Ontology-based Semantic Relatedness Measures: Applications and Calculation

Alexander Gelbukh

Natural Language Processing Laboratory,
Centro de Investigación en Computación,
Instituto Politécnico Nacional.
07738 México D.F.
gelbukh@gelbukh.com

Abstract. We propose a procedure for measuring semantic relatedness of two words using an ontology, or semantic network dictionary. We discuss applications of this procedure in detail for lexical, syntactical, and co-reference disambiguation in natural language processing as well as in machine translation. In addition, we use a simplified version of this procedure for automatic translation of the semantic network itself into other languages. This simplifies creation and maintenance of semantic network dictionaries for different languages, thus enabling the described methods for processing of texts in languages other than English.

Keywords. Natural language processing, text processing, syntactic analysis, disambiguation, semantic network.

1 Introduction

Natural language processing is a branch of Artificial Intelligence and Computational Linguistics that studies methodologies and algorithms for automatic analysis of natural language, in the form of text or speech, and applications of linguistic processing of texts to many tasks important in practice.

Among various problems that arise in such automatic processing is the problem of ambiguity: the algorithm should make a choice between two or several possible variants of interpretation of the same linguistic unit, such as a word or syntactic dependency in a sentence. A promising way of resolving such ambiguities is to select the interpretation most consistent with the context. A specific notion of consistency is given by a so-called semantic relatedness measure: a numerical measure between the given unit (say, a word) and similar units located nearby in the text. Given a suitable semantic relatedness measure, the algorithm should measure the relatedness between all variants of interpretation of the given unit and all units in its vicinity and select the one that on average gives the best result (the strongest relationship).

Thus, the study of disambiguation methodologies can be largely reduced to the study of different definitions of semantic relatedness measures, and suitability for a particular task, and their calculation basing on the available lexical resources. In this paper, we describe a particular semantic relatedness measure calculated using a semantic network dictionary such as WordNet [Miller, 1990] or FACTOTUM SemNet [Bolshakov et al., 1995b].

1.1 The curse of ambiguity

Probably the most difficult problem that nearly any algorithm dealing with the natural language faces is the curse of ambiguity. Be it just one word, or a phrase, or a text, very often there are several possible interpretations of what it means or what structure it has. We consider ambiguity resolution at all language levels the most important problem of natural language processing. To resolve the ambiguity, in much larger number of cases than it seems at the first glance, complicated reasoning or deep knowledge is necessary, often of semantic, pragmatic, or extra-linguistic nature.

A large number of works on ambiguity resolution employ manually crafted marked up text corpora, dictionaries [Luk, 1995], thesauri [Yarowsky, 1992], semantic networks [Sussna, 1993; Voorhees, 1993], or a combination of such lexical information sources [Yarowsky, 1995]. Still the problem is far from being satisfactorily solved.

In an ideal case, ambiguity resolution should be a side effect of some kind of "understanding," by which we mean construction of some detailed model of the whole situation described in the text and embedding it in the world model based on pre-existing knowledge, experience, or other texts read. The "true" linguistic knowledge, mostly lexical, ideally should be stored in vast dictionaries, such as combinatory dictionaries developed in frame of the Meaning $\Leftrightarrow$ Text theory [Mel'cuk, 1974; Steel, 1990], or programmed in sophisticated procedures, such as in the Word Expert Parser model [Berleant and Daniel, 1995]. However, either manual or automatic compilation of such resources is extremely labor consuming and is hardly affordable in the nearest decades. On the other hand, such "true understanding" is too demanding computationally to be considered now; what is more, there seems to exist evidence that such a way is too computationally demanding even for human brain.

A way that is less computationally demanding is to use some pre-constructed pieces of "typical" situations and first of all to check the ambiguous constructions against them, addressing to a deeper analysis only when the choice cannot be made with simpler processing. Such pre-constructed pieces of information can be of different nature, such as syntagmatic, semantic, pragmatic, etc.

For instance, syntagmatic patterns could be represented by frequently used or "meaningful" word combinations, such as *take a bus*, *take a pen*, as opposed to **take weather* [Bolshakov *et al.*, 1995a]. Such a simplified set of syntagmatic patterns can be used (and probably is used by a human) in syntactic analysis instead of the much more expensive "true understanding."

In a similar manner, instead of a computationally demanding reasoning, a set of simplified "typical" semantic patterns can be used for disambiguation. Such semantic patters could describe some atomic pieces of typical situations involving the words of the text. One of the form of representation of such knowledge is a semantic network, a set of semantic relationships between words in their specific senses.

1.2 Applications of semantic relatedness measures

This measure of relatedness is useful for resolving ambiguities of different types as well as for related tasks such as automatic translation of texts or even dictionaries. For instance, to resolve **syntactic** ambiguity, a variant of parsing should be chosen in which syntactically related words are more closely related semantically. To resolve **lexical** ambiguity between word senses in a context, the lexical variant should be chosen that is most closely related to the global or local topic of the document, or to the nearest words in the context [Banerjee and Pedersen, 2002; Patwardhan et al. 2003].

Similarly, to resolve **referential** ambiguity, the closest candidate is chosen to the words in the local context. In **text translation**, if the homonyms are not separated in the bilingual dictionary used for translation, the procedure of lexical disambiguation can be applied in the target language at the stage of text generation. Finally, in **translation of dictionaries** including the semantic network itself, lexical disambiguation can be performed on the reverse translation of the results back to the source language.

Various semantic relatedness measures have been proposed [Budanitsky and Hirst, 2001]; some of them are implemented in the freely available WordNet::Similarity package [Pedersen et al., 2004]. In this paper, we show how a semantic network dictionary can be used to measure the degree of semantic relatedness in a typical context between two given words [Gelbukh 1998].

The paper is organized as follows. In Section 2, we discuss a methodology for distance measurement in a semantic network. In Section 3, we present the applications of this methodology in various tasks related to computational linguistics and natural language processing. In Section 4, we discuss computational aspects of our methodology. Finally, Section 5 concludes the paper.

2 Distance measurements in a semantic network

In this section, we discuss the basic notions of measuring the semantic distances between two given words. The specific algorithms are given in the last section of the paper.

2.1 The structure of a semantic network dictionary

In our research, we used the FACTOTUM SemNet semantic network dictionary [Bolshakov et al., 1995b]. It is an English dictionary, though below we describe how to use it for other languages, our target language being Spanish. There exist other semantic networks—most notably WordNet [Miller, 1990], which has been widely used for natural language processing because of its availability; a Spanish version is available in frame of European WordNet project. SemNet, however, has a larger number of types and a more flexible representation of semantic relationships, making it more suitable for natural language processing applications. However, our methodology can be applied to WordNet, too.

In its logical structure, the SemNet dictionary is a set of so-called *relationships* between pairs of concepts (in rare cases between sets, here we omit the corresponding details for simplicity).

In SemNet, a **concept** is usually a word, e.g., *book*, or a word combination, e.g., *address book*, referring to a specific thing or idea. In most cases textual words have several meanings; in this case they are marked with different numbers, e.g., *bill*$_1$ (banknote), versus *bill*$_2$ (check), *bill*$_3$ (declaration), *bill*$_4$ (pike), or *bill*$_5$ (ax). Often such word senses have different translations to other languages: in Spanish, *bill*$_1$ is *billete*, *bill*$_2$ is *cuenta*, *bill*$_3$ is *declaración*, *bill*$_4$ is *pico*, and *bill*$_5$ is *hacha*.

All such senses of any word, even closely related ones, have different identification numbers in the dictionary, are located at different positions, and often have different sets of relationships to other words; if needed, they can be connected with each other explicitly by a relationship. Thus, one word (character string) can represent different concepts.

In turn, one concept can be represented by several words. In this case, they are considered synonymous in these particular meanings, and are listed together in a *synset* to represent and disambiguate the concept, e.g., {*bill*$_1$, *note, banknote*}. Thus, generally by a concept we always mean a group of synonymous word senses. However, for convenience we name the concepts with just one of the words of the synset.

Relationships are used to connect two (or, rarely, more) concepts. They are labeled with different type names, such as IS_A, USES, CAUSES, etc. A relationship can be viewed as a simple statement expressing a "typical" fact, e.g., *computer IS_A equipment*, *explosion* CAUSES *damage*. There are some attributes, or properties, of individual relationships, like MAYBE, USUALLY, RARELY, etc., e.g., *seeing MAYBE* USES *telescope*. For human convenience, there are different ways to express the same fact in the SemNet dictionary, e.g., *telescope* IS_USED_FOR *seeing*, though all such expressions can be formally converted to a common internal representation. A fragment of such a network is shown in the Fig. 1. In this example, we can see that a telescope is a tool to see, an animal can have an object, etc.

In the human-readable form of the dictionary, the most extensive set of relationships—namely, most of the IS_A relationships—is represented implicitly by placing the concepts in hierarchical order in the dictionary. This does not imply that a concept may not be a subtype of several concepts, for example, a *girl* IS_A *child* and IS_A *female*; in this case, one of the relationships is indicated in the dictionary explicitly.

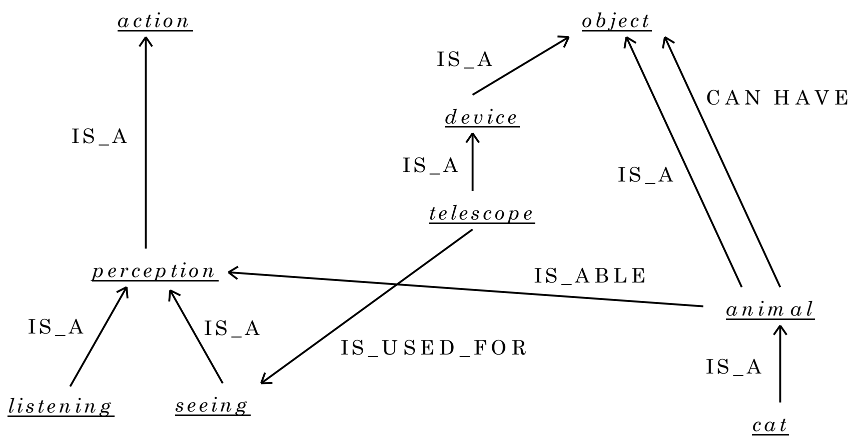

Fig. 1. A fragment of a semantic network.

Many possible relationships can be easily inferred by some general rules from other relationships; e.g., transitive relationships like IS_A and IS_PART_OF: if *a* IS_PART_OF *b* and *b* IS_PART_OF *c*, then *a* IS_PART_OF *c*. In such cases, only some of the relationships—the immediate ones—are explicitly included in the dictionary, to keep its size maintainable, e.g., *car* HAS_PART *motor* and *motor* HAS_PART *screw* implies *car* HAS_PART *screw*.

Other rules of inference involve particular relationships or groups of relationships. E.g., the most obvious one is that if *a IS_A b* and *b R c*, then *a R c*, where R is any relationship. In some cases, such inheritance of characteristics from higher categories is defeasible, i.e., it may be blocked explicitly by a special notation in the definition of a concept, or it may be canceled where contradictory information is inherited from more than one higher node.

2.2 Paths in the semantic network

The semantic network can be viewed as a graph. A **path** in such a graph is a chain of relationships r_1, ..., r_n such that r_i and r_{i+1} have exactly one common word, $i = 1, ..., n - 1$. If a word A is the beginning of the path and the word B is its end, we say that the path leads from A to B. There are several reasons to use paths for measuring the semantic closeness of words in the network.

First, since some of the relationships are present in the network only implicitly and can be inferred by application of the inference rules, a problem arises of generating all the relationships, including the implicit ones, between, say, two given words. The problem can be formulated as enumerating all the network paths with some special condition that lead from one given word to another.

Second, in some cases important commonality between words may not be expressed in terms of any existing named type of relationship. E.g., on Fig. 1, it can be seen that "a *cat* CAN HAVE something that IS_USED_FOR *seeing*." There is no named type of relationship expressing this fact, so we have to represent the commonality between these two words by just a path of the two relationships.

Third, some rules of inference may have fuzzy character, being rather common-sense observations. Therefore, applying too many rules can make the result less reliable. In some cases, we can express this loss of reliability by adding the MAYBE attribute to the resulting inferred relationship, though in general case we should use some kind of lengths, or weights, of relationships and paths, as described below.

Finally, some participants of the situation described in the text may not be mentioned explicitly. E.g., in the phrase "*The seller asked the buyer for too high price*" there may be no explicit relationship in the dictionary between the words *seller* and *buyer*, though the relationship between them can be found through the implied actant, *goods*: *buyer* CONSUMES *goods* HAS_SOURCE *seller*, a path of two relationships.

2.3 Lengths of the relationships and paths

In general, we need to assign some weight, or "length," to each relationship and calculate the "length" of a path based not just on the number of links in it, but also on their individual lengths. This value gives the quantitative estimation of how closely related are the two given words, while the path itself with the labeled links gives the qualitative estimation of exactly how the two given words are related.

For **explicit** relationships, such a length can reflect the degree of the importance of the relationship, e.g., IS_A relationships indicate that the words are close and probably substitutable for each other in most contexts. On the other hand, CONSUMES relationship reflects much less degree of closeness, i.e., it is "longer."

For **inferred** relationships, their lengths can depend on the kind of relationships involved in the logical inference (i.e., be an attribute of the corresponding rule) and on the length of the logical chain. E.g., many applications of the transitivity rule for IS_A relationships can increase the length of the resulting IS_A relationship. In Fig. 1, it is true but "less reliable" that *cat* CAN HAVE *telescope*, due to too many applications of IS_A transitivity rule. The fuzzy character of the inference rules is obvious for such relationships as IS_SIMILAR_TO, which is "to some degree" transitive.

For a **path**, its length should increase with the total length of the constituting links. The longer the path between two words, the less semantically related they are.

An easy way to assign the lengths to the links is to relate a specific length value to each **type** of the links, e.g., 1 to IS_A, 5 to SIMILAR_TO, and 20 to CAUSED_BY. Such assignment may be context-dependent or may vary according to the type of information being retrieved from the text. E.g., in a text with the principal topic “toys” the relationship SIMILAR_APPEARANCE can be more important than in a text with another principal topic. We leave such considerations for the future.

Sometimes it might be desirable to assign a length to **individual** links or groups of links. Ideally, each individual link should have some specific length expressing the degree of commonality between the two specific words. Assignment of these values hardly can be done by hand; instead, some procedure for training on a large corpus might be used in the future. There are cases, though, when additional individual coefficients can be assigned automatically.

For instance, special precautions are to be taken in order to prevent the algorithm from abuse of hierarchical links like IS_A. Namely, any concept referring to an object IS_A *object*: *car* IS_A *object*, *book* IS_A *object*. Thus, any two objects are connected with a path of two (usually implicit) links: *car* IS_A *object* HAS_SUBTYPE *book*, which normally should not imply a great degree of commonality between them. On the other hand, in some cases a path of two IS_A relationships does imply commonality: *Ford* IS_A *car* HAS_SUBTYPE *BMW*.

This problem is already mitigated by assigning to the implicit relationship a greater length (corresponding to a weaker relationship) than the length of an explicit relationship, when the implicit relationship is computed by application of the inference rules. However, the precision of the procedure can be improved by assigning the greater length to the hierarchical links located near the top of the hierarchy, thus, the length of the link *thing* IS_A *object* is more than that of the link *Ford* IS_A *car*. Namely, on the stage of preparation of the relationships database, the *maximum* number of links is determined from each node to the top of the hierarchy, and the links leading to this node are scaled correspondingly.

For example, in Fig. 2, the distance between *Ford* and *linguist* is much greater than between *Ford* and *BMW*, though both ones are subtypes of *object*. The distance between *car* and *book* is greater than the distance between *Ford* and *BMW*, though in both pairs there are exactly two explicit links between the nodes.

2.4 Shortest paths problem

To determine the semantic distance between two words, the shortest possible path in the network is to be found; its length can be used as an estimation of the degree of their semantic nearness. Since not all paths can be acceptable in a specific context, in some, and supposedly many, cases the next, then next, etc., shortest path should be used to determine the measure of the semantic distance between two words in a specific *context*. E.g., if the context suggests that the possible relationship between the two words is USES (expressed by the preposition *with*), then even a shorter relationship of the type IS_A cannot be used as a measure of closeness between these words in this context.

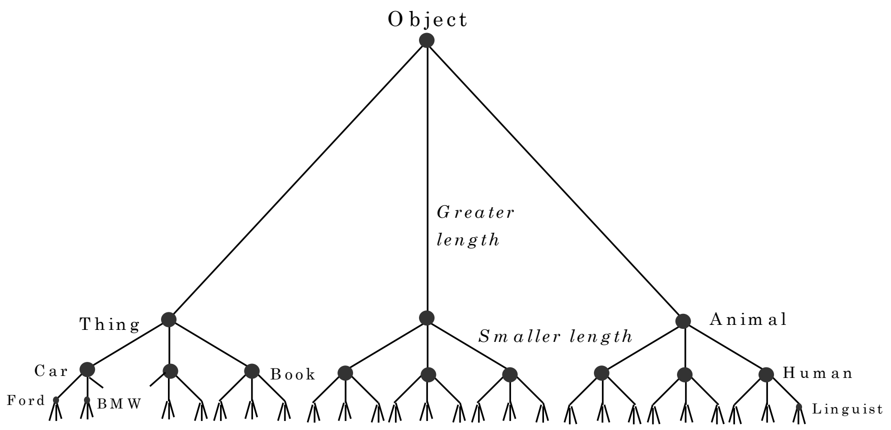

Fig. 2. Different lengths of hierarchical links.

The curse of ambiguity manifests itself in the full degree in the task of finding such paths. There is virtually infinite number of paths in the network, connecting the two given words. The computational aspects of the problem of finding the shortest path are discussed in the last section of the article; here it is enough to mention that the problem has well-known solutions and the only mathematical issue is computational efficiency. Thus, we will first discuss the linguistic applications of such an algorithm, assuming that it operates on a large enough semantic network dictionary.

3 Applications in computational linguistics tasks

Finding the shortest paths in the semantic network between two given concepts and measuring their relatedness in the network in a specific context has numerous applications for disambiguation in language processing and automatic translation. Note that in practice one can adopt a methodology where different sources of evidence and measures of semantic relatedness “vote” for the final decision; here we propose one of such “voters.”

3.1 Syntactical disambiguation

Consider a phrase “*John sees a cat with a telescope*.” The phrase is syntactically ambiguous: does it mean ‘John uses a telescope to see a cat’ or ‘John sees a cat that has a telescope,’ or ‘John sees a cat and a telescope,’ or maybe ‘John that has a telescope sees a cat,’ etc.? This ambiguity cannot be resolved using only lexical or syntactical information, since all the interpretations are syntactically plausible.

Most methods employed currently for solving this ambiguity, such as probabilistic grammars, rely on supervised machine learning to learn probabilities of different syntactic links, or, in the case of lexicalized grammars, the probabilities of combining specific words. With this information, a parse variant that contains most probable links is preferred to other variants and is chosen as the output of the parser.

While such methods give excellent results, they have certain disadvantages. The first disadvantage that we can mention here is the need in manually marked up corpora, called treebanks. Such corpora are expensive in development, and they do not yet exist for all languages; in fact, such corpora of considerable size exist only for a few major languages.

Another important disadvantage of statistical methods for this task is the data sparseness effect: while such training corpora have plenty of examples for frequent phenomena, due to the Zipf distribution law they lack a reliable number of examples for less frequent cases. In contrast, manually crafted linguistic resources tend to pay attention to linguistic phenomena irrespectively of their frequency, and thus provide information for both frequent and infrequent usage cases.

Therefore, in this paper we will assume so-called symbolic approach, in contrast to more widespread statistical approach. The symbolic approach relies on manually crafted dictionaries and grammars. In particular, it allows for exploiting existing lexical resources and dictionaries, including those created in pre-computer era for the use of human readers and not automatic procedures. The dictionary we used for this work, FACTOTUM SemNet, is based on the classical Roget thesaurus, which to some degree guarantees high quality of the information it contains.

In Fig. 3, the first two of abovementioned variants for the analysis of the phrase "*John sees a cat with a telescope*" are presented. The syntactic dependencies in question are *see* → *telescope* and *cat* → *telescope*; what are the semantic relationships between these words? There is a relatively short path between *seeing* and *telescope* in the semantic network dictionary. What is more, we can note that the type of the relationship(s) constituting this path agrees with the supposed instrumental syntactic relationship between these words in the phrase.

On the other hand, the best path between any sense of *cat* and *telescope* that agrees with the type of the supposed syntactic dependency is much longer. Thus, the variant (1) should be chosen here. This, though, should not prevent the linguistic processor from being able to backtrack and revise this decision later if the subsequent sentences disagree with this choice.

Sometimes just the quantitative measure of the nearness (the weighted length of the path) can be used for comparison. However, for better quality of analysis the whole path should be checked against the expected syntactical type of the relationship. E.g., in a phrase "*John sees a cat with a boy"* there is a short path between *seeing* and *boy*: *boy* IS_ABLE *see*, but the type of the relationship contradicts with the hypothesis that *boy* here is a tool to *see* with. This is why the procedure for finding the paths should be able to enumerate the paths until an acceptable one is found.

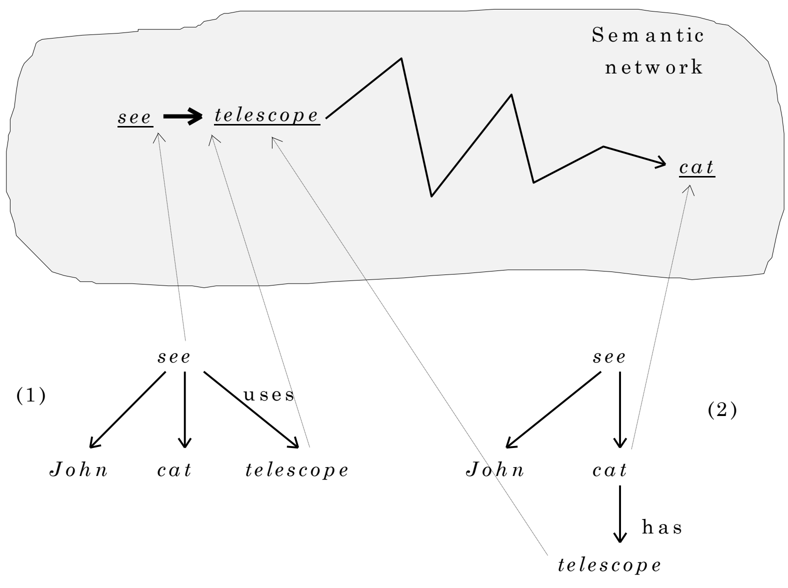

Fig. 3. Example of syntactic disambiguation process.

3.2 Lexical disambiguation

Ambiguity also arises in selection of a particular sense of a word in a phrase. Sometimes they can be resolved at syntactic level, usually when the choice is made between different parts of speech, e.g., in the phrase "*John tables the dishes"* the word *tables* is clearly a verb. However, in many cases, especially when a word has different meanings within the same part of speech, semantic information has to be employed.

Compare, for example, the phrases "*There were fruits and drinks on the table*" and "*The numbers were arranged in a table*." By addressing a semantic network, it can be determined that in the first phrase the shortest path exists between other words and the sense 'table as a furniture,' while in the second phrase, the shortest path leads from *numbers* to 'table as a picture.'

It is not as clear as it is with syntactic ambiguity, with what words in the phrase the given word is to be compared. Good candidates can be words close in the syntactical structure to the given one. Other good candidates are the words describing the main global or local topics of the document. For example, if the document in general is on mathematics, the word *table* will likely be used in it as 'table as a picture,' even if the nearest words do not suggest this directly.

Local and global topics of the document can be determined with the approach called CLASITEX [Guzmán-Arenas, 1997]. In this approach, all the textual words of the document without any preliminary disambiguation are clustered with the help of a semantic network dictionary. The centers of the largest clusters represent the main topics of the document. The noise is canceled out since the wrong senses of the words form smaller clusters. This approach can be applied to a part of the document, revealing the local topics. These topics can also be used in the disambiguation process: the distance is to be measured between them and the current word being disambiguated.

Since in the process of disambiguation, many words or word senses (global and local topics, surrounding words, etc.) possibly have to be tried and the results have to be accumulated, the procedure may be computationally demanding. However, in comparison with, say, Word Expert Parser model [Small and Rieger, 1982], our procedure requires easier available data and can be used in frame of the traditional text processing algorithms.

3.3 Referential disambiguation

The problem of referential disambiguation arises each time a pronoun, ellipsis, or zero subject (very common in such languages as Spanish) is used in the text. In general, at the stage of text analysis such a reference must be replaced with another word probably used somewhere in the text. Though there are linguistic considerations on selecting the candidates to fill the valence, they usually give ambiguous results when only lexical and syntactic information is considered.

However, it is possible to resolve this task into the task of lexical disambiguation. Namely, when several candidates are to be tried to fill the valence, they can be just treated as different "senses" of the pronoun in this particular context. Then the procedure described in the previous section can be applied with nearly no modifications. The only difference is that neither global nor local topics are used in the comparison.

3.4 Machine translation

In general, text translation is a quite different task from text understanding. Ideally, translation should include the steps of text understanding in the source language and then text generating in the target language. If the ambiguity is resolved at the stage of analysis, and if the bilingual dictionary is good enough, there should be no problems with ambiguity during text generation. However, in real life it is not the case, for both practical and theoretical reasons [Narin'yani, 1997].

In practice, less sophisticated methods are currently used, working mostly at the syntactic level. Some of commercially available symbolic-based translation systems distinguish the senses of the words only by a limited number of semantic classes or by literal recognition of some number of idioms. E.g., this phrase was translated from Spanish by Globalink's Power Translator Professional: "*El artista realiza bien el papel*" ⇒ "*The artist accomplishes well the paper*" (instead of *role*). This program, though, does distinguish these senses in some contexts: it seems to make choice based on literal recognition of the idiom "*jugar un papel*," e.g., "*El diputado juega un papel importante*" gives "*The deputy* *plays a role important*," but: "*El diputado juega el papel más importante*" gives "*The deputy plays the most important paper*." For good symbolic-based automatic translation, there must be available (1) a good disambiguation procedure in the source language, (2) a good bilingual dictionary that translates one-to-one senses to senses, not textual words to sets of words. Both conditions are very difficult to satisfy. For example, there might not be available a Spanish dictionary to disambiguate the two senses of the word *papel*.

In addition, the most elaborated up to date dictionaries, including academic dictionaries, usually provide translations of a word into several possible words in the target language, e.g.: "*papel*: paper; document; role; <...>" [Spanish-English, 1963]. In this case, even if the senses have been disambiguated in the source language, the dictionary anyway does not contain the information necessary to translate them one-to-one into words of the target language.

Our methodology permits to disambiguate the words after translation in the target language. As in the previous section, we can treat the ambiguous position as a word with several "senses" and then apply the procedure of lexical disambiguation to the generated phrase in the target language.

For instance, in the example above, there is a shorter path in the English semantic network between *artist* and *role* than between *artist* and *paper* or *document*. This allows us to use a semantic network to improve the results of translation made with existing bilingual dictionaries, rather than developing new sense-to-sense dictionaries, which are expensive to create and difficult to share between different systems due to their tight integration with the other modules of a linguistic processor.

3.5 Automatic translation of the semantic network

Our disambiguation procedure can be applied to automatic or semi-automatic translation of the semantic network itself into other languages. Since we have taken part in such a translation project (though the work was mainly done by hand), we are aware of all the deficiencies of the very idea of translation of a semantic network, and of low quality of the resulting dictionary [Bolshakov *et al.*, 1995b]. Still there are at least three reasons to translate semantic networks.

First, creating a semantic network from zero is a very difficult and expensive work. If the way the results are used is tolerant to the incompleteness and minor inaccuracies, it may be more efficient to use a lower quality dictionary translated from an existing resource than to wait for a better dictionary to be created in the far future. Actually, we believe that due to the nature of the functioning of natural language, any language processing software must be tolerant enough to incomplete and inaccurate information. However, since the semantic network contains mostly the facts about real-world objects and ideas, and in part due to commonality between the languages, most of the relationships tend to be translated correctly (though this may depend on the languages and subject area).

Second, the linguistic resources for such languages as English, French, Japanese, etc., are maintained by many people and groups all over the world, with much money spent on their development, enlargement, and refinement. It would be a waste of effort to repeat all this work in full size for each language. Thus for groups that work on, say, Spanish language, to take advantage of the efforts spent in the world on development of English semantic networks, it is necessary not only to translate the first draft of the dictionary from English, but to be able to repeat such translation automatically as new versions of the English dictionaries become available. There is no need to mention that the existing machine translation programs designed for translation of phrases in a discourse are not appropriate to translate structured resources such as dictionaries; thus the necessity to create specialized dictionary translation software.

Third, existing ontologies can be used, such as so-called T2 Reference Ontology for English. Automatic or semi-automatic procedure for translation of this resource can be very useful in maintaining compatibility between semantic networks in different languages and the ANSI standard.

Details of the translation procedure are beyond the scope of this paper. Here we only discuss the application of the procedure for enumerating the paths in the network between two given points to the task of translation of the semantic network itself. However, automatic translation of a semantic network faces the same main problem: ambiguity. Each word in each its occurrence in the text of the dictionary, presumably in different senses, is translated by an ordinary bilingual dictionary to several different words of the target language.

The following procedure is proposed to choose the correct variant of the translation, using the same (English in our case) semantic network. Each variant of translation of a word is translated back to the source language. Then the distance in the source semantic network is measured between the source word and each variant of such a reverse translation. The variant(s) of translation are chosen, at least one of whose reverse translations is located near the source word sense in the network, i.e., there is a “short” enough path from this variant to the source word sense, see Fig. 4.

The copy of the source word is removed from the set of the reverse translations. Words having only one reverse translation, namely the same source word, are treated as special cases. They are inserted in the resulting dictionary, and if the source word has different senses, such words are marked when automatically inserted in the dictionary, and then checked by hand.

In theory, only the words with a reverse translation within the same concept, i.e., at the zero distance from the source word, should be accepted. However, in practice, a bilingual dictionary in most cases does not gives such accurate results; therefore, the paths of nonzero length should be taken into account.

For the set of the paths in the network to be considered, for each textual word all its senses should be tried unless any disambiguating information is available in the bilingual dictionary; usually it is not. The choice of the candidate is made in two steps. First, the weights, i.e., the lengths of the corresponding paths, of the reverse translations of each candidate are combined to calculate the weight of the candidate itself. Second, the candidate(s) are chosen with the best such weight.

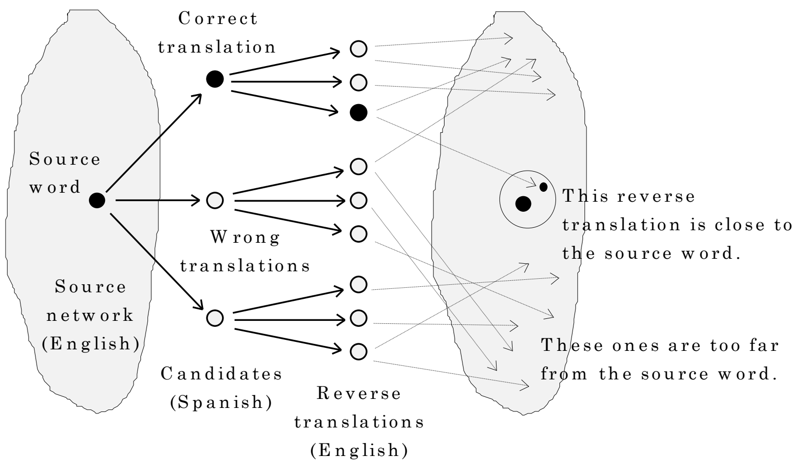
Correct translation
Source word
Wrong translations
Source network (English)
Candidates (Spanish)
Reverse translations (English)
This reverse translation is close to the source word.
These ones are too far from the source word.

Fig. 4. Translation of a semantic network (two copies of the same network are shown to simplify the picture)

Various procedures can be used for both calculations. To combine the weights of the reverse translations for one candidate, in the simplest case a maximum (but not average) can be taken. In a more sophisticated procedure, the values for all the reverse translations better than some threshold should be accumulated. To choose the acceptable candidates of translation, in the simplest case only the best one is taken for each word, or all the candidates are accepted that are better than some threshold value. More sophisticated procedures can also be tried. For example, all candidates better than some threshold value should be accepted, all candidates worse than some threshold value should be rejected, or the best one should be chosen from those candidates whose weight falls between these two thresholds. The obtained semantic network dictionary may be then post-edited by hand. To be able to repeat the translation as new versions of the source dictionary become available, the changes made by hand should be saved in a special protocol.

As compared with the procedure of enumerating the paths used for text processing, the procedure used for translation of the dictionary itself can be simplified by ignoring completely the inference rules, since in this case the meanings should be preserved much more precisely. The length of the path can be calculated as just the number of links in it. This makes the implementation of the procedure for translation much more straightforward than for text understanding.

Inference rules can be used for better results. However, application of each rule should substantially increment the length of the path. E.g., a chain of transitive relationships like IS_A should be considered long enough, whereas the procedure used for text understanding would use the length near to 1 for such a path. The choice here, as well as the selection of the thresholds mentioned above, is made on the basis of desired compromise between the accuracy of the translation (less usage of inference, higher thresholds) and the number of words that will get any translation at all (more usage of inference, lower thresholds).

Only basic relationships, such as IS_A and possibly IS_PART_OF and a few other, should appear in the paths, but not such relationships as USES, etc.

4 Computational aspects

Here we describe the mathematical problem statement and possible algorithms for its solution. Generally, a simple modification of Dijkstra's shortest path-finding algorithm [Dijkstra, 1959] could suffice, though we present a more sophisticated modification, adapted to large sparse networks.

4.1 Problem statement

Finding paths in the network is important for computational linguistic applications, primarily to measure the distance in the network between the two given words in a specific context. Usually to measure such a distance in a specific context, the shortest paths between the two points are to be found; however, it is not always true that necessarily the very short, optimal, path should be found first. There are several reasons for this.

First, there are many **restrictions** on suitability of particular types of paths in a specific context. If, for example, the syntactic relationship between the two given words suggests, say, instrumental relationship, then the semantic relationships of any other types, even very short, are useless in this context. Such restrictions can be applied after the path is found in the network. Thus, it is probable that the very short path will prove unusable in a specific context; in this case, the next path should be retrieved and examined. Thus, there is no point to apply a computationally demanding procedure to optimize completely the search process.

Second, the rules for calculating the length of a path in a specific context can be **context-dependent** and complicated. They may be better applied after the path has been found. This problem is discussed in more detail below at the end of this section.

Third, the **low precision** of all language data, including in the first place the text itself being analyzed, but also the dictionaries, grammars, etc., makes very precise procedures not so necessary. Everything in language is vague; any text is full with hints, omissions, implied information, metaphors, rather than being a collection of clear and simple logical statements. This makes too precise procedures of text processing in many cases useless.

On the other hand, since automatic procedures are applied to huge amounts of texts, **performance** is important, as long as the result of the analysis fits in the same confidence interval. Performance is especially important since the procedure for semantic distance measurement is invoked very many times in typical applications, such as referential disambiguation.

Therefore, we can consider the following problem: to enumerate the paths in a network between two given nodes, as a tendency starting from shorter ones, under the following conditions:

– Various timeouts apply, e.g., a threshold on the length of the paths: only the paths of the length less than a threshold value are considered.
– Computational effectiveness is a priority.
– Accuracy is the second priority. Better paths should go first, but only on average.
– The network is large and stored in a database, so that retrieval of the links leading from a given point is the most time-consuming operation.

At each step, we estimate—again, with some probability—the lower limit of the length of the paths the procedure can find yet. The importance of this latter requirement will be discussed below. We suppose that the calling routine at some will moment stop the enumerating process, or some kind of time-out is used to prevent the algorithm from infinite work, such as a restriction on the number of paths, or on their lengths, so that if the procedure cannot find any paths shorter than some threshold, it should stop. In addition, some qualitative restrictions may be imposed on the desirable paths, e.g., not to contain a particular relationship.

The problem is very similar to the well-known problem of finding the shortest paths in sparse graphs, e.g., [Shier, 1976; Johnson, 1977; Minieka, 1978]. However, there are some differences in the goals and conditions with the classical problems of optimization. The main differences between the two problems are summarized in Table 1.

Table 1. Comparison of the considered problem and the classical one.

Classical Optimization	**Our problem**
There are no restrictions on the length of the path.	Only the paths shorter than some threshold value (i.e., short enough ones) are considered.
Only one path is searched for (in some variants K paths).	Paths must be enumerated until the caller “accepts” one.
The very best path must be found.	Better paths should generally go first, but not necessarily the very best one is to be the first.
The length of a path is a mere sum of the lengths, or weights, of the individual links.	The length is calculated according to the fuzzy rules of combination of the relationships.
No previously prepared data is usually used.	Some data can be prepared in the database in advance.
The graph is small enough to be kept in memory.	The graph is very big and is stored on the hard disk.

By better paths, in Table 1 we mean the ones with smaller length, which usually means ones that contain fewer links or links with smaller lengths. This measure is computed as a combination of the lengths of individual links, with application of, or taking into account, the rules of inference. E.g., a chain of five IS_A relationships may be considered "shorter" than a chain of two IS_SIMILAR_TO relationships. In general, such an estimation is a complex problem by itself, and we will not describe it here in more detail.

The measure of length used by the algorithm can differ from the measure that is used by the calling procedure, the latter being probably calculated or refined by the caller itself with the application of some specific, possibly complex, rules, for example, making use of the logical structure of the situation described in the text itself.

This difference arises from our intention to separate the information internal to the semantic network from the information used in various applications, and to provide a general procedure (probably implemented as a separate module) that permits the caller to treat the semantic network as a black box. However, some minimal adjustments of the procedure will anyway be necessary for some applications; they are discussed below in the sections devoted to the corresponding applications.

We assume, therefore, that the algorithm should find the paths just good in some general meaning, and the caller will check if the path is in fact good for it, though the "generally good" paths should be usually good enough for the caller.

Therefore, the algorithm should not even try to optimize completely the enumerating process, since anyway chances are little that the very best in general sense path will be the very best for the caller, and we expect usually it will not. This changes the approach to the algorithm as compared with the classical optimization problems.

4.2 Algorithms

While there is extensive research devoted to the shortest paths problem, we are not aware of any known algorithm for solving exactly our task. It is not our goal in this paper to propose a mathematically refined algorithm, since at the current stage of the research we are mostly interested in the linguistic applications of the idea itself. However, we describe here some variants of the algorithms we currently use.

4.2.1 The case of equal lengths of the links

Here we consider a non-weighted graph. A simple algorithm of enumerating all the paths, a modification of Dijksatra's one, is as follows. Define a sphere S_r (A) around the point A as the set, actually a tree, of all the paths of the length *r* leading from the point A. (Each path in the tree is compactly represented by the ending point, additional characteristics such as the length, and a pointer to the previous path in the tree. When a new path is formed by adding one link to the previous one, only such a data structure is to be created in memory.) Since we consider here the length of each link in the network to be just 1, the radii of the spheres are natural numbers; we can also consider S_0 (A) being the empty path, i.e., the point A itself. We call the set of ends of all the paths of the sphere S (A) its surface.

The next sphere S_{i+1} (A) can be formed by adding to S_i (A) each link leading from each its surface point. If necessary, obvious precautions can be taken to prevent the paths being formed from cycles, at least from the cycles formed by two copies of the same link passed in the opposite directions; this can be done by comparison of the link being added to a path with the immediately previous link in the path. Other types of cycles in a sparse network usually do not present much problems for our task.

The algorithm alternates between increasing the spheres S (A) and S (B), starting from, say, S (A). At each step, the intersection of the *surfaces* of the spheres S_i (A) and S_j (B), $j = i$ or $j = i - 1$, gives the paths of the length $i + j$. This algorithm enumerates all the paths between A and B, starting from the shortest ones.

In case of an oriented graph, when only the oriented paths from A to B are to be found, a simple modification of this algorithm can be used. The sphere S^+_{i+1} (A) should be formed only with the links leading *from* the points of S^+_i (A), while the sphere S^-_{i+1} (B) should be formed only with the links leading *to* the points of S^-_i (B). If the paths both from A to B and from B to A are to be found, two spheres S^+ and S^- are maintained for A and B, consisting of the links leading to and from the points, correspondingly. If there are other restrictions on the types of the paths, they also can be taken into consideration at the step of increasing the spheres.

4.2.2 Different lengths and inference rules

The algorithm described in the previous section can be generalized to the case of weighted graphs. We consider here a modification that not always gives the shortest paths first, but does so as a tendency. This algorithm can be easily modified to enumerate the paths in the proper order, but with slightly lower efficiency.

In this algorithm, the sets of paths, which we will still call spheres, actually are not spheres, i.e., the paths in such “spheres” do not have the same lengths. We define these spheres just recursively, the sphere S_{i+1} (A) being formed by adding to S_i (A) all the links leading from some of its surface points (we chose to add all the links here since the operation of retrieval of the links is the most time-consuming). In this case, not all the surface points of S_i (A) are expanded, instead, expanded are only the paths, usually one path, with the minimal length among all the paths of S_i (A). The surface of S_{i+1} (A) is defined by replacing the expanded points of the surface of S_i (A) with the ends of the newly added links.

Similarly, the spheres S (A) and S (B) are increased in turn, and the intersection of their surfaces gives the different paths between A and B. It is easy to prove that this algorithm enumerates all the paths. Namely, let us call the minimal length of a path in the sphere its *minimal radius*. Each step of the algorithm increases the minimal radius of one sphere, and if the current minimal radii of S (A) and S (B) are r_1 and r_2, then all the paths with the lengths of $r_1 + r_2$ have been already enumerated by this moment.

Our algorithm does not enumerate the paths in the exact order of their length. A counter-example can be constructed when two points are connected by two long links (thus the length of the path is large) and, in addition, are connected by three short links (so that the length of the resulting path is small). In this case, the algorithm finds the former path first, while the latter path is shorter.

However, generally it tends to enumerate the shorter paths before the longer ones. It is possible to store the found paths temporarily without reporting them to the output, until the value $r_1 + r_2$ reaches the length of a temporarily stored path. With this modification, the algorithm will enumerate the paths in the proper order. However, for our goals we chose to use the path as soon as the algorithm finds it.

The inference rules and the rules for determining the lengths of different combinations of the relationships can be taken into account at the step of increasing the spheres and at the step of determining the intersection of the spheres. Namely, when a link is added to the path, the length of the path being formed is determined accordingly to the rules of combination of links. Similarly, when a complete path between A and B is formed by connecting together two paths, one of S (A) and another of S (B), the length of the combination may differ from the sum of the length of the two paths. This contributes in the lack of order in enumeration process. However, the shorter paths still tend to be enumerated before the longer ones.

4.2.3. Use of pre-calculated data

If the network is not a small world graph, the methods described above are good only to find short enough paths, since spheres of big radii are too large. In practice, it may not be a problem if only short paths are important for the applications.

However, if longer paths are required, an additional network of “pivot nodes” with pre-calculated information about their connections with each other may be used. This is similar to the idea of a cellular telephone system, where two phones, instead of communicating with each other, communicate with nearby nodes of a dense enough network, while those nodes can then communicate with each other in a predefined manner.

For this, at the stage of preparation of the database, nodes are added, or existing nodes are used, at nearly equal distances from each other and not further than some threshold distance from any node in the network. The number of such control nodes should be much less than the total number of nodes in the network. Information is stored with those nodes to help finding the paths leading from each of them to each another. To find a path from an arbitrary point A in the network to another point B, first, the paths from each one of these points to the nearest pivot node are determined using the method of increasing spheres described above. Then, the path between the control nodes is retrieved from the database. Finally, the complete resulting path can be varied or optimized locally around the retrieved one.

4.2.4. Multiple comparisons

Sometimes not only the distance between two given nodes is to be determined. Instead, the questions to be answered are as follows: (1) which point in some set of points is the nearest to a given one, or (2) what are the two nearest points in a two given sets. These problems arise in disambiguation of the binding of a prepositional phrase and in referential disambiguation, correspondingly.

A simple modification of our algorithm allows us to take advantage of alternating between increasing the spheres in turn and of using the same sphere to determine the distances from the given point A to each of the points $B_1, ..., B_n$. All the spheres are increased, each one in its turn. Suppose we find at least one path between A and, say, B_1 such that its length is smaller than the sum of the minimal radii of the spheres S (A) and S (B_2), and no shorter paths have yet been found between A and B_2. Then the distance between A and B_1 is smaller than the distance between A and B_2. Note that since the paths are retrieved not in a completely precise order, the check against the minimal radii is important.

Given the complicated rules of link combination, the latter criterion is not precise, since the length of a path can be different from the sum of the lengths of its parts. Currently, we ignore this complication, since we consider the loss of precision to be less than normal fuzziness of all data related to natural language. In case of serious problems arising in understanding of a particular phrase, backtracking can be used later to calculate the distances in question more precisely.

5 Conclusions

A number of semantic network dictionaries and ontologies are available nowadays, mostly for English language, such as WordNet or the FACTOTUM SemNet dictionary.

We have presented in this paper a simple procedure, namely the search for the shortest paths in a sparse network, that can be used for determining the measure of semantic relatedness of two given word senses in a very large semantic network. This measure is useful for disambiguation in a variety of important tasks of natural language processing, such as lexical, syntactic, and referential disambiguation, as well as in text generation and machine translation.

In addition, this procedure can be used to translate automatically the semantic network dictionary itself into other languages. This makes our methods usable for processing of languages other than English. This also simplifies creation and maintenance of semantic network dictionaries for these languages. What is more, such automatic translation of a semantic network will be useful in development and maintenance of semantic networks in languages other than English, which would conform to the ANSI Standard Ontology (T2).

In our future work, we plan to consider more detailed information for syntactic disambiguation that can be extracted from existing dictionaries [Castro-Sánchez and Sidorov, 2012]. We also plan to combine methods for improving the translation results developed in this paper with statistical methods based on alignment of parallel bilingual text corpora [Sidorov et al., 2011].

Acknowledgments

The work was partially supported by CONACYT grant 50206-H, SNI, and FP7-PEOPLE-2010-IRSES: Web Information Quality – Evaluation Initiative (WIQ-EI) European Commission project 269180.

References

1. [Banerjee and Pedersen, 2002] Banerjee, Satanjeev, and Ted Pedersen. *An Adapted Lesk Algorithm for Word Sense Disambiguation using WordNet*. In: Proceedings of CICLing 2002, the Third International Conference on Intelligent Text Processing and Computational Linguistics, pp. 136–145, 2002, Mexico City.
2. [Berleant and Daniel, 1995] Berleant, Daniel. *Engineering "word experts" for word sense disambiguation.* In Natural Language Engineering 1, 1995: pp. 339-362.
3. [Bolshakov *et al.*, 1995a] Bolshakov, I.A., P.J.Cassidy, A.F.Gelbukh. *CrossLexica -- a dictionary of collocations and thesaurus of the general Russian lexicon* (in Russian, abstract in English). In Proceedings of International Workshop Dialogue'95: Computational Linguistics and its Applications, Khazan, 1995.
4. [Bolshakov *et al.*, 1995b] Bolshakov, I.A., P.J. Cassidy, A.F. Gelbukh. *Parallel English and Russian hierarchical thesauri with semantic links, based on an enriched Roget's thesaurus* (in Russian, abstract in English). In Proceedings of International Workshop Dialogue'95: Computational Linguistics and its Applications, Khazan, 1995.
5. [Budanitsky and Hirst, 2001] Budanitsky, Alexander and Graeme Hirst. *Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures*. In: Workshop on WordNet and Other Lexical Resources, Second meeting of the North American Chapter of the Association for Computational Linguistics, Pittsburgh, June 2001.
6. [Castro-Sánchez and Sidorov, 2012] Castro-Sánchez, Noé Alejandro and Grigori Sidorov. *Extracción automática de los patrones de rección de verbos de los diccionarios explicativos*. Polibits, vol. 45, 2012, pp. 67–74.
7. [Dijkstra, 1959] Dijkstra, E.W. *A note of two problems in connection with graphs*. Numerische Matematik, 1959, V. 1, pp. 269-271.
8. [Gelbukh, 1997] Gelbukh, A. *Using a Semantic Network for Lexical and Syntactic Disambiguation*. In Proc. of the International Symposium "CIC-97: Nuevas Aplicaciones e Innovaciones Tecnológicas en Computación," November 12-14, 1997, Mexico D.F.

9. [Gelbukh, 1998] Gelbukh, A. *Using a Semantic Network Dictionary in Some Tasks of Disambiguation and Translation*. Technical report, Serie Roja, N 36. CIC, IPN, 1998.
10. [Guzmán-Arenas, 1997] Guzmán-Arenas, A. *Determining principal themes in a Spanish article* (in Spanish). In Proc. of the International Symposium "CIC-97: Nuevas Aplicaciones e Innovaciones Tecnológicas en Computación," November 12-14, 1997, Mexico D.F.
11. [Johnson, 1977] Johnson, D.B. *Efficient algorithms for shortest paths in sparse networks*. J. ACM, 1997. Vol. 24, N 1, pp. 1-13.
12. [Luk, 1995] Alpha K. Luk, *Statistical Sense Disambiguation with Relatively Small Corpora Using Dictionary Definitions*. In Proceedings of the 33rd Annual Meeting of the Amer. Soc. for Comp. Ling., 1995, pp. 181–188.
13. [Mel'cuk, 1974] Mel'cuk, Igor A. *Experience in theories of Meaning ⇔ Text linguistic models* (in Russian). Moscow: Nauka, 1974.
14. [Miller, 1990] Miller, George A., ed. *WordNet: An on-line lexical database*. International Journal of Lexicography, 3, 1990: pp. 235-312.
15. [Narin'yani, 1997] Narin'yani, A.S. *Automatic text understanding – new perspective* (in Russian, abstract in English). In Proceedings of International Workshop Dialogue'97: Computational Linguistics and its Applications, Moscow, 1997.
16. [Patwardhan et al. 2003] Patwardhan, Siddharth, Satanjeev Banerjee, and Ted Pedersen. *Using Measures of Semantic Relatedness for Word Sense Disambiguation*. In Proceedings of CICLing 2003, the Fourth International Conference on Intelligent Text Processing and Computational Linguistics, pp. 241–257, 2003, Mexico City.
17. [Pedersen et al., 2004] Pedersen, Ted, Siddharth Patwardhan, and Jason Michelizzi, J. 2004. *WordNet::Similarity: Measuring the relatedness of concepts*. In HLT-NAACL 2004, Association for Computational Linguistics, pp. 38–41.
18. [Shier, 1976] Shier D.R. *Iterative methods for determining the* K *shortest paths in a network*. In Networks, 1976. Vol. 6, N 3, pp. 205-229.
19. [Sidorov et al., 2011] Sidorov, Grigori, Juan-Pablo Posadas-Durán, Héctor Jiménez-Salazar, Liliana Chanona-Hernández. *A New Combined Lexical and Statistical based Sentence Level Alignment Algorithm for Parallel Texts*. International Journal of Computational Linguistics and Applications, Vol 2 (1-2), 2011, pp. 257–263.
20. [Small and Rieger, 1982] Small, S.L., and C.J. Rieger. *Parsing and comprehending with word experts (a theory and its realization)*. In Lehnert and Ringle (eds.), Strategies for Natural Language Processing, 1982, pp. 89-147.
21. [Spanish-English, 1963] *Spanish-English, English-Spanish dictionary*, Pocket books, Inc. NY, 1963.
22. [Steel, 1990] Steel, James, ed. *Meaning – Text Theory. Linguistics, lexicography, and implications.* University of Ottawa press, 1990.
23. [Sussna, 1993] Sussna, M. *Word Sense disambiguation for free text indexing using a massive semantic network*. In: Proceedings of CIKM, 1993.
24. [Voorhees, 1993] Voorhees, E.M. *Using WordNet to disambiguate word sense for text retrieval*. In Proceedings of ACM SIGIR Conference, 1993, pp. 171-180.
25. [Yarowsky, 1995] Yarowsky, David. *Unsupervised Word Sense Disambiguation Rivaling Supervised Methods*. In Proceedings of the 33rd Annual Meeting of the Amer. Soc. for Comp. Ling., 1995, pp. 189-196.
26. [Yarowsky, 1992] Yarowsky, David. Word Sense Disambiguation Using Statistical Models of Roget's Categories Training on Large Corpora. In Proceedings of COLING-92, 1992, pp. 454-460.