Online Presentation of an Upper Ontology

Michal Ševčenko

Department of Computer Science
Czech Technical University in Prague
sevcenko@vc.cvut.cz

Abstract. This article presents the SUMO Browser—online tool that can be used for browsing the Suggested Upper Merged Ontology, SUMO, and its connection to the WordNet lexicon. The Browser facilitates the process of getting familiar with SUMO content. A brief introduction into SUMO and WordNet is also presented.

1 Introduction

New engineering tasks like information retrieval, natural language processing, knowledge representation, and data interoperability require new resources. Among important resources needed for designing knowledge systems belong *ontologies*—formal descriptions of the structure of knowledge bases. Special class of ontologies is formed by *upper ontologies*—domain-independent ontologies intended to be reused and extended for particular domain to form a *domain ontology*. Another important resources are *electronic lexicons*. Lexicons provide a bridge between the knowledge represented in knowledge systems, and natural language.

This article presents a short introduction into one particular upper ontology—the SUMO [1, 3], and one particular lexicon—the WordNet [2, 4]. A discussion about the usability of the WordNet lexicon in automated natural language processing tasks is included. The next section gives the idea of interconnecting SUMO and the WordNet. The last sections introduce the SUMO Browser [5]—an online tool for exposing the content of SUMO and the WordNet to the user in an appropriate form. Separate sections are dedicated to the paraphrasing feature of the Browser, and to sample application demonstrating programmatical access to the SUMO and WordNet content.

2 SUMO—the Suggested Upper Merged Ontology

SUMO is a collection of approximately 1000 well-defined and well-documented concepts, interconnected into semantic network and accompanied by a number of axioms. The concepts range from very general ones, such as Quantity, to very specific, such as Bird. The axioms mostly reflect common-sense notions that are generally recognized among the concepts. SUMO is intended as a domain-independent substrate for designing domain ontologies.

Axioms help to constraint interpretation of concepts, and provide guidelines for automated reasoning systems that process knowledge bases conforming to the SUMO ontology. An example of such an axiom is:

"If *c* is an instance of combustion, then there exist heating *h* and radiating light *l* so that both *h* and *l* are subprocesses of *c*".

This rather complicated, but logical, sentence says that a process of heating and a process of emitting light accompany each process of combustion (burning). Moreover, this axiom is coded in SUMO in a formal logical language.

Concepts in SUMO are organized into a single hierarchy rooted at Entity, representing the most general concept. The first two levels of the hierarchy are depicted in Figure 1. You can see, for example, that Entities are divided into physically existent stuff (Physical), and abstract, mentally represented stuff (Abstract). Physical things are further distinguished as objects and processes, etc.

Subclasses of a class are usually mutually exclusive, i.e. they do not share common instances. For example, nothing can be both an abstract and a physical, neither both an object and a process. This property is explicitly specified in SUMO. However, some classes can have multiple superclasses. For example, a Human is both Hominid (a member of certain class of animals) and a CognitiveAgent (an entity with the ability to reason).

One of the drawbacks of SUMO is its relatively low coverage that does not allow its employment for open-domain applications. It also lacks a connection between its concepts and natural language words. These limitations have been partially overcome by connecting SUMO to the WordNet lexicon.

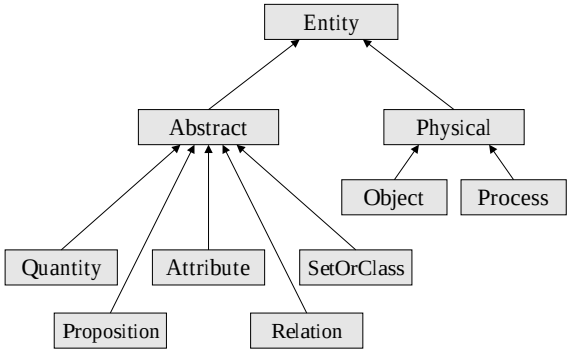

Fig. 1. Top-level concepts in SUMO

3 WordNet—an Online Lexical Database

WordNet is a freely available on-line lexicon. Linguists at the Princeton University have created it as a result of their psycholinguistic research. However, in the last decade WordNet proved to be very valuable resource for automated processing of natural language.

Technically, WordNet is an electronic thesaurus, defining large set of word meanings, interlinked with semantic pointers. The logical structure of WordNet is shown in Figure 2.

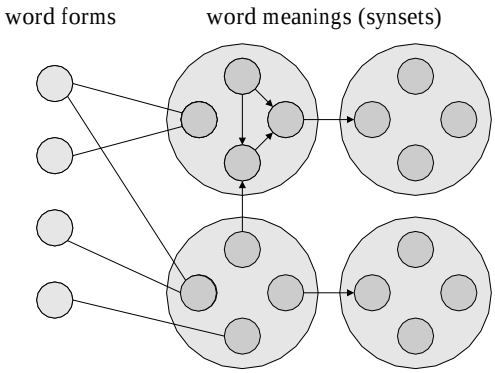

Fig. 2. The logical structure of WordNet

Word meanings are associated with word forms that can express them. We can see on the figure that the relation between word forms and word meanings is *m* to *n*—word form can have many meanings, and many word forms can refer to the same meaning. The former phenomenon is called polysemy, the latter is called synonymy. Dealing with such an ambiguity of natural language is the key challenge in automated natural language processing.

Each word meaning entry (also called synonym set, or *synset*), is accompanied with short informal definition (called *gloss*), and list of word forms that can represent the synset in spoken or written language. Synsets are kept separately for different parts of speech: there are databases of nouns, verbs, adjectives and adverbs. It should be noted that the semantic relations between synsets are different for different part of speech. For example, for nouns a chief relation between synsets is an is-a relation, well known from data modeling. In WordNet, this relation is called *hypernymy*/*hyponymy*.

It may seem at the first sight that synsets in WordNet built up a large semantic network, as we know it as a knowledge representation paradigm of artificial intelligence. However, a closer look reveals that semantic relations in WordNet are sometimes very vague and non-logical, and cannot be used for logical inference. The relations were coded by lexicographers, and were meant as a resemblance of humans' understanding of relations between word meanings. Moreover, due to the overwhelming size of the semantic network, the design rationale for semantic relations was rather local, without paying attention to the overall structure of the whole network.

Consider an example of the synset corresponding to the word blues. The WordNet defines blues as 'a type of folk song that originated among Black Americans at the beginning of the 20th century; has a melancholy sound from repeated use of blue notes'. The hypernymy hierarchy of this synset is shown in Figure 3.

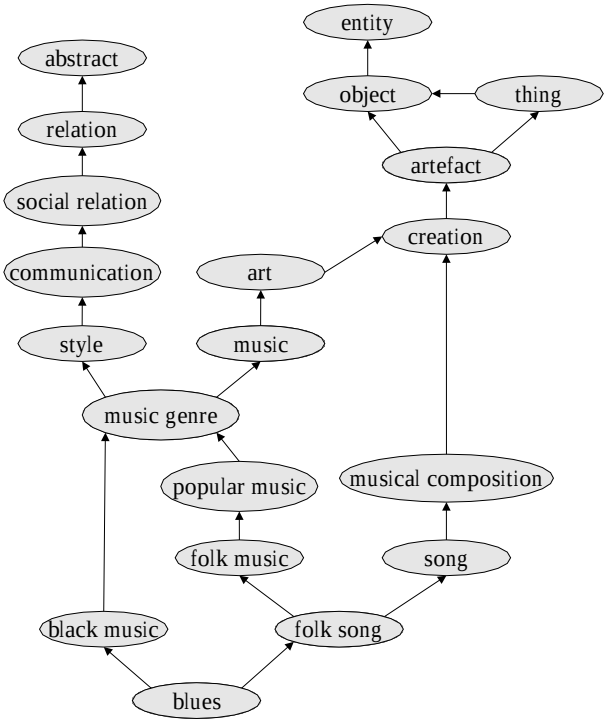

Fig. 3. The hypernymy hierarchy of synset blues

It can be seen from this figure that this small semantic network is quite ill formed, due to loose interpretation of concept meanings. For example, blues is both abstract (mental, nonexistent) and thing (physically existing). Similarly, the concept folk song has two senses: one sense denotes a class of songs which are folk, and is kind of song. However, the concept can be also understood as an attribute of songs, being kind of music genre. In common language these two concepts are usually not explicitly distinguished, and thus this distinction is not handled in WordNet. Similar situation occurs with multiple hypernyms of the concept music genre, where it is omitted the distinction between a process (music), and its role (social relation). Of source, there are more similar problems that complicate exploitation of WordNet as a resource for automated natural language processing.

It is interesting to ask whether these discrepancies are intrinsic to resources such as WordNet, or if they can be avoided by more careful design. The problem is that logic and linguistic rationales for organizing word meanings are quite different, especially for very general concepts, which are close to the root of the hierarchy.

4 Mapping WordNet to SUMO

It should be clear from the previous sections that both SUMO and WordNet address the similar problem, although with different focus. Both SUMO and WordNet define conceptualizations (simplifications) of our world. WordNet with the chief purpose to map these conceptualizations into natural language terms, and SUMO with the purpose to organize them into a logical structure. It thus makes sense to create a mapping between these two resources.

SUMO authors have developed such a mapping. The mapping enriches WordNet database files by tagging each synset with the corresponding SUMO concept. Moreover, a kind of relation between WordNet synset and SUMO concept is presented—the WordNet synset may be declared as equivalent to the SUMO term, as subsumed by it, or as an instance of it[1]. For example, the synset {*animal*, *beast*, *fauna*} is marked as equivalent to the SUMO concept Animal. The synset *scavenger*, however, is declared as subsumed by the concept Animal (if there were an equivalent class for this synset in SUMO, it would be a subclass of Animal). Similarly, the WordNet synset *Pythagoras* is marked as an instance of SUMO concept Human.

These mapping files allow mapping natural language words into SUMO terms, using WordNet synsets as an intermediate layer.

5 The SUMO Browser

Although SUMO authors spent considerable effort to make the structure of concepts and axioms transparent and unambiguous, it takes a while for a novice reader to get familiar with it and to understand its philosophy. To facilitate this process, an online tool, the SUMO Browser, has been developed. Using this tool, users may browse both SUMO and WordNet hierarchies, and navigate from a SUMO concept to the corresponding WordNet synsets and vice versa.

The Browser has been designed with the intention to present the content of SUMO in a user-friendly way, so that it is legible even for non-expert users. One of its unique features is natural language paraphrasing of axioms, which is described in more detail in section 6. Moreover, although SUMO implementation is internally just a list of axioms, the Browser understands and interprets some of the axioms and displays them in an appropriate way.

The Browser can display information about one concept (class) at a time. It displays the following information:

- the ontology section to which the class belong
- the list of immediate subclasses
- the list of all superclasses
- the list of instances
- the list of coordinate terms (other subclasses of the immediate superclass)
- equivalent WordNet synsets

[1] It should be noted that WordNet hypernymy/hyponymy relations cover subclass-superclass relations as well as class-instance relations. In SUMO, these relations are strictly distinguished.

– all related WordNet synsets (on separate page)
– list of relevant axioms, both in logic and in natural language paraphrase

An example of a sheet for concept Animal can be seen in Figure 4. Where appropriate, the contents of the concept information is converted into active hypertext link leading to another SUMO concept or WordNet synset.

One axiom relevant for the concept Animal is displayed on the sheet. As you can see, the axiom is presented both in natural language and in prefix logical notation.

Alternatively, the Browser allows navigating through the subclass–superclass hierarchy of concepts, similarly like file managers allow to navigate a hierarchy of folders on a disk drive. You can see an example of a hierarchy view in Figure 5.

The third way how to access the content of SUMO or WordNet is by looking up a particular concept or English word given a text. If an English word is searched, a list of corresponding WordNet synsets is displayed. These synsets can be navigated along WordNet semantic pointers, or a corresponding SUMO concept may be displayed.

Other online ontology browsers, such as Ontolingua [6], Chimæra [7], or the original SUMO browser, prefer simplicity and logical consistency of the user interface over its user-friendliness. They try to avoid using ad-hoc visualization patterns for different axioms. Our Browser, on the other hand, frequently employs such ad-hoc patterns. Doing so generally produces more legible output, but clearly requires more work during design and implementation of the Browser.

An example of ad-hoc visualization pattern is visualization of type restrictions of functions or relations. For instance, instead of displaying axioms

`(domain ListOrderFn 1 List)`, `(domain ListOrderFn 2 PositiveInteger)` and `(range ListOrderFn Entity)`

in their raw form, they are interpreted and displayed as a single expression

Entity **ListOrderFn**(List, PositiveInteger).

The source code of the Browser is freely available. This allows users to either install it on their computer, or to reuse the source code to access either SUMO or WordNet programmatically, as described in section 7.

6 English Paraphrasing of Axioms

This section briefly describes the natural language paraphrasing feature of the Browser. Although this feature merely transcripts logical expressions encoded in KIF[2] language [8] into English-like sentences, it may be of great help for users that are not familiar with particular KIF syntax. The paraphrases are also good feedback for axiom authors.

SUMO axioms are predicate logic expressions. These expressions are internally represented as trees, where internal nodes are logical conjunctions and quantifiers, and tree leaves are predicates. The paraphrasing algorithm is a straightforward recursive traversal, which applies single rule to each node to produce natural language text. For example, expression $E_1 \wedge E_2$ is translated into "$N(E_1)$ and $N(E_2)$", where $N(x)$ is a natural language representation of expression x, which is obtained applying the same algorithm. Besides these simple rules, the following techniques are used:

[2] KIF = Knowledge Interchange Format, the language in which SUMO ontology is encoded

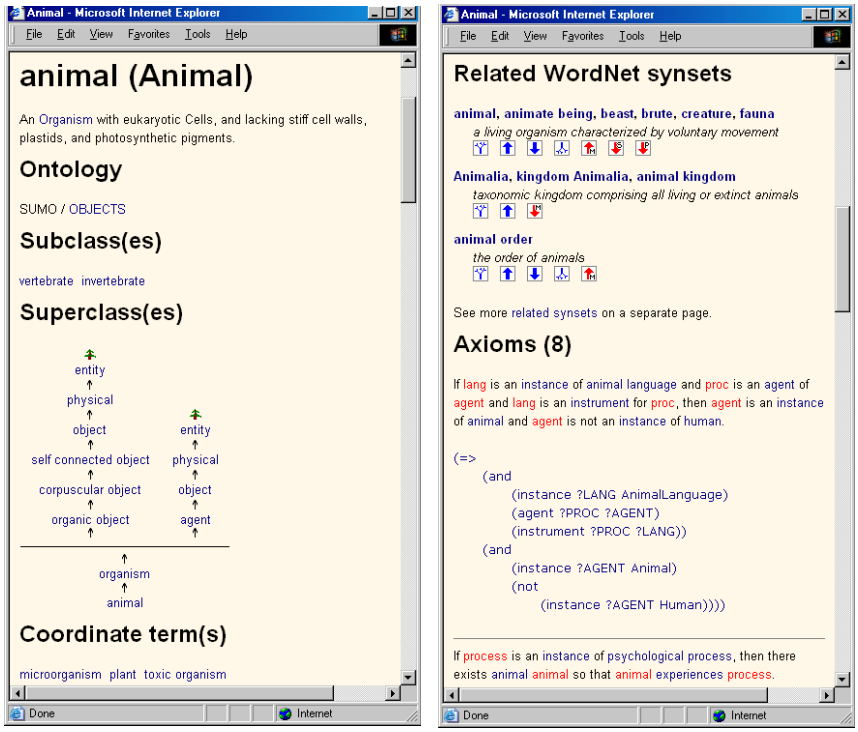

Fig. 4. Example of a SUMO browser sheet for the concept Animal

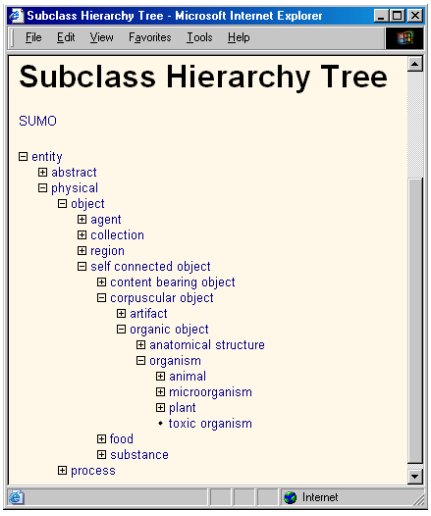

Fig. 5. Example of a hierarchy view

– If the input expression contains a negation in some internal node, the negation is always distributed to the leaf node using simple transformation rule; for example, expressions $\neg(A \wedge B)$ and $\neg(\forall x : P(x))$ are transformed into $\neg A \vee \neg B$ and $\exists x : \neg P(x)$, respectively. This improves the legibility of resulting English sentences.
– If the expression structure is too complex, and mere linear representation might lead to ambiguous interpretations, indented lists are used to structure the English paraphrase. For example, expression $(A \wedge B) \Rightarrow (C \vee D)$ is translated into
  - if A and B
  - then C or D

  This provides legible and unambiguous output even for very complex expressions.
– The following frequently occurring expression pattern:

  $\exists x, y : instance(x, Person) \wedge instance(y, Animal) \wedge pet(x, y)$

  is translated into more concise representation "There exists Person x and Animal y so that x has a pet y", rather than "There exists x and y so that x is an instance…".

The remaining issue is how to paraphrase leaf nodes of expressions, i.e. predicates and functions. The paraphrasing algorithm relies on the presence of special formatting strings that are embedded in the ontology and define the natural language output of predicates and functions. Special relation format associates each predicate or function with its formatting string. For example, axiom associating the predicate parent with its formatter reads "`(format parent "%2 is %n a &%parent of %1")`". During paraphrasing, %1 tag is replaced with natural language output of first predicate's argument, %2 with the second one, and %n tag with the word 'not' if the predicate is being rendered as negative, or with an empty string otherwise. The '&%' tag means that the word 'parent' is made into a hypertext link that leads to the SUMO concept parent.

This approach has been extended to support multilingual paraphrases. It is possible to associate more formatters to single predicate, each for different language. In cooperation with SUMO authors, formatters for five languages have been developed: English, German, Czech, Italian and Hindi.

7 Demonstration of SUMO Browser API

As it has been mentioned, the SUMO browser is distributed with its source code. A part of the source code are libraries that can access SUMO and WordNet files programmatically. This section presents a sample application that makes use of these libraries.

One of the problems with WordNet database is that it is very large, as it contains words from all areas of human interest. One might want to get rid of concepts from uninteresting domains to make the database more manageable, by means of pruning the WordNet hierarchy. The SUMO-WordNet mapping is an excellent resource for such a task, and SUMO Browser API enables us to do it in minutes.

Let's assume that we are not interested in biological concepts in WordNet. We might try to prune all synsets that are associated with SUMO concept Organism or one of its subclasses. The code fragment in Figure. 7 demonstrates how to do this procedure using SUMO Browser API.

```
01  // wordnet noun synsets
02  set<SynSet> synsets;
03  // SUMO ontology
04  OntologyInfo ontology;
05
06  // read wordnet noun database (special version annotated with SUMO concepts)
07  SynSet::ReadDataFile(synsets, "noun.dat");
08  // read SUMO ontology
09  ontology.ParseOntology("merge.txt");
10
11  set<SynSet>::iterator i, next;
12  // go through all wordnet synsets
13  i = synsets.begin();
14  while (i != synsets.end())
15  {
16      next = i; ++next;
17      // prune flag
18      bool prune = false;
19      // go through all SUMO terms associated with the current synset
20      for (list<SynSet::SumoTerm>::iterator k = i->SumoTerms.begin();
21          !prune && k != i->SumoTerms.end(); ++k)
22      {
23          // is the current associated SUMO term a subclass of "Organism"?
24          if (ontology.IsSubclassOf(k->Concept, "Organism"))
25              // if so, mark it for pruning
26              prune = true;
27      }
28      // erase synset if prune flag is set
29      if (prune)
30          synsets.erase(i);
31      i = next;
32  }
33
34  // write remaining synsets back to the database file
35  SynSet::WriteDataFile(synsets, "noun.dat");
```

Fig. 6. C++ Code fragment for pruning WordNet noun database

Code at lines 1–9 loads SUMO ontology and WordNet noun database into memory. The main cycle at lines 14–32 considers each noun synset for pruning. The synset is pruned if any of its associated SUMO concepts is a subclass of SUMO concept Organism (lines 20–27). Remaining synsets are written back to disk at line 35 in the WordNet native format.

8 Conclusions

SUMO, WordNet and the SUMO Browser have been presented in this article. SUMO is intended as a domain-independent substrate for designing domain ontologies. The WordNet lexicon provides a link between formal content expressed in SUMO and natural language. WordNet also proved to be useful in development of domain ontologies built up on the top of SUMO, as well as in testing the coverage of SUMO or SUMO compliant domain ontologies. The SUMO Browser facilitates these tasks by exposing the content of SUMO, WordNet, and eventual user-defined domain ontologies in a user-friendly way. One of its unique features is paraphrasing hard-to-read logical inscription of axioms into natural language. SUMO Browser source code contains libraries for accessing SUMO and WordNet files programmatically. They can be used for experimenting with these resources.

References

1. Pease, A., Niles, I., Li, J.: The Suggested Upper Merged Ontology: A Large Ontology for the Semantic Web and its Applications. To appear in Working Notes of the AAAI-2002 Workshop on Ontologies and the Semantic Web
   http://projects.teknowledge.com/AAAI-2002/Pease.ps
2. Miller, G.A., Beckwith, R., Fellbaum, C. Gross, D., Miller, K.J.: Introduction to WordNet: an on-line lexical database. In International Journal of Lexicography 3 (4), 1990, pp. 235 - 244.
   ftp://ftp.cogsci.princeton.edu/pub/wordnet/5papers.ps
3. SUMO home page. http://ontology.teknowledge.com
4. WordNet home page. http://www.cogsci.princeton.edu/~wn/
5. SUMO Browser home page. http://virtual.cvut.cz/kifb/en/
6. Farquhar, A., Fikes, R., Rice, J.: The Ontolingua Server: a Tool for Collaborative Ontology Construction; Proceedings of the Tenth Knowledge Acquisition for Knowledge-Based Systems Workshop; Banff, Canada, 1996.
7. McGuinness, D. L., Fikes, R., Rice, J., Wilder, S.: The Chimaera Ontology Environment. Proceedings of the 17th National Conference on Artificial Intelligence (AAAI 2000). Austin, Texas, 2000.
8. Genesereth, M.: Knowledge Interchange Format. Proceedings of the 2nd International Conference on the Principles of Knowledge Representation and Reasoning (KR-91), 1991