Between chunk ideology and full parsing needs[1]

Petya Osenova and Kiril Simov
BulTreeBank project
http://www.BulTreeBank.org
Linguistic Modelling Laboratory - CLPPI, Bulgarian Academy of Sciences
Acad. G.Bonchev Str. 25A, 1113 Sofia, Bulgaria
Tel: (+3592) 979 28 25, (+3592) 979 38 12, Fax: (+3592) 70 72 73
petya@bultreebank.org, kivs@bultreebank.org

Abstract

This paper discusses the intended balance between shallow parsing strategies and full parsing needs in the context of a treebank creation. Two kinds of mechanisms are described, which make the output of the shallow processors compatible for further stages of analyses. These mechanisms are divided into two groups: adapting and repairing ones.

1 Introduction

In recent years the applicability of shallow-parsed vs fully parsed texts has been discussed a lot in the NLP literature. Various attempts have been made to combine the advantages of shallow and full parsing processors with respect to different tasks and domains. This integration is viewed at least in the following ways: finding an adequate algorithm for extending the partial analysis to a deep one (Kübler and Hinrichs 2001) or finding a way to support both techniques within one formalism, thus having at disposal two different levels of granularity (Balfourier et al 2002) and (Crysmann et al 2002).

In all the mentioned works, however, the experiments rely on existing resources, tools for their processing, wide-coverage formal grammars. One challenge for the NLP community is the integration of the two mentioned mechanisms in the context of a less-processed language like Bulgarian. Our proposal is to enrich the shallow-parsing engine with additional features in order to facilitate the constraint grammar-based parser. For this purpose we apply certain strategies and achieve one, so to say, 'mediating level' of granularity.

The structure of the paper is as follows: Section 2 describes the annotation architecture within BulTreeBank Project. Section 3 concentrates on the main features of the CLaRK System. Section 4 gives a brief critical overview on chunks and chunking. In Section 5 two adapting strategies are described with respect to the lexicon and the NP chunker. Section 6 outlines other repairing mechanisms. The last section summarizes our conclusions.

2 Annotation Architecture

The syntactic annotation process within our project on Bulgarian (Simov, Popova and Osenova 2001, Simov et al 2002) relies on the combination of two formal grammars: 1) a partial grammar module, which produces shallow parsed texts; 2) an HPSG-based general grammar, which supplies a mechanism for deep linguistic analyses.

Partial grammar for shallow parsing

It is a multi-layered processor, organized in a cascaded manner, which means that the output from one tool becomes an input for the next. The module consists of the following tools: a dictionary-based POS tagger, which covers around 100 000 lexemes; a neuron-base disambiguator, which achieves precision of 93 % for all morphological features; named-entity recognition tool, which pre-identifies proper

[1] The work on the system is currently supported by BulTreeBank project funded by the Volkswagen-Stiftung, Federal Republic of Germany under the programme "Cooperation with Natural and Engineering Scientists in Central and Eastern Europe" contract I/76 887.

names, numerical expressions and abbreviations via pattern-maching techniques and gazeteers; multiword processing, which deals with idiomatic constructions of different kinds (introducing expressions, parentheticals etc.). The output from these tools is the input for the chunkers.

We compiled several reliable regular-expression chunk grammars for automatic identification of the non-recursive phrases (NPs, VPs, APs, AdvPs and PPs). They are applied in a cascaded order. For example, AP chunks and PP chunks are identified after the NP chunk recognition. Thus AP chunks contain only the elements, which are not a part of an NP chunk and PP chunks are formed by combining a preposition with a following NP. In accordance with Abney's ideas (see (Abney 1991) and (Abney 1996)) about 'easy-first parsing' and 'islands of certainty', we relied on the presence of clear indicators for specifying the chunk boundaries. For example, the adverb is not a clear indicator for the beginning of an NP, unless it is trapped between a preposition and a noun.

HPSG grammar

It is a symbolic parser, which is responsible for the adequate full analysis. It incorporates three elements: universal principles, language specific principles for Bulgarian and a lexicon. The HPSG grammar component is used in the project as a general constraint over the possible output syntactic structures of one and the same input sentence. The constraint is activated during the editing stage of the annotation process, checked by human experts. We also envisage using information from it as a top-down filtering over the partial grammars. It is encoded into an HPSG grammar development tool, ensuring the appropriateness conditions within the generated analyses.

The linguistic knowledge, encoded in the HPSG sort hierarchy, forms the backbone of the syntactic descriptions of the Bulgarian data. The sort hierarchy defines all possible linguistic structures that are further constrained by the grammar and/or the information, entered by the annotators. The annotation schemata, employed during the project, allow for composite tag definitions. Thus each tag in the syntactic structure, being decomposable, ensures the distribution of the grammatical information to the relevant substructures.

The HPSG grammar itself is viewed as a definition platform for the annotation scheme of the treebank. Hence, the elements of the treebank are not really trees, but feature graphs. Being exemplifications of the phrase-structure backbone, the tree-like representations are kept just for reasons of compatibility with other syntactic theories.

The graphical HPSG representation of the sentences is encoded in XML format. It turned out that within XML *element : attribute* structuring we can adequately present all the interesting linguistic phenomena like discontinuity, coreferences, ambiguity and dependency relations.

Thus, in principle, the preparation of the annotation scheme requires a proper handling of the following specific tasks:

1. Identification of the relevant for Bulgarian part of the HPSG sort hierarchy as described in (Pollard and Sag 1994); its modification with respect to the language-specific phenomena. It should be noted that the Bulgarian-specific part of the sort hierarchy will be subject to change during the development of the treebank according to the demands of the Bulgarian data.
2. Representation of the HPSG Universal Grammar principles from (Pollard and Sag 1994). This very general grammar is being used as a top constraint during the annotation of the trees in the treebank. For example, each headed phrase in the treebank has to satisfy the Head Feature Principle of HPSG. This grammar is being further extended by Bulgarian specific principles. These specific principles handle, for instance, the relatively free word order in Bulgarian, the clitic behaviour and distribution etc.
3. Compiling a fine-grained lexicon. It will extend the current available morphological lexicon with semantic and valency information.
4. Reusing and mapping the information from the shallow-based language tools and resources for deep linguistic analysis and in HPSG compatible format.

Given the very general nature of the sort hierarchy (compare, for example, the rich sort hierarchy of 5, 069 lexical and phrasal types in the German HPSG grammar (Crysmann 2002 et al, p. 3)), the lexicon and the principles, we consider a priority the fourth task, which re-distributes the focus from pure

resource compilation to the mapping strategies between the shallow parser output and the deep analysis.

Thus in this paper we concentrate on the fourth task, which presupposes the previous three and ensures a platform for deeper syntactic processing. We discuss the interface between the constraint-based linguistic theory and the principles of shallow parsing from two points of view: linguistic and implementation. For processing we use the CLaRK System (Simov et al 2001).

However, a problem arises when moving from the shallow stage to the deep one. The strategies, applied successfully at lower levels of processing are often inadequate for the consistency of deeper analyses. For example, the chunkers rely on pure syntactic information, ignoring attachment decisions and semantics, which are needed at next level. Thus at some point both - the procedures and the results need some tuning. It is supposed to be done in the following directions:

- adapting the resources and tools for theory-dependent purposes, and
- introducing some repairing mechanisms.

3 CLaRK System

In the process of annotation two software tools are relied upon: the CLaRK system, which supports the shallow processing stage and the manual annotation, and the TRALE system, which operates over the output from CLaRK and produces deep analyses.
The CLaRK System is an XML-based system for corpora development. The main aim behind the design of the system is the minimization of human intervention during the creation of language resources. It incorporates several technologies:

1. XML technology;
2. Unicode;
3. Regular Cascaded Grammars;
4. Constraints over XML Documents.

For document management, storing and querying, we chose the XML technology because of its popularity and its ease of understanding. The core of CLaRK is an Unicode XML Editor, which is the main interface to the system. Besides the XML language itself, we implemented an XPath language for navigation in documents and an XSLT engine for transformation of XML documents. The XSL transformations can be applied locally to an XML element ant its content.

The basic mechanism in CLaRK for linguistic processing of text corpora is the cascaded regular grammar processor. The main challenge to the grammars in question is how to apply them on XML encoding of the linguistic information. The system offers a solution using an XPath language for constructing the input word to the grammar and an XML encoding of the categories of the recognised words. This tool has been used for the development of the partial grammars, listed above. The rules of the grammar have the following form:

```
LC   =   RELC
R    =   RE
RC   =   RERC
RM   =   XMLF
```

Where RE is a regular expression which matches the sub-words of the input word for which the rule is applicable; RELC and RERC are regular expressions imposing constraints over the left and right context of the recognized sub-word; XMLF is an XML fragment which defines the category of the sub-word. XMLF can contain the recognized sub-word and its place in the fragment is determined by the variable \w which can be used zero or more times. The recognized sub-word is substituted with the fragment.

Several mechanisms for imposing constraints over XML documents are available. The constraints cannot be stated by the standard XML technology. The constraints are used in two modes: checking the validity of a document regarding a set of constraints; supporting the linguist in his/her work during the

building of a corpus. The first mode allows the creation of constraints for the validation of a corpus according to given requirements. The second mode helps the underlying strategy of minimisation of the human labour. The constraints are used to facilitate the manual annotation (disambiguation and HPSG annotation).

We envisage several uses for our system: Corpora markup. Here users work with the XML tools of the system in order to mark-up texts with respect to an XML DTD. This task usually requires an enormous human effort and comprises both the mark-up itself and its validation afterwards. Using the available grammar resources such as morphological analyzers or partial parsing, the system can state local constraints reflecting the characteristics of a particular kind of texts or mark-up. One example of such constraints can be as follows: a PP according to a DTD can have as parent an NP or VP, but if the left sister is a VP then the only possible parent is VP. The system can use such kind of constraints in order to support the user and minimize his work. Dictionary compilation for human users. The system will support the creation of the actual lexical entries whose structure will be defined via an appropriate DTD.

The XML tools will be used also for corpus investigation that provides appropriate examples of the word usage in the available corpora. The constraints incorporated in the system will be used for writing a grammar of the sublanguages of the definitions of the lexical items, for imposing constraints over elements of lexical entries and the dictionary as a whole. Corpora investigation. The CLaRK System offers a rich set of tools for searching over tokens and mark-up in XML corpora, including cascaded grammars, XPath language. Their combinations are used for tasks such as: extraction of elements from a corpus - for example, extraction of all NPs in the corpus; concordance - for example, give me all NPs in the context of their use ordered by a user defined set of criteria. The system is implemented in JAVA.

4 The chunk strategies

The chunker relies on the preference of accuracy to coverage. It aims at recognizing segments, which are not complete, but easy and reliable to capture. For this reason chunking supports non-recursiveness. It avoids the attachment problems (PPs, modifying clauses etc) and semantic information. Chunkers recognize phrases with clear indicators of their beginning and ending.
All these strategies, however, cause the well-known problems like the following:

1. Maximal recursive phrases cannot be captured.
2. Some non-recursive groups are partially identified. It happens in cases when the phrase starts with an unclear indicator. For NPs it is the adverb or the non-definite participle. As a disambiguating context we consider the preceding preposition.
3. Misinterpretations are often caused: within the conjuncts of the coordination, prepositional phrases or NPs of type NN. In the coordination case, it is impossible to deal with scoping, because of the relatively free order in Bulgarian. Consequently, the modifier's scoping cannot be interpreted adequately; in the PP one, within sequences of two prepositional phrases the second one could modify the noun within the first NP phrase. In the NN case the recognized entities are sometimes either implausible linguistically, or just two different NPs. See the following pair examples, which demonstrate both – the chunk output and the aimed correct analysis:

   (1a) Toj vidja [NP hubavi mazhe] i [NP zheni]
   he see-aorist beautiful-pl man-pl and woman-pl
   He saw beautiful men and women
   (1b) Toj vidja [NP hubavi [NP mazhe i zheni]]
   he see-aorist beautiful-pl man-pl and woman-pl
   He saw beautiful men and women

   (2a) Vlajzoh [PP v magazina] [PP na chicho si]
   came-1p,sg in shop-the-m,sg of uncle-n,sg my
   I entered my uncle's shop
   (2b) Vlajzoh [PP v [NP magazina [PP na chicho si]]]
   came-1p,sg in shop-the-m,sg of uncle-n,sg my
   I entered my uncle's shop

(3a)*[*[edna biznes] [sreshta]]
one-fem,sg business-m,sg meeting-fem,sg
a business meeting
(3b) [edna [biznes sreshta]]
one-fem,sg business-m,sg meeting-fem,sg
a business meeting

Note that in (3a) the leftmost premodifiers usually do not agree with the first noun, but with the second. For this reason the analysis is not appropriate.

As the chunker relies on compromise decisions with respect to linguistic adequacy, its output needs to be tuned and supported by other mechanisms, which to add the missing information or to pre-interpret the existing ones. Such mechanisms are presented in the next section.

5 Adapting procedures

The adapting procedures are needed for several reasons. One of them is due to the lack of a large-scale HPSG (or any other formal) grammar and lexicon for a less-processed language like Bulgarian. Another reason is that shallow parsing output has to be integrated for solving more complex linguistic tasks, such as word order and discontinuity, correct attachments etc. For that reason we rely on the pre-processing stage not only to cover the data, but to be reliable enough for further theory-dependent exploration. We expect our morphological dictionary to meet the basic requirements of an HPSG-oriented lexicon and the chunkers to encode the needed for an HPSG parser knowledge.

5.1 Adapting the tagset into an HPSG-based lexicon

Recall that during the preprocessing stage of the corpus we rely on the morphological tagger, which assigns all possible analyses and the neuron-network based disambiguator, which reduces these possible analyses to the most probable one. As the tags contain not only POS identification, but information about all grammatical characteristics, we decided to convert the positional tagset into an HPSG compatible lexical database. We aim at two main things: 1. entry structures, which are maximally close to the sort hierarchy encodings and relations in the theory and 2. enriching the lexical level (respectively - lexicon) with multiword expressions. In the first case we keep the original POS categories. Phenomena like substanivization, nominalization or type shifting are handled by special principles. For the moment the valency of non-verbs is underspecified. It is presented separately for a small semantic class of quantity nouns, for example. The argument structure of the verbs is derived from their transitive-intransitive characteristics, but it is still not sufficient, because some intransitive verbs can have complements as well. For this reason we envisage to use the information from the machine-readable valency dictionary of Bulgarian. In the second case we try to store the most frequent multiword constructions, which either show their own specific linguistic features, or have to be captured better as functional units than as decomposed ones (complex conjunctions or complex PPs). Problems with agreement features arise in some idiosyncratic complex proper names, because some of them participate only with their own gender/number – *Varna-f e krasiva-f* (Varna is beautiful), while others can trigger an external one (for example, the gender of the missing noun adjunct, in this case – village, which is a neuter noun): *Dikanite-pl e krasivo-n,sg* (Dikanite is beautiful). In such cases the semantic classes are consulted in order to define the correct agreement possibility/possibilities.

Viewing some phrasal units as lexical items helps us to compensate the lack of elaborate HPSG grammar engine and at the same time to improve robustness and to pre-repair the possible chunk errors.

The idea of using attribute:value specifications for compositional representation has been explored at different linguistic levels and for different purposes. See for example the supertag idea in (Joshi and Srinivas 1994) or the morphological tagset for Turkish (Hakkani-Tur et al 2000). Our approach is similar to the so called typological marking, i.e. replacing the output information of the tagger with more fine-grained feature-based one (Nioche and Habert 2001). At this stage we have worked out mapping rules, which view the positioned in a certain order symbols within the tags as values of some atomic or complex, hierarchically structured attributes. Here is an example:

Ncmsi -> <1=pos>=noun
<2=type>=common
<3=agreement gender>=masculine
<4=agreement number>=singular
<5=definiteness>=definite

The mapping rules are supposed to be of different ranks as to generality. Thus, except for using the positional information, we developed more general rules, capturing the groups of all the nouns, verbs etc. For example all the nouns receive the following lexical specification as a default:

N -> <sign>=word
<cathead>=noun
<catvalency>=valency
<head>=substantive

Once the levels of abstraction are properly defined, the information is mapped in appropriate way without any overgeneralization or undergeneralization.

5.2 Adapting nominal chunks to HPSG-oriented analyses

Chunking is considered to be a theory independent step, which precedes the deep syntactic analyses. But we have to take into account its further integration into the HPSG-based implementation. Such an attempt was introduced in (Richter-Sailer 1996) for German with respect to the word order, for example.
This integration can be done within several mutually constraining directions:

1. The grammar rules present the main grammatical patterns within NPs, such as gender and number agreement or specific element order. This information can be used in producing the HPSG analyses. One advantage to be explored here is that chunking relies on the notion of head. Thus the head-adjunct structures are predicted easily during the chunking stage.
2. The compiled 'golden standard' set of sentences, which are assumed to represent the main NP patterns, can be used as a top-down filtering on the NP chunk patterns. Especially in cases, where the NP parses remained incomplete due to vague starting indicators. Thus the 'golden standard' patterns would add the relevant information to the incomplete parses. The evaluation showed precision of 92 % and recall of 87.7 %
3. The corpus itself can be used as a correcting mechanism with its data-driven frequency NP patterns. These patterns will help for modelling the typical grammatical relations within NPs.

Nominal chunks and head-lexicalized relations

The morphosyntactic tags of the NP elements show the terminal categories of the constituent, but we aim at deriving explicit head-modifier information. This task is not trivial, because: 1. it is theory dependent and 2. it relies on the language specific decisions over certain linguistic phenomena. Such tasks have already been targeted at using, for example, memory-based methods for assigning function labels to the constitents (Kübler and Hinrichs 2001) and (Buchholz et al 1999).

For the base, non-recursive Bulgarian NPs we assume that the internal head-lexicalized elements are: *heads*, *adjuncts of different kinds*, *lexicalized prosodic groups like 'a definite adjective plus a possessive clitic'*.

Our tasks are: 1. to use chunks for predicting the head-dependant roles and 2. to incorporate the chunks into deeper analyses.

Transforming nominal chunk rules into HPSG-based dependency relations

For the first task we use the CLaRK system and for the second we rely on the inreface between CLaRK and TRALE. Thus the chunk output from the CLaRK system has to be in accordance with HPSG relational specifications. We decided to pre-encode the grammatical information into head-lexicalized features in the following way:

First, we have created the basic NP chunk grammar. As it was shown above, it consists of number of rules. Here two examples of shorter rules are given for simplicity:

Rule1
(target types: shapkata mi)

LC = empty
R = "N#",("Psx#t"|"Ps#t")
RC = empty
RM = <np type="common">\w<np>

The rule encodes the following: the left and right context areas are empty; *N#* stands for any noun; *Psx#t* stands for any reflexive-possessive pronoun clitic and *Ps#t* stands for any possessive pronoun clitic.

Rule2
(target types: vsichki moi shapki)

LC = empty
R = ("Pc#"|"Pf#"|"Pd#"),("Ps#"|"Psx#")?,"N#"
RC = empty
RM = <np type="common">\w<np>

The rule encodes the following: the left and right context areas are empty; *N#* stands for any noun; *Pc#* stands for any collective pronoun, and *Pf#* stands for any indefinite pronoun, *Pd#* stands for any demonstrative pronoun, *Ps#* stands for any possessive pronoun, *Psx#* stands for every reflexive-possessive pronoun.

As a second step, we pre-encode every chunk rule into subrules and apply the new rules within NP chunks. The following transformations are made: every conjunct from the rule (no matter complex or not) occupies the position of the regular expression area from left to right direction. In this way when the last element of the chunk rule is in the regular expression area, all the others are available in the left context area. From the second new subrule on, the right context area becomes occupied as well. The category of every subrule, encoded as return mark-up next obeys the HPSG-based and modified for Bulgarian dependency relations as head, adjunct, complement, clitic etc. Here we give an example of such a decomposing strategy over just the first simple rule:

Rule1 - new subrule1

LC = "N#"
R = "Psx#t"|"Ps#t"
RC = empty
RM = <clitic>\w<clitic>

After the application of the first sub-rule, the possessive clitic receives the special clitic tag.

Rule1 - new subrule2

LC = empty
R = "N#"
RC = "Psx#t"|"Ps#t"
RM = <head>\w<head>

After the application of the second sub-rule, the noun receives the head tag.

In this way we receive output structures in XML like the one below (note that the morphological tags are omitted for clarity):

<np><head>shapkata</head><clitic>mi</clitic></np>

Similarly the result from the second rule is:

<np><adjunct>tova</adjunct><head>neshto</head></np>

Hence, by this strategy, the set of the subrules, constituting one rule within a grammar, gives dependencies and their linearity, valid for the recognized structure.

Incorporating the nominal chunks into a deeper HPSG analysis

First, the base nominal groups are captured by the chunker. The rules are assigned different return mark-up categories according to the targeted elements. Thus the bare nouns as well as nouns, post-modified by possessive clitics, are captured as lexicals – N. Other recognized structures are assigned the category NPA, which means NP of type *head-adjunct*. Manual post-editing is needed in cases of NN types, for attachment decisions and for cases where the NP is incomplete due to a non-clear starting indicator. Here is an example, in which the two nouns are first identified as head-complement structure and then the head-adjunct label is assigned.

```
<analysis>
<S>
   <VPA>
         <VPS>
               <NPA>
                     <A><w ana="Afsi">Дълга</w></A>
                     <NPC>
                           <N><w ana="Ncfsi">колона</w></N>
                           <N><w ana="Ncmpi">автомобили</w></N>
                     </NPC>
               </NPA>
               <V><w ana="Vpit+f-r3s">потегля</w></V>
         </VPS>
         <PP>
               <Prep><w ana="R">към</w></Prep>
               <N><w ana="Npnsi">Делхи</N>
         </PP>
   </VPA>
   <pt>.</pt>
</S>
</analysis>
```

Here *Afsi*, *Ncmpi*, *Ncfsi*, *Npnsi*, *Vpit+f-r3s*, *R* are tags from our tagset encoding POS information, but also gender, number, definiteness, tense, person etc. The XML tags *VPA*, *VPS*, *NPA*, *PP* encode the type of the constituent. The annotation scheme we use here is described in (Simov and Osenova 2003) and behind the type of the constituents provides mechanisms for expressing co-reference of different kinds between the constituents. In this way it ensures the relevant HPSG analyses.

6 Repairing mechanisms

The repairing mechanisms deal with the flaws of the partial parsing which can not be corrected by adding information as it was discussed in the previous section. Generally, these are cases, in which a constituent is partially analyzed in two (or more) chunks. These chunks cannot be grouped together in the true constituent by using monotonic parsing system. We envisage these cases to be repaired by transformations of the already constructed chunk analysis.

The process of repairing includes the following steps:

1. *Problem identification.* This is a formal procedure which determines the possible places in the chunk analysis where the flaw appears.
2. *Transformation.* This includes a definition of an appropriate transformation which is able to restructure all instances of the problematic case into the correct one.

As a means for description of problematic cases we use the cascaded grammars, the constraints system and XPath language engine of the CLaRK system. The cascaded grammars are used to group the problematic chunks into a bigger group that contains the chunks that have to be transformed into a new configuration of chunks. Then, if necessary, we use the constraints of the CLaRK system in order to add information to the newly introduced chunks. This information will be context dependent. After this operation we can accept that each problematic case is marked-up with appropriate information. Each of these new chunks can be pointed to by an XPath expression.

The repairing transformation is defined as an XSL Transformation over the local tree which represents a problematic case. Then the local tree is shown to the annotator and s/he is asked whether the transformation to be applied. When all problematic cases are inspected by the annotator and the real errors are transformed into the right one, the auxiliary information added during the problem identification step is removed with the removal tool of the system.

This repairing mechanism can work for the moment only in combination with a human annotator. Thus it is applicable to such chunk grammars which have a relatively high level of precision and the repairing is needed for a small amount of easy detectable problematic cases. In our project such cases are the PP attachment problem, the NP coordination problem and NN types, as discussed above. In this cases it is simple to write a regular grammar that groups together the consecutive PPs or the two parts of the coordination. Then the annotator inspects them and applies the transformations.

7 Conclusion

In this paper we described the mapping strategies between the tools for shallow parsing and the grammar-based annotation engine. These strategies are of different kinds and they aim at the following: (1) adapting the existing resources for deeper analyses and (2) creating a reliable base for consistent linguistic analyses.

In our future work we envisage to use additional linguistic information in order to narrow the selection of problematic cases and even for automatic repairing of some of the cases. Such linguistic information will be taken from the valency dictionary of Bulgarian and the noun lexicon, which is under development within the project. The valency dictionary determines the possible valency frames of about 1000 verbs and for each argument in a given frame the morphological and semantic restrictions are defined. We mapped the semantic restrictions of the arguments to the core ontology of the SIMPLE project. At moment we are working on classification of the Bulgarian nouns to the same ontology. Then we will try to use this information for recognition of the problematic cases.

References

Abney St 1991 *Parsing By Chunks*. In: Robert Berwick, Steven Abney and Carol Tenny (eds.), *Principle-Based Parsing*. Kluwer Academic Publishers, Dordrecht.

Abney St 1996 *Partial Parsing via Finite-State Cascades.* In: *Proceedings of the ESSLLI'96 Robust Parsing Workshop*. Prague, Czech Republic.

Balfourier J-M, Philippe B, van Rullen T 2002 *From Shallow to Deep Parsing using Constraint Satisfaction*. In: Proceedings of COLING-2002

Buchholz S, Veenstra J, Daelemans W 1999 *Cascaded Grammatical Relation Assignment*. In *Proceedings of EMNLP/VLC-99*, University of Maryland, USA, pp. 239-246.

Crysmann B, Frank A, Kiefer B, Müller St, Neumann G, Piskorski J, Schäfer U, Siegel M, Uszkoreit H, Xu F, Becker M, Krieger H-U 2002 *An Integrated Architecture for Shallow and Deep Processing*. In *Proceedings of the 40th Annual Meeting of the ACL*, Philadelphia, July 2002, pp. 441-448.

Götz Th, Meurers D 1997 *The ConTroll system as large grammar development platform*. In *Proceedings of the ACL/EACL post-conference workshop on Computational Environments for Grammar Development and Linguistic Engineering*. Madrid, Spain.

Hakkani-Tur D, Oflazer K, Tur G 2000 *Statistical Morphological Disambiguation for Agglutinative Languages* In: *Proceedings of COLING 2000*, Saarbrucken, Germany.

Joshi AK, Srinivas B 1994 *Disambiguation of Super Parts of Speech (or Supertags): Almost Parsing*. In: *Proceedings of the 17th International Conference on Computational Linguistics (COLING '94)*. Kyoto, Japan.

Kübler S, Hinrichs E 2001 *From Chunks to Function-Argument Structure: A Similarity-Based Approach*. In *Proceedings of ACL 2001*, Toulouse, France.

Nioche J, Habert B 2001 *Using feature structures as a unifying representation format for corpora exploration* In Paul Rayson, Andrew Wilson, Tony McEnery, Andrew Hardie, Shereen Khoja (eds), *Proceedings of the CORPUS LINGUISTICS 2001 Conference*, eds. Lancaster, pp. 401-412.

Pollard C, Sag I 1994 *Head-Driven Phrase Structure Grammar* University of Chicago Press, Chicago, Illinois, USA.

Richter F, Sailer M 1996 *Regions and Word Order* Handout (9 pages) for a talk given at the *International Center Workshop on Computational Linguistics in Tuebingen* on September 20th, 1996.

Simov K, Popova G, Osenova P 2001 *HPSG-based syntactic treebank of Bulgarian (BulTreeBank)*. In: "*A Rainbow of Corpora: Corpus Linguistics and the Languages of the World*", edited by Andrew Wilson, Paul Rayson, and Tony McEnery; Lincom-Europa, Munich, pp. 135-142.

Simov K, Peev Z, Kouylekov M, Simov A, Dimitrov M, Kiryakov A 2001 *CLaRK - an XML-based System for Corpora Development*. In Paul Rayson, Andrew Wilson, Tony McEnery, Andrew Hardie, Shereen Khoja (eds), *Proceedings of the CORPUS LINGUISTICS 2001 Conference*, eds. Lancaster, pages: 558-560.

Simov K, Osenova P, Slavcheva M, Kolkovska S, Balabanova E, Doikoff D, Ivanova K, Simov A, Kouylekov M 2002 *Building a Linguistically Interpreted Corpus of Bulgarian: the BulTreeBank*. In: Proceedings from the LREC conference, Canary Islands, Spain.

Simov K, Osenova P 2003 *Practical Annotation Scheme for an HPSG Treebank of Bulgarian*, In *Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora (LINC-03),* Budapest, Hungary.