# Unsupervised Learning for Syntactic Disambiguation

Alexander Gelbukh

Centro de Investigación en Computación,
Instituto Politécnico Nacional,
Mexico

www.gelbukh.com

**Abstract.** We present a methodology framework for syntactic disambiguation in natural language texts. The method takes advantage of an existing manually compiled non-probabilistic and non-lexicalized grammar, and turns it into a probabilistic lexicalized grammar by automatically learning a kind of subcategorization frames or selectional preferences for all words observed in the training corpus. The dictionary of subcategorization frames or selectional preferences obtained in the training process can be subsequently used for syntactic disambiguation of new unseen texts. The learning process is unsupervised and requires no manual markup. The learning algorithm proposed in this paper can take advantage of any existing disambiguation method, including linguistically motivated methods of filtering or weighting competing alternative parse trees or syntactic relations, thus allowing for integration of linguistic knowledge and unsupervised machine learning.

**Keywords.** Natural language processing, syntactic parsing, syntactic disambiguation, unsupervised machine learning.

## Aprendizaje no supervisado para la desambiguación sintáctica

**Resumen.** Se presenta un marco metodológico para la desambiguación sintáctica de textos en lenguaje natural. El método se aprovecha de una gramática no probabilística y no lexicalizada existente compilada manualmente, y la convierte en una gramática lexicalizada probabilística a través del aprendizaje automático de una especie de los marcos de subcategorización o preferencias de selección para todas las palabras observadas en el corpus de entrenamiento. El diccionario de los marcos de subcategorización o preferencias de selección, obtenido en el proceso de entrenamiento, se puede utilizar posteriormente para la desambiguación sintáctica de nuevos textos no vistos previamente por el algoritmo. El proceso de aprendizaje es no supervisado y no requiere de marcaje manual alguno. El algoritmo de aprendizaje propuesto en este artículo se puede aprovechar de cualquier método de desambiguación existente, incluyendo métodos lingüísticamente motivados, para la filtración o ponderación de los árboles sintácticos alternativos o relaciones sintácticas alternativas, lo que permite la integración del conocimiento lingüístico y el aprendizaje automático no supervisado.

**Palabras clave.** Procesamiento del lenguaje natural, análisis sintáctico, desambiguación sintáctica, aprendizaje automático no supervisado.

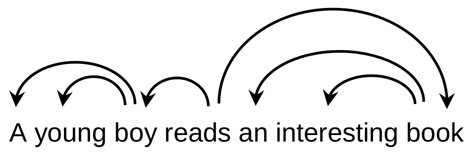
## 1 Introduction

While there exist different definitions of syntactic structure, generally speaking, by syntactic structure of a natural language sentence it is understood a description of which words can be grouped together to form meaningful phrases or which words add details to other words. For example, in a sentence *A young boy reads an interesting book*, the following structure of nested phrases can be observed:

[[*A young boy*] [*reads* [*an interesting book*]],

where, for example, [*A young boy*] and [*an interesting book*] are names of an entities and [*reads* [*an interesting book*] is a name of an action that one of these entities performs.

Another way of describing relationships between words in a sentence is specifying which words add more details to other words:



Here we indicate that *a* and *young* add more details to *boy*, thus forming for the entity a more specific name than just *boy*. The two ways of representing syntactic information are roughly equivalent, and in this paper, we will use them interchangeably.

For many sentences, however, linguistic rules allow more than one interpretation in terms of such syntactic structure. A notable example is prepositional phrase attachment: a sentence such as *John sees a cat with a telescope* can be interpreted as *John sees* [*a cat with a telescope*] (John sees a cat that has a telescope) or as *John sees* [*a cat*] [*with a telescope*] (John uses a telescope to see a cat).

This phenomenon is called syntactic ambiguity. It is one of the most difficult problems of natural language processing. In many languages syntactic ambiguity is greatly increased by ambiguity of attachment of prepositional phrases, or, more generally, of clauses in specific grammatical cases (grammatical cases roughly correspond to English prepositions, but can be expressed cumulatively within a word by changing its morphological form).

Due to space limitations, in this paper we simplify the discussion by omitting the treatment of a number of syntactic constructions such as English-specific attributive chains, the passive transformations, Spanish impersonal and reflexive constructions, and the handling of morphological ambiguity. Instead, we mostly concentrate on prepositional phrase attachment and similar phenomena (grammatical cases in synthetic languages that have them, such as Turkish, Finnish, or Russian).

In addition, in frame of our methodology we construct a special data set of lexical nature that is useful to resolve the ambiguity related to the use of prepositions and grammatical cases. The same data set, namely a kind of a combinatorial dictionary, is also useful for automatic text generation and even for foreigners learning the language or composing texts in this language.

For this end, we describe an iterative procedure to automatically learn such a data set from a large text corpus and simultaneously resolve the syntactic ambiguity in the same corpus. The data set can later be used for disambiguation of other, unseen texts. We have also used it as a raw material for manual compilation of a human-oriented dictionary of subcategorization frames.

The problem of prepositional phrase attachment, mainly in English, has been addressed using both rule-based approaches [8] and statistical lexical approaches [17]. Correct syntactic analysis is crucial in many important tasks of natural language processing, such as in concept extraction [20–22], which in turn has been demonstrated to be very important in such tasks as sentiment analysis [23, 24] and recognizing textual entailment [18], among others.

On the other hand, various iterative and re-estimation methods are actively used in the field of machine learning to calculate the probabilities used in hidden Markov models, such as Baum-Welch re-estimation method [3], to fit data models, such as the expectation-maximization method, or for grammar induction from very large corpora [19].

While these previous works are based mainly on “tagging” approach to parsing, in our paper we address the problem from the point of view of general task of syntactic disambiguation, i.e., of choosing one of the possible syntactic trees for the whole sentence.

We also connect the technical task of disambiguation with well-known linguistic notions of subcategorization frames and show that the data obtained in disambiguation of a large corpus can be used in semi-automatic compilation of a kind of a combinatory dictionary or selectional preferences dictionary. We also discuss the idea of taking into account the probabilities of the typical errors made by the parser, in addition to the probabilities of some natural language constructions.

The paper is organized as follows. In Section 2, we discuss in more detail the main problem addressed in this paper: the ambiguity of prepositional phrase attachment. In Section 3, we explain the data structure central to our methodology framework: the subcategorization dictionary. In Section 4, we show how the problem of syntactic ambiguity can be solved using the subcategorization dictionary. In Section 5, we explain how such a dictionary can be automatically learnt from syntactically parsed and disambiguated text.

Given that Section 4 and 5 seemingly present a chicken-and-egg problem pattern (we need the dictionary to analyze the training corpus from which we extract the same dictionary), in Section 6 we present an iterative procedure that by bootstrapping solves both problems, building the dictionary and disambiguating the syntactic structure of the corpus's sentences.

In Section 7, we briefly discuss experimental results. Section 8 outlines some ways to generalize our framework and gives some directions for future work. Finally, Section 9 concludes the paper.

## 2 Syntactic Ambiguity

Consider a simple English phrase: *They moved their office from the town to the capital*. Syntactic parsers usually use a morphological representation of the sentence in question, which in this case is:

*NP V NP P NP P NP*,

where *NP* is a noun phrase (or a pronoun), *V* is a verb, and *P* is a preposition. Possible syntactic interpretations of a sentence with such parts of speech of words are as follows (with some simplifications):

1. [*They* [*moved* [*their office*] [*from the town*] [*to the capital*]]]: they moved it from the town; they moved it to the capital.
2. [*They* [*moved* [*their office* [*from the town*]] [*to the capital*]]]: there is an office from the town, and they moved it to the capital.
3. [*They* [*moved* [*their office* [*from the town*] [*to the capital*]]]: there is an "office from the town to the capital," and they moved it.
4. [*They* [*moved* [*their office*] [*from* [*the town* [*to the capital*]]]]]: there is a "town to the capital," and from it, they moved the office.
5. [*They* [*moved* [*their office* [*from* [*the town* [*to the capital*]]]]]]: there is a "town to the capital" and an office from this town, and they moved this office.

In Fig. 1, we show these five structures in a graphical way using the dependency grammar notation introduced in Section 1, which we consider more appropriate for our discussion in the rest of this paper.

Of these five alternative interpretations, a native speaker would most probably choose the structure 1 as the only possible interpretation. Such a decision can be made by means of taking into account some additional information. As we will

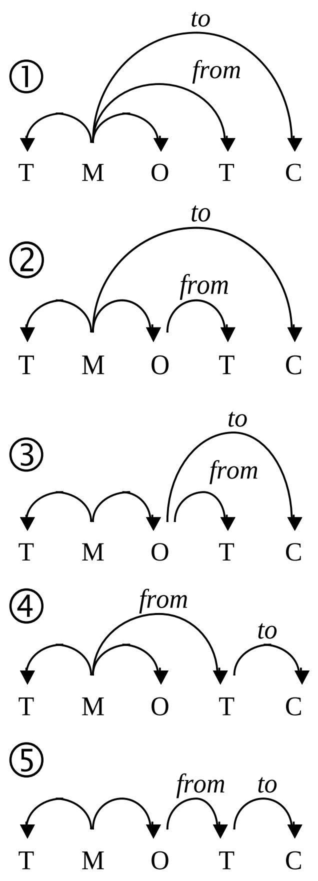


**Fig. 1.** Possible alternative syntactic structures for the sentence *They* (T) *moved* (M) *their office* (O) *from the town* (T) *to the capital* (C), in a simplified dependency representation.

show below, this information can be represented using lexical or syntactic, but not semantic, representation, and a special dictionary can help the parser to resolve the syntactic ambiguity in this sentence.

This type of ambiguity is very common in syntactic analysis. The number of variants grows exponentially with the number of prepositions [10], e.g., the phrase *They moved their office from the town in the North to the capital of the country* has 42 such variants of syntactic structure, etc.

Such ambiguity cannot be resolved by general grammar rules related to the word order even taking into account specific prepositions. Indeed, all the five patterns shown in Fig. 1 are possible in

English with the same prepositions; here are legitimate examples corresponding to each one of the five variants:

1. [*They* [*moved* [*their office*] [*from the town*] [*to the capital*]]]: they moved it from the town; they moved it to the capital.
2. [*They* [*told* [*the news* [*from the town*]] [*to the neighbor*]]]: there is news from the town, and they told it to the neighbor.
3. [*They* [*prohibited* [*any movement* [*from left*] [*to right*]]]: there is a "movement from left to right," and they prohibited it.
4. [*They* [*excluded* [*this word*] [*from* [*the preface* [*to the book*]]]]]: there is a "preface to the book," and from it, they excluded this word.
5. [*They* [*published* [*an excerpt* [*from* [*the preface* [*to the book*]]]]]]: there is a "preface to the book" and an excerpt from this preface, and they published this excerpt.

Therefore, the problem can only be solved by taking into account some *lexical* properties of the words. The example *They moved their office to the capital from the town* shows that these properties are not related with word order in the sentence: the starting and ending points of the movement are still the same, though the word order is different. However, these lexical properties are not of semantic nature. Indeed, if we know that the following seemingly incorrect sentences:

- **They moved their office from the dog to the idea*
- **They sold a book to ten dollars for the customer*

were written by a literate native speaker, we will have to admit that their syntactic interpretation is the pattern number 1, and the semantic interpretation of the second phrase is that *ten dollars* is the buyer and *the customer* is the price, despite of absurdity of such a meaning (this reasonong follows Chomsky's example *Colorless green ideas sleep furiously*).

On the other hand, with other prepositions and the same main words, the sentence has a different structure: *They moved the lawyer's office of the town near the capital* has a legitimate interpretation with the pattern number 5, for example, with the meaning of *move* 'to stir'.

As we see, a dictionary that would list the usage of specific prepositions with specific words is necessary to resolve the cases of ambiguity of this kind. Since no semantic information has to be included in such a dictionary, it is possible to learn such data from a large text corpus using purely statistical methods.

## 3 Subcategorization Frames

While there exists a well-established notion of the dictionary that presents the kind of information discussed here, we will use a simpler structure enough for our purposes, which allows for automatic learning from a text corpus.

### 3.1 Traditional Government Patterns Dictionary

In the Meaning $\Leftrightarrow$ Text Theory [16, 30], a government pattern is defined as a table that enumerates the syntactic valences, called actants, of the word, all possible ways of their expression in sentences, and the limitations on compatibility between them. Such dictionaries are intended in the first place for text generation. In addition, they are very useful for foreigner that learn the language or compose texts in this language.

Fig. 2 shows a greatly simplified example of the entry for the word *move*$_1$ 'to change position' (here the index 1 refers to a specific sense, other senses being *move*$_2$ 'to stir', *move*$_3$ 'to excite', etc.). This example follows the structure of the dictionary that our group has developed for Spanish [6]. In other languages, the representation can slightly vary: for example, a Russian dictionary, instead of, or together with, prepositions, indicates the grammatical cases of the words. The ways of expression of the actants are mutually exclusive: the same actant can be expressed with one of these ways in a specific sentence, mostly depending on the author's stylistic choice. However, not all actants and even not all specific ways of expression of the actants are compatible; such incompatibilities, called restrictions, are indicated in the dictionary (omitted in our example for simplicity).

In the dictionary, the actants are supplied with explanations of their semantic roles. The semantic

| Word: move$_1$ | | |
|---|---|---|
| Agent A transfers object O from starting point S to destination point D by the trajectory T. | | |
| Actant | Ways of expression | Example |
| A = 1: agent | 1.1: Noun (agent) | *a man / the Government moves* |
| O = 2: object | 2.1: Noun (object) | *move their office* |
| S = 3: starting point | 3.1: *from* Noun (location) | *move from the town* |
| | 3.2: *out of* Noun (location) | *move out of the town* |
| D = 4: destination point | 4.1: *to* Noun (location) | *move to the capital* |
| | 4.2: *into* Noun (location) | *move into a new apartment* |
| | 4.3: *towards* Noun (location) | *move towards the exit* |
| T = 5: trajectory | 5.1: *by* Noun (location) | *move by the shore* |
| | 5.2: *through* Noun (location) | *move through the forest* |
| Restrictions: none (for simplicity here we combine the transitive and intransitive senses. For a transitive sense only, the restrictions would mention that the second actant is obligatory). | | |

**Fig. 2.** Example of a government patterns dictionary entry

marks such as *location* or *agent* are intended to help disambiguation. In addition, they can be used in text generation if they are different within the same actant. Such a dictionary enables a program that does not have any semantic information to recognize the structure and the semantic roles in the phrases. It is also necessary for text generation or composition, both by a program or by a person [9].

The knowledge on preposition usage or, more generally, of the ways of expression of syntactic valences (actants), is language-dependent and therefore it cannot be automatically inferred by an algorithm basing solely on semantic information such as sense definitions. For example, for the second actant of the word 'to marry', English uses no preposition at all, Spanish uses the preposition *con* 'with', and similarly in Bulgarian, while Russian (which is a language very similar to Bulgarian) uses the preposition *na* 'on'. Therefore, such information should be explicitly provided by a corresponding language-dependent dictionary.

A traditional government patterns dictionary does not include the ways of expression of circumstances of the words, since this is not lexical knowledge; instead, these ways are fixed for the language in general. For example, in the sentence *They moved their office from the town to the capital at five o'clock on Monday for their convenience*, the ways of expression of the circumstances do not depend on the main word *to move*.

While such dictionaries are of great importance, few attempts have been made to compile such dictionaries or to consistently provide the information on the ways of expression of valences in the common general-purpose dictionaries. The dictionary [4] is the closest to this type for English. Manual compilation of such a dictionary is a very labor-consuming task. Thus, in this paper we present a framework for automatically learning a simpler but still useful similar data structure from raw unprepared text corpora.

## 3.2 Subcategorization Frames for Syntactic Disambiguation

A much simpler structure is enough for purposes of syntactic disambiguation. Such simpler structure can be obtained automatically using the methodology described in this paper. Fig. 3 shows an abridged example of such a structure (this is not a real output of the program: since we worked with

| move | | | |
|---|---|---|---|
| $p^+$ | $p^-$ | Combination | Example |
| 8892 | 3782 | — | *Jill moved impatiently.* |
| 3478 | 921 | to | *John moved to the new apartment.* |
| 372 | 123 | ∅ + from + to | *The firm moved its office from the town to the capital.* |
| 135 | 342 | ∅ + out of | *She moved the table out of the room.* |
| 83 | 58 | ∅ + into | *We moved the device into the house.* |
| 76 | 782 | to + for | *The family moved to the South for sake of the child.* |
| 34 | 89 | ∅ + from + through | *He moved the table from the room through the door.* |
| 30 | 219 | to + at | *Jack moved to the new apartment at five o'clock.* |
| 25 | 38 | to + through | *She moved to the South through the forest.* |
| 9 | 13 | towards | *The group moved towards the mountain.* |
| 1 | 463 | of | *She moved the table of John's friend.* |

**Fig. 3.** An entry of the dictionary used for disambiguation (the last line illustrates an error in the automatically compiles dictionary)

Spanish, we have to use artificial English examples in this paper, with figures chosen manually and only for illustration purposes).

The meaning of the first two columns is discussed in Section 3.3 below. The symbol '∅' denotes the absence of any preposition, i.e., direct object; the symbol '—' denotes the absence of any arguments at all. In this example, we do not consider the first actant, the subject, since it is always attached to the verb in a predictable manner.

Fig. 3 shows the data that can be obtained automatically from raw texts with the procedure discussed in Section 3.4, along with the examples that also can be obtained automatically. There are significant differences between this data structure and the traditional government pattern dictionary shown in Fig. 2:

- Only possible combinations found in the texts are shown in the table, and the valences are not grouped together into actants.
- No information on the semantic roles is provided.
- In Fig. 2, no semantic types of the words, such as *agent* or *location*, are shown. However, if the information on the semantic types is available, it can be added to the table in the same way, e.g., "∅ (object) + *out of* (location)," though this would make the table much larger.
- Because this dataset is meant to be obtained automatically, some erroneous combinations may be present in the list, as it is shown in the last line of the table. However, the weights (see Section 3.3) of these combinations are usually very low—lower than the threshold of eliminating the combinations from the dictionary. Thus, they appear in the final list in very few cases.

On the other hand, the dataset has additional information, and in the first place, the statistical weights discussed in Section 3.3 below.

Because of all these differences, and because the term "government pattern" has a very specific and elaborate meaning in frame of the Meaning ⇔ Text Theory, we prefer to use less specific term for the dataset shown in Fig. 3. Namely, we will use the term subcategorization frame, which seems to less precisely defined in existing literature and used with greater variation.

### 3.3 Positive and Negative Statistical Weights of the Frames

Along with each combination, the number of its occurrences observed in the text corpus, denoted by $p^+$, is included in the table. On the one hand, this number shows the reliability of the information on this combination. On the other hand, in text generation or composition it allows choosing the most common way of expression of the actants. However, the main use of this number for disambiguation is discussed in Section 4 below.

More precisely, this number is not exactly the number of occurrences; instead, it is weighted by the probability of each occurrence to be the true variant of the syntactic structure given its ambiguity, as discussed in Section 2. This is a technical trick, the intended meaning of this figure being just the number of occurrences in the correct structures—however, the weighted number is the best we can obtain without knowing exactly which of the variants of the syntactic structure of the sentences are the correct ones.

More interesting is the second column, $p^–$. This is the number of occurrences of the given combination in the incorrect variants of the syntactic structure of sentences built by a specific parser (we assume here that the parser produces all the possible variants of syntactic interpretation of the same sentence). More precisely, the figure $p^–$ is, similarly to $p^+$, weighted by the probability of each specific variant to be false.

For example, suppose the parser builds all the five possible variants for the phrase shown in Fig. 1, the corpus consists of only one phrase, and we know that the first variant is the correct one. Then the pattern *town to* is assigned the values $p^+ = 0$ and $p^– = 2$, the former one because this combination does not occur in the correct variants of parsing, and the latter one from the variants 4 and 5 known in this case to be incorrect.

If, on the other hand, we have no information on which variant is correct, then we have $p^+ = p^– = 2/5$, i.e., there is no way to extract meaningful disambiguation information from only one phrase. However, below we will show that such information can be automatically extracted from a larger text corpus.

This information—the weights $p^+$ and $p^–$—is used for disambiguation: When a specific combination is observed in one of the variants generated by the parser, is it more probable that this combination is found in a real (correct) structure or that this variant of the structure should be discarded in favor of other variants? Does the parser more frequently detect this combination in correct or in incorrect variants of syntactic structure? The disambiguation procedure that answers these questions is discussed in Section 4.

### 3.4 Computer-aided Compilation of the Traditional Dictionary

While the data structure shown in Fig. 3 is intended primarily for automatic syntactic disambiguation, it is possible to use these data for semi-automatic compilation of a classic human-oriented dictionary such as the one shown in Fig. 2. For this, we used a dialogue procedure. The algorithm of partitioning of the set of prepositions into actants for one entry performs the following steps:

1. The prepositions are grouped together, so that no group contains two prepositions that belong to the same combination. These groups correspond to the hypothetical actants of the word.
2. Of all such possible partitions, the ones that result in the minimum number of groups are chosen.
3. All the possible orders of the set of the groups are considered, with the restriction that the group that contains the direct object must be the second actant. For each such order, a quantitative measure is calculated according to the word order in the combinations: Those variants for which a greater number of combinations agree in the order with the ordering of the groups are scored better.
4. The ordered partitioning with the best score is presented to the human annotator. The annotator can remove some of the prepositions, if he or she considers that they are related to circumstances rather than to actants, or move a preposition to another group. After each action of the annotator, the calculations are repeated taking into account the restrictions introduced by the annotator, and an improved version of partitioning is presented to the annotator.

The process repeats until the annotator accepts a presented version or else chooses to continue manually if the algorithm fails to converge to a satisfactory solution. At each stage, the annotator is presented with the examples, which are also included in the final dictionary.

After the actants have been identified, two kinds of hypotheses are presented to the annotator:

- First, the hypotheses on the obligatory actants. If some actant is present in all the available examples, then the program suggests to the annotator to mark it as obligatory. The verbs with obligatory second actant are called transitive, such as *to give smth.* In English, there exist verbs with two obligatory actants, such as *to tell smth. to smb*.: both sentences **He told this news*, **He told to Jack* are incomplete.
- Second, the hypotheses on incompatibility of actants or individual variants of their expression, e.g., individual prepositions. Only pairs of prepositions are currently considered by our algorithm. If two prepositions belonging to different actants are not found together in the available examples, the algorithm suggests that they are incompatible. Since the number of such hypotheses is often very large, special heuristics are used to order them according to their plausibility. However, we have to admit that this feature was not very useful so far.

While in Fig. 3 only one example is shown, our program collects up to 10 examples for the same combination. They are chosen from the text corpus based on a combined criterion: (1) they cover the corpus approximately proportionally and (2) the examples are kept with the best scores assigned by the procedure described in Section 4. The examples are ordered by the latter scores and the best one is the first to be presented to the annotator. However, the annotator can view all of them and choose the best ones, remove some of them, search the corpus for other examples, or enter new examples manually.

With this, the algorithm can help the annotator to identify syntactic features of words. However, semantic interpretation of the word and the semantic roles corresponding to the syntactic valences are to be added by the annotator manually.

## 4 Disambiguation with Weighted Government Patterns

We assume that the syntactic parser is based on a manually crafted grammar [14], which allows for multiple interpretation of the same sentence with insufficient or no means for deciding which of them is the correct one. This creates syntactic ambiguity.

By disambiguation, we mean assigning to the variants the weights according to the probability (interpreted as plausibility) for the given variant to represent the correct structure of the sentence intended by the author or at least the one that most of native speakers would choose for this sentence.

We consider such a notion of disambiguation better than just to choosing one of the variants and discarding all other variants. In particular, these weights can be used for ordering the variants. The variant with the highest weight is considered first by the subsequent modules of the text processing chain; if for some reason (say, semantic inappropriateness in the given context) it cannot be accepted, the next variant is considered—which is impossible with the parsers that always generate only one output variant thath they consider most likely correct.

In addition, these weights can be combined with other possible estimations of the correctness of the variants. Finally, these weights are used internally by our procedure as described in Section 6.

Therefore, we suppose that a parser is available that builds for each phrase one or more (possibly very many) variants of the syntactic structure. We assume that the parser always builds the correct structure for a sentence, plus possibly some additional, incorrect variants. In this section, we assume that the subcategorization dataset shown in Fig. 2 is given (in Sections 5 and 6 we will discuss how it is constructed).

As an underlying statistical model, we consider the model of information transmission in the presence of noise. The set of the variants generated for one sentence is considered an observable mixture of variants generated by two different sources: the source $S^+$ produces only the correct variants, and the source $S^-$ only incorrect

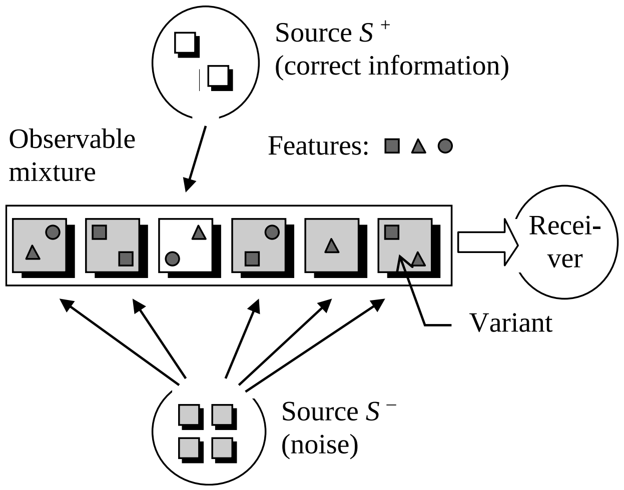


**Fig. 4.** A model of observable mixture of variants from two sources

ones. Each time we receive such a mixture, exactly one variant in it was emitted by the source $S^+$ and all the others, the noise, by the source $S^-$; see Fig. 4.

There is a set of “features” of the variants; for each feature, a variant can have an occurrence of such a feature, several occurrences of it, or none. In our case, each feature is a specific combination of prepositions related with a specific word. The variant has this feature if in the given variant of the syntactic structure this word is connected by dependency relations (Fig. 1) with exactly this set of prepositions (though see Section 8 for generalizations).

The receiver can observe the features, and the task of disambiguation is to guess which variant was issued by the “correct” source $S^+$. In this section, the probabilities for each of the sources, $S^+$ and $S^-$, to assign a given feature to the variant that it issues are considered to be known: these are the values $p^+$ and $p^-$, correspondingly (after normalization so that they sum up to a unity).

We do not describe here the handling of morphological ambiguity [13] that involves the frequencies of specific words and a slightly different handling of the probabilities $p^+$ and $p^-$. In fact, morphological ambiguity is rare in the variants of parsing, since it is usually resolved by the syntactic grammar itself. Thus, we suppose that all variants include the same set of words and prepositions, though connected differently to each other, as in Fig. 1.

Now it is easy to estimate the probability of the hypothesis $H_i$ that the variant number $i$ in the observed mixture of variants is the correct one, i.e., that it was issued by the source $S^+$, we denote this probability by the weight $w_i$ of the hypothesis $H_i$. We assume that the features are numerous so that for each specific feature, the probability *not* to occur in a specific variant is almost 1. A simple reasoning based on the Bayes theorem [12] proves that the weight $w_i$ is a product

$$w_i = C \times p_i^{apr} \times \prod \frac{p_j^+}{p_j^-}$$

by all the features $j$ found in the variant number $i$, where $p_i^{apr}$ is the a priori probability of this hypothesis; $C$ is the normalizing constant, since the total probability of the hypotheses $H_k$ is to be equal to 1. The problem of division by zero never occurs due to the additional constant $\lambda$ introduced in Section 5.

When we use the dataset extracted from a corpus to disambiguate other texts, some combinations are absent in the dictionary. In this case the corresponding factor should be set to a small value $\varepsilon$, since we the probability for the parser to generate an erroneous variant of a new type is greater than the probability to find a new correct combination. This value should be non-zero to allow comparison between variants by other factors.

As we show in Section 5 below, even while learning the probabilities from a corpus, the process speeds up considerably if the combinations with small quotient $p^+ / p^-$ are eliminated from the dictionary.

This policy agrees with the rule of using small values for those combinations that were not found in the dataset; the threshold for eliminating the combinations should be set approximately to $\varepsilon$. With this, each time when the analyzer observes a feature that is not present in the dataset, it can safely assume that this feature was present there with a very small weight but was eliminated in order to save memory and speed up the learning process.

Finally, we can formulate the procedure for assigning the weights to the hypotheses of the syntactic structure of one phrase. It proceeds as follows:

1. All the variants permitted by the grammar used by the parser are built, these are considered as the hypotheses $H_i$.
2. Any available knowledge and procedures are applied to estimate a priori the "quality" $p_i^{apr}$ of each such hypothesis. These procedures can take into account, for example, the length of the dependency links (shorter links are generally scored better), semantic coherence of the structure [11], weights of the grammar rules used in it [2, section 7.6], etc. If no information of such kind is available, then equal weights are assigned to all available hypotheses.
3. For each variant of the syntactic tree, the features of the given variant are looked up in the dictionary. In our case, for each word, the combination of prepositions attached to this word in the current variant of the syntactic tree of the sentence is retrieved from the list. If the combination is found, then the weight $w_i$ of the variant is multiplied by its $p^+ / p^-$, otherwise it is multiplied by $\varepsilon$.
4. The weights $w_i$ are normalized so that $\Sigma\ w_i = 1$ for the variants of the structure of the same sentence.
5. The variants are ordered by the weights $w_i$, and the variant with the greatest such weight is selected as the result of the analysis if only one result is required.

Some generalizations of the proposed framework are discussed in Section 7. However, the changes discussed there only concern the nature of the features operated upon and do not concern the procedure itself.

## 5 Learning the Weighted Subcategorization Frames from a Corpus

In the previous section, we assumed that we have a dictionary of weighted subcategorization frames, and showed how it is used for syntactic disambiguation, that is, for estimating the quality of the variants of the syntactic structure buit by the parser.

Consider now the opposite task: Assume that there is a parser and a disambiguation procedure that assigns the plausibility weights to the hypotheses (variants), and we need to find the frequencies of occurrence of each feature in the correct and wrong variants. In our case, a feature is a word along with the set of dependency links that lead from this word, labeled with their types; see Fig. 5.

If the available disambiguation procedure directly choses the correct variant, then the only thing to do is to increment the counter $p^+$ for all the combinations found in the correct variant, and $p^-$ for all the combinations found in all incorrect ones.

However, if the disambiguation procedure only determines the weights of the variants, then we should consider the same model as shown in Fig. 1, and again apply the Bayes theorem. For each variant, the probability of that it was issued by the "correct" source $S^+$ is $w_i$, and the probability that it was issued by the source of incorrect variants $S^-$ is $1 - w_i$; these values are accumulated over all variants for all sentences in the corpus.

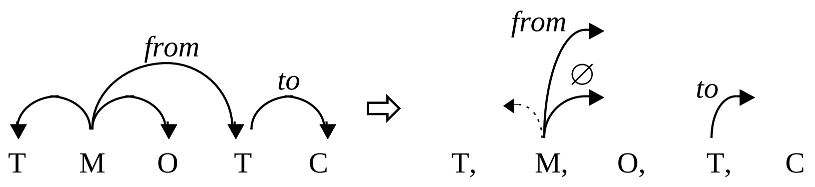


**Fig. 5.** The features of a variant of the syntactic tree: the tree on the left, the five extracted features on the right

To calculate the average values, the total should be divided by the number of variants generated by the sources $S^+$ and $S^-$. Let $V$ be the number of variants generated by the parser and $S$ the number of sentences in the corpus, then the total number of the correct variants is $S$ and of incorrect is $V - S$. Thus, the obtained formulae are as follows:

$$p_j^+ = \frac{\sum w_k}{S},$$
$$p_j^- = \frac{\sum (1 - w_k) + \lambda}{V - S},$$

where the summation is performed over all the occurrences of the feature $j$ in the variants $w_k$, and the meaning of λ is described below.

The formulae work for the ideal case, when the corpus is so large that any possible type of combination or error occurs many times during its analysis. In reality, due to infinite variety of the constructions in open texts, all the possible words and combinations cannot occur in any corpus, even a very large one. What is more, very numerous are the cases that occur in the corpus very few times, or even one time.

Such cases introduce instability in the model since the quotient $p^+ / p^-$ for them is either very big or very small, this value being almost random, since each additional occurrence would greatly affect it. (Though in [8] the significance of rare cases is especially emphasized, we did not observe such an effect, so that smoothing of the rare cases gave much better results.)

There are different methods to suppress such rare cases in the statistical results. We have chosen to artificially add some number λ of occurrences of each combination in the false variants, thus assuming that anything wrong can happen with some small but non-zero probability, and has not yet been observed only because of a small size of the training corpus; because of the Zipf law, no size is large enough.

Our experiments have shown that this method works best. The value of λ was also chosen experimentally: the best results happened to be achieved with λ = $S$.

With this, our procedure for accumulating the statistical weights for the combinations is as follows:

1. All the variants of the structure are built for each phrase of the corpus.

2. The variants for each phrase are evaluated, i.e., are given the probability weights $w_i$ such that $\Sigma\ w_i = 1$ for each phrase, by an external procedure—for example, the one described in Section 4.

3. For each combination found in each variant, the counters $p^+$ and $p^-$ are incremented by the values $w_i$ and $1 - w_i$, respectively. The initial values are zeroes.

4. Finally, the value of λ = $S$ is added to each $p^-$ and the values are divided by S and $V - S$, respectively.

After these values have been determined, the combinations with the quotient $p^+ / p^-$ smaller than some threshold value $\varepsilon$ can be eliminated from the dictionary, as described in Section 6. To speed up the procedure, after Step 2 the variants with the probability $w_i$ lower than some threshold can be ignored.

## 6 Iterative Disambiguation and Learning

In Sections 40 and 5, we have described two procedures, which work in the mutually opposite directions and apparently represent a case of chicken-and-egg problem arrangement: each one of them requires the other to have been solved for its correct functioning.

As a solution of this chicken-and-egg problem, we use these two procedures iteratively, given that we have a large enough training corpus. The work starts from, say, an empty dictionary of subcategorization frames. The disambiguation procedure will then assign equal weights to all the variants or will keep their a priori weights.

At the next step, these (equal) weights of the variants are used to train the model, i.e., to determine the frequencies $p^+$ and $p^-$. However, now the model learns from unequal distribution of features by variants: some features are more frequent. Of special importance are the cases when the sentence has only one or few variants: the features that occur in these variants are likely indicators of a correct variant.

Then the process is repeated: the dictionary of features constructed at the previous step is used to assign new weights to the variants, and these weights are in turn used to construct a better dictionary. Our experiments show that the iterative process converges very quickly.

In this process, the information does not appear from nothing. Throughout the corpus, the variants that share the same combinations are “interconnected” to each other in the model in the sense that re-evaluation of one of them indirectly affects the evaluation of other: they either “help” each other to win the competition within their sets of variants for one sentence, or suppress each other when they lose this competition. Thus the model optimizes itself to the state when the winners in each set have as much as possible in common.

Since the sentences are different, the errors are random, and at the same time, the grammatical sentences have some combinations in common. Thus, these sentences, all together in the corpus, tend to win the competition. The key difference in this process as compared to other machine learning algorithms is that the variants of the analysis of the same sentence are normalized to 1. Thus, there is a positive feedback between variants from different sentences that share the same combinations (features), but negative feedback (inhibition in the terminology of neural network) between variants of analysis of the same sentence.

As to implementation, once the set of the variants and the set of combinations found in these variants have been built, the data structures used in the iterative procedure can be fixed in the computer memory, because all the operations in our procedures are arithmetical and do not produce any new objects. However, depending on the implementation, the time of access to the dictionary can be significantly reduced at the later iterations by eliminating the combinations with the value of $p^+ / p^-$ lower than a pre-set threshold, or by ignoring the variants with the weight lower than some another threshold. After the first iteration, the dictionary is usually very large, but after two or three iterations, nearly only the correct combinations are left in the dictionary, which greatly reduces its size.

## 7 Experimental Results

We experimented with a Spanish corpus and, accordingly, a handcrafted Spanish grammar. In our experiments, two values were measured. One was the similarity between the dictionary built by the program and the manually compiled gold standard dictionary, and the other was the percentage of correctly parsed sentences. By correctly parsed sentences we mean the ones for which the variant with the highest weight was the true one identified by human annotators. The techniques of experiments to measure these two values were different.

To measure the similarity between the dictionaries, a real text corpus could not be used because the “true” dictionary that the authors of the texts had in mind was unknown. Therefore, to check our methodology, we modeled the process of text generation to obtain a quasi-text corpus built with a known dictionary [5]. Only the statistical characteristics of the text were modeled, such as the length of the phrase and, of course, the preposition usage; we paid the main attention to the constructions common in Spanish texts. In addition, with this method, we could measure the percentage of the correctly parsed phrases as well.

In various experiments, we observed all the three patterns of convergence mentioned in the similar context in [15], depending on the formulae and parameters we used, as well as on the size of the corpus. In the *initial maximum* pattern, the dictionary obtained after the first iteration, i.e., with equal weights of the variants, was the best, as well as the percentage of the variants guessed correctly with this dictionary. At the subsequent iterations, both estimations were getting worse.

In the *early maximum* pattern, the best values were achieved after several iterations, and then they slightly degraded. Finally, with the formulae described here and the parameter $\lambda$ of the order of $S$, as described in Section 5, the *classical* pattern was achieved: the values tended to grow and quickly stabilize at the relatively high level. However, even in this case we observed slight elements of the *early maximum* pattern: after reaching the maximum, the percentage of correctly guessed variants fell insignificantly, usually within 1%, and stabilized at that value.

As a measure of similarity between the two dictionaries, the one found by our method and the true one know a priori, we used several measures: the percentage of incorrect combinations, the coverage, and the difference of the probabilities of usage $p^+$ for the correct combinations. After a few iterations these values stabilized at the level around 5% of incorrect combinations and 80% of similarity of the probabilities. The coverage was rather low in our experiments (about 30%) since due to the technical limitations of our program so far we used relatively small corpus.

A typical sequence of the percentages of the correctly guessed variants at consecutive iteration was 37%, 85%, 89%, 90%, 90%, etc., or, taking into account only the phrases for which the parser generated more than one variant, 16%, 80%, 86%, 87%, 87%, etc. As one can observe, the results quickly stabilized. The first figures in both sequences were obtained with the equal weights, by picking an arbitrary variant for each phrase. The last figures of in the sequences show the accuracy reached with the method that is presented here.

Our second set of experiments was carried with real Spanish and Russian text corpora. As a Spanish corpus we used mainly the texts kindly provided to us by the publisher of Gazeta UNAM, the newspaper of UNAM University, Mexico City; the corpus contained approximately 8 million words. We used a very simple context-free parser to build the initial set of the variants; the grammar contained only 41 rule in a language similar to a context-free grammar (with phrase heads marked in each rule).

Then we performed a selective check of the results. This check showed good convergence of the method with the best value reached so far being 78% of correctly parsed phrases; on unseen data analyzed with the dictionary built at the training stage, this figure so far was 69%. Note that all the figures reflect not the number of correctly attached prepositions, but instead the number of correctly parsed entire sentences: if any part of the sentence was not parsed correctly, then the whole sentence was considered parsed incorrectly. In the future, with a more elaborated grammar we expect to achieve better results.

In our experiments, we did not observe any advantage of using some nontrivial initial values for the weights of the hypotheses or for the dictionary—for example, using an initial dictionary close to the true one, or assigning the variants initial weights close to the gold standard. The best results were obtained with equal weights, i.e., with initially empty dictionary. However, the a priori information can be used at each iteration, as it was described in Section 6.

## 8 Generalizations and Future Work

Our methodology method has many possible variants. For example, different kinds of information can be taken into account in the list of combinations. If there is any lexical information available from the parser, such as:

- part of speech (in Spanish prepositions can precede verbs),
- animacy (in Russian this is a morphological characteristic),
- semantic class (such as *person*, *agent*, *living being*, *organization*, *object*, *action*),

etc., then they can be added to the subcategorization frames, provided that the corpus is large enough to avoid the data sparseness problem. In this case, some method of merging the subcategorization frames with similar structure but with different characteristics of the governed word should be applied. For example, if two patterns have comparable weights and differ only in animacy of one of the valences, they should be merged in a common entry without the animacy mark.

Conversely, counting each preposition separately will very significantly reduce the data sparseness problem. For example, instead of one frame such as *move* + *from* + *to* + *through*, three independent frames can be considered: *move* + *from*, *move* + *to*, *move* + *through*, though this may reduce the accuracy when the model is trained on a large enough corpus.

Other generalizations concern the very nature of the objects for which the statistics is gathered, this is why throughout the paper we preferred to refer to them as to abstract "features." First, by such features the grammar rules used in the parsing process can be considered. This will turn the grammar used by the parser into a probabilistic grammar [2]. Second, we expect that with a very large corpus, a similar approach can be applied to word combinations, for both syntactic disambiguation and composition of the dictionary useful for human readers [7]. In particular, all kinds of recently introduced syntactic n-grams [25–29] can be used as features in our method, instead of a specific kind of (incomplete) syntactic n-gram presented in Fig. 5.

All the three methods, namely, the one based on the weights of the subcategorization frames, grammar rules, and word combinations can be used simultaneously as described in Section 4.

Finally, the method can be translated into the language of neural networks. Indeed, the variants of the parsing can be viewed as neurons, a features common to two variants can be viewed as a mutually exciting link, while any two variants belonging to the same phrase can be viewed as mutually inhibiting; disambiguation is viewed as excitement of exactly one neuron in each set of the mutually inhibiting ones. We plan to investigate whether the neural network techniques can increase the performance of the method.

## 9 Conclusions

We have presented a methodological framework for syntactic disambiguation based on the use of a data structure similar to subcategorization frames with statistical weights. The subcategorization frames and their weights are automatically learned from a text corpus. The dictionary of obtained frames can be used for disambiguation of new unseen texts. The method has the following advantages:

- The learning is unsupervised: no manual preparation is required to train the model, apart from writing a small grammar. However, a morphological analyzer or tagger and a syntactic grammar are required, since the method aims to disambiguate the results of an existing parser.
- The method is compatible with other methods of disambiguation, especially with methods that produce an estimation of probability for each variant. This provides an opportunity for incorporating linguistic knowledge and linguistically motivated procedures into a purely statistical unsupervised learning method.
- The results of the application of the method are tuned to a specific parser, taking into account the balance between the correct and wrong assignments of prepositions to words. Note that training the dictionary for any other parser comes at no cost because the method relies on unsupervised learning.
- The data built by the algorithm is lexicalized, so that the amount of processed data does not increase with the growth of the number of rules in the grammar.
- The data set learned from the corpus is useful for semi-automatic compilation of a traditional government patterns dictionary, which is used both for semantic analysis in natural language processing and as learning and authoring aid by the foreigners that compose texts in the given language.
- The subcategorization frames used by the method correspond to some linguistic reality—unlike, say, the probabilities used by the Hidden Markov Model or neural network methods.

We believe that the latter means that the native speakers are aware of such a reality and, according to Grice's Cooperative Principle, in text composition intentionally try to avoid constructions that would be misleading with respect to subcategorization frames of the words, cf.: ?*They laughed at this place* vs. *They laughed here*, ?*He spoke with the director of the new plan* vs. *He spoke of the new plan with the director*.

## Acknowledgements

This work was partially supported by the Government of Mexico via the Instituto Politécnico Nacional grant SIP 20144534 and SNI, and the European Union via the European Commission project 269180 FP7-PEOPLE-2010-IRSES: Web Information Quality–Evaluation Initiative (WIQ-EI).

## References

1. **Alfared, R. & Béchet, D. (2012).** POS taggers and dependency parsing. *International Journal of Computational Linguistics and Applications*, 3(2), 107–122.
2. **Allen, J. (1995).** *Natural language understanding*. The Benjamin/Cummings Publishing Company, Inc.
3. **Baum, L.E. (1972).** An inequality and associated maximization technique in statistical estimation for probabilistic functions of a Markov process. *Inequalities*, 3, 1–8.
4. **Benson, M., Benson, E., & Ilson, R. (1986).** *The BBI Combinatory dictionary of English*. John Benjamins Publishing Co.
5. **Bolshakov I.A., Gelbukh, A., & Galicia-Haro, S. (1998).** Simulation in linguistics: assessing and tuning text analysis methods with quasi-text generators. *International workshop on computational linguistics and its applications, Dialogue-98*, Khazan, Russia.
6. **Bolshakov, I.A., Gelbukh, A., Galicia Haro, S., & Orozco Guzmán, M. (1998).** *Government patterns of 670 Spanish verbs*. Technical report, Serie Roja, N 35. CIC, IPN, 1998, 65 pp.
7. **Bolshakov I.A., Cassidy, P.J., & Gelbukh, A. (1995).** CrossLexica: a dictionary of word combinations and a thesaurus of Russian (in Russian). *International workshop on computational linguistics and its applications, Dialogue-95*, Khazan, Russia.
8. **Brill, E., & Resnik, P. (1994).** A rule-based approach to prepositional phrase attachment disambiguation. *ACL*, Kyoto, Japan.
9. **Castro-Sánchez, N.A., & Sidorov, G. (2010).** Analysis of definitions of verbs in an explanatory dictionary for automatic extraction of actants based on detection of patterns. *Lecture Notes in Computer Science*, 6177, 233–239.
10. **Church, K., & Patil, R. (1982).** Coping with syntactic ambiguity, or how to put the block in the box on the table. *American Journal of Computational Linguistics*, 8(3–4), 139–149.
11. **Gelbukh, A. (2012).** Ontology-based semantic relatedness measures: Applications and calculation. *Research in Computing Science*, 47, 117–138.
12. **Gelbukh, A., Bolshakov, I.A., & Galicia-Haro, S. (1998).** Statistics of parsing errors can help syntactic disambiguation. *CIC-98, Simposium Internacional de Computación*, Mexico, 405–515.
13. **Gelbukh, A., Sidorov, G., & Velásquez, F. (2003).** Análisis morfológico automático del español a través de generación. *Escritos*, 28, 9–26.
14. **Gelbukh, A., Sidorov, G., Galicia Haro, S., & Bolshakov, I.A. (2002).** Environment for development of a natural language syntactic analyzer. *Acta Academia*, 206–213.
15. **Elworthy, D. (1994).** Does Baum-Welsh re-estimation help taggers? *Fourth Conference on Applied Natural Language Processing*, Germany.
16. **Mel'čuk, I.A. (1974).** An *experience of the theory of Meaning ⇔ Text models* (in Russian). Nauka, Moscow.
17. **Merlo, P., Crocker, M., & Berthouzoz, C. (1997).** Attaching multiple prepositional phrases: Generalized backed-off estimation. *EMNLP-2*, Brown University Providence, Rhode Island, USA.
18. **Pakray, P., Poria, S., Gelbukh, A., & Bandyopadhyay, S. (2011).** Semantic textual entailment recognition using UNL. *Polibits*, 43, 23–27.
19. **Pereira, F., & Schabes, Y. (1992).** Inside-outside reestimation from partially bracketed corpora. *ACL*, University of Delaware, Newark, Delaware, USA.
20. **Poria, S., Agarwal, B., Gelbukh, A., Hussain, A., & Howard, N. (2014).** Dependency-based semantic parsing for concept-level text analysis. *15th International Conference on Intelligent Text Processing and Computational Linguistics, CICLing 2014*. *Lecture Notes in Computer Science*, 8403, 113–127.
21. **Poria, S., Cambria, E., Winterstein, G., & Huang, G.-B. (2014).** Sentic patterns: Dependency-based

rules for concept-level sentiment analysis. Knowledge-Based Systems, in press.

22. **Poria, S., Gelbukh, A., Agarwal, B., Cambria, E., & Howard, N. (2014).** Sentic Demo: A hybrid concept-level aspect-based sentiment analysis toolkit. *ESWC 2014*, Crete, Greece.

23. **Poria, S., Gelbukh, A., Cambria, E., Hussain, A., & Huang, G.-B. (2014).** EmoSenticSpace: A novel framework for affective common-sense reasoning. *Knowledge-Based Systems*, in press.

24. **Poria, S., Gelbukh, A., Hussain, A., Das, D., & Bandopadhyay, S. (2013).** Enhanced SenticNet with affective labels for concept-based opinion mining. *IEEE Intelligent Systems*, 28(2), 31–38.

25. **Sidorov, G. (2013).** Non-continuous syntactic n-grams. *Polibits*, 48, 67–75.

26. **Sidorov, G. (2013).** Syntactic Dependency Based N-grams in rule based automatic English as second language grammar correction. *International Journal of Computational Linguistics and Applications*, 4(2), 169–188.

27. **Sidorov, G. (2013).** *Non-linear construction of n-grams in computational linguistics: syntactic, filtered, and generalized n-grams*, 166 p.

28. **Sidorov, G., Velasquez, F., Stamatatos, E., Gelbukh, A., & Chanona-Hernández, L. (2012).** Syntactic Dependency-based n-grams as classification features. *Lecture Notes in Artificial Intelligence*, 7630, 1–11.

29. **Sidorov, G., Velasquez, F., Stamatatos, E., Gelbukh, A., & Chanona-Hernández, L. (2012).** Syntactic n-grams as machine learning features for natural language processing. *Expert Systems with Applications*, 41(3), 853–860.

30. **Steel, J. (ed.). (1990).** *Meaning – Text Theory. Linguistics,* lexicography*, and implications.* University of Ottawa press.

**Alexander Gelbukh** received MSc in mathematics from the Lomonosov Moscow State University, Russia, and PhD in computer science from VINITI, Russia. He is currently a Research Professor and Head of the Natural Language Processing Laboratory of the Centro de Investigación in Computación (CIC) of the Instituto Politécnico Nacional (IPN), Mexico, and the President of the Mexican Society of Artificial Intelligence (SMIA). He is a Member of the Mexican Academy of Sciences and National Researcher of Mexico (SNI) at Excellence level 2. He is author, co-author, or editor of more than 500 research publications in natural language processing and artificial intelligence.

*Article received on 12/03/2014, accepted on 02/06/2014.*