Dependency Analysis and CBR to Bridge the Generation Gap in Template-Based NLG

Virginia Francisco and Raquel Hervás and Pablo Gervás

Departamento de Ingeniería del Software e Inteligencia Artificial
Universidad Complutense de Madrid, Spain
{virginia,raquelhb}@fdi.ucm.es,pgervas@sip.ucm.es

Abstract. The present paper describes how dependency analysis can be used to automatically extract from a corpus a set of cases - and an accompanying vocabulary - which enable a template-based generator to achieve reasonable coverage over conceptual messages beyond the explicit scope of the templates defined in it. Details are provided on the actual process of partial automation that has been applied to obtain the case base, together with the various ingredients of the template-based generator, which applies case-based reasoning techniques. This module resorts to the taxonomy of concepts in WordNet to compute similarity between concepts involved in the texts. A case retrieval net is used as a memory model. The set of data to be converted into text acts as a query to the system. The process of solving a given query may involve several retrieval processes - to obtain a set of cases that together constitute a good solution for transcribing the data in the query as text messages - and a process of knowledge-intensive adaptation which resorts to a knowledge base to identify appropriate substitutions and completions for the concepts that appear in the cases, using the query as a source. We describe this case-based solution for selecting an appropriate set of templates to render a given set of data as text, we present numeric results of system performance in the domain of press articles, and we discuss its advantages and shortcomings.

1 Introduction

A classic problem in natural language generation is the "generation gap" described by Meteer [1], a discrepancy between what can be expressed in the text plan and what the particular realization solution can actually convert into text. This is particularly apparent in template-based generators, which have recently achieved widespread acceptance. Template-based solutions for natural language generation rely on reusing fragments of text extracted from typical texts in a given domain, having applied to them a process which identifies the part of them which is common to all uses, and leaving certain gaps to be filled with details corresponding to a new use. For instance, when conveying the information that *Alice married Christopher in Birmingham*, a template such as _ *married* _ *in* _ may be used, filling in the gap with appropriate strings for Alice, Christopher and Birmingham.

In terms of templates, the "generation gap" occurs when the input calls for messages not explicitly contemplated in the set of templates in use. Whereas more complex natural language generation systems based on the use of grammars can have rich stages devoted to selecting fresh combinations of words to convey the same meaning, template-based systems are faced with an additional difficulty. Meanings not explicitly contemplated may be conveyed by a combination of templates whose meanings overlap to cover the full meaning required. However, the fact that templates are made up of words that are not accessible to the system makes the system blind to the possible ways of combining them. Annotating the templates with tags that indicate the circumstances under which it is appropriate to use the template would solve the problem, but it eliminates some of the advantages of the template solution over more knowledge-rich approaches.

Case-based reasoning (CBR) is a well established problem solving technique that searches for solutions to new problems in terms of how similar problems have been solved in the past. This is very close to template-based generation, which can be basically understood as reusing fragments of text extracted from typical texts in a given domain, having applied to them a process which identifies the part of them which is common to all users, and leaving certain gaps to be filled with details corresponding to a new use. Applying specifically a case-based solution to template-selection presents the advantage that the information needed to solve the problem can be obtained from the original examples of appropriate use that gave rise to the templates. By associating a case with each template, with case attributes consisting of conceptual descriptions of the arguments that were used for the template in the original instance, a case-based reasoning solution can be employed to select the best template for realizing a particular message.

The present paper describes a case-based solution for the task of selecting adequate templates for realizing messages describing actions in a given domain. The goal is to achieve coverage of a broad range of messages by combining instances of a restricted set of templates, and providing automated means for dealing with overlaps between the information conveyed by the templates found.

2 Case Based Reasoning Techniques and Dependency Analysis

This section provides a brief outline of the basic CBR techniques employed in the paper, the lexical database used to provide the reference taxonomy of concepts, and the dependency analysis tool employed for the automatic construction of the case base.

2.1 Case Based Reasoning Techniques and Technologies

Case-based Reasoning (CBR) [2] is a problem solving paradigm that uses the specific knowledge of previously experienced problem situations. Each problem

is considered as a domain case, and a new problem is solved by retrieving the most similar case or cases, reusing the information and knowledge in these cases to solve the problem, revising the proposed solution, and retaining the parts of this experience likely to be useful for future problem solving. General knowledge about the domain under consideration usually plays a part in this cycle by supporting the CBR processes.

Case based reasoning solutions must rely on efficient storage and retrieval of the cases. A good solution for this problem is to store cases in a Case Retrieval Net (CRN) [3]. Case Retrieval Nets are a memory model developed to improve the efficiency of the retrieval tasks of the CBR cycle. They are based on the idea that humans are able to solve problems without performing an intensive search process, but they often start from the given description, consider the neighbourhood, and extend the scope of considered objects if required. CRNs organize the case base as a net of Information Entities (IEs) which represent any basic knowledge item in the form of an attribute-value pair. A case then consist of a set of such IEs, and the case base is a net with nodes for the entities observed in the domain and additional nodes denoting the particular cases. IE nodes may be connected by similarity arcs, and a case node is reachable from its constituting IE nodes via relevance arcs. Different degrees of similarity and relevance are expressed by varying arcs weights. Given this structure, case retrieval is carried out by activating the IEs given in the query case, propagating this activation according to similarity through the net of IE nodes, and collecting the achieved activation in the associated case nodes.

Case-based reasoning techniques have been applied in the past [4, 5] to the problem of selecting specific verb templates as lexical realizations for actions described conceptually in the input. The system described in that work operated over manually built resources (vocabulary, case-base, and taxonomy used as reference). This restricted greatly the coverage that it could achieve. The solution presented in this paper involves a similar application of CBR methodology, but relies on state-of-the-art techniques of linguistic analysis to automate the necessary processes of building vocabulary and case base from domain corpora, as well as resorting to an existing lexical database to provide the taxonomy.

2.2 The Role of Taxonomies in Computing Similarity

A crucial operation in any CBR system is establishing similarities between query and cases, which can usually be reduced to searching for similarities between particular domain items that make up the query and the corresponding items in the cases. A popular solution is to rely on a taxonomy of concepts to deal with this task. Similarity between concepts is computed in terms of the distance traversed over the taxonomical structure to reach one from the other.

Currently, a number of efforts in the area of language engineering are aimed to the development of systems of basic semantic categories (often called "upper-level ontologies"), to be used as main organizational *backbones*, suitable to impose a structure on large lexical repositories. Examples of such systems are the PENMAN Upper Model [6], the Mikrokosmos ontology [7], and the WordNet

[8] upper structure. Machine learning techniques have been used to build *mapping dictionaries*, lexicons of elementary semantic expressions and corresponding natural language realizations [9].

WordNet is by far the richest and largest database among all resources that are indexed by concepts. For this reason, WordNet has been chosen as initial lexical resource for the development of the module presented in this paper. WordNet is an on-line lexical reference system organized into synonyms sets - or *synsets* -, each of them representing one underlying lexical concept, linked by semantic relations like synonymy or hyponymy. This organization makes it possible to use WordNet as a knowledge source. The hypernymy/hyponymy relation can be considered equivalent to the "isa"/"r-isa" relation, and it induces a taxonomical hierarchy over the set of available concepts.

WordNet is not organized according to individual *words*, it is organized according to *concepts*. Due to linguistic phenomena such as polysemy and synonymy, there is potentially a many-to-many mapping between concepts and words. This raises the important problem of Word Sense Disambiguation [10] (WSD), which has by itself deserved the attention of many researchers. At this point, WordNet provides some help: the *tag count* field for synsets. This field allows us to order, within a synset, which of the nouns is more *usual* in a generic corpus (in this case, the Brown Corpus [11]).

```
0(
E0  ()          fin        C     *
1   President   ~          N     2     s          (gov accuse)
2   accused     accuse     V     E0    i          (gov fin)
E2  ()          president  N     2     subj       (gov accuse)
3   Georgia     ~          N     2     obj        (gov accuse)
4   of          ~          Prep  2     mod        (gov accuse)
5   terrorism   ~          N     4     pcomp-n    (gov of)
)
```

Fig. 1. Example of dependency tree for the sentence *President accused Georgia of terrorism*

2.3 Dependency Analysis

The basic idea of the dependency analysis is that the syntactic structure of a sentence is described in terms of dependency relations between pairs of words (a parent and its child). These relations compose a tree (the dependency tree). Dependency analysis has been used succesfully for several applications: multilingual machine translation [12], recognising textual entailment [13], and automatic evaluation of question-answer systems [14].

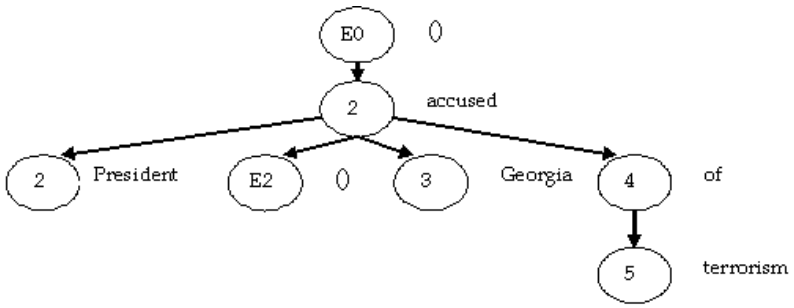

Fig. 2. Example of the graphical representation of the dependency tree for the sentence *President accused Georgia of terrorism*

MINIPAR [15] analyses English texts with high accuracy and efficiency in terms of time. An example of the dependency tree generated by MINIPAR for the sentence *President accused Georgia of terrorism* is given in Figures 1 and 2.

3 Using Dependency Analysis to Build a Case-Base for Template Selection

A template-based generator selects a set of string fragments - which are deemed suitable to describe the concepts to be transmitted - and then composes them in a particular way to produce a final string corresponding to a sentence. The string fragments may correspond to atoms - strings that corresponds to words or phrases which will appear in the final text as they are - or templates - strings with place holders at positions where other string are to be inserted. This is an acceptable method when operating in restricted domains, but results can be poor if complex concepts or actions have to be expressed. Such complex structures may require the introduction of lexical chains that are employed exclusively for a specific referent or verb in some context. This introduces an unwanted rigidity in the system, because it makes the task of extending the vocabulary an arduous one.

To facilitate this task, we have defined an automated process of jointly building the vocabulary and the case-base for a case-based template-selection module. This module relies on subsequent processing of its output by an accompanying surface realization module. This module is in charge of putting together the selected terms and templates. Additionally, it carries out a basic orthographic transformation of the resulting sentences. Templates are converted into strings formatted in accordance to the orthographic rules of English - sentence initial letters are capitalized, and a period is added at the end.

3.1 Basic Operation of the Case-Based Template-Selection Module

The case-based reasoning module implements two of the four basic stages of a classic CBR cycle: retrieval and reuse of cases from the case base. No automated solution to the revision and retainment stages is contemplated so far, due to the fact that a very complex set of linguistic, cognitive and pragmatic constraints must be taken into account when validating any natural language solutions generated in this manner. The contribution of an expert in the domain is required to revise the results achieved by the module.

The retrieval task starts with a partial or complete problem description - a partial description of the action -, and ends when a matching previous case has been found. In our module, the retrieval of cases is directly handled by the Case Retrieval Net and its method of similarity propagation. Starting from a partial description of the action we need to lexicalise, the retrieval of the more similar cases is done by calculating an activation value for each case in the case base. The ones with higher activation are the more similar ones to the given query. This calculation is performed in three steps: (1) the IE nodes that correspond to the query are activated, (2) the activation is propagated according to the similarity values of the arcs, and (3) the achieved activations in the previous step are collected in the associated case nodes. Once we have the final activation in the cases, the one with the higher value is returned by the net. It would be possible to take not only the most similar one, but a set with the most similar cases to the query.

Each retrieved case has an associated template from the vocabulary for the verb or action it represents. In the process of reusing the case we have obtained from the net, we have to substitute the attribute values of the past case with the query values.

3.2 The Resources Required: Case Base and Vocabulary

The vocabulary contains all the lexical information essential to write the final text. A lexical tag made up of one or more words is assigned to each concept in the domain. This is used for lexicalising individual concepts, with little choice given. The vocabulary for actions or verbs becomes more complex: it is stored in the form of cases, where each case stores not only the corresponding template - solution of the case -, but also additional information concerning the elements involved in the action and the role that those elements play in the action - description of the case. A case is not an abstract instance of a verb or action, but rather a concrete instance in which specific actors, locations and objects appear.

Examples of cases are given below. The associated templates are shown below for each case:

LEX:	ACTOR:	OBJECT:	OF:
accuse_of	president	Georgia	terrorism

_ *accused* _ *of* _

```
LEX:         ACTOR:   LIKE:
behave_like  leaders  Stalinists
```

_ *behave like* _

It is important to take into account that the structure of the cases is not rigid. They will not always have the same elements, nor in the same order. A clear example is provided by the verbs 'leave' and 'go', both involving some kind of movement. The first one has an attribute `From` to indicate where the actor is coming from, whereas the second one has an attribute `To` that indicates his destination.

Cases are stored in a Case Retrieval Net. This model is appropriate for the problem under consideration, because on one hand our cases consist of attribute-value pairs that are related with one another, and on the other hand the queries posed to the module will not always be complete. To find a lexical tag for a given action, the CRN is queried with the class of elements involved in the action.

The vocabulary of the module is built from the case base. For each attribute-value pair in the cases an information entity is created. For each case, a node is created which holds references to the information entities that are contained. When introducing an IE, if that entity has already appeared in another case it is not duplicated. Instead, another association is created between the new case and the existing information entity.

As IEs are inserted to form the net, it is necessary to establish a measure of similarity between them. The hyponymy/hypernymy relation of WordNet can be seen as a "isa" relation. WordNet can therefore be used as a taxonomy over which to automatically calculate the similarity between the concepts appearing in the cases. This requires some additional measures when creating the case base, to ensure that all elements appearing as arguments anywhere in the case base are adequately covered by WordNet. A preliminary filter is applied to the automatically generated cases, so that if one of the elements of a case is not found in WordNet, the case is discarded. From our initial corpus 297 cases were generated, and 179 of them were discarded using WordNet, being our final case base formed by a total of 118 cases.

The similarity between two entities is calculated by taking into account the distance between them and using Formula 1.

$$sim(c1, c2) = 1 - (distance(c1, c2)/20) \tag{1}$$

The distance between two concepts is calculated by finding their first shared ancestor or hypernym, and adding up the distance between this ancestor and each of the concepts. It is also necessary to have a similarity value for each entity with itself. This value is always the maximum possible, because the distance between the entity and itself is 0.

Each of the IEs is related to the cases to which it belongs with a certain value of relevance. In the implemented module we have chosen that all the elements in a case has relevance 0.5.

3.3 Constructing the Case Base from the Dependency Trees for the Corpus

In order to obtain the case base automatically we have developed a method based on MINIPAR, which gives a dependency tree for every sentence, and based on this tree we select every verb and the words related to it. This section explains the process followed to obtain the different cases involved in a text and their templates. Firstly MINIPAR processes the texts and generates a dependency tree for each of the sentences. Each tree is analysed in order to obtain the following elements:

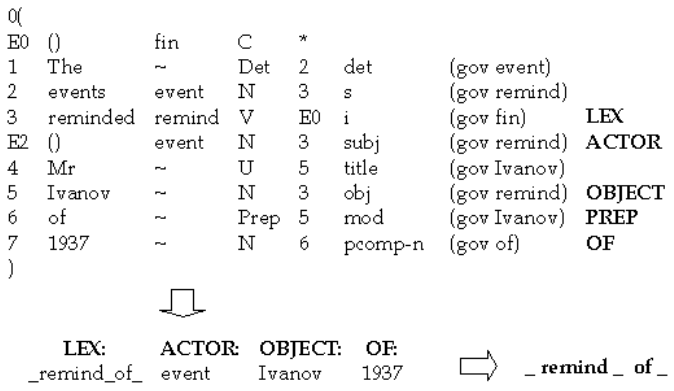

Fig. 3. Dependency tree, template and related case for *The events reminded Mr Ivanov of 1937*

- The verbs involved in each sentence. The process looks for every node in the tree marked as a verb during lexical analysis. Each of the verbs found in the sentence will give rise to a new case. The stems of the verbs as identified by MINIPAR are stored in the LEX attribute field of the case.
- The nodes which depend on each of the verbs. Once we have the children of every verb we process them in search for the rest of the elements required to build the cases and templates:
  - Subject of the verb. MINIPAR identifies for each verb a special node that is marked as subject of that verb during lexical analysis. The stem of the subject node is stored in the ACTOR attribute field of the case.
  - Objects of the verb. In a similar way, MINIPAR identifies objects of the verb. The stem of the object node is stored in the OBJECT attribute field of the case.
  - Prepositions. MINIPAR identifies with a special label the prepositions that appear in the sentence. Each preposition found in the sentence gives rise to a new field in the case. This new field is labelled with the preposition itself as name of the attribute.

- Words related to prepositions. MINIPAR indicates dependency relations for every word. The nouns that act as head of the nodes related to the prepositions identified in the previous step are used as values for the preposition attributes discovered in the previous step.

An example of dependency tree and the case and template generated for the sentence *The events reminded Mr Ivanov of 1937* is given in Figure 3.

4 Evaluation and Discussion

In order to evaluate the feasibility of our proposal we carried out some preliminary tests over a set of news items in four different domains: politics, sport, science, and health. These news items contain 96 sentences in total which generate 297 cases. To evaluate the automated generation of cases we have generated the cases for every news item and then we have checked the correctness of each of the cases. The average percentage of success is over 50%. This is not related to the acurracy of MINIPAR but to the fact that our first approximation to the problem only uses the basic elements of the resulting dependency tree, as described above.

Analysing instances where the process produced incorrect cases indicated five main reasons for failure:

- **Nested cases.** There are some cases that have as object or as actor another case. The current representation does not allow nesting of cases, so these subcases are not being recognized. Our first solution to this problem is to represent the super-case and the sub-case as two different cases. In the super-case the nested case has a special representation which is considered during the retrieval as "every word", having maximum similarity with any other concept. An example is the sentence "Russian President accuses Georgia of acting like Stalinists". Here, we have two cases: one for the sentence "Russian President accuses Georgia of" where the value of the attribute *of* is "NC" (which represent the nested case) and another case for the sentence "acting like Stalinists".
- **Actor mistakenly identified.** In some cases the actor is not identified or the word MINIPAR points as subject is not the correct one. An example is the sentence "Foetuses as early as 12 weeks appearing to "walk" in the womb", where MINIPAR has decided that the subject for the verb "appear" is the word "weeks", although the correct choice is "foetuses".
- **Object mistakenly identified.** In some cases the object is not identified as object in the lexical analysis. An example is the sentence "foetuses become viable and potentially self-aware", where MINIPAR has not taken as object of the verb "become" any word. The correct choice would have been "viable" and "self-aware".
- **Verb mistakenly identified.** In some sentences the verb is not well identified by MINIPAR. An example is the sentence "This testing and spectacular track built a lead of more than 20 seconds over Schumacher", where "track" has been identified as a verb.

– **Prepositions mistakenly identified.** In some sentences the preposition is not well identified because MINIPAR considers that the preposition is not related to the verb. An example is the sentence "Mr Saakashvili has accused the Kremlin of hysteria", where "of" is not considered a preposition related to "accused" but is related to "Kremlin".

Figure 4 shows the relative contribution to the total error of each source of failure in terms of percentages of the total number of processed cases.

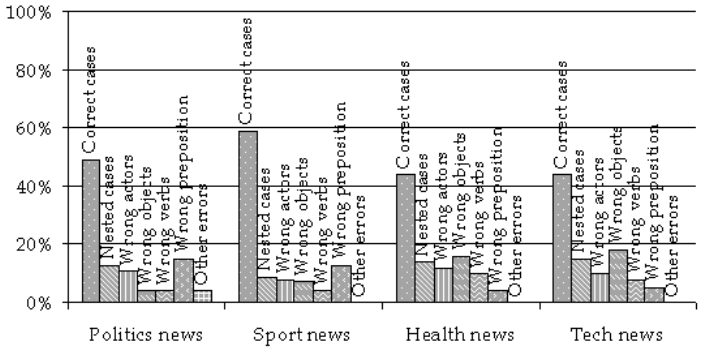

Fig. 4. Percentage of wrong cases group by reasons

5 Conclusions and Future Work

Dependency analysis provides a good first approximation for extracting automatically the information needed for case-based template selection. Full coverage of the initial corpus is not a priority since texts to be generated need not match those in the corpus precisely. Even with the current restrictions imposed by the internal representation, the success rate for that stage of the process is close to 50%. This indicates that a portion of the corpus can be converted into cases from the point of view of the information appearing in the sentences. The proposed solution is therefore easily scalable to larger corpora.

Even when the incorrect cases are not representing the exact information extracted from the corpus, they can be valid cases for the CBR module, not disturbing its functionality. Some of them would be discarded by WordNet's filter, and most of the remainder will have low similarities with the queries whenever they have resulted in nonsensical information.

The use of WordNet as a taxonomical knowledge base provides acceptable means for validating input lexical items. However, if used as the only validation mechanism, it lowers effective system coverage, largely because WordNet does

not include proper nouns. This leads to the elimination of more than half of the cases extracted from the documents in the initial sample because the elements appearing in them were not covered by WordNet. A possible addition to the system would be a knowledge base for proper nouns as well as general concepts.

An issue that needs to be addressed is whether dependency analysis is the most adequate tool for the particular needs of the extraction process required. Similar processes to those presented in this paper must be tested using constituent analysis as means for accessing the linguistic structure of sentences in the corpus, and the results compared with those presented here. Further work will consider alternative language analysis tools and lexical resources.

One of the points to take into account in the future versions is the resolution of pronominal references. In the current version the pronoun are taken as value of the different fields (actor, object, ...). A method for anaphora resolution must be developed in future versions in order to solve this problem.

The resulting texts would improve significantly if a more complex set of templates were considered. Template-based generators have obtained results comparable to more elaborate solutions by resorting to recursive use of templates [16]. In our approach, this would correspond to allowing actions to be represented as nested cases, where a case would be constructed not only of attribute-value pairs, but also attribute-case pairs, where the value for some attribute may itself be a complete case - with an associated template. Recursive nesting of cases would allow recursive use of templates. MINIPAR provides sufficient information to identify nested structures, but the retrieval and adaptation stages would have to be adapted to deal with this recursive nature. This issue is related to the scalability of the solution in the sense that scalating the solution to more complex linguistic constructs would need to address the problem of improving the complexity of the cases.

The similarity being employed in the current version establishes a normalising upper limit independent of the depth of WordNet as a taxonomy. This should be corrected in subsequent versions.

The automatic process of acquiring the cases leads to situations where the sentences *"someone has something"* and *"someone says something"* give rise to cases with only two elements: an actor and an object. For the system these two cases are in principle equivalent. However, the CBR process ensures their correct use by resorting to the contextual information available in the original sentences from which the cases were extracted: both of them will probably have had a person as subject, but the kind of element that is had will be conceptually different from the kind of element that can be said. This allows the system to perform reasonably well in spite of the apparent sparsity of explicit knowledge employed.

Acknowledgements

Partially supported by the Spanish Ministry of Education and Science project TIN2006-14433-C02-01, and research group grant UCM-CAM-910494, jointly

funded by Universidad Complutense de Madrid and the Comunidad Autónoma de Madrid (D.General de Universidades e Investigación).

References

1. Meteer, M.W.: The generation gap: the problem of expressibility in text planning. PhD thesis, Amherst, MA, USA (1990)
2. Aamodt, A., Plaza, E.: Case-based reasoning : Foundational issues, methodological variations, and system approaches (1994)
3. Lenz, M., Burkhard, H.D.: Case Retrieval Nets: Basic Ideas and Extensions. In: KI - Kunstliche Intelligenz. (1996) 227–239
4. Hervás, R., Gervás, P.: Case Retrieval Nets for Heuristic Lexicalization in Natural Language Generation. In Cardoso, A., Bento, C., Dias, G., eds.: Progress in Artificial Intelligence (EPIA 05). Number LNAI 1036, Covilha, Portugal, Springer-Verlag (2005)
5. Hervás, R., Gervás, P.: Case-based reasoning for knowledge-intensive template selection during text generation. In: Proc. of the 8th European Conference on Case-Based Reasoning, Springer-Verlag (2006)
6. Bateman, J.A., Kasper, R.T., Moore, J.D., Whitney, R.A.: A General Organization of Knowledge for Natural Language Processing: the PENMAN upper model (1990)
7. Mahesh, K.: Ontology development for machine translation: Ideology and methodology. Technical Report MCCS-96-292 (1996)
8. Miller, G.A.: Wordnet: a lexical database for English. Commun. ACM **38** (1995) 39–41
9. Barzilay, R., Lee, L.: Bootstrapping lexical choice via multiple-sequence alignment. In: Proc. of the EMNLP'02. (2002) 164–171
10. Ide, N., Veroni, J.: Word Sense Disambiguation: The State of the Art. Computational Linguistics (1998) 1–40
11. Nelson Francis, W., Kucera, H.: Computing Analysis of Present-day American English. Brown University Press, Providence, RI (1967)
12. Maxwell, D., Schubert, K.: Metataxis in Practice: Dependency Syntax for Multilingual Machine Translation. Foris Publications (1989)
13. Kouylekov, M., Magnini, B.: Tree edit distance for recognizing textual entailment: Estimating the cost of insertion. In: Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, Venezia, Italia. (2006)
14. Herrera, J., Peñas, A., Rodrigo, A., Verdejo, F.: UNED at PASCAL RTE-2 Challenge. In: Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, Venezia, Italia. (2006)
15. Lin, D.: Dependency-based evaluation of MINIPAR. In: Proc. of Workshop on the Evaluation of Parsing Systems, Granada, Spain (May 1998)
16. McRoy, S., Channarukul, S., Ali, S.: A Natural Language Generation Component for Dialog Systems. In Cox, M., ed.: Working Notes of the AAAI Workshop on Mixed-Initiative Intelligence (AAAI99). (1999)