# DILUCT: An Open-Source Spanish Dependency Parser based on Rules, Heuristics, and Selectional Preferences*

Hiram Calvo, Alexander Gelbukh

Natural Language Processing Laboratory,
Center for Computing Research, National Polytechnic Institute
Mexico City, 07738, Mexico
likufanele@likufanele.com, gelbukh@gelbukh.com
www.Likufanele.com, www.Gelbukh.com

**Abstract.** In several natural language applications such as text mining, information retrieval or question answering, it is convenient to have a structured representation of sentences so that formal transformations on language (queries, database storing, etc.) can be done. We present a simple and robust dependency parser for Spanish which produces such structure. The algorithm uses heuristic rules to infer dependency relationships between words, and word co-occurrence statistics (learnt in an unsupervised manner) to resolve ambiguities such as prepositional phrase attachment. If a complete parse cannot be produced, a partial structure is built with some (if not all) dependency relations identified. Evaluation shows that in spite of its simplicity, the parser's accuracy is superior to the available existing parsers for Spanish. Though certain grammar rules, as well as the lexical resources used, are specific for Spanish, the suggested approach is language-independent.

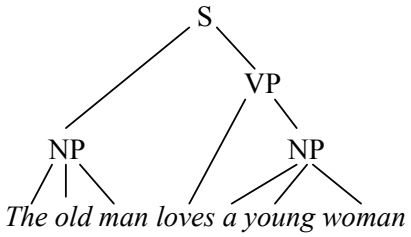
## 1 Introduction

Many natural language applications require identifying properly structures present in sentences, namely entities, compound verbs, and not less importantly, the relations among them. Obtaining such structure representation can be seen as performing a full syntactic analysis with semantic elements helping to build it. There are several approaches to syntactic analysis: those oriented to the constituency and dependency structure, respectively. In the constituency approach, the structure of the sentence is described by grouping words together and specifying the type of each group, usually according to its main word [13]:

$$[[\text{The old man}]_{NP}\ [\text{loves}\ [\text{a young woman}]_{NP}]_{VP}]_{S}$$

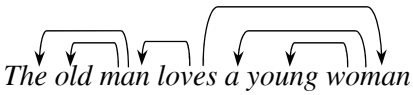
Here NP stands for noun phrase, VP for verb phrase, and S for the whole sentence. Such a tree can also be represented graphically:

* This work was done under partial support of Mexican Government (SNI, CGPI-IPN, COFAA-IPN, and PIFI-IPN). The authors cordially thank Jordi Atserias for providing the data on the comparison of TACAT parser with our system.

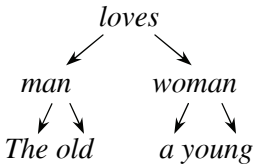



where the nodes stand for text spans (constituents) and arcs for "consists of" relationship.

In dependency approach, words are considered "dependent" from, or modifying, other words [13]. A word modifies another word (governor) in the sentence if it adds details to the latter, while the whole combination inherits the syntactic (and semantic) properties of the governor: *old man* is a kind of *man* (and not a kind of *old*); *man loves woman* is a kind of (situation of) *love* (and not, say, a kind of *woman*). Such dependency is represented by an arrow from the governor to the governed word:




or, in a graphical form:




where the arcs represent the dependency relation between individual words, the words of the lower levels contributing details to those of the upper levels while preserving the syntactic properties of the latter.

In spite of the 40-year discussion in literature, there is no consensus as to which formalism is better. Though combined formalisms such as HPSG [28] have been proposed, they seem to bear the heritage of the advantages as well as disadvantages of both approaches, the latter impeding their wide use in natural language processing practice. Probably the pertinence each approach depends on a specific task.

We had two-fold motivation for this work. One task we had in mind was the study of lexical compatibility of specific words, and in particular, compilation and use of a dictionary of collocations—stable or frequent word combinations, such as *eat bread* or *deep sleep* as opposed to **eat sleep* and **deep bread* [4]. Such combinations were shown to be useful in tasks ranging from syntactic analysis [33] and machine translation [5] to semantic error correction [6] and steganography [2]. Dependency approach to syntax seems to be much more appropriate for such task.

Our second motivation was the construction of semantic representation of text, even if partial, for a range of applications from information retrieval and text mining [24, 25] to software specifications [17]. All known semantic approaches—such as conceptual graphs [29], Minimal Recursion Semantics [15], or semantic networks [22]—roughly resemble a set of predicates, where individual words represent predi-

cates or their arguments (which in turn can be predicates). The resulting structures are in much closer direct correspondence with the dependency tree than with a constituency tree of the sentence in question, so that dependency syntax seems to be more appropriate for direct translation into semantic structures. Specifically, dependency structure makes it much easier matching—say, in information retrieval—paraphrases of the same meaning (such as active/passive voice transformation) or transforming from one such synonymous structure to another one.

In addition, we found that a dependency parser can be much easier made robust than a constituency parser. The known approaches to dependency parsing much easier cope with both incomplete grammars and ungrammatical sentences than the standard approaches to context-free parsing.

Indeed, a standard context-free parser builds the structure incrementally, so that failure of constructing a constituent implies the impossibility to construct all the further constituents that should have contained this one. What is more, an incorrect decision on an early stage of parsing leads to completely or largely incorrect final result.

In contrast, in dependency parsing the selection of a governor for a given word, or the decision on whether the given two words are connected with a dependency relation, is much more (though not at all completely) decoupled from the corresponding decision on another pair of words. This makes it possible to continue the parsing process even if some of such decisions could not be made successfully. The resulting structure can prove to be incomplete (with some relationships missing) or not completely correct (with some relationships wrongly identified). However, an incorrect decision on a particular pair of words usually does not cause a snowball of cascaded errors at the further steps of parsing.

In this paper we present DILUCT, a simple robust dependency parser for Spanish. Though some specific rules, as well as the lexical resources and the preprocessing tools used, are specific for Spanish, the general framework is language-independent. An online demo and the source code of the system are available online.[1]

The parser uses an ordered set of simple heuristic rules to iteratively determine the dependency relationships between words not yet assigned to a governor. In case of ambiguities of certain types, word co-occurrences statistics gathered in an unsupervised manner from a large corpus or from the Web (through querying a search engine) is used to select the most probable variant. No manually prepared tree-bank is used for training.

We evaluated the parser by counting the number of correctly identified dependency relationship on a relatively small tree-bank. The experiments showed that the accuracy of our system is superior to that of existing Spanish parsers, such as TACAT [12] and Connexor.

The rest of the paper is organized as follows. In Section 2 we discuss the existing approaches to dependency parsing that have influenced our work. In Section 3 we present our algorithm, and in Section 4 give the evaluation results. Section 5 concludes the paper.

[1] www.likufanele.com/diluct

## 2 Related Work

Dependency approach to syntax was first introduced by Tesnière [32] and further developed by [22], who extensively used it in his Meaning ⇔ Text Theory [21, 30] in connection to semantic representation as well as with a number of lexical properties of words, including lexical functions [23, 3].

One of the first serious attempts to construct a dependency parser we are aware about was the syntactic module of the English-Russian machine translation system ETAP [1]. The parsing algorithm consists of two main steps:

1. All individual word pairs with potentially plausible dependency relation are identified.
2. So-called filters remove links incompatible with other identified links.
3. Of the remaining potential links, a subset forming a tree (namely, a projective tree except for certain specific situations) is chosen.

In ETAP, the grammar (a compendium of situations where a dependency relation is potentially plausible) is described in a specially developed specification language describing the patterns to be searched for in the sentence and the actions on constructing the tree that are to be done when such a pattern is found. Both the patters and the actions are expressed in semi-procedural way, using numerous built-in functions (some of which are language-dependent) used by the grammar interpreter. An average pattern–action rule consists of 10–20 lines of tight code. To our knowledge, no statistical information is currently used in the ETAP parser.

Our work is inspired by this approach. However, we made the following main design decisions different from those of ETAP. First, our parser is meant to be much simpler, even if at the cost of inevitable loss of accuracy. Second, we do not rely on complex and detailed lexical recourses. Third, we do rely on word co-occurrences statistics, which we believe to compensate for the lack of completeness of the grammar.

Indeed, Yuret [33] has shown that co-occurrence statistics (more precisely, a similar measure that he calls *lexical attraction*) alone can provide enough information for highly accurate dependency parsing, with no hand-made grammar at all. In his algorithm, of all projective trees the one that provides the highest total value of lexical attraction of all connected word pairs is selected. However, his approach relies on huge quantities of training data (though training is unsupervised). In addition, it only can construct projective trees (a tree is called projective if it has no crossing arcs in the graphical representation shown in Section 1).

We believe that a combined approach using both a simple hand-made grammar and word co-occurrence statistics learned in an unsupervised manner from a smaller corpus provides a reasonable compromise between accuracy and practical feasibility.

On the other hand, the mainstream of current research on dependency parsing is oriented to formal grammars [16]. In fact, the HPSG grammar [27] was perhaps one of the first successful attempts to, in effect, achieve a dependency structure (necessary for both using lexical information in the parser itself and constructing the semantic representation) by using a combination of constituency and dependency machinery. As we have mentioned, low robustness is a disadvantage of non-heuristically-based approaches.

Of syntactic parsers with realistic coverage available for Spanish we can mention the commercially available XEROX parser[2] and Connexor Machinese Syntax[3] and the freely available parser TACAT.[4] We used the latter two systems to compare their accuracy with that of our system. Only Connexor's system is really dependency-based, relying on the Functional Dependency Grammar formalism [31], the other systems being constituency-based.

## 3 Algorithm

Following the standard approach, we first pre-process the input text—which basically includes tokenizing, sentence splitting, tagging, and lemmatizing—and then apply the parsing algorithm proper.

### 3.1 Preprocessing

**Tokenization and sentence splitting:** The text is tokenized into words and punctuation marks and split into sentences.

We currently do not distinguish punctuation marks; thus each punctuation mark is substituted with a comma (in the future we will consider different treatment for different punctuation marks).

Two compounds of article and preposition are split: *del* → *de el* 'of the', *al* → *a el* 'to the'.

Compound prepositions represented in writing as several words are jointed into one word, for example: *con la intención de* 'in order to', *a lo largo de* 'throughout', etc. Similarly treated are a few adverbial phrases such as *a pesar de* 'in spite of', *de otra manera* 'otherwise', etc., and several pronominal phrases such as *sí mismo* 'itself'. The list of such combination is small (currently including 62 items) and closed. Though we currently do not perform named entity recognition, we plan this for the future.

**Tagging:** The text is POS-tagged using the TnT tagger [7] trained on the Spanish corpus CLiC-TALP.[5] This tagger has a performance of over 94% [26].

We also correct some frequent errors of the TnT tagger, for example:

| Rule | Example |
|---|---|
| Det Adj V → Det S V | *el inglés vino*<br>'the English(man) came' |
| Det Adj Prep → Det S Prep | *el inglés con*<br>'the English(man) with' |

[2] which used to be on www.xrce.xerox.com/research/mltt/ demos/spanish.html, but seems to be removed recently.

[3] www.connexor.com/demo/syntax.

[4] www.lsi.upc.es/~nlp/freeling/demo.php.

[5] clic.fil.ub.es.

**Lemmatizing:** We use a dictionary-based Spanish morphological analyzer [18].[6] In case of ambiguity the variant of the part of speech (POS) reported by the tagger is selected, with the following exceptions:

| Tagger predicted | Analyzer found | Example |
|---|---|---|
| Adjective | Past participle | *dado* 'given' |
| Adverb | Present participle | *dando* 'giving' |
| Noun | Infinitive | *dar* 'to give' |

If the analyzer does not give an option in the first column but does give one in the second column, the latter is accepted.

If an expected nouns, adjective, or participle is not recognized by the analyzer, we try removing a suffix removal, e.g., *flaquito* → *flaco* 'little (and) skinny → skinny.' For this, we try removing a suspected suffix and check whether the word is recognized by the morphological analyzer. Examples of the suffix removal rules are:

| Rule | Example |
|---|---|
| *-cita* → *-za* | *tacita* → *taza* |
| | 'little cup → cup' |
| *-quilla* → *-ca* | *chiquilla* → *chica* |
| | 'nice girl → girl' |

### 3.2 Rules

Parsing rules are applied to the lemmatized text. Following an approach similar to [1,9], we represent a rule as a sub-graph, e.g., N ← V. Application of a rule consists in the following steps:

1. A substring matching the sequence of the words in the rule is searched for in the sentence.
2. Syntactic relations between the matched words are established according to those

| Sentence | Rule |
|---|---|
| *Un*(Det) *perro*(N) *grande*(Adj) *ladra* (V) | Det ← N |
| *perro*(N) *grande*(Adj) *ladra* (V)<br>↓<br>*Un*(Det) | N → Adj |
| *perro*(N) *ladra* (V)<br><br>*Un*(Det) *grande*(Adj) | N ← V |
| *ladra* (V)<br>↓<br>*perro*(N)<br><br>*Un*(Det) *grande*(Adj) | Done |

**Figure 1. Applying rules to parse *Un perro grande ladra* 'a big dork barks**

[6] www.Gelbukh.com/agme.

specified in the rule.

3. All words that have been assigned a governor by the rule are removed from the sentence in the sense that they do not participate in further comparisons at step 1.

For example, for the sentence *Un perro grande ladra* ‘a big dog barks’ see Figure 1.

As it can be seen from the example, the order of the rule application is important. The rules are ordered; at each iteration of the algorithm, the first applicable rule is applied, and then the algorithm repeats looking for an applicable rule from the first one. The processing stops when no rule can be applied.

Table 1. Grammar rules for parsing

| Rule | Example |
|---|---|
| Auxiliary verb system and verb chains | |
| *estar* \| *andar* ← Ger | *estar comiendo* ‘to be eating’ |
| *haber* \| *ser* ← Part | *haber comido* ‘to have eaten’ |
| *haber* ← *estado* ← Ger | *haber estado comiendo* ‘have been eating’ |
| $ir_{pres}$ *a* ← Inf | *ir a comer* ‘to be going to eat’ |
| $ir_{pres}$ ← Ger ← Inf | *ir queriendo comer* ‘keep wanting to eat’ |
| V → *que* → Inf | *tener que comer* ‘to have to eat’ |
| V → V | *querer comer* ‘to want to eat’ |
| Standard constructions | |
| Adv ← Adj | *muy alegre* ‘very happy’ |
| Det ← N | *un hombre* ‘a man’ |
| N → Adj | *hombre alto* ‘tall man’ |
| Adj ← N | *gran hombre* ‘great man’ |
| V → Adv | *venir tarde* ‘come late’ |
| Adv ← V | *perfectamente entender* ‘understand perfectly’ |
| Conjunctions (see explanation below) | |
| N Conj N V(pl) ⇒ [N N] V(pl) | *Juan y María hablan* ‘John and Mary speak’ |
| X Conj X ⇒ [X X]<br>(X stands for any) | (*libro*) *nuevo e interesante* ‘new and interesting (book)’ |
| Other rules | |
| N → *que* V | *hombre que habla* ‘man that speaks’ |
| *que* → V | *que habla* ‘that speaks’ |
| N X *que*<br>(X stands for any) | *hombre tal que* ‘a man such that’; *hombre , que* ‘man, which’ |
| Det ← Pron | *otro yo* ‘another I’ |
| V → Adj | *sentir triste* ‘to feel sad’ |
| N , Adj | *hombre , alto* ‘man , tall’ |
| N , N | *hombre , mujer* ‘man , woman’ |
| N → Prep → V | *obligación de hablar* ‘obligation to speak’ |
| V , V | *comer , dormir* ‘eat , sleep’ |
| V Det ← V | *aborrecer el hacer* ‘hate doing’ |

peated modifiers. For example, in the phrases *el otro día* 'the other day' or *libro nuevo interesante* 'new interesting book' the two determiners (two adjectives, respectively) will be connected as modifiers to the noun by the same rule Det ← N (N → Adj, respectively) at two successive iterations of the algorithm.

Our rules are not yet fully formalized (this is why we call our approach semi-heuristic), so in what follows we will give additional comments to some rules. Currently our grammar includes the rules sown in **Table 1**[7]

Coordinative conjunctions always have been a pain in the neck of dependency formalisms and an argument in favor of constituency approaches. Following the idea of Gladki [20], we represent coordinated words in a constituency-like manner, joining them in a compound quasi-word. In the resulting "tree" we effectively duplicate (or multiply) each arc coming to, or outgoing from, such a special node. For example, a fragment [*John Mary*] ← *speak* (*John and Mary speak*) is interpreted as representing two relationships: *John* ← *speak* and *Mary* ← *speak*; a fragment *merry* ← [*John Mary*] ← *marry* (*Merry John and Mary marry*) yields for dependency pairs: *merry* ← *John* ← *marry* and *merry* ← *Mary* ← *marry*. We should note that currently this machinery is not yet fully implemented in our system.

Accordingly, our rules for handling conjunctions have are rewriting rules rather than tree construction rules. The first rule forms such a compound quasi-word out of two coordinated nouns if they precede a plural verb. The rule eliminates the conjunction, since in our implementation conjunctions do not participate in the tree structure. Basically what the rule does is to assure that the verb having such a compound subject is plural, i.e., to rule out the interpretation of *John loves Mary and Jack loves Jill* as *John loves* [*Mary and Jack*] *loves Jill*.

### 3.3 Prepositional Phrase Attachment

This stage is performed after the stage of application of the rules described in the previous section.

For any preposition that have not yet been attached to a governor, its compatibility with every noun and every verb in the sentence is evaluated using word co-occurrence statistics (which can be obtained by a simple query to an Internet search engine). The obtained measure is combined with a penalty on the linear distance: the more distant is a potential governor from the preposition in question the less appropriate it is for attachment. More details on the statistical technique of prepositional used here can be found in [10].

### 3.4 Heuristics

The heuristics are applied after the stages described in the previous sections. The purpose of the heuristics is to attach the words that were not assigned any governor in the rule application stage.

---

[7] The bar | stands for variants: *estar* | *andar* ← Ger stands for two rules, *estar* ← Ger and *andar* ← Ger.

The system currently uses the following heuristics, which are iteratively applied in this order, in a manner similar to how rules are applied:

1. An unattached *que* 'that, which' is attached to the nearest verb (to the left or to the right of the *que*) that does not have another *que* as its immediate or indirect governor.
2. For an unattached pronoun is attached to the nearest verb that does not have a *que* as its immediate or indirect governor.
3. An unattached N is attached to the most probable verb that does not have a *que* as its immediate or indirect governor. For estimating the probability, an algorithm similar to the one described in the previous section is used. The statistics described in [11] are used.
4. For an unattached verb *v*, the nearest another verb *w* is looked for to the left; if there is no verb to the left, then the nearest one to the right is looked for. If *w* has a *que* as direct or indirect governor, then *v* is attached to this *que*; otherwise it is attached to *w*.
5. An unattached adverb or subordinative conjunction (except for *que*) is attached to the nearest verb (to the left or to the right of the *que*) that does not have another *que* as its immediate or indirect governor.

Note that if the sentence contains more than one verb, at the step 4 each verb is attached to some another verb, which can result in a circular dependency. However, this does not harm since such a circular dependency will be broken in the last stage of processing.

### 3.5 Selection of the Root

The structure constructed at the steps of the algorithm described in the previous sections can be redundant. In particular, it can contain circular dependencies between verbs. The final step of analysis is to select the most appropriate root.

We use the following simple heuristics to select the root. For each node in the obtained digraph, we count the number of other nodes reachable from the given one through a directed path along the arrows. The word that maximizes this number is selected as the root. In particular, all its incoming arcs are deleted from the final structure.

## 4 Evaluation

We present in this section a comparison of our parser against a hand-tagged gold standard. We also compare our parser with two widely known parsers for Spanish. The first one is Connexor Machinese Syntax for Spanish, a dependency parser, and TACAT, a constituency parser.

We have followed the evaluation scheme proposed by [8], which suggests evaluating parsing accuracy based on grammatical relations between lemmatized lexical heads. This scheme is suitable for evaluating dependency parsers and constituency parsers as well, because it considers relations in a tree which are present in both formalisms, for example [Det *car the*] and [DirectObject *drop it*]. For our purposes of evaluation we translate the output of the three parsers and the gold standard into a

Table 2. Triples extracted for the sentence: *El más reciente caso de caridad burocratizada es el de los bosnios, niños y adultos.*

| Spanish triples | gloss | 3LB | Connexor | DILUCT | TACAT |
|---|---|---|---|---|---|
| adulto DET el | 'the adult' | x | | | |
| bosnio DET el | 'the bosnian' | x | x | x | |
| caridad ADJ burocratizado | 'bureaucratized charity' | x | | x | x |
| caso ADJ reciente | 'recent case' | x | | x | x |
| caso DET el | 'the case' | x | | x | x |
| caso PREP de | 'case of' | x | x | x | x |
| de DET el | 'of the' | x | | | x |
| de SUST adulto | 'of adult' | x | | | |
| de SUST bosnio | 'of bosnian' | x | | x | |
| de SUST caridad | 'of charity' | x | x | x | x |
| de SUST niño | 'of children' | x | | | |
| niño DET el | 'the child' | x | | | |
| reciente ADV más | 'most recent' | x | | | x |
| ser PREP de | 'be of' | x | | x | x |
| ser SUST caso | 'be case' | x | | x | x |
| recentar SUST caso | '*to recent* case' | | x | | |
| caso ADJ más | 'case most' | | | x | |
| bosnio SUST niño | 'bosnian child' | | | x | |
| ser SUST adulto | 'be adult' | | | x | |
| de , | 'of ,' | | | | x |
| , los | ', the' | | | | x |
| , bosnios | ', Bosnian' | | | | x |

series of triples including two words and their relationship. Then the triples of the parsers are compared against the triples from the gold standard to find a correspondence.

We have chosen the corpus Cast3LB as our gold standard because it is, until now, the only syntactically tagged corpus for Spanish that is widely available. Cast3LB is a corpus consisting of 100,000 words (approximately 3,700 sentences) extracted from two corpora: the CLiCTALP corpus (75,000 words), a balanced corpus containing literary, journalistic, scientific, and other topics; the second corpus was the EFE Spanish news agency (25,000 words) corresponding to year 2000. This corpus was annotated following [14] using the constituency approach, so that we first converted it to a dependency treebank. A rough description of this procedure follows. For details, see [11].

1. Extract patterns from the treebank to form rules. For example, a node called NP with two children, Det and N yields the rule NP  Det N

2. Use heuristics to find the head component of each rule. For example, a noun will always be the head in a rule, except when a verb is present. The head is marked with the @ symbol: NP  Det @N.
3. Use this information to establish the connection between heads of each constituent
4. Extract triples for each dependency relation in the dependency tree-bank.

As an example, consider Table 2. It shows the triples for the sentence taken from Cast3LB. *El más reciente caso de caridad burocratizada es el de los bosnios, niños y adultos.* 'The most recent case of bureaucratized charity is the one about the Bosnian, children and adult.' In some cases the parsers extract additional triples not found in the gold standard.

We extracted 190 random sentences from the 3LB tree-bank and parsed them with Connexor and DILUCT. Precision, recall and F-measure of the different parsers against Cast3LB are as follows.

| | Precision | Recall | F-measure |
|---|---|---|---|
| Connexor | 0.55 | 0.38 | 0.45 |
| DILUCT | 0.47 | 0.55 | 0.51 |
| TACAT[8] | – | 0.30 | – |

Note that the Connexor parser, though has a rather similar F-measure as our system, is not freely available and of course is not open-source.

## 5 Conclusions

We have presented a simple and robust dependency parser for Spanish. It uses simple hand-made heuristic rules for the decisions on admissibility of structural elements and on word co-occurrence statistics for disambiguation. The statistics is learned from a large corpus, or obtained by querying an Internet search engine, in an unsupervised manner—i.e., no manually created tree-bank is used for training. In case if the parser cannot produce a complete parse tree, a partial structure is returned consisting of the dependency links it could recognize.

Comparison of the accuracy of our parser with two the available systems for Spanish we are aware of shows that our parser outperforms both of them.

Though a number of specific rules of the grammar are specific for Spanish, the approach itself is language-independent. As future work we plan to develop similar parsers for other languages, including English, for which the necessary preprocessing tools—such as POS tagger and lemmatizer—are available.

As other future work direction we could mention in the first place improvement of the system of grammar rules. The current rules sometimes do their job in a quick-and-dirty manner, which results in just the right thing to do in most of the cases, but can be done with greater attention to details.

[8] Results for TACAT were kindly provided by Jordi Atserias.

Finally, we plan to evaluate the usefulness of our parser in real tasks of information retrieval, text mining, and constructing semantic representation of the text, such as conceptual graphs.

## References


1. Yuri D. Apresyan, Igor Boguslavski, Leonid Iomdin, Alexandr Lazurski, Nikolaj Pertsov, Vladimir Sannikov, Leonid Tsinman. 1989. *Linguistic Support of the ETAP-2 System* (in Russian). Moscow, Nauka.
2. Igor A. Bolshakov. 2004. A Method of Linguistic Steganography Based on Collocationally-Verified Synonymy. *Information Hiding 2004*, *Lecture Notes in Computer Science*, 3200 Springer-Verlag, pp. 180–191.
3. Igor A. Bolshakov, Alexander Gelbukh. 1998. Lexical functions in Spanish. Proc. *CIC-98, Simposium Internacional de Computación*, Mexico, pp. 383–395; www.gelbukh.com/CV/Publications/1998/ CIC-98-Lexical-Functions.htm.
4. Igor A. Bolshakov, Alexander Gelbukh. 2001a. A Very Large Database of Collocations and Semantic Links. Proc. NLDB-2000: 5th Intern. Conf. on Applications of Natural Language to Information Systems, France, 2000. *Lecture Notes in Computer Science* N 1959, Springer-Verlag, pp. 103–114; www.gelbukh.com/ CV/Publications/2000/NLDB-2000-XLex.htm.
5. Igor A. Bolshakov, Alexander Gelbukh. 2001b. A Large Database of Collocations and Semantic References: Interlingual Applications. *International J. of Translation*, V.13, No.1–2, pp. 167–187.
6. Igor A. Bolshakov, Alexander Gelbukh. 2003. On Detection of Malapropisms by Multistage Collocation Testing. *NLDB-2003, 8th Int. Conf. on Application of Natural Language to Information Systems*. Bonner Köllen Verlag, 2003, pp. 28–41.
7. Thorsten Brants. 2000. TNT—A Statistical Part-of-Speech Tagger. In: Proc. *ANLP-2000, 6th Applied NLP Conference*, Seattle.
8. Ted Briscoe, John Carroll, Jonathan Graham and Ann Copestake. 2002. Relational evaluation schemes. In: *Proceedings of the Beyond PARSEVAL Workshop at the 3rd International Conference on Language Resources and Evaluation,* Las Palmas, Gran Canaria, 4–8.
9. Hiram Calvo, Alexander Gelbukh. 2003. Natural Language Interface Framework for Spatial Object Composition Systems. *Procesamiento de Lenguaje Natural*, N 31; www.gelbukh.com/CV/Publications/2003/sepln03-2f.pdf.
10. Hiram Calvo, Alexander Gelbukh. 2004. Acquiring Selectional Preferences from Untagged Text for Prepositional Phrase Attachment Disambiguation. In: *Proc. NLDB-2004*, *Lecture Notes in Computer Science*, N 3136, pp. 207–216.
11. Hiram Calvo, Alexander Gelbukh, Adam Kilgarriff. 2005. Distributional Thesaurus versus WordNet: A Comparison of Backoff Techniques for Unsupervised PP Attachment. In: *Computational Linguistics and Intelligent Text Processing* (CICLing-2005). *Lecture Notes in Computer Science* N 3406, Springer-Verlag, pp. 177–188.
12. Xavier Carreras, Isaac Chao, Lluis Padró, Muntsa Padró. 2004. FreeLing: An Open-Source Suite of Language Analyzers. *Proc. 4th Intern. Conf. on Language Resources and Evaluation (LREC-04)*, Portugal.
13. Noam Chomsky. 1957. *Syntactic Structures*. The Hague: Mouton & Co.
14. Montserrat Civit, and Maria Antònia Martí. 2004. Estándares de anotación morfosintáctica para el español. *Workshop of tools and resources for Spanish and Portuguese*. IBERAMIA 2004. Mexico.

15. Ann Copestake, Dan Flickinger, and Ivan A. Sag. 1997. *Minimal Recursion Semantics. An introduction.* CSLI, Stanford University.
16. Ralph Debusmann, Denys Duchier, Geert-Jan M. Kruijff, 2004. Extensible Dependency Grammar: A New Methodology. In: *Recent Advances in Dependency Grammar. Proc. of a workshop at COLING-2004*, Geneve.
17. Isabel Díaz, Lidia Moreno, Inmaculada Fuentes, Oscar Pastor. 2005. Integrating Natural Language Techniques in OO-Method. In: Alexander Gelbukh (ed.), *Computational Linguistics and Intelligent Text Processing* (CICLing-2005), *Lecture Notes in Computer Science*, 3406, Springer-Verlag, pp. 560–571.
18. Alexander Gelbukh, Grigori Sidorov, Francisco Velásquez. 2003. Análisis morfológico automático del español a través de generación. *Escritos*, N 28, pp. 9–26.
19. Gelbukh, A., S. Torres, H. Calvo. 2005. Transforming a Constituency Treebank into a Dependency Treebank. Submitted to *Procesamiento del Lenguaje Natural* No. 34, Spain.
20. A. V. Gladki. 1985. *Syntax Structures of Natural Language in Automated Dialogue Systems* (in Russian). Moscow, Nauka.
21. Igor A. Mel'čuk. 1981. Meaning-text models: a recent trend in Soviet linguistics. *Annual Review of Anthropology* 10, 27–62.
22. Igor A. Mel'čuk. 1988. Dependency Syntax: Theory and Practice. State University Press of New York.
23. Igor A. Mel'čuk. 1996. Lexical Functions: A Tool for the Description of Lexical Relations in the Lexicon. In: L. Wanner (ed.), *Lexical Functions in Lexicography and Natural Language Processing*, Amsterdam/Philadelphia: Benjamins, 37–102.
24. Manuel Montes-y-Gómez, Aurelio López-López, and Alexander Gelbukh. 2000. Information Retrieval with Conceptual Graph Matching. *Proc. DEXA-2000, 11th Intern. Conf. on Database and Expert Systems Applications, England. Lecture Notes in Computer Science*, N 1873, Springer-Verlag, pp. 312–321.
25. Manuel Montes-y-Gómez, Alexander F. Gelbukh, Aurelio López-López. 2002. Text Mining at Detail Level Using Conceptual Graphs. In: Uta Priss *et al*. (Eds.): *Conceptual Structures: Integration and Interfaces*, 10th Intern. Conf. on Conceptual Structures, ICCS-2002, Bulgaria. *Lecture Notes in Computer Science*, N 2393, Springer-Verlag, pp. 122–136; ccc.inaoep.mx/~mmontesg/publicaciones/2002/DetaiedTM-iccs02.pdf.
26. Raúl Morales-Carrasco and Alexander Gelbukh. 2003. Evaluation of TnT Tagger for Spanish. Proc. *4th Mexican International Conference on Computer Science*, Mexico. IEEE Computer Society Press.
27. Carl Pollard and Ivan Sag. 1994. *Head-Driven Phrase Structure Grammar*. University of Chicago Press, Chicago, IL and London, UK.
28. Ivan Sag, Tom Wasow, and Emily M. Bender. 2003. *Syntactic Theory. A Formal Introduction* (2nd Edition). CSLI Publications, Stanford, CA.
29. John F. Sowa. 1984. *Conceptual Structures: Information Processing in Mind and Machine*. Addison-Wesley Publishing Co., Reading, MA.
30. James Steele (ed.). 1990. *Meaning-Text Theory. Linguistics, Lexicography, and Implications*. Ottawa: Univ. of Ottawa Press.
31. Pasi Tapanainen. 1999. *Parsing in two frameworks: finite-state and functional dependency grammar*. Academic Dissertation. University of Helsinki, Language Technology, Department of General Linguistics, Faculty of Arts.
32. Lucien Tesnière. 1959. *Eléments de syntaxe structurale.* Paris: Librairie Klincksieck.
33. Deniz Yuret. 1998. *Discovery of Linguistic Relations Using Lexical Attraction*, PhD thesis, MIT.