Chapter 7
The Lexical Bridge: A Methodology for Bridging the Semantic Gaps between a Natural Language and an Ontology

Kent D. Bimson, Richard D. Hull and Daniel Nieten

7.1 Introduction

Recently, a significant amount of research has been focused on extracting knowledge from natural language (NL) text and transforming it into an ontology-based semantic representation (Bimson 2012). The purpose of this research is to find ways to translate meaningful information in NL sources into a standardized, structured knowledge representation for the purposes of semantic normalization, integration, analysis, and reasoning.

However, a major obstacle to successfully translating NL meaning into ontology representations is that languages are semantically much more expressive than ontologies, resulting in significant meaning loss when translating NL semantics into ontology semantics. In Chapter 6 of this book, we characterize the kinds of meaning that get lost in the translation of NL semantics into ontology semantic structures, the reasons for that loss, and the impacts that these "semantic gaps" have on an ontology's representation of NL semantics.

The purpose of this chapter is to present a methodology that serves as a first step in spanning those semantic gaps, which we call "building a lexical bridge" (LB) between the NL and ontology representations of meaning. The goal of building the LB is to capture more of the meaning expressed in NL within an ontology. Our

K. D. Bimson (✉) · R. D. Hull
Intelligent Software Solutions, Inc., 5450 Tech Center Drive, Suite 400, Colorado Springs, CO 80919, USA
e-mail: kent.bimson@issinc.com

K. D. Bimson
Department of Electrical & Computer Engineering, The University of New Mexico, Albuquerque, NM, 87131-0001, USA

R. D. Hull
e-mail: richard.hull@issinc.com

D. Nieten
Red Lambda, Inc., 2180 FL-434, Longwood, FL 32779, USA
e-mail: dnieten@redlambda.com

M. Workman (ed.), *Semantic Web,* DOI 10.1007/978-3-319-16658-2_7

objective is to "lexicalize the ontology" by parsing ontology literals (i.e., class and property string names) into lexical items that can be used to generate an ontology lexicon (OL). The OL is used by our semantic equivalency algorithm (SEA) to compare the lexical meaning embedded in ontology literals to NL sources, in order to find synonymous and paraphrastic expressions in text. Together, the OL and SEA can be used for a number of high-value purposes, such as:

- Enriching the semantics of the ontology
- Improving semantic search of text sources based on the ontology
- Improving the results of ontology-based text extraction algorithms
- Enhancing our ability to compare the semantics of one ontology to that of another, and
- Identifying and eliminating redundant knowledge, such as synonymous Resource Description Framework (RDF) assertions

Each of these benefits is discussed in more detail in the *Potential Applications* section below.

7.2 Technical Approach: The LB

In this chapter, we present a method for *lexicalizing an ontology,* by which we mean *building a LB between an NL and an ontology*. The purpose of building a LB is to enhance the ability of ontologies to represent the lexical meaning hidden in class and property literals (or string names). By doing so, we prepare ontology constructs for word and phrase-based comparison with NLs.

Lexicalized Ontology Example

A simple example of "lexicalizing ontology constructs," would be translating the ontology property name:[1]

```
records-observed-results-of[1]
```

into its component NL lexical item stem forms:

```
[lexical item stems: record, observed, result, of]
```

and then marking them with their individual meanings, parts of speech and synonyms:

```
[lexical item: record
      (part of speech: transitive verb)
      (sense: set down in writing)
      (synonyms: write, pen, jot)]
```

A table-type representation of these constructs is illustrated in Fig. 7.1.

[1] Example from the approved *Joint Command, Control and Consultation Information Exchange Data Model (JC3IEDM)* triple [Action records-observed-results-of Action-Effect].

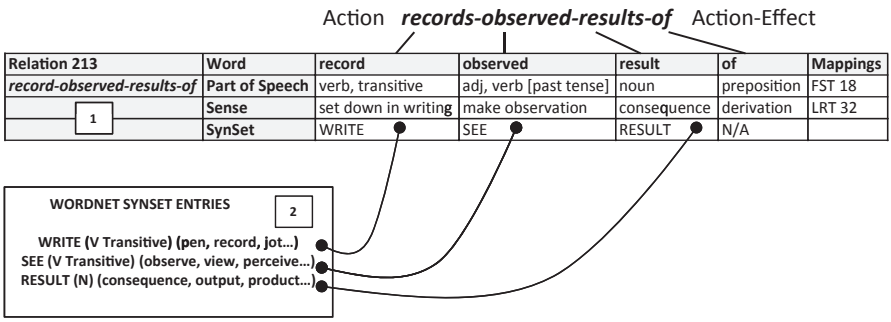

Fig. 7.1 The first step building a semantic bridge between the meaning expressed in a NL and an ontology is to lexicalize the ontology, parsing its class and property strings into constituent NL words, and then mapping the ontology lexicon to text. *NL* natural language

In order to build a LB, we need a way to transform ontology class and property literals (strings) into the NL words that modelers used to define these literals. Although NL-sounding string names are not needed to form unique Web Ontology Language (OWL) or RDF literals[2], the fact is that NL words and phrases are usually used by human modelers as the basis for these strings in order to make ontology class and property names more understandable to humans. This modeling approach can be exploited to our advantage in bridging the semantic gaps between an ontology and a NL.

Once the ontology string is lexicalized, each word extracted from the string can then be mapped to its sense (or meaning) and to its synonyms, using a thesaurus-style application—such as WordNet or FrameNet—each of which provides a database of English words and their synonym sets (also called synsets).

LB Components

A number of components are needed in developing the LB. These components, illustrated in Fig. 7.2, include:

1. *An ontology string parser*: The algorithms needed to parse ontology literals, or string names, into NL words in order to populate the OL.[3]
2. *A NL parser*: The algorithms needed to parse NL text into words, parts of speech, and senses (meanings).

[2] Literals can be any string in RDF and OWL, so *records-observed-results-of* could be represented as the string "x," as long as it is unique, though this is relatively useless for humans.

[3] As will be discussed later, this is not as straightforward as simply applying a NL parser to ontology strings, since the strings are often formatted differently (hyphenation in our example) and are often ungrammatical, in a NL sense, which leads to unsuccessful parsing results when using standard NL parsers.

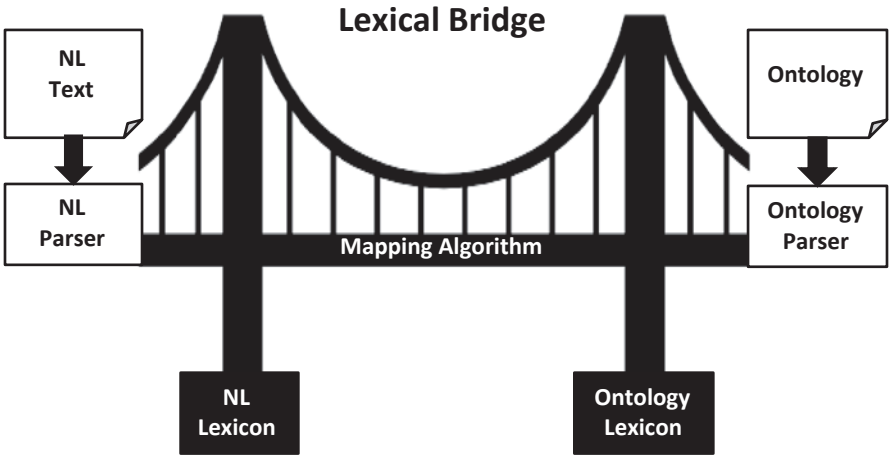

Fig. 7.2 The LB provides a semantic crosswalk from NL semantics to ontology semantics based on parsing ontology literals into NL words and creating an ontology lexicon. *LB* Lexical Bridge, *NL* natural language

3. *An ontology lexicon*: A lexicon of the NL words used to construct ontology literals, along with their intended parts of speech and ontology senses.
4. *A NL lexicon*: An online NL dictionary and thesaurus, providing access to synonyms for the OL.
5. *An ontology-to-NL mapping algorithm*: The algorithm for comparing words (and meanings) in ontology strings to NL words and phrases with similar meanings.

Once NL words (and their meanings, or "senses") have been extracted from ontology strings—and archived in the OL—they can be used as a basis for mapping the semantics of the ontology to the semantics of NL on a word-for-word basis. Our approach to this process is discussed below.

Building the LB

The first target application for our LB is to translate RDF triple class and property literals into NL words and senses. This application was developed on a project sponsored by the US Navy SPAWAR—called the RDF Find, Filter and Format (RDF-F3) project—under the Navy's Small Business Innovation Research (SBIR) Program.

The goal of RDF-F3 is to prevent redundant RDF triples—when extracted from text—from being asserted to the knowledge base. In order to do so, we must be able

to map the meaning of RDF literals in the knowledge base to the meaning in text from which new assertions will be extracted. The LB will accomplish this goal by translating RDF triple literals into NL words and phrases, allowing the meaning of the words embedded in RDF literals to be semantically compared to words from NL text. The technical objectives of the project were to:

1. Define RDF redundancy in a formal, semantic way
2. Develop the LB methodology
3. Design the LB architecture
4. Develop an LB prototype

Our accomplishments in each of these areas are discussed in the following sections.

Potential Applications of the LB

A lexicalized ontology provides a technical basis for significant improvements in ontology-based knowledge extraction from text sources. These improvements include, but are not limited to, the following potential applications, each of which contributes significant benefits to deployed semantic solutions.

1. *RDF redundancy identification and prevention*. The assertion of redundant RDF triples to a knowledge base creates excessive knowledge growth as well as disconnected graphs. The LB technology can be used to compare the meaning of RDF triple strings to NL words and phrases, identifying potentially redundant knowledge in text before it is asserted to the knowledge base. This use case is the primary objective of the RDF-F3 project and is discussed further below as the initial application of the LB.
2. *Improving semantic search in text*. The LB's NL words representation of RDF constructs can be used to find synonyms and paraphrases in text sources that are beyond the scope of current ontology-driven search engines.
3. *Learning new classes and properties*. By lexicalizing class and property names, we will be able to significantly improve the identification of semantically similar, but nevertheless different, phrases in text. These can be recommended as a "new ontology relation" or as a "new class" (as mentioned above). In other words, our architecture provides a solid foundation for "learning" new ontology constructs. This can be done much more effectively using the LB than with native RDF constructs, since it does word-based analysis and comparison.
4. *Bridging the semantic gap between ontology semantics and NL semantics*. Our research (Chapter 6) has shown that significant meaning is lost in transforming NL semantics into ontology structures. By adding a "lexicon," "thesaurus," and "paraphrase" data structures to ontologies, we provide a significant LB between the rich semantics of an NL and the simple semantics of an ontology.
5. *Improving service-based semantics*. By providing service-based access to an ontology's lexicalized structures, we expose the meaning within class and prop-

erty string names for exploitation by other applications and analysis algorithms, improving an external application's "understanding" of what the ontology really means.

6. *Semantic comparative analysis of ontologies*. It is often quite difficult to automatically compare the semantic content of one ontology to that of another ontology. This is because modelers use different words and phrases to create ontology class and property string names. By lexicalizing ontologies, we translate these string names into their component NL words and phrases, thereby improving inter-ontology comparative analysis based on NL semantics.

In the remainder of this chapter, we focus on applying our LB methodology and algorithms to RDF redundancy prevention as a first target application.

RDF Redundancy Definition

One of the keys to growing robust, lean, nonredundant knowledge bases is identifying text that is semantically equivalent with knowledge already in the triple store, as well as identifying new, ontology-relevant knowledge that should be asserted to the knowledge base. In other words, we must find ways to differentiate between redundant and nonredundant knowledge, using the ontology as a reference semantic data structure. The major challenge of this task is comparing the meaning of text words (TW) and phrases to the meaning of words embedded in ontology literals, the purpose for which the LB is being designed. For this reason, we are applying the LB methodology and technology to RDF redundancy prevention as a first target application.

The first step in addressing this challenge was to formally define RDF redundancy and to use that formal definition as a basis for developing the LB use cases, architecture, and prototype. For purposes of brevity, we only summarize the definition of RDF redundancy in this chapter. RDF redundancy must be defined along two axes: (1) graph equivalence, and (2) semantic equivalence. Standards, such as RDF Primer (2004) and RDF Semantic Web Standards (2004), focus on the former, wherein equivalent "meaning" is based on equivalent RDF graphs. Graph equivalence, however, does not fully address semantic equivalence, which is based on linguistic synonymy and paraphrase rather than intersecting nodes and edges. In lexicalizing the ontology, we provide a lexical basis for comparing the "intended lexical meaning" of ontology class and property names by parsing them into their constituent NL words. These words can then be compared for similar meaning, either within the same triple store (RDF to RDF), across different triple stores (RDF to RDF′) or between a triple store and an NL corpus (RDF to NL text). The meanings could be identical (same words) or equivalent (synonyms) or different. In the former case, the RDF is redundant. In the second case, the RDF may be redundant or it may add valuable information, as discussed above. In the last case, the semantics of each triple is different, representing new knowledge.

We define *semantic redundancy* in RDF triples on both a class and an instance level, as follows:

Formal Definitions of Redundancy:
Class Redundancy: An RDF triple T at the class level (T box), represented as <C1 Relation C2>, is redundant with respect to a specific knowledge base KB IFF there exists a triple T, in KB, represented as <C3 Relation' C4>, for which Relation is identical to Relation' , AND C1 is a synonym for C3, AND C2 is a synonym for C4, as defined by the KB lexicon, L.
Instance Redundancy: An RDF triple T at the individual level (A box), represented as <I1 Relation I2>, is redundant with respect to a specific knowledge base KB IFF there exists a triple T' in KB, represented as <I3 Relation' I4>, for which Relation is identical to Relation', AND I1 and I3 identify an equivalent real work referent, AND I2 and I4 identify an equivalent real world referent, as defined by the KB lexicon L.

These definitions add a NL (semantics-based) equivalence definition to World Wide Web Consortium's (W3C's) RDF (graph-based) definition, providing a lexical basis for identifying identical, equivalent, and (potentially) redundant RDF assertions in the knowledge base, as well as semantically equivalent NL statements. This semantic definition will improve our ability to identify and filter out NL equivalents before assertion of the RDF to the knowledge base. It also provides us with the formal foundation needed to develop the LB methodology, use cases and architecture.

LB Methodology Applied to RDF Redundancy Evaluation

Our methodology focuses on using the LB to prevent the assertion of redundant RDF triples to the knowledge base, particularly when RDF triples are extracted from NL text sources. The concept is to parse RDF ontology class and property names into NL lexical items, using the latter as a basis for comparing the meaning of the embedded words in RDF strings to words and phrases in text sources, identifying potential synonyms and paraphrases. A lexicalized RDF triple that means the same thing as an NL statement identifies that NL text as identical (redundant) or equivalent (potentially redundant) relative to the existing knowledge in the RDF triple store.

Our methodology, illustrated in Fig. 7.3, is a step-by-step process that the analyst will use for redundancy prevention. The methodology is based on both the graph-based and semantics-based definitions of RDF redundancy. In this process, we use the LB Parser to parse ontology literals, lexicalize them, and apply our lexico-semantic analysis to the parsed set of terms to determine lexical equivalency for the RDF triples. This information can then be used to determine redundancy with respect to the individual RDF and associated graphs.

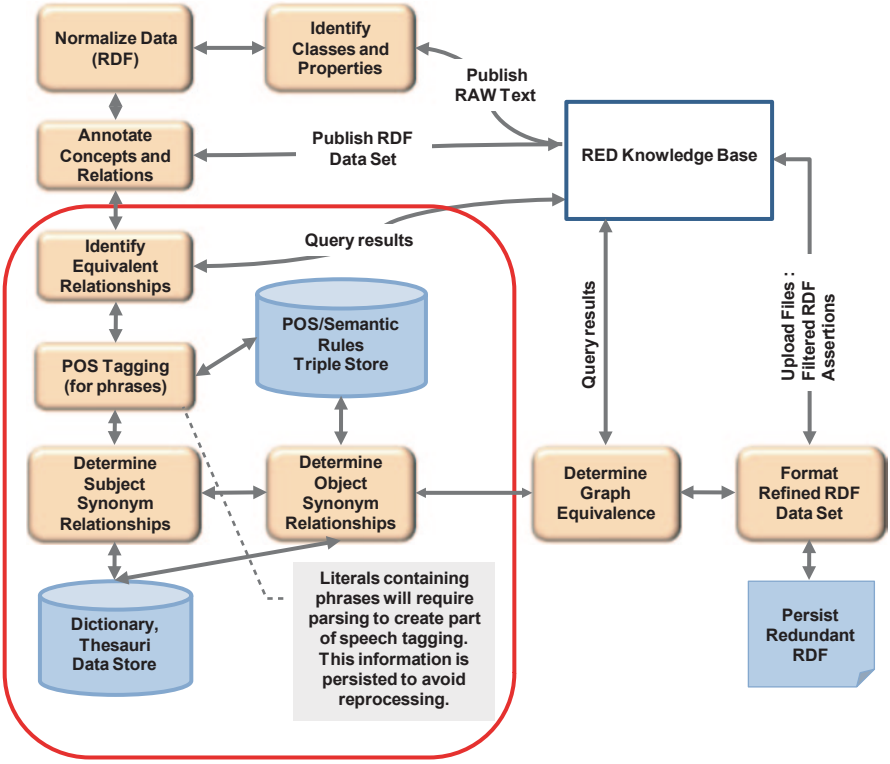

Fig. 7.3 LB methodology for lexicalizing RDF strings as a basis for semantic comparison with text sources with the goal of preventing the assertion of redundant RDF statements extracted from text. *LB* Lexical Bridge, *RDF* Resource Description Framework

The steps in our RDF redundancy analysis and prevention process are as follows.

Step 1 First, we apply the most straightforward criteria for determining RDF equivalency based on graph comparison, which involves leveraging OWL and RDFS constructs such as `owl:sameAs and owl:equivalentClass`.

Step 2 Second, we determine the lexical equivalence for class and property string names in one or more ontologies, beginning with the subject and object (classes) and ending with the verb (object property).

Step 3 Third, we determine the class or property literal equivalence, based on the W3C graph-based rules of equivalence.

Step 4 We then lexicalize literals and apply our rules of semantics-based equivalence as expressed in our lexical definition of RDF redundancy and our RDF equivalency algorithm, discussed below.

Methodology Data Products New phrase structure parse trees and rules are created during steps 2–4 and persisted, as are the subset of triples and associated graphs that have satisfied the equivalence criteria. At this point there are two artifacts of

interest. The first is the filtered data set resulting from the removal of *identical* RDF (provably redundant) and the second is the collection of RDF and associated graphs that have been identified as *equivalent RDF* (potentially redundant), based on both graph-based and semantic-based algorithms. These artifacts provide additional data sets for analysis and potential human vetting to confirm or reject an RDF triple and its associated graph as redundant. The vetted and/or non-vetted data sets can be published back to the knowledge base for use by any other RDF analysis tools.

LB Architecture

The LB's conceptual architecture, illustrated in Fig. 7.4, leverages a number of components from a portfolio of text parsing, extraction, transformation, and analysis technologies used to develop ontology-driven NL solutions. Components labeled 2, 3 and 4 represent mature components. Those labeled 1 represent prototype components added to complete our LB and RDF-F3 prototype. This framework provides a mechanism for connecting processing resources across a message bus construct. The processing components are connected to the message bus, where each subscribes to one or more input topics and can publish processed results on one or more output topics. The processing components are focused on part-of-speech (POS) tagging for RDF literals, identifying the lexico-semantic information for each word and phrase in the literal.

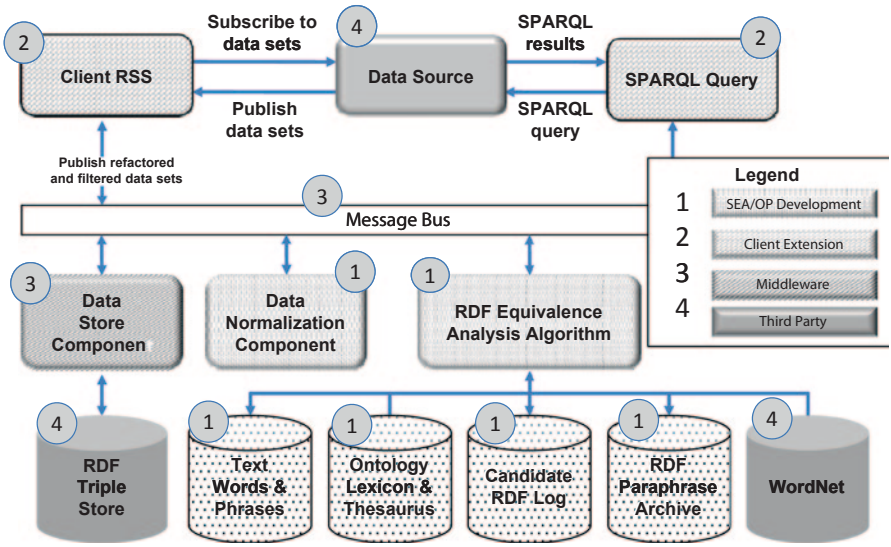

Fig. 7.4 RDF-F3 architecture lexicalized RDF triple literals for word-by-word comparison with text. *RDF* Resource Description Framework

LB Prototype and Results

Our initial prototype was designed to demonstrate three fundamental components of LB processing within the RDF-F3 application:

1. Ontology parser: Automated parsing of an RDF literal into its component lexical items
2. NL parser: Parsing semantically equivalent NL text into its component lexical items
3. Semantic mapping algorithm: Comparison of RDF lexical components to NL text lexical components to determine semantic equivalency.

As previously discussed, RDF class and property string names are not composed of words in the lexical sense. It is therefore difficult to compare a relation literal, such as JC3IEDM's relation `records-observed-results-of` (Multilateral Interoperability Programme (2009)), with words and phrases extracted from text, on a word-for-word basis, as illustrated in Table 7.1.

To do so, the literals must be parsed into individual words. In addition, candidate NL phrases must also be parsed into their individual words, as illustrated in the same table. The words from each must then be compared for equivalent meanings, or "senses."

Step 1: Parse Ontology Literal The first step in our prototype was to parse the string representing an RDF class or property literal, determining the constituent words and their parts of speech. Initial parsing of literals into words is illustrated in Table 7.2, rows 2–5 for the literal `records-observed-results-of.` We used an extended version of the Brill tagger (1992) to perform the parsing and the POS tagging, though we experimented with others as well.

Each word, together with its POS tag, was then processed to identify its potential meaning, which in WordNet means identifying the synonym sets (synset) to which it could belong. In this example, we show both the noun and verb synsets for the word `records`, as provided by WordNet.

It is important to note that none of the POS taggers performed correctly on this literal string. All identified `records` as a noun rather than a verb, very likely due to the fact that there were no other "subject nouns" in this literal, which parsers will treat as a grammatically well-formed sentence. However, this literal is an ungrammatical structure, linguistically speaking, at least in its ontology form. We therefore propose to tailor the Brill parser to account for "ontology literals" differently from "text sentences" to account for the linguistically ungrammatical structure of ontology literals.

Table 7.1 JC3IEDM object property string name and semantically equivalent NL used in the lexical bridge prototype

Ontology object property	Natural language paraphrases
`records-observed-results-of`	Wrote results about
	Will report observation concerning
	Are archiving observed evidence of

Table 7.2 Results of applying text parsing to ontology literals (records-observed-results-of) and to text (e.g., wrote results about). The text parser erroneously identifies "records" as the noun rather than the verb

Sentence/ ontology concept	Word #	Word	POS tag	POS	Root/lemma
Records observed results of	1	records	NNS	Noun, plural common	record
	2	observed	VBN	Verb, past participle	observe
	3	results	NNS	Noun, plural common	result
	4	of	IN	Preposition or subordinating conjunction	of
Wrote results about	1	wrote	VBD	Verb, past tense	write
	2	results	NNS	Noun, plural common	result
	3	about	IN	Preposition or subordinating conjunction	about
Will report observation concerning	1	will	MD	Modal verb	will
	2	report	VB	Verb, base form	report
	3	observation	NN	Noun, singular common	observation
	4	concerning	VBG	Verb, present participle	concern
Are archiving observed evidence of	l	are	VBP	Verb, present tense, not 3rd person singular	be
	2	archiving	VBG	Verb, present participle	archive
	3	observed	VBN	Verb, past participle	observe
	4	evidence	NN	Noun, singular common	evidence
	5	of	IN	Preposition or subordinating conjunction	of

Although not yet prototyped, our next objective for this lexicalization step to is to lexicalize an entire ontology in this manner, building a core Ontology Vocabulary by parsing class and property literals into NL words, together with their senses (or

meanings). We will then add synonyms for each of the ontology's lexical items, creating an Ontology Thesaurus. The vocabulary and thesaurus, taken together, form the OL, as defined in our RDF redundancy definition.

Step 2: Parse Text Paraphrases The second step in our prototyping effort was to parse NL paraphrases for their lexical content, as illustrated for three paraphrases in Table 7.2, rows 5 to the end. Each of these words, together with its POS, was then used to identify potential senses, or synsets, in WordNet, as illustrated in Fig. 7.5 for both the noun and verb senses of "records."

Step 3: Compare Lexical Semantics in Ontology Literal to Lexical Semantics in Text The next step in our algorithm is to process the individual words in the ontology literal parse trees, comparing each ontology word (OW) to text words (TWs) in the text parse trees. Specifically, this involves comparing each OW's POS and synset to a candidate TW's POS and synset (Table 7.2 and Fig. 7.6). We call this the *SEA*. For this specific application, it is an *RDF SEA*.

Although we have developed the logical algorithm for RDF semantic equivalency analysis (discussed in the next section), it has not been prototyped because we have not yet developed a robust OL, which it needs to do the word-to-word comparative analysis.

Steps 2 and 3 beg the question of how paraphrase candidates are identified in text sources in the first place, since this needs to be an automated process. Our algorithm uses the OL as a filter to identify candidate synonymous/paraphrastic NL expressions after POS tagging and sense disambiguation have been performed. In other

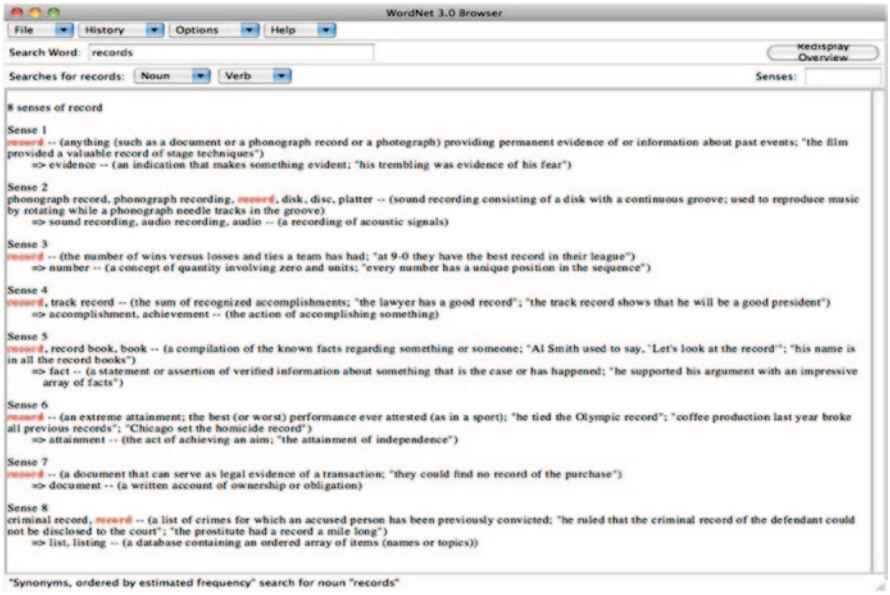

Fig. 7.5 A good lexicon, like WordNet, provides all senses for a word based on its part of speech

```
Algorithm: RDF-Equivalency algorithm
Function RDF-Equivalent (T: list of terms)
    for each {sentence || phrase ∈ T } do
         W := ParseSentence(Phrase)
         M := FindSynsets(W)
    end for
    for each { syn chain ∈ M } do
         P := SearchForSynsetChain(Phrase)
         s := BestSense(p,I,∅)
    end for

% Execute the Part of Speech tagging
Function ParseSentence (T: phrase)
    P := PartofSpeechTagging(T)

% Find WordNet Synsets – retrieves all the synsets associated with
the words
Function FindSynsets(W: list of word)
    M := 0
    for each {noun ∈ W || verb ∈ W || adj ∈ W || adv ∈ W} do
         s := Synsets(W)
         if s length > 1 then
              M := AddSynsets(M, FindSynsets(s))
         else
              M:= AddSynets(M,s)
         end if
    end for

% Determine the Best Synset chains
Function BestSense (t: term, I: list of senses, P: list of terms)
    BestSense := ∅ MinDistance := 0
    for each {sense s ∈ t} do
         d := MinDistanceS(s, I)
         if MinDistance = 0 or d < MinDistance then
              BestSense := s
         MinDistance := d end if
    end for

% Minimum distance using dijkstra's
Function MinDistance (s: sense, I: list of senses)
    d := 0
    for each {sense s' ∈ I} do
         d := d + DijkstraShortestPath(s, s')
    end for
```

Fig. 7.6 RDF equivalency algorithm uses the lexical bridge method to parse ontology literals into NL words and senses for comparison with text words and senses in order to identify potentially redundant triples in text sources. *RDF* Resource Description Framework, *NL* natural language

words, a TW from a text source is either in the OL or not, based on a look-up. This means that the TW is semantically equivalent to an OW in the lexicon. If the TW is not in the OL, then it is eliminated from consideration as a possible synonym. If the TW is in the OL, then it is retained for consideration as part of a synonymous

(or paraphrastic) text expression. Once all candidate words are identified in the text source, N-grams will be used to determine whether individual words (like "wrote" in Table 7.2) occur in a *paraphrastic or synonymous context* with other OW's (such as "results" in Table 7.2), a process explained more fully in the next section about our SEA. This analysis involves iterative comparison of word senses among OW's and TW's. In this way, the OL will be used to filter out TW's with no meanings in common with OL words and to identify sequences of words from text that are synonymous with word sequences representing ontology literals (Fig. 7.5).

Semantic Equivalency Algorithm Our algorithm uses *synset chains,* or graphs of sense relationships, to compare the lexical semantics of ontology literals to the lexical semantics of text paraphrases. Since these chains are graphs, we can apply standard graph algorithms, such as Dijkstras (1959) and Hart (1968), to our semantic equivalency analysis (SEA). The SEA is responsible for assembling the synset chain for each noun, verb, adjective, and adverb contained in a literal (from the ontology), sentence or phrase (from text). The algorithm then searches the knowledge base to determine if the synset chain is already asserted. If the chain does not exist then the algorithm will assert the new chain and the associated RDF triple reference. If the synset chain does exist for one word in a given phrase, then the algorithm searches for each subsequent word's synset chain as well. An equivalent phrase will have the same synset chains. A version of the SEA tailored for RDF analysis, called the RDF-Equivalency Algorithm, is presented in Fig. 7.6.

7.3 Conclusion and Next Steps

In Chapter 6, we summarized the mismatches between natural language and ontology semantics. As a result of these mismatches, many kinds of NL meanings will be lost when attempting to represent them in standard ontology representations, severely limiting the effectiveness of ontology-based semantic search, knowledge extraction, knowledge representation, knowledge discovery and ontology-to-ontology comparative analysis.

In Chapter 7, we have proposed that the first step in overcoming these limitations is to build a Lexical Bridge between a NL and an ontology. The Lexical Bridge is composed of the lexical and phrasal data structures and algorithms needed to compare the words and phrases in NL text to those embedded in ontology literals. We demonstrated the use of the Lexical Bridge in determining semantic redundancy in an RDF triple store, with the goal of ensuring that ontology-based knowledge extracted from text was represented only once within the knowledge base, thereby limiting knowledge growth to uniquely different pieces of information.

Our early prototype demonstrated that this approach has promise. Though the Lexical Bridge has its limitations, such as its inherent complexity, it is an important step in providing a more structured ontology-based representation of NL meaning, which is an important goal in most aspects of semantic processing.

In summary, we propose that NL *words and phrases*, together with their basic *meanings*, be used to provide the basic building blocks for a semantic bridge between a NL and an ontology.

References

Bechhofer, S., et al. (2004). *OWL Web Ontology Language Reference*, World Wide Web Consortium (W3C) recommendation, Feb. 2004. http://www.w3.org/TR/owl-ref/.

Bimson, K. (2009) *Principles of interontology development, research supporting lingua franca requirements and design*, tech. report, Modus Operandi.

Bimson, K., Teresa N., Jon N., & David M. (2012). *Tactical Semantics: Extracting Situational Knowledge from Voice Transcripts using Ontology-Driven Text Analysis*, Semantic Technology Conference, San Francisco, CA, June 2012.

Brill, E. (1992). A simple rule-based part of speech tagger, HLT '91: Proceedings of the workshop on Speech and Natural Language, Morristown, NJ, USA: Association for Computational Linguistics, pp. 112–116, doi:10.3115/1075527.1075553, ISBN 1-55860-272-0.

Dictionary.com. (2011). www.dictionary.com. http://dictionary.reference.com/browse/conflation.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1, 269–271. doi:10.1007/BF01386390.

Hart, P. E., Nilsson, N. J., Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics* (SSC), *4*(2), 100–107. doi:10.1109/TSSC.1968.300136.

Lowe (1990). The Berkeley framenet project.

Miller, G. A. (Ed.) (1990). WordNet: An on-line lexical database. *International Journal of Lexicography 3*(4), 235–312.

Mulilateral Interoperability Programme. (2009). *The Joint C3 information exchange data model metamodel*. Greding, Germany.

RDF Primer. (2004). W3C recommendation, Feb. 2004. http://www.w3.org/TR/rdf-primer/#rdfmodel.

RDF Semantic Web Standards. (2004). W3C recommendation, Feb. 2004. http://www.w3.org/RDF/.