{-1}$, $I\left(R^{LNote}\right)^{-1}$, $I\left(R^{SNote}\right)^{-1}$ and $I\left(R^{Usage}\right)^{-1}$ are functions
- $I\left(R^{ili}\right) = \left\{I(q) \times I(q) \mid q \in B^{ili}\right\} \setminus \left\{I(q) \times I(q) \mid q \in B^{lang}\right\}$
- $I(\equiv) = I\left(R^{Literal^{-1}}\right) \circ I\left(R^{Literal}\right)$

Definition: We define the truth of a formula of WN language at point $x \in W$ over WordNet structure by induction on the formula construction:

- $x \Vdash c$ iff $x = I(c)$ for any nominal c
- $x \Vdash c$ iff $x \in I(c)$ for any boolean constant c
- $x \Vdash p$ iff $x \in V(p)$ for any $p \in Var$
- $x \Vdash (\neg\varphi)$ iff $x \nVdash \varphi$
- $x \Vdash (\varphi \vee \psi)$ iff $x \Vdash \varphi$ or $x \Vdash \psi$
- $x \Vdash (\varphi \wedge \psi)$ iff $x \Vdash \varphi$ and $x \Vdash \psi$
- $x \Vdash (\varphi \rightarrow \psi)$ iff $x \Vdash \varphi \Rightarrow x \Vdash \psi$
- $x \Vdash (\varphi \leftrightarrow \psi)$ iff $x \Vdash \varphi \Leftrightarrow x \Vdash \psi$
- $x \Vdash ([R]\varphi)$ iff $\forall y \in W\,(xI(R)y \Rightarrow y \Vdash \varphi)$, where $R \in Rel$
- $x \Vdash (\langle R\rangle\varphi)$ iff $\exists y \in W\,(xI(R)y$ and $y \Vdash \varphi)$, where $R \in Rel$

3.3. WN language in practice

The WN language defined above is used to perform simple and advanced queries in WN that are needed in wordnet exploration and validation, as well as in the work of annotators using wordnet, e.g. in word-sense disambiguation (WSD).

3.4. Example queries

- **Return all synsets which contain the word *cat***

  $$\langle R^{Literal}\rangle q_{cat}^{word}$$

  q_{cat}^{word} retrieves the Literal objects representing the word 'cat', then the modality $\langle R^{Literal}\rangle$ gives the Synsets which contain them.

- **Return all hypernyms of the synset identified by the nominal** q

  $$\left\langle R^{hypernym}\right\rangle q$$

  q is a name for an object in WordNet Structure. The modality returns all the synsets which are hyperonyms of the object.

- **Return all synsets which contain the word *cat* and their correspondences in Bulgarian.**

  $$\left\langle R^{Literal}\right\rangle q_{cat}^{word} \vee \left(q_{bg}^{lang} \wedge \left\langle R^{ili}\right\rangle \left\langle R^{Literal}\right\rangle q_{cat}^{word}\right)$$

  $\langle R^{ili}\rangle\langle R^{Literal}\rangle q_{cat}^{word}$ retrieves the synsets in other languages which are connected to the synsets containing the word 'cat'(see the previous example).

  q_{bg}^{lang} gives the synsets in the Bulgarian wordnet.

  $q_{bg}^{lang} \wedge \left\langle R^{ili}\right\rangle \left\langle R^{Literal}\right\rangle q_{cat}^{word}$) returns the intersection of the upper two sets.

  Finally the disjunction is interpreted as union of the sets returned by its members.

3.5. Representation of WordNet structure

Although there are other alternatives, the relational nature of WordNet and the necessity of fast concurrent access to large amount of data determine the choice of RDBMS. SQL is the background of the query system that uses the defined modal language. A modal formula φ is automatically translated into an equivalent SQL query ϕ. In other words, for any $x \in W$, $x \Vdash \varphi$ if and only if $\tilde{x}$ belongs to the results of the query ϕ evaluated in the database, where $\tilde{x}$ is the representation of x in the database.
The database is organized in such a way as to be self-explanatory. For example information about the binary relations in the wordnet representation is stored in the database and is used in visualization, editing and validation.
Hydra uses three types of WN objects.

— `Synset` (representing the synonym sets in a WordNet structure)

— `Literal` (representing the graphical words)

— `Note` (representing some text data in a WordNet structure as usage examples and explanatory notes)

We call these objects `linguistic units` (LU). The literals in a synset are in the relation 'literal' with it. Notes are found in a number of relations with Synsets and Literals, such as Usage, LNote, SNote. Every LU is associated with a single synset.

3.5.1. The tables of the relational database

SYNSET – the table representing the synsets.

LITERAL – the table representing the literals

NOTE – the table representing the notes

REL – the table storing the binary relations between LUs

There are several tables which make the database self explanatory.

3.6. Data retrieval

Hydra enables search in the WordNet database by means of formulae in the WN language. The engine returns all linguistic units at which the formula is true in the WordNet structure.
We define the translation of the formulae in SQL queries by induction.
The result of each query produced by formula is a table, containing the identifiers (id) of the LU at which the formula is true. The identifiers are natural numbers, belonging to three disjoint intervals, so that the type of the LU is recognizable by its identifier.

- c is nominal

  ```
  select c as id;
  ```

- For the constants $q^{Literal}, q^{Synset}, q^{Note}$ the corresponding queries are:

```
select id from Literal;
select id from Synset;
select id from Notes;
```

- Let $B \in \{B^{ili}, B^{pos}, B^{def}, B^{bcs}, B^{lang}\}$, $q^{type}_{value} \in B$. For q^{type}_{value} we have:

```
select id from Synset where type = value;
```

- Let $B \in \{B^{word}, B^{lemma}, B^{sense}\}$, $q^{type}_{value} \in B$. For q^{type}_{value} we have:

```
select id from Literal where type = value;
```

- Let $q^{type}_{value} \in B^{note}$. For q^{type}_{value} we have:

```
select id from Notes where type = value;
```

- Let VT be the table presenting the valuation of the variable x, then:

```
select id from VT;
```

- We retrieve the universe of the model using the following formula:

$$q^{Literal} \vee q^{Synset} \vee q^{Notes}$$

Let P, Q and R be the translations of the formulae φ, ψ and the relation R while F is the translation of the above formula (defining the universe). Then we have:

- $\neg\varphi$

```
select id from F
    where not exists
(select 1 from P where P.id = F.id)
```

- $\varphi \wedge \psi$

```
select P.id from Synset
    inner join Q
on P.id = Q.id;
```

- $\varphi \vee \psi$

```
P union Q;
```

- $\langle R \rangle \varphi$

```
select distinct Rel.id1 as id from Rel
inner join P
on Rel.id2 = P.id and rel = R;
```

$\varphi \rightarrow \psi, \varphi \leftrightarrow \psi, [R]\varphi$ are expressed by means of the already defined formulae.

4. Novel features of Hydra

4.1. Searching

Hydra has a Searcher with options for searching by formula, by literal and by literal using regular expression. With formulae, one may also use regular expressions when providing values for constants. Below is expressed the first of the example queries ($\langle R^{Literal} \rangle q^{word}_{cat}$).

```
<Literal>word('cat')
```

The following formula uses a regular expression[4] to find the synsets containing words *cat* or *cut* (the use of regular expression is marked with #):

```
<Literal>word(#'^c[au]t')
```

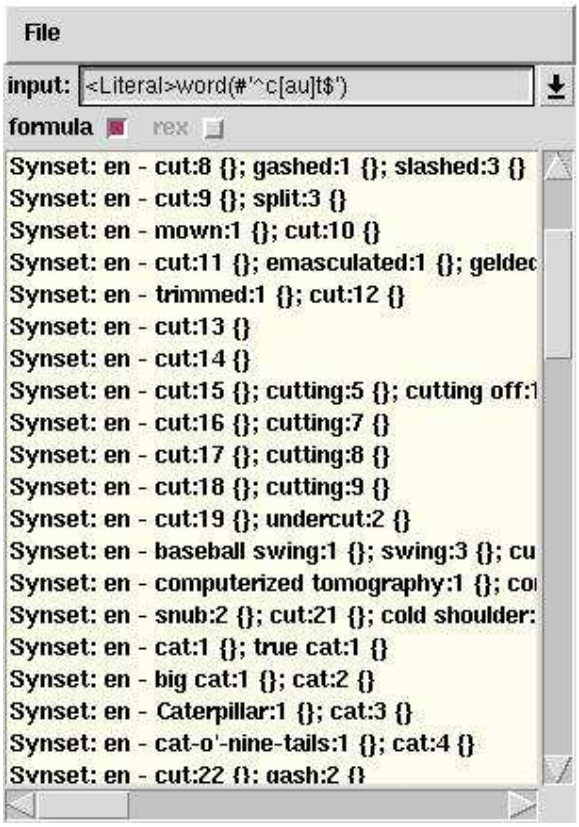

Figure 1: Search by a formula using regular expression

4.2. Visualization of WordNet

Hydra synchronizes different languages through the synsets encoding equivalent senses. Hydra has several views for the display of LUs in any language. The most sophisticated one – **MainView** provides edit functions over LUs, functions for addition and removal of relations, as well as creation and deletion of LUs and cloning of synsets from other languages available. Edit is performed in a user interface which uses both the modal language and SQL. Another important feature of the tool is the recursive presentation of the WordNet relational structure, visualized as a tree structure. Every node of the tree that represents a LU

[4]The system uses MySql regular expressions

is expandable (displaying its data and relations). The edges represent the relations between LUs and are named accordingly. This view has configurable 'look and feel' through an XML configuration file (where data and relation can be edited for specifying control type, order and color).
Another important view is the **TreeView** where relations are shown as tree structures. This view visualizes only acyclic relations. Let R be such relation. A successor of a node l in the tree is each neighbour LU – $\{x \mid lRx\}$. This view contains two columns. The tree on the right side shows LUs, while the left column displays the antecedents of the corresponding LU. If the antecedents of a LU are more than one, the antecedent to be used for the path upward can be selected by the user.
The **SimpleView** displays the characteristic data of a synset to be selected for visualization. The neighbours of the synset are shown, as well.

4.3. Undo/Redo

Hydra has a high quality system for managing user operations in a way that can be easily cancelled and redone.

4.4. Concurrent access

Undo/redo operations complicate the maintenance of concurrent access since different users may happen to edit simultaneously the same objects in the structure. Special strategies are used to take care of such cases. Consistency checks after user's modifications in the wordnet structure must be made. A database manager is not capable of coping with these problems alone. To this end, in addition to the manager Hydra uses locking of the currently edited LU and its neighbours, and other strategies.

5. Implementation

The tool is implemented in Python [5]. GUI is in Tkinter and Tix. As a platform-independent system, Hydra has been successfully tested under Linux and Windows. The RDBMS used in the implementation is MySQL (¿5.0). Almost all of the design patterns proposed by GoF (Gamma et al., 1995) are used in the implementation. The resulting system is a robust and extendable one. Hydra's data retrieval engine is used in several other applications. UTF-8 database encoding makes Hydra language-independent. The system has been tested successfully on Bulgarian, English and French.

6. Conclusion

As shown in the present paper, Hydra is an innovative system for managing (development, validation and exploration) WordNet database that uses modal logic. Hydra's development is still in progress. Future work will include exploitation of the propositional variables. One option is for variables to be evaluated over the result of a query. Another option is for them to be initialized with a value represented by a table (this table may be retrieved outside the system). A Web Interface GUI implementation is also considered.

Hydra is currently used at the Department of computational linguistics, IBL-BAS in the development of the Bulgarian wordnet.

7. Acknowledgements

I am indebted to my scientific advisor Assoc. Prof. PhD. Tinko Tinchev for the valuable help and comments in my work on Hydra's design and development, as well as in discussing versions of this paper.

8. References

E. Gamma, R. Helm, R. Johnson, and J. Vlissides. 1995. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.

S. Koeva, S. Mihov, and T. Tinchev. 2004. Bulgarian wordnet - structure and validation. *Romanian J. Of Inf. Sci. And Technology*, 7, No. 1-2:61–78.

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, No 4:235–244.

Borislav Rizov. 2006. Relational structures for wordnet. a modal approach. Master's thesis, Sofia University. In bulgarian.

[5] http://www.python.org

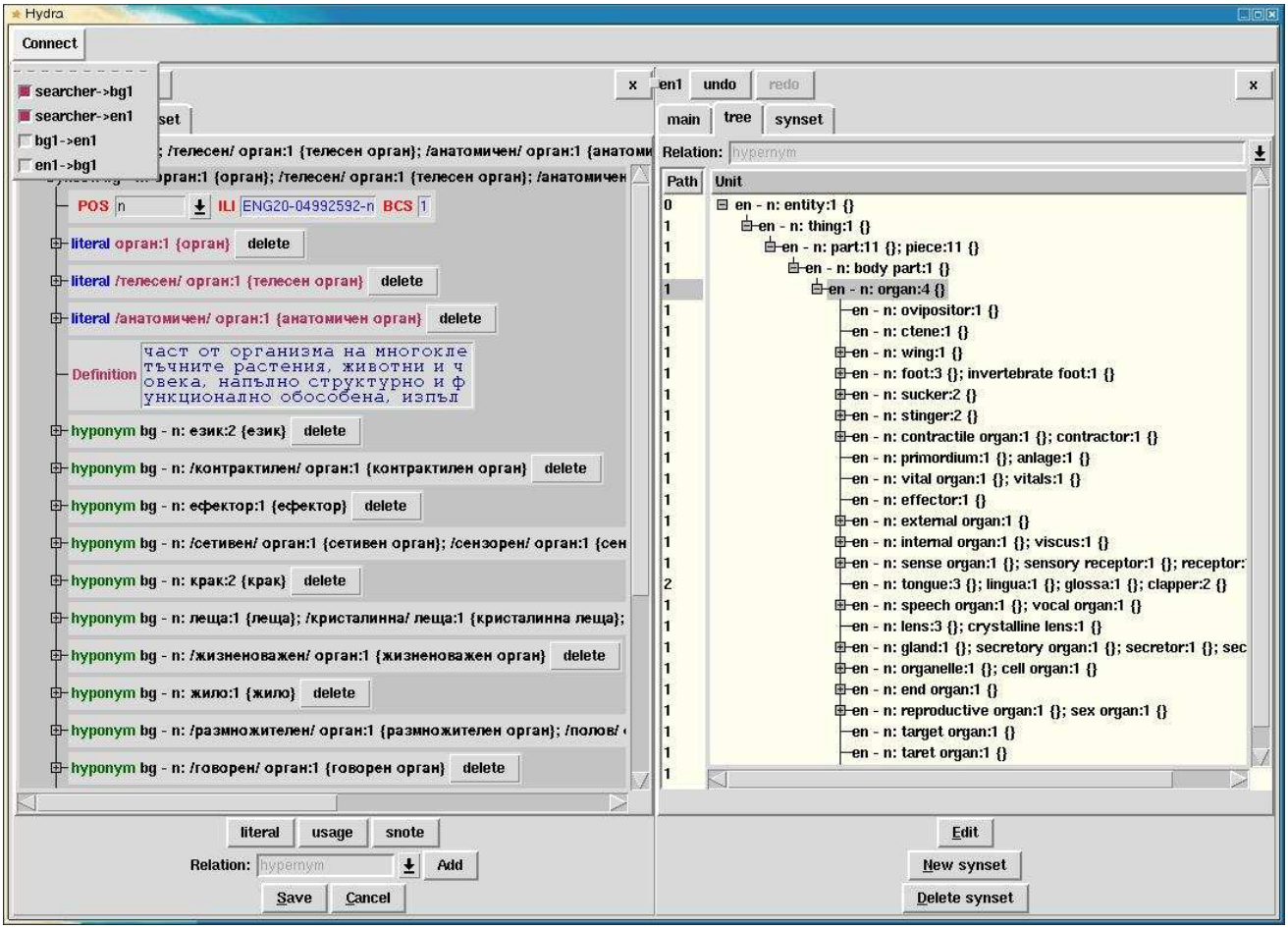

Figure 2: Hydra