AUTOMATIC EXTRACTION OF SYNTACTIC PATTERNS FOR DEPENDENCY PARSING IN NOUN PHRASE CHUNKS

Mihaela Colhon[*] and Dan Cristea[**]

Abstract: In this article we present a method for automatic extraction of syntactic patterns that are used to develop a dependency parsing method. The patterns have been extracted from a corpus automatically annotated for tokens, sentences' borders, parts of speech and noun phrases, and manually annotated for dependency relations between words. The evaluation shows promising results in the case of an order-free language.

Keywords: dependency parsing, syntactic patterns, noun phrases

1. Introduction

Reliable dependency parsing is a notorious difficult problem in Natural Language Processing (NLP). We describe in this paper[1] a pattern-based approach in dependency parsing that addresses only some syntactic chunks of texts. The chunk isles that we concentrate on are a class of noun phrases (NPs) displaying a rather limited recursivity, for which reliable chunkers are known to exist. The idea is that, generally, NP constituents cover a significant part of a sentence and if their dependency structures are known, we would be closer to recuperate the whole structure of a sentence. Moreover, often, NPs, sometimes augmented to prepositional phrases (PPs), fulfil semantic roles around verbs. The ambiguity of attaching an NP/PP to a verb can be reduced by benefiting from a semantic roles parser[2]. Therefore, an approach for depicting the hidden dependency structure of NPs can be combined to other syntactic and semantic methods, on the way to build the whole dependency structure of a sentence.

It is worth noting that quite often there is no consensus on what the correct dependency structure for a particular sentence should be. To build a dependency treebank, the human annotators must decide for each word which is the one it depends on. To determine dependencies often involves a deep interpretation process and this is why for the same word sequence, sometimes, different dependency structures could be negotiated among annotators. In contrast, the decisions the human annotators should take while building phrase-structure treebanks usually offer much less ambiguity. Generally,

[*] University of Craiova, Departament of Computer Science, mcolhon@inf.ucv.ro.
[**] "Alexandru Ioan Cuza" University, Iaşi, Faculty of Computer Science, Romanian Academy, Institute for Computer Science, Iași, dcristea@info.uaic.ro.

[1] The first author received support for this research from the strategic grant POSDRU/89/1.5/S/61968, Project ID 61986 (2009), co-financed by the European Social Fund within the Sectorial Operational Program Human Resources Development 2007-2013. Our thanks go to Radu Simionescu for realising the automatic annotation of the *1984* corpus with markers for SENTENCE, TOK, POS and NP, to Augusto Perez – who did the manual annotation of the dependency relations, and to the METANET4U project – which supported the creation of the automatic annotation tools.

[2] When lexical-semantic inputs are exploited, the outcome, of course, goes beyond the syntactic level of representation.

for treebanks reflecting the syntax of languages that have a free word order (as are for instance Czech, Romanian, etc.) a dependency structure representation is preferred to a constituent structure representation, e.g. the Prague Dependency Treebank (Hajič et al. 2000) in the case of Czech. Following this line, at the Faculty of Computer Science of the "Alexandru Ioan Cuza" University of Iaşi a dependency treebank for the Romanian language was recently built (Perez 2012). We have used this resource in the training and evaluation stages of the proposed mechanism.

The parsing mechanism presented in this paper is based on the morpho-syntactic descriptions (henceforth MSD[3]) of the words involved in dependency relations but takes also into account the MSDs of the nearby words in order to create a specific syntactic-motivated context in which the dependency relations occur.

More precisely, we have implemented a system for generating and applying corpus-based patterns in dependency parsing mechanism. For each dependency relation identified in the training corpus, the resulted MSD description of the dependent words but also of the neighbour words, are then generalised in a try to identify the syntactic invariant properties requested by a specific syntactic function. The morpho-syntactic features upon which the generalisation is made specify, besides the Part-of-Speech data (henceforth PoS), the morphological information of the corresponding word such as gender, tense, person, etc. We can conclude that such specifications include all the relevant grammatical features of the Romanian words. The syntactic specifications resulted after applying the generalisation on the MSD extracted from the training corpus can serve as a set of syntactic models that can be further used for the analysis of new texts.

The proposed method works as follows: starting from flat NP sequences and using a set of syntactic patterns extracted from a training corpus, we identify dependency links between the words of the NPs based on their MSD data. Thus we transform flat NP structures into dependency trees, not necessarily complete. The knowledge enclosed in this dependency parsing is structured as syntactic patterns defined in terms of morpho-syntactic specifications.

The motivation of this study was the need to create a dependency parsing technique as accurate as possible. We have noticed that statistical methods usually fail in cases where the grammatical function could be easily determined upon certain syntactic information. These errors could be determined by the noise in the corpus but also by some linguistic phenomena which are hardly identified by a statistical model. Thus, a better approach would be to use this dependency parsing technique before any statistical approach and, only in case of no result, to initiate a statistical parsing technique.

The article is structured as follows. We start by presenting the state of the art in the domain of dependency parsing, by focusing on the pattern-based approaches. In the following section we present the structure of the corpus used in the system training and in the evaluation process. In section 4 we describe the patterns construction method and in the last two sections we discuss the obtained evaluation scores and formulate some conclusions.

[3] The morpho-syntactic description term was introduced in the literature during the MULTEXT-East Project (Erjavec 2010).

2. State of the art

In this paper we present a method for automatic construction of dependency grammar rules that cover NP sequences, based on a collection of records of the form:

(1) <<NP chunk> <dependency link> <idx1> <idx2>>

extracted from a corpus, where <NP chunk> is a sequence of MSD tags representing an NP, <dependency link> represents a dependency relation, and <idx1> and <idx2> give the positions of the head word and the dependent word in the chunk between which the dependency link should occur. We deterministically associate dependency rules to the syntactic structures of NP sequences by using a matching method based on regular expressions that take into consideration also contextual information. As such, the parsing problem has a lot to do with the pattern-matching problem: each sequence of MSD tags given as input is matched against such an <NP chunk> pattern that has been previously extracted from the corpus. If the matching succeeds, then the <dependency link> attached to the pattern will be instantiated between the two words, belonging to the chunk, indicated by the indexes. In case of failure, of course, no dependency relation will be identified.

In pattern-based approaches, it is important to properly choose the pattern description formalism and the pattern structures (Kurc et al. 2010). Hearst (1992) defines a scheme for patterns that was often taken as a reference point: patterns are regular expressions with lemmatized word forms as the alphabet and variables corresponding to noun phrases (for example: *NP1 such as NP2* and *NP1 is a/an NP2* are patterns defined in order to encode the hypernymy relation). Snow et al. (2004) report that these patterns have been successful at identifying some examples of WordNet relationships (only that, applied to the construction of lexical thesauri the method has a low proficiency due to the small number of patterns typically employed).

In the approach we present in this paper we do not rely on lexical information. Our syntactic patterns include only MSD tags that have been automatically marked in the corpus. We are aware that in many cases of ambiguity, adding lexical information should result in an improvement of the performances and therefore is an option to be considered in the future. However, lexical information implies also a much larger training corpus, which was not in our hands at present.

3. The corpus

The corpus used in this study contains the text of the first chapter of George Orwell's novel *1984*. In order to make it useful as a Romanian Dependency Treebank, three levels of annotation have been added to the raw text and encoded in XML, by adopting a simplified form of the XCES standard (Ide et al. 2000):
(i) segmentation and lexical information: sentences have their boundaries marked and each token has attached its part of speech, lemma, and morpho-syntactic information (gender, number, person, case, etc.) by running an automatic processing chain that

includes: sentence segmentation, tokenisation, POS-tagging and lemmatisation (Simionescu 2012a);
(ii) noun phrases: NP chunks are automatically marked by a NP chunker (Simionescu 2012b) and manually corrected; although the NPs could be recursive, their complexity is small because they do not include relative clauses;
(iii) dependency data: tokens in sentences have been manually annotated (Perez 2012) for their head-words and the corresponding dependency relations.

As will be shown, these levels of annotation are sufficient to extract surface patterns and to associate them with dependency rules that will finally configure a dependency parser.

The corpus thus acquired contains 374 sentences, 6778 tokens and 650 NP structures. A different section of the same corpus was used in the evaluation process.

4. Extracting the patterns

A syntactic chunker implements a shallow parser on the natural language texts. Today, the shallow parsing techniques are dominated by fast computational mechanisms, such as regular expression analysis.

The outputs of a chunker signal the borders of the recognized syntactic groups. In the case of a NP chunker the outputs are the frontiers of the recognized nominal groups. The syntactic chunkers do not necessarily indicate the internal structure of the recognized chunks or their role inside the sentence they are part of.

Noun phrase chunking of a text represents the process of dividing a text or a phrase into semantic units which are usually centered around a noun or pronoun. The precission of such technique often depends on how accurately some semantic relations between the words of the text are identified. Let us take the following sequence: *fata cu ochi albaştri şi cu rochia de dantelă neagră* 'the girl with blue eyes and black lace dress'. We could divide it into several NP chunks as follows:

(2) [$_{NP}$ *fata cu ochi albaştri şi cu rochia de dantelă neagră*]
[$_{NP}$ *fata cu ochi albaştri*] + *şi cu* [$_{NP}$ *rochia de dantelă neagră*]
[$_{NP}$ *fata cu ochi albaştri*] + *şi cu* [$_{NP}$ *rochia*] + *de* [$_{NP}$ *dantelă neagră*]
[$_{NP}$ *fata*] + *cu* [$_{NP}$ *ochi albaştri*] + *şi cu* [$_{NP}$ *rochia de dantelă neagră*]
[$_{NP}$ *fata*] + *cu* [$_{NP}$ *ochi albaştri*] + *şi cu* [$_{NP}$ *rochia*] + *de* [$_{NP}$ *dantelă neagră*]

The above example shows only non-recursive noun phrases, but larger ones can also be assembled, each of them being made of one or more shorter noun phrases. In the training corpus, such cases are properly annotated, such that, for each recursive noun phrase, all the included noun phrases are also marked. As a direct consequence, from the training corpus we could extract not only the maximal noun phrases recognized by the NP chunker but also their internal NP components.

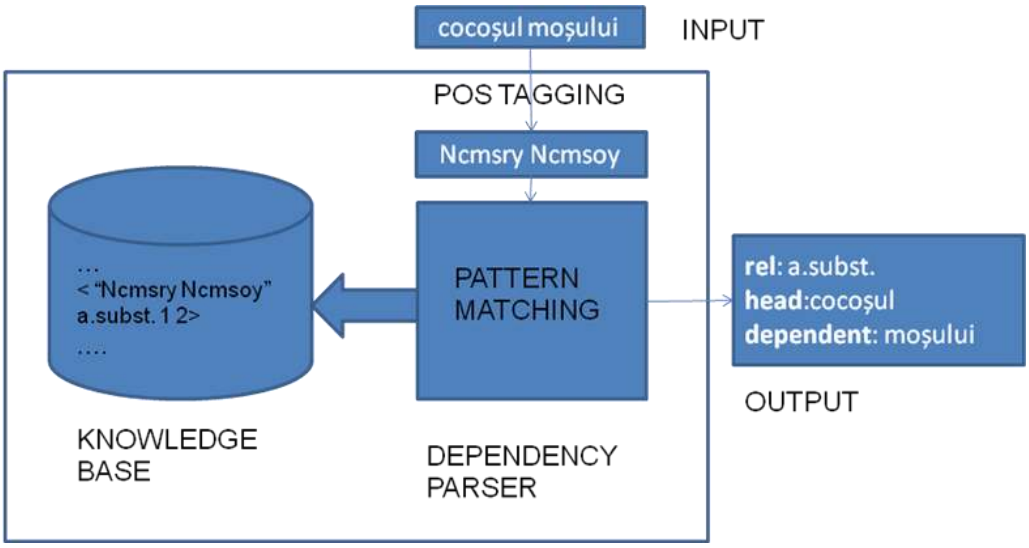

Figure 1: Architecture of the Pattern-based Parsing System

The primary data given by the corpus is used to associate dependency structures with each sequence of MSD tags corresponding to NPs extracted from the corpus and which will be called in the following morphological structures. As we have already said, the corpus used in this study puts in evidence three levels of annotations: POS tags, NP chunks and dependency relations. For each NP chunk found in the corpus, we extracted the morphological structure and all the dependency relations marked by the human annotator between the words of the NP chunk. The collection of syntactic patterns thus acquired covers all the syntactic structures found in the corpus. For example, if we take the following bracket representation for a Romanian noun phrase *un fond de rezervă* 'a reserve fund":

(3) [NP [Timsr un] [Ncms-n fond] [Spsa de] [Ncfsrn rezervă]]

its morphological structure is: Timsr Ncms-n Spsa Ncfsrn[4]. A sequence of *N* MSD tags should be described by *N*-1 patterns, each putting in evidence one of the *N*-1 internal heads[5] (see Figure 1). Thus, if the above sequence of words belongs to the corpus, 3 patterns will be attached to the 4 MSD tags: one attaching *un* to *fond*, one *de* to *fond* and one *rezervă* to *de*.

The morphological structures become the syntactic patterns of the dependency parser by means of a generalization process described further down. To each such pattern we associate the corresponding dependency information, more precisely, the dependency relations and their involved head and dependent components.

[4] See Appendix A for the meaning of the MSD tags in this paper.
[5] This is because we do not consider discontinued NPs.

The structure of the nominal phrase depends on the category of the head-word of the group. NPs are generally well individuated on FDG trees as sub-trees having as root nominal categories (mostly *nouns*, but also *pronouns*, *numerals* can act as NP heads).

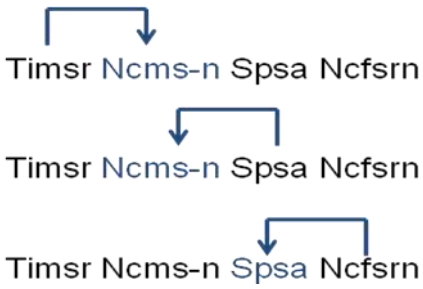

Figure 2: Example of a pattern of 4 MSD tags and 3 internal dependency relations

To the morphological structures found in the corpus we have applied a generalization process. This was necessary in order to cover similar syntactic constructions for which no occurrences have been found in the corpus, but also for reducing the total number of patterns. During the generalization, syntactic patterns are induced from the extracted morphological structures, by collapsing, for the same dependency relation, the contextual information or the MSD tags of the dependency elements involved into regular expressions.

Consider, for instance, a syntactic pattern of the form[6]:

(4) Nc[fm]sry Ncfsoy (Pp3fso.|Afpfson)?

codifies six NP morphological structures:

(5) Ncfsry Ncfsoy Pp3fso- or Ncmsry Ncfsoy Pp3fso- or Ncfsry Ncfsoy Afpfson or Ncmsry Ncfsoy Afpfson or Ncfsry Ncfsoy or Ncmsry Ncfsoy.

In order to derive the dependency structure of NP sequences, only the dependency relations occurring internally in NP chunks have been used in the parser rules. The most frequently encountered dependency relations found in the training corpus are:
(i) nominal modifier (marked as *a.subst.*[7]) – linking two nouns, as in *casa plăcerii* ‘house of pleasure’;
(ii) adjectival modifier (marked as *a.adj.*[8]) – linking an adjective to the noun it modifies, as in *frumoasa fată* ‘the beautiful girl”;
(iii) determinant (marked as *det.*) – linking an article or a determiner to the noun it modifies, as in *o carte* ‘a book’.

[6] See Appendix B for a glossary of the notations used in the regular expressions in this paper.
[7] For Romanian *atribut substantival*.
[8] For Romanian *atribut adjectival*.

For this phase of the dependency NP-parsing based on syntactic patterns we were concentrated only on the above 3 most common relations and we leave as future work the generation of patterns for the rest of the dependencies that could be identified in a nominal phrase group.

The main idea of the parser is that words with similar tags appearing in similar contexts to the ones found in the corpus will be linked by the same dependency relations.

The following examples show patterns covering structures with 3 MSD tags:

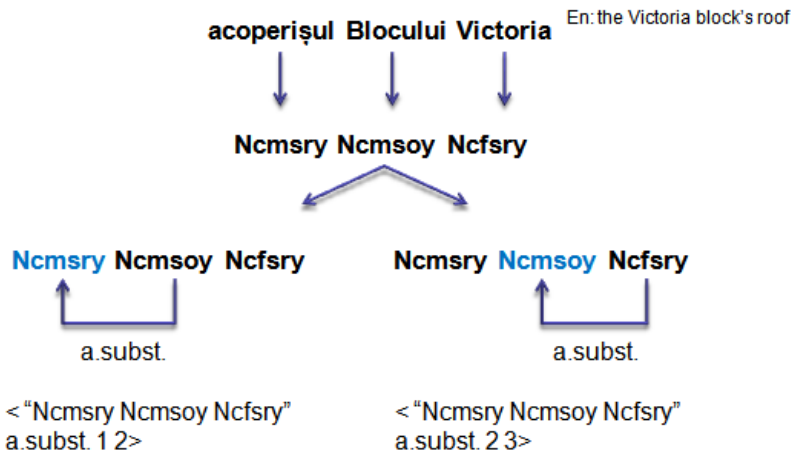

Figure 3: The NP sequence *acoperişul Blocului Victoria* and the resulting patterns
pattern: (structure "Ncmsry Ncmsoy Ncfsry")
(rel_name a.subst.) (head 1) (dependent 2)
pattern: (structure "Ncmsry Ncmsoy Ncfsry")
(rel_name a.subst.) (head 2) (dependent 3)

These patterns apply to the sequence *acoperişul Blocului Victoria* 'the Victoria Building's roof' illustrated in Figure 3. Both match identical sequences of tokens, containing MSD tags which include the *a.subst.* relations: in the first pattern, the relation occurs between the first two noun tags (that is Ncmsry and Ncmsoy) while, in the second pattern, the same *a.subst.* relation connects the last two tags (more precisely, Ncmsoy and Ncfsry).

Consider next the patterns below:

(6) pattern: (structure "Tdfpr Mcfp-l Ncfp-n")
(rel_name det.)(head 2) (dependent 1)
pattern: (structure "Tdfpr Mcfp-l Ncfp-n")
(rel_name a.adj.) (head 3) (dependent 2)

These patterns apply to the sequence *cele Două Minute* 'the Two Minutes' in Figure 4. The patterns treat sequences involving an article, a numeral and a noun.

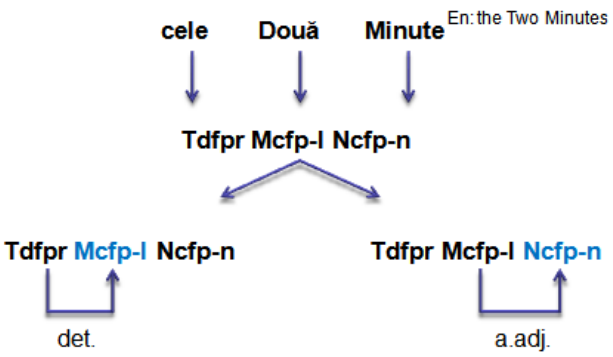

Figure 4: The NP sequence *cele Două Minute* and the resulting patterns
pattern: (structure "(Tsms)? Ncms[or]y Afpms-n")
(rel_name a.adj.)(head 1) (dependent 2)

A sequence matched by the pattern Tsms Ncmsoy Afpms-n, which is an instance of this pattern including the first, optional, element, is *al Partidului Interior* 'of the Inner Party', while a sequence matching the pattern without the first element is *pieptul solid* 'the solid breast'. Here, the *a.adj.* relation connects the tokens having the MSD tags Ncmsoy and Afpms-n or Ncmsry and Afpms-n. As can be seen, the pattern has a non-compulsory left context for this relation – the tag Tsms. Optional elements (is this and similar notations) are not counted.

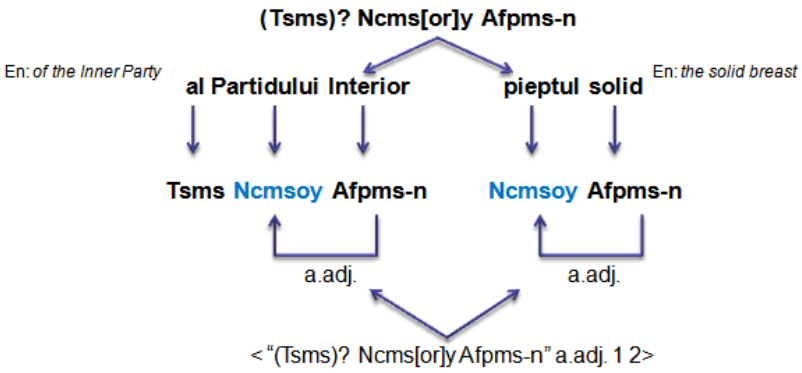

Figure 5: The same dependency relation in two different NP sequences
pattern: (structure "Tsms Ncmsoy Afpms-n")
(rel_name det.)(head 2)(dependent 1)
pattern: (structure "Tsms Ncmsoy Afpms-n")
(rel_name a.adj.)(head 2)(dependent 3)

These patterns match the same sequence *al Partidului Interior* 'of the Inner Party' as above, but the pattern now identifies the *det*. relation connecting the first two tokens (with the head in the second position). The pattern has a right context for this relation – the tag

Afpms-n. The second pattern indicates an *a.adj*. relation between the last two tokens, and this relation has a left context given by an article tag Tsms.

It is of course possible that the corpus will reveal more than exactly one relation to occur between identical pairs of MSD tags. In this case the parser will include them all in the result. As mentioned already, the disambiguation between these cases should be done at a lexical level – not treated in this study. The reverse case is when the NP morphological structure is not completely parsed by the dependency model, because the rules in the collection do not include the complete combination of tags in the input. Such misses affect negatively the recall.

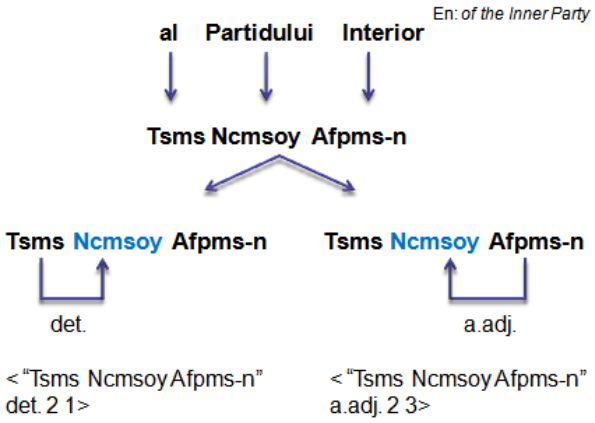

Figure 6: The NP sequence *al Partidului Interior* and the resulting patterns

5. Evaluation

We adopted Hajič's (2000) formulas in the evaluation of the dependency parser:

(7) Dependency Recall

$$R_D = \frac{Correct(D)}{|S|}$$

where *Correct*(*D*) is the number of correct dependencies found by the parser (the word is attached to its true head and the relation has got the correct label) and |*S*| is the size of test data in words (since |dependencies| = |words|)

(8) Dependency precision

$$P_D = \frac{Correct(D)}{|Generated(D)|}$$

where *Generated*(*D*) is the number of dependencies found by the parser.

Because of the small size of the corpus (374 sentences) we decided to apply a 3-fold cross validation, which ensures that all the sentences of the corpus are used for both training and testing. This procedure gave us a training corpus of 282 sentences and a testing corpus of 92 sentences. The results of the parser evaluation are given in the Table 1. As it can be seen from Table 1, the evaluation shows a rather high precision (above 83%) for the most frequent dependency relations found in NP constructions. There is a difficult trade in designing proper generalization rules because making them too lax could trigger false instances. On the other hand, making them too straight, leads to fewer applications, therefore a drop of recall, as in this case. We are aware of the fact that this is a critical issue on which we will have to insist in further research.

Dependency relation	Precision	Recall
nominal modifier	0.83	0.90
Adjectival modifier	0.97	0.88
determinant	0.88	0.97

Table 1: Parser evaluation scores

6. Conclusions

We presented here a rule-based syntactic parser for analyzing dependencies inside NPs, whose rules have been automatically inferred from a corpus that resulted from a human-machine collaboration. Usually the pattern-based approaches are developed for fixed-order languages, like English. Still, we have shown that good results can be obtained also for free-order languages, such as Romanian. We consider that the overall evaluation scores of this rule-based parser shows that the method can be applied satisfactorily, even if the training corpus is reduced. Moreover, we are confident that a refinement of the generalization mechanism could bring further enhancements.

References

Erjavec, T. 2010. Multext-east version 4: Multilingual morphosyntactic specifications, lexicons and corpora. In N. Calzori, K. Choukri, B. Maegaard, J. Mariani, J. Odjik, S. Piperidis, M. Rosner, D. Tapias (eds.), *Proceedings of the 7th International Conference on Language Resources and Evaluation, Valetta, Malta (LREC 2010)*, 2544-2547. Paris: European Language Resources Association.

Hajič, J. 2000. Introduction to natural language processing: Treebanks, treebanking and evaluation. Course notes. http://www.cs.jhu.edu/~hajic/courses/cs465/cs46526/ppframe.htm.

Hajič, J., Böhmová, A., Hajičová, E., Vidová-Hladká, B. 2000. The Prague dependency treebank: A three-level annotation scenario. In A. Abeillé (ed.), *Treebanks: Building and Using Parsed Corpora*, 103-127. Dordrecht: Kluwer.

Hearst, M. A. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceeedings of COLING-92: Proceedings of the 14th International Conference on Computational Linguistics (COLING-92), Nantes, France, 23-28 August 1992*, 539-545. San Francisco: Morgan Kaufmann.

Ide, N., Bonhomme, P., Romary, L. 2000. Xces: An xml-based encoding standard for linguistic corpora. In *Proceedings of the Second International Language Resources and Evaluation Conference*, 825-830. Paris: European Language Resources Association.

Kurc, R., Piasecki, M., Szpakowicz, S. 2010. Corpus-based extraction of morpho-syntactic patterns for the automatic acquisition of hypernymy. In M. A. Kłopotek, A. Przepiórkowski, S. T. Wierzchoń, and K. Trojanowski (eds.), *Intelligent Information Systems. New Approaches*, 77-90. Siedlce: Wydawnictwo Akademii Podlaskiej

Perez, C. A. 2012. Casuistry of Romanian functional dependency grammar. In M. A. Moruz, D. Cristea, D. Tufiş, A. Iftene, H. N. Teodorescu (eds.), *Proceedings of the 8th International Conference "Linguistic Resources and Tools for Processing of the Romanian Language"*, 19-28. Iaşi: Editura Universităţii "Alexandru Ioan Cuza".

Simionescu, R. 2012a. Graphical grammar studio as a constraint grammar solution for part of speech tagging. In M. A. Moruz, D. Cristea, D. Tufiş, A. Iftene, H. N. Teodorescu (eds.), *Proceedings of the 8th International Conference "Linguistic Resources and Tools for Processing of the Romanian Language"*, 109-118. Iaşi: Editura Universităţii "Alexandru Ioan Cuza".

Simionescu, R. 2012b. Romanian deep noun phrase chunking using graphical grammar studio. In M. A. Moruz, D. Cristea, D. Tufiş, A. Iftene, H. N. Teodorescu (eds.), *Proceedings of the 8th International Conference "Linguistic Resources and Tools for Processing of the Romanian Language"*,135-143. Iaşi: Editura Universităţii "Alexandru Ioan Cuza".

Snow, R., Jurafsky, D., Ng, A. Y. 2004. Learning syntactic patterns for automatic hypernym discovery. *Advances in Neural Information Processing Systems* (*NIPS)* 17: 1297-1304.

Appendix 1: Glossary of MSD notation

MSD tag	**The meaning of the notation (according to MULTEXT-East lexical specifications)**
Afpfson	Adjective qualifier positive feminine singular oblique –definiteness
Afpfsrn	Adjective qualifier positive feminine singular direct –definiteness
Afpms-n	Adjective qualifier positive masculine singular –definiteness
Ncfsoy	Noun common feminine singular oblique +definiteness
Ncfsrn	Noun common feminine singular direct –definiteness
Ncfsry	Noun common feminine singular direct +definiteness
Ncmsoy	Noun common masculine singular oblique +definiteness
Ncmsry	Noun common masculine singular direct +definiteness
Ncms-n	Noun common masculine singular –definiteness
Pp3fso-	Pronoun personal third feminine singular oblique
Spsa	Adposition preposition simple accusative
Tifsr	Article indefinite feminine singular direct
Timsr	Article indefinite masculine singular direct
Tsms	Article possessive masculine singular

Appendix 2: Glossary of notations used in patterns

Meta-character	Meaning
[ ]	Match anything inside the square brackets for ONE character position once and only once.
( )	The open parenthesis and close parenthesis are used to group parts of the included expression together.
?	The ? (question mark) matches the preceding character 0 or 1 times only.
|	The | (vertical bar or pipe) means OR logical between the left hand and the right hand of its values.