Combining the named-entity recognition task and NP chunking strategy for robust pre-processing[1]

Petya Osenova and Sia Kolkovska
BulTreeBank Project
http://www.bultreebank.org
Linguistic Modeling Laboratory, Bulgarian Academy of Sciences
Acad. G. Bonchev St. 25A, 1113 Sofia, Bulgaria
petya@bultreebank.org, sia@ibl.bas.bg

Abstract

In this paper we aim at outlining the joint exploitation of two nominal grammars – named-entity grammar and chunk grammar in the process of building a treebank. Their contribution towards unified and effective NP shallow parser is stressed upon. Taking into account their specific underlying principles, the points of interrelation are discussed and related problems are pointed out.

1 Introduction

Within a treebank usually three things are of great importance: its size, the detailness of the syntactic descriptions and the desired high level of quality. Thus well-designed semi-automatic facilities are required for the annotators during the stage of annotation. Such a minimization of human labour is achieved maily by exploiting all the possibilities for providing automatic partial analyses of the input strings before the stage of the 'attachment-resolution' annotation.

The syntactic annotation process within BulTreeBank Project relies on the development of two formal grammars: 1) a partial grammar for shallow parsing and 2) an HPSG-based general grammar.

To put it more precisely, the annotation architecture of the HPSG-based treebank includes, generally speaking, the following pre-processing steps: tokenization, POS tagging, morphosyntactic disambiguation, named entity recognition and partial parsing (Simov et al. 2001; Simov et al. 2002; Osenova, Simov 2002). These steps prove out to be necessary when using a symbolic grammar as an annotation scheme, because the grammar-based parsing ensures deep and detailed analyses at the cost of lower coverage and robustness (Dipper 2000, p. 59). More elaborate description of the pre-processing components in BulTreeBank annotation architecture is presented in Table 1.

In this paper we view the place of two nominal grammars – Named-entity grammar and General chunk grammar - within the overall stage of pre-processing. Despite the fact that they follow different ideologies, we claim that these grammars are complementary and together can contribute to at least three things: 1) improving the accuracy and coverage of the partial parsing; 2) facilitating the stage of deeper syntactic analyses and 3) improving the recognition of named entities.

Table 1

Ordering of the pre-processing steps	**Supportive Knowledge-based sources or features**	**Output**
1. Symbol-based classification	← token categories and rules	Set of tokens, categorized

[1] This work is funded by the Volkswagen Stiftung, Federal Republic of Germany under theProgramme "Cooperation with Natural and Engineering Scientists in Central and Eastern Europe" contract I/76 887.

	using specific orthographical features	by distinctions like ALLCAPITALS, Firstcapital, lowercased etc.
2. General token classification	← word categories, such as common words, proper names and abbreviations	Pre-compiled word-lists of potentially classified tokens, which result from the additional application of a statistical approach
3. Morphological tagging	← morphological dictionary, guessers	All possible analyses for the tokens are given in accordance with the BulTreeBank positional tagset
4. Disambiguation	← neural network-based disambiguator	The true analysis for a given token is assigned with certain probability values. The analysis with the highest probability is chosen as the true one. The accuracy achieved is about 93-95 % depending on the complexity of the tag (POS only or containing all grammatical information)
5. Partial grammars engine: sentence determination detection of abbreviations numerical expressions	← pre-compiled lexicons from stages 2 and 3	Detected and classified sentence types, abbreviations and numerical expressions
6. Named Entities Grammar for persons, organizations, places and miscellaneous names	← gazeteers and pre-compiled lexicons from stages 2 and 3	Detected and classified proper names of different kinds with grammatical features assigned
7. base NPs chunker	← grammatical features, partial grammars, Named entities grammar	Non-recursive nominal chunks

Traditionally, the named-entity recognition (NER) module consists of two sub-modules: 1) numerical expressions, dates, special symbols and 2) names. The former is considered to be a trivial and, more or less, language-independent task, while the latter appears to be more complex. Interesting for us is the incorporation of these sub-modules within a consistent and accurate nominal parser, because these modules appear to operate on the nominal groups, but in slightly different way. The most important thing for us is the overall result to be good enough for the next stage – HPSG grammar parsing. For this reason we put a special stress on the recognition, the grammatical and ontological interpretation of names (words and phrases), and their relation to the general NP chunker.

In computational linguistics area NER is usually discussed with respect to two different tasks: 1) Information Extraction and Information Retrieval; 2) Building an input module to a robust shallow parsing engine (Grover, McDonald et al, p. 3). As it was mentioned above, we concentrated on named-entity recognition preferably in accordance with the second task.

In our pre-processing annotation architecture NER is assumed to be solved prior to the stage of NP chunking. The motivation is as follows:

1. a good chunk grammar relies on the tagger accuracy and on efficient recognition of the unknown words. Most of these unknown words appear to be different kinds of names. Hence their impact on overall texts is significant. Therefore, with the general recognition component of the named-entity task included, the efficiency of the chunking stage is improved regarding the quality of the input data.
2. NER module pre-solves some of the attachment problems, which are encountered by an NP chunker and thus - usually left for further stages.

In the paper we aim at outlining the joint exploitation of two nominal grammars – named-entity grammar and chunk grammar in the process of building a treebank. Their contribution towards a unified and effective NP shallow parser is stressed upon. Taking into account their specific underlying principles, the points of interrelation are discussed and related problems are pointed out.

The structure of the paper is as follows: The next section discusses the general architecture and encoding of the two nominal grammars. Section 3 describes in detail the Named-entity recognition grammar (NER) within the BulTreeBank pre-processing module. In Section 4 the base NP chunker (NPCG) for Bulgarian is presented briefly. In section 5 the points of overlapping between the two grammars are outlined. Section 6 presents our conclusions and ideas on future work.

2 The architecture and encoding of the grammars

From linguistic point of view the two nominal grammars are reflections of the language-specific nominal phenomena. For this reason their core information is reusable with respect to different purposes.

From implementational point of view the grammars are constructed with respect to the CLaRK system design. For this reason we first present some general information about CLaRK grammar engine and then discuss the grammars themselves.

2.1 The regular XML-based grammars within the CLaRK system – a general overview[2]

CLaRK is an XML-based software system for corpora development implemented in JAVA.The core of the system is an XML editor. There are, however, additional language processing modules, which support the linguistic work: a tokenizer, a finite-state engine, an Xpath query language and constraints engine. Here we briefly sketch the functions of the cascaded regular grammars over XML documents.

The notion of cascaded regular grammars

In the CLaRK system the definition of regular grammars follows (Abney 1996).

The general idea underlying cascaded regular grammars is that there is a set of regular grammars. The grammars in the set are in particular order. The input of a given grammar in the set is either the input string if the grammar is first in the order or the output string of the previous grammar. Another specific feature of the cascaded grammars is that each grammar tries to recognize only a particular category in the string but not the whole string. The parts of the input word that are not recognized by the grammar are copied to the output word. The rules follow the formula below:

C -> R

where **R** is a regular expression and **C** is a category of the words recognized by **R**.

A *regular grammar* is a set of rules. It works over a word and tries to segment it into a sequence of subwords in such a way that each of these subwords is recognized by a regular expression rule within the grammar.

An additional requirement suggested by (Abney1996) is the so-called *longest match*, which is a way to choose one of the possible analyses for a grammar. The longest match strategy requires that the

[2] The explanations in this subsection are based on (Simov, Kouylekov, Simov 2002).

recognized sub-words from left to right have the longest length possible. Thus the segmentation of the input word starts from the left and tries to find the first longest sub-words that can be recognized by the grammar and so on to the end of the word.

The representation of tokens within the regular grammars

First of all, it is accepted that each grammar works on the content of an element in an XML document. Therefore when a text is considered an input word for a grammar, it is represented as a sequence of tokens. We can refer to the tokens in the regular expressions in the grammars by tokens. In the CLaRK system, however, the means for describing tokens are enlarged with the so-called *token descriptions,* which correspond to the letter descriptions in the above section on regular expressions. In the *token descriptions* the following means are used: strings (sequences of characters), wildcard symbols (# for zero or more symbols, @ for zero or one symbol, and token categories). Each token description matches exactly one token in the input word.

The *token descriptions* are divided into two types - those that are interpreted directly as tokens and others that are interpreted as token types first and then as tokens belonging to these token types.

The first kind of *token descriptions* is represented as enclosed in double quotes. The string is interpreted as one token with respect to the current tokenizer. If the string does not contain a wildcard symbol then it represents exactly one token. If the string contains the wildcard symbol # then it denotes an infinite set of tokens depending on the symbols that are replaced by #. This is not a problem in the system because the token description is always matched by a token in the input word. The other wildcard symbol is treated in a similar way, but zero or one symbol is put in its place. One token description may contain more than one wildcard symbol. Within the regular expressions the angle brackets are used in order to denote the boundaries of the element values. Inside the angle brackets we could write a regular expression of arbitrary complexity in round brackets. As letters in these regular expressions we use again token descriptions for the values of textual elements and the values of attributes. For tag descriptions we use strings which are neither enclosed in double quotes nor preceded by a dollar sign. We can use wildcard symbols in the tag name. Thus

<p> is matched with a tag p;

<@> is matched with all tags with length one.

<#> is matched with all tags.

The encoding of the category of the recognized elements

With respect to the XML encoding it was decided that the category for each rule in the CLaRK system is a custom mark-up that substitutes the recognized word. Since in most cases we would also like to save the recognized word, we use the variable **\w** for the recognized word. For instance, the following tokens are assigned their morphosyntactic characteristics:

<Det>\w</Det> -> "the"|"a"

<N>\w</N> -> "telescope"|"garden"|"boy"

<Adj>\w</Adj> -> "slow"|"quick"|"lazy"

<V>\w</V> -> "walks"|"see"|"sees"|"saw"

<Prep>\w</Prep> -> "above"|"with"|"in"

The left and the right context

One of the extensions of the regular grammars within the grammar is the description of the left and the right context of a given recognized subword. Hence, the rule is the same as above, but the left and the right contexts are added as well:

C -> LC : R : RC

Where **C** is the category, **LC** is a regular expression describing the left context of the recognized words, **R** is a regular expression describing the set of all words, which have to be recognized by the rule, **RC** is a regular expression describing the right context of the recognized words. Note that the regular expressions **LC** and **RC** can be empty and in this case there are no constraints over the left and the right context.

After introducing the main ideology of the CLaRK grammar engine, we can outline how the nominal grammars fit in.

2.2 The nominal grammars and CLaRK

As it was mentioned above, both grammars are supported by the grammar engine within the CLaRK system (for more details see - Simov, Kouylekov, Simov 2002). The general NP chunker operates over the content of the **ta** element, which presents the true morphosyntactic tag of the token. The Named entity grammar operates over the content of the **ph** element, which presents the orthographical form of the token as well as on the content of the **ta** element. The chunker composes its rules on token descriptions, which are interpreted directly as tokens, while NER grammar combines them with token types in order to match its targets. It is necessary, because NER grammar relies on capitalization as well. Below we give two example rules, taken from both grammars. Note that the regular expression rules of the item are presented as well as their categories:

(1) a chunk grammar rule

Item: <("A#"|"Pd#")>,<("Pneos-n"|"Pfeos-n")>

goliamo nishto, tova neshto

'big nothing', 'this something'

Category: <np type="common" gram="Nnsi"></np>

The regular expression matches all sequences of tokens, which are tagged as adjectives ("A#") or demonstrative pronouns ("Pd#"), followed by a singular, neuter indefinite ("Pneos-n") or negative pronoun ("Pfeos-n").

(2) a named-entity grammar rule

Item:<("Zapad#"|"Sever#"|"Iztoch#"|"Ujg#"|"Sred#"),"A#">, <$CYRwc, ("N#"|"unknown")>

Zapadna Evropa

'West Europe'

Category: <np type="LocNE"></np>

This rule underspecifies the grammatical characteristics as to gender and number. For this reason the 'gram' attribute is not present within the tag. When divided into more sub-rules, each of which bearing agreement features for number and gender, then the 'gram' attribute will bear the relevant morphosyntactic information.

The regular expression matches all sequences of tokens, which are: tagged as adjectives and at the same time it aims at certain lexemes (in this case – the geographical directions), followed by a capitalized noun or an unrecognized token.

As it was demonstrated in the above examples, the return XML mark-up encodes as many similarities and peculiarities of the matched expressions as necessary. We aim at having as much grammatical/semantic information at mother nodes as possible. For example, the chunker has information about the grammatical features of the given NP: <np type="common" gram="Nmsi">\w</NP>, where *Nmsi=Noun, masculine, singular, indefinite*. The named entity grammar supplies additionally information about named-entity category: <np type="PersonNE" gram="Nmsi">\w</np>.

The cascadedness is very important for the grammars application. For one thing, it is obeyed between the grammars:

- The NE grammar is run before the chunker, but then there are special chunk rule operations over some of the NE grammar output tags.

Secondly, the cascadeness is kept within the two grammars:

- The chunker runs first its module with left context included, and then – the main module, which is irrelevant to the context.
- The NE grammar runs first its rules, which capture the longest string of capitalized words and then runs the rules, capturing the shorter ones. It consists of rules that successfully restrict the overgeneration in some cases.

2.3 The nominal grammars and linguistically processed data

The two grammars are hand-crafted and they rely on models, derived from the collected data. The NE grammar has to do with proper names and aims at their ontological interpretation. Needless to say, it assigns the grammatical characteristics to the names as well. Recursive NPs are also identified, because the NEs are considered to be atomic expressions at this level despite their internal structural complexity. The base NP chunk grammar handles all nominals. It excludes recursivity and semantics, and relies on constituency. Let us discuss these interactions in more detail here.

One good test for the external atomism and internal complexity of the Named entities is the fact that many of them are extensions of acronyms. For example: *savremennata [BAN]* ('contemporary-the BAN')=*savremennata [Balgarska akademija na naukite]* ('contemporary-the Bulgarian Academy of Sciences'). Hence, if such cases were treated as just the usual string of words, not as a whole entity, then the chunker would analyze them in the following way: *[savremennata Balgarska akademija] na [naukite]* ('[contemporary-the Bulgarian Academy] of [Sciences]'), thus posponing the PP attachment of the second part for the next stage. But unlike the common NPs, where the usual linguistic treatment follows the heuristic: *take the other modifiers first and then the PP modifiers ([np [np the boy] [pp with [np the hat]]])*, in these cases it is very important to keep the conceptual wholeness of the structure. Thus, another linguistic strategy seems to be more justified, namely – identifying such recursive structures first and then combining them with the pre-modifiers.

Below we present the underlying methodologies of the two grammars and then, their interrelation is discussed.

3 BulName - Named Entity Recognition module of the BulTreeBank project

As a module of BulTreeBank project, BulName is closely connected to the other sub-components and grammars, developed within the project. Simultaneously the output of BulName supports the partial parsing. Additionally, it can contribute to the development of a semantic-based annotation layer over the treebank (together with other knowledge-based resources, such as Bulgarian electronic valency dictionary) and can additionally support extraction of linguistic ontologies.

Linguistic pre-processing and the BulName module

BulName efficiency is supported by the previous pre-processing steps in the following specific way:

1. Recall that the output of the *General token classification stage* are lists with words, categorized on the base of the token types as common words, proper names and abbreviations. The lists serve as additional means for recognition of NEs in sentence initial positions and as preliminary probabilistic prompts, derived from the whole corpus.

2. The efficiency of the NER grammar relies on the accuracy of the POS tagger as well. POS tagging includes not only part-of -peech annotation, but more detailed morphological information in addition (gender, number, tense etc.).
3. For the recognition of abbreviations a list of abbreviations is used together with a technique developed for their identification (Osenova, Simov 2002) and processing (Ivanova, Dojkoff 2002). This step is necessary for removing the errors, caused by category ambiguity within a graphical feature like capitalization.

After using the *General token classification* for initial prompts on the possible names detection, NER grammar relies heavily on the following pre-processing stages:

- lexicon lookup module
- gazetteer lookup module

In the *lexicon lookup module* all words with first position in the sentences are checked in the lexicon with respect to their possible membership in the list of the common words. When a word occurs in the lexicon, it is considered a non-named entity and is excluded from the list of potential names. The common words lexicon, employed in the BulTreeBank project, is based on "Bulgarian Inflectional Morphology Dictionary" (Popov, Simov, Vidinska 1998).

The *gazetteer lookup module* includes matching against lists with names and then assigning them the appropriate category and the grammatical characteristics, if possible. The ambiguous names are marked by more than one tag in gazetteers. For instance, the name *Rakovski* is marked as a name of person and as a location (town, street). The ambiguity of such names is resolved later by a submodule within the BulName module.

At present the gazetteers consist of 11 000 words – Bulgarian and foreign person names, locations from the whole world and organizations. The lists have been compiled from appropriate Internet sites and then linguistically processed by experts.

Description of the BulName module

There exists a great number of literature, discussing different approaches for NER task. Here only some of them are mentioned:

- computational linguistic approaches (Stevenson and Gaizaukas 2000)
- statistical and machine learning techniques (Buchholz, S. et al. 2000; Soderland S. 1997; R. Yangarber et al. 2000; Borthwick A. et al.1998; Baluja S. et al. 2000; Zhou G. et al. 2000).
- matching rules - LaSIE II (Humphreys K. et all.1998), the MUSE system (Diana Maynard et al.)
- hybrid approaches - Mikheev et al. 1998; Mikheev et al. 1999; FACILE project (Black W. et al. 1998); NERC subpart of the CROSSMARC project (Glover C., McDonald et al.); Swedish Named Entity Recognizer (Dalianis H. et al. 2001); Farmakioutou et al. 2000.

The NER module within the BulTreeBank is a hybrid one in the following sense: it relies on combination of hand-crafted rules describing patterns to match, and partial matching techniques.

It includes two basic layers: *the general recognition layer* (which just identifies that certain tokens/sequences of tokens are names) and the more fine-grained *classifying one* (which assigns to these names ontological labels like locations, organizations, persons etc.). Thus the recognizing layer is the one, whose efficiency is crucial for the consistent linguistic NP processing. The categorizing information at this stage is optional for the syntactic parsing. It will be employed at the further stages when dealing with the semantic restrictions on the arguments of the verbs.

BulName is a module, which relies on language specific rules compiled by taking into account three factors: *capitalization, morpho-syntactic structures* and *keywords*. Both - the typical internal

structure of NE-phrases as well as the regular local contexts the NEs occur in - are described by morpho-syntactic patterns.

We also rely on keywords or identifiers, following the ideas of (McDonald 1996) for internal and external evidence. Identifiers are common words, which are part of the internal structure of NE-phrases (internal identifiers) and they either precede or follow a given NE (external identifiers). For instance, the noun *bank* is an internal identifier for the name *Balgarska narodna banka* (Bulgarian National Bank). However, the same noun is an external identifier for one-word named entity *Biohim* in the phrase *banka Biohim* (bank Biohim).

The identifiers are listed and classified into semantic classes. The grouping goes into four types: (1) identifiers for names of persons, (2) identifiers for locations, (3) identifiers for organizations and (4) miscellaneous identifiers. For example, the type "identifiers for persons" includes nouns from semantic classes *professions, titles, military titles, relative relations, ranks* etc.

While some of the rules are based on the capitalization only, other ones are compiled by considering all three factors mentioned above. The patterns including identifiers simultaneously recognize and categorize the NEs. For instance, the mentioned above identifier *bank*, in combination with a morphological pattern, recognizes the string *Bulgarian National Bank* as *NE* and assigns to it a tag for *organization NE* (OrgNE).

The application of BulName involves two stages. (1) first the *rule-based grammar* is applied, because it recognizes "sure-fire" NEs (according to the terminology of A. Mikheev), with a high precision score obtained with their identification; (2) the *partial matching* is carried out at the second stage. Then unclassified NEs are matched against already classified ones.

3.1 The ideology of the NER grammar module

The rule-based module of BulName consists of three sub-modules, designed for recognition and categorization: (1) NEs, which are sequences of two or more capitalized words, for ex. *Ivan Ivanov, Elton John, East Europe.* They aim at recognizing the cases, which lack in the lists; (2) of NE-phrases such as *Demokratichna partija* (Democratic Party), *Balgarska narodna banka* (Bulgarian National Bank), *Dunavska nizina* (The Danube lowland); (3) of one-word NEs.

The three sub-modules are not applied arbitrarily, but in a fixed order. For instance, the set of rules, finding *one-word NEs* follows the rules for *NE-phrases*.

Processing of NEs, which are sequences of capitalized words

They are recognized and classified by rules based both - on capitalization and on POS characteristics. See, for example, the following rule:

<$CYRwc, ("N#"|"unknown")>,<$CYRwc, ("N#"|"unknown")>+

The rule identifies and categorizes the string as <np type="PersNE">. The string refers to NEs such as *Ivan Ivanov, Elton John*.

Of course, we have in mind the possibility the first word in such strings to be a common noun in a sentence initial position as in the sentences like *Prezidentyt Parvanov kaza…* (The president Parvanov said…) or *Reka Dunav minava prez…* (River Danube passes along…). The common nouns in such strings belong to several semantic classes (titles, ranks – president, minister, some professions – doctor etc., geographical objects – pick, mountain, river etc.). We use such cases for the compilation of the next rule:

<("title"|"rank"|"profession"|"geographical object"), "N#">, <$CYRwc, "N#"|"unknown")+

where the common nouns belong to one of the semantic classes mentioned above. We apply this rule before the previous one in order to pick up names like that, preventing them from being recognized as part of a PersNE.

The following rule:

<(“East”|“West”|”North”|”South”|”Middle”), “A#”>, <$CYRwc, (“N#”|”unknown”)>

handles NEs, which are <np type=”LocNE”>. See for example the named entities *East Europe, South Korea.*

Three features of the processed NEs are taken into account for compiling the above rule: the capitalization of their first element, the NEs morpho-syntactic pattern and the internal identifiers, which are adjectives for geographical direction.

Processing of NEs-phrases

The NE-phrases such as *Agenzija za privatizacija* (Agency for privatization), *Cherno more* (Black sea), *Balgarska akademija na naukite* (Bulgarian Academy of Sciences) are relatively easily recognized because of their predictable internal structure in Bulgarain. One specific feature of Bulgarian, in contrast to English, for example, is the capitalization only of the first word in NE-phrases.

A set of rules is defined for recognition and categorization of NE-phrases. For instance, the following rule:

<$CYRwc,“A#”>,
<(”beach”|”sea”|”village”|”town”|”lowland”|”pick”|“mountain”|”pass”|etc.), “N#”>

captures NEs like *Cherno more* (Black sea), *Slanchev brjag* (Sunny beach), *Cherni vrah* (Black pick), assigning them a tag <np type=”LocNE”>.

Processing of one-word NEs

One-word NEs are relatively easily found when they are in non-initial position in the sentence. All capitalized words in such position are annotated as NEs. There are some exceptions, but they can be listed and treated separately. For example: *Velichestvo* (Majesty) and *Svetejshestvo* (Holiness).

We can further use as an identifying feature the suffixes of Bulgarian family names *-ov, -ova, -ev, -eva, -ski, -ska* etc[3]. These suffixes are employed within the guesser and the general token classification as well, but the NE grammar can serve as a further corrective mechanism. All capitalized nouns, ending in such suffixes, are tagged as <np type=”PersNE”>. This rule overgenerates, so we combine it with a restrictive rule in order to exclude non-named entities, which are capitalized adjectives with possessive meaning derived from family names. They might coincide with family names. Compare, for instance, a PersNE *Ivanova* (Ivanova) and a possessive adjective *Ivanova* (Ivanov’s) in the phrase *Ivanova kola* (Ivanov’s car).

In other cases, we rely on prompting right and left contexts. The crucial for the categorization of one-word NEs is the role of identifiers. NEs, which follow an identifier from semantic classes *title, profession, rank, relative name* etc. belong to the PersNE category. The same tag is assigned to the NEs, preceding verbs of perception, mental verbs, verbs of communication or verbs like *to be born, study, teach, work (at), to marry* etc., which may take only PersNE as a subject.

As to one-word NEs in sentence initial position, after the application of *general token classification*, for their recognition we additionally rely on the local document-based approach. The lexicon and the lists are checked for local-based predictions of the tokens. For example, when the common nouns in such position are excluded, the rest of them can be regarded as potential NEs. Of course, as the BulTreeBank lexicon does not comprise all the words in the language and as the local-based approach could fail due to context insufficiency, the possibility some of the candidates for NEs to be non-named entities is not absolutely excluded.

In order to categorize the recognized NEs in the sentence initial position, we rely on a set of reliable rules. Such as the following rule:

[3] Note that in our tagset these family names have one and the same tag with the possessive adjectives of proper name origin. This underspecification helps us to handle the ambiguities at the pre-processing stage.

Item Area: <$CYRwc,(“N#”|”unknown”)>

Left Context Area: <”,”>,<$CYRws,“Ps#”>?,<$CYRws,“A#”>?, <”sister”|”brother”|”mother” |”father”|”director”|”chief”|etc.), “N#”)>

where the lowercased noun is an identifier from the semantic classes *family relations, ranks* etc. The rule categorizes the first noun in the string as <np type=”PersNE”>. See, for example, the phrase *Amelia, my pretty sister*, where *Amelia* is classified as a *name of person*.

Partial matching stage of BulName module

The NEs, left uncategorized by the rule-based grammar, are processed further at the second stage of the processing. The partial matching of classified NEs with unclassified ones is carried out following the ideas of (Mikheev et al. 1998). The partial matching serves to categorize already recognized NEs (usually one-word NEs), which are shortened forms of names. Alternatively it can include variables or be used without identifiers. For example, the named entity *Blair* is tagged as <np type=”PersNE”> because it partially matches already categorized named entity *Tony Blair*. In the same way all occurences of the named entity *Biohim* in a text are tagged as <np type=”LocNE”> by analogy with already classified NE, used with the identifier *bank*.

3.3 Evaluation

BulName module is still under development and for that reason it has been tested over relatively small number of documents. It was tested over manually tagged texts, comprising about 50 000 words. The texts are from Bulgarian newspapers.

For the sake of accuracy and convenience we calculate recall and precision measures separately for the two layers of the system – the *recognizing layer* and the *categorizing one*. This division was imposed by two reasons. Firstly, there are cases of correctly recognized NEs but wrongly categorized. Secondly, our system processes some one-word NE partially, only recognizing them and this peculiarity has to be taken in mind in the evaluation stage. Note that here we do not present any evaluation on the accuracy of the grammatical information assignment. The metrics are based on detection and ontological categorization so far. As our near future task we plan to extend the evaluation procedure further with the grammatical information included. It is necessary because the NE part of the nominal system and their grammatical interpretation on word and phrasal level is crucial for deeper parsing.

The highest are the precision and the recall scores achieved in identifying and classifying NEs, which comprise two or more capital words. We obtained 94,4 % recall score and 97,5 % precision score in their recognition and 94,4 % recall score and 94,6 % precision score in their categorization (See Table 2). The recognition recall score and the categorizasion recall score are equal because all recognized words are categorized.

For the NE-phrase we achieve 87,6 % recall score and 100 % precision score in their recognition and 87,6 % recall score and 100 % precision score in their categorization.

The scores for one-word NEs are as follows: 94,3 % recall score and 92,1 % precision score in their recognition and 71,7 % recall score and 89,2 % precision score in their categorization.

As it was seen above, we obtained a relatively high recall score in recognition of such type NEs (94,3 %), but lower recall score (71,7) in their categorization. We failed to categorize them in the cases, when both - an identifier and a predictable local context lack and the NEs are mentioned just once in the text. Obviously the context-based rules approach, even combined with partial matching approach, is insufficient for that problem to be resolved. To find a technique for increasing the categorization recall score of one-word NEs is the main problem we have to handle. But for our present purposes we give a priority to the efficient recognition stage. The classifying one is complementary.

Table 2

Types NEs	Recognition Recall Score	Recognition Precision Score	Categorization Recall Score	Categorization Precision Score
NE,comprising Two or more Capitalized words	94,4 %	97,5 %	94,4 %	94,6 %
NE-phrases	87,6 %	100 %	87,6 %	100 %
One word NEs	94,3 %	92,1 %	71,7 %	89,2 %

The average recall and precision score of the system are respectively 88,3 % and 95,5 %. The f-measure (= 2 *(recall * precision) / (recall + precision)) is 91,76.

3.4 Errors and problematic cases

This section focuses on some difficulties, which the BulName module faces in named-entity recognition.

- Part of the mistakes in NE category tags is due to the ambiguity of NEs. The family NEs may refer to organizations or to be used for naming of locations (streets, cities etc.). This kind of metonymy is regular. If identifier lacks, the assigned ontological NE tag might be incorrect, while the grammatical information remains appropriate.
- In some cases the NER module recognizes only the part of NE. That problem arises when NE is a phrase, comprising a conjunction, e. g. *Institut po kriminalistika i kriminalogija* (Institute on Criminal Law and Criminalogy). BulName identifies as NE only the part of phrase before the conjunct – *Institut po kriminalistika* (Institute on Criminal Law). A possible repairing for the most common cases would be matching with the acronyms, whose extentions the names are.
- Some omissions in NE recognition are due to word ambiguities. For example, there are several Bulgarian first names, which are ambiguous with a common word - adjectives, such as *Vesela, Rumen*. Here we rely on the general token classification, which would tell us that such words belong to two lists: of common words and proper names. Then their usage in ambiguous contexts is either resolved by local techniques or context-sensitive NE rules, either remain ambiguous.
- Another part of the omissions in the categorization of one-word NEs is caused by an incompleteness of our lists with identifiers. In such cases NER system cannot categorize the NE, although it is used with an identifier because the identifier lacks in our lists. In testing BulName over new texts we are continuously enriching the lists with new identifiers, improving in such a way the rule-based module of the system.

4 The Bulgarian NP general chunk grammar[4]

In this section we outline the basic assumptions which underline the general NP chunker module.

Basicly, it relies on 1) information, based on local combinations of relevant grammatical features (concerning, for example, agreement, definiteness) and 2) the support of the context, when problematic cases are to be solved. The chunks are defined as 'the non-recursive core of an intra-clausal constituent to its head, but not including post-head dependents' (Abney 1996). The

[4] For a more detailed description of this module and for more problematic discussion see (Osenova 2002).

Bulgarian NP general chunk grammar keeps the idea of non-recursivity, but at the same time it includes rules, which handle structures with post-head dependents (see below). The strategy relies on three heuristics: *islands of certainty, easy-first parsing* and *prefer syntax to semantics*.

The strategy applied takes into account the presence of *clear indicators*, pointing to the unambiguous beginning of an NP. For example, the adverb and the adjectivally used participles without definite article are not clear indicators for the beginning of an NP. Hence, here we rely on the presence of a preposition as right context in order to detect such cases 100 %. For example, the following rule recognizes two- or more-componential NPs beginning with an adverb/adverbs:

Right Context Area: <"R">
Item Area:
<"D">*,
<("Ps#sml#"|"Psx#sml#"|"Pd#sm"|"Pi#sm"|"Prp#sm"|"Pf#sm"|"Pf#sm#"|"Pn#sm"|"Pc#sm"|
"Pc#sm#"|"M@ms#"|"Ams@"|"V#car@sm#"|"V#cv#sm#"|"V#cao@sm#")>?,
<("Psx#t"|"Ps#t")>?,<"D">?,<"V#c#s#">?,<"Ams@">*,<"M#ms#">?,<"N@msi">
Return mark-up area: <np>\w</np>

Here the general morpho-syntactic tags are as follows: **R** stands for a preposition, **D** for an adverb, **P** for pronouns of different kinds, **A** for adjectives, **M** for numerals, **V** for participles and **N** for the head noun. Additionally the rule requires that all the flectional words within the string agree in number (singular) and gender (masculine).

An unambiguous NP for the chunker can be of two kinds: (1) if *mono-componential*, it includes the *nouns, pronouns* and *nominalized parts of speech, only when assigned a noun category* and (2) if *multi-componential*, it can begin with the following words: *an adjective, a numeral, a pronoun* or a *participle with a definite article in adjectival use*.

As the BulName modules, described above, capture the specific types of NPs, the task of the general NP chunker is to handle:

1. the common nouns: the chunk grammar operates on the content of the morpho-syntactic tags. Therefore, its accuracy depends on the tagger. Here is an example of a rule, which matches vocative phrases with postpositive modifiers like: *Boje moj* ('God-my'=My God):

   <"N#v">,<("Ps#li"|"Ps#li#"|"A#")>

2. cases, in which the proper names are modified, but the modifiers are outside their scope: the chunk grammar operates additionally on the BulName module output NPs. Here is an example of the rule, which recognizes pre-modified NP phrases of different named-entity types, such as *[hubavata [Maria]]* ('pretty-the Maria'=the pretty Maria) or *[vashata [Demokratichna partija]]* ('your-the Democratic party'=your Democratic party).

   <("Ps#l#"|"Psx#l#"|"Pd#"|"Pi#"|"Prp#"|"Pf#"|"Pf#"|"Pn#"|"Pc#"|"M#"|"A#"|"V#car#"|"V#cv#"|"
   V#cao#")>,<("Psx#t"|"Ps#t")>?,<",">?,<"C">?,<"D">?,<"V#c#">?,<"A#">*,<"M#">?,<"NEorg"|"NEpers"|"NEloc">

The rule could operate on both - the relevant attributes of the recognized NPs, i.e. names of organizations, persons, locations or the assigned grammatical characteristics.

However, our strategy for detecting only clearly indicated NPs is preserved. Thus we would have the expected omissions in attachment-ambiguous contexts. Compare, for instance, the following two sentences.

(1) *Postignahme **dobre** [np organizirano [np-org Ministerstvo na obrazovanieto]]*

'Achieved-we **well** organized Ministry of education'

We achieved a well-organized Ministry of education.

(2) *Postignahme [np edno **dobre** organizirano [np-org Ministerstvo na obrazovanieto]]*

'Achieved-we one **well** organized Ministry of education'

We achieved a well-organized Ministry of Education.

In the first one the NP starts with an adverb, which is not a clear indicator of an NP, and thus the NP is partially captured, while in the second one the adverb is within the scope of a non-ambiguous indicator, therefore the NP is captured with all its modifying elements.

Note that if the NP with an unclear starting indicator comes after a preposition, the whole NP is unproblematically captured.

One peculiarity when operating over the output NP tags of the BulName grammar is the fact that not always we can rely on agreement features between the modifier and the NE element, especially in cases with names of foreign origin, in which the grammatical gender is underspecified. For example: *Dobrata Smith ('nice-the-fem Smith')*

The pre-modified NEs seem to be not very frequent. In newspaper texts, containing about 50 000 words, we met only 6 occurences. The test showed that most of them are organizations. But nevertheless, such constructions might be productive. It is necessary to test them on cross-genre and cross-newspaper data in order to get more precise picture of their real distribution.

5 The points of merging between BulName module and General nominal chunker

As it was shown above, the two nominal grammars possess somewhat different ideologies and criteria for NPs identification. But sometimes they are forced to 'borrow' from each other differenet approaches, which are typical for the 'other' grammar. This usually happens, when one starts to think about the global efficiency of an NP shallow parser and how to combine the power of different tools. We list only two points of merging here, because they are aimed at solving some complex or peculiar NPs, which otherwise should be treated (problematically however!) at deeper stages of analyses.

5.1 The strategies

- The strategy of preparing lists for assigning a semantic class of nouns, which is the usual practice for the NE grammar, becomes suitable for handling some recursive common nouns structures (NN) at the subchunk level such as *chasha voda* ('glass water'=glass of water).
- Some pure syntactic contexts, together with identifiers, are considered disambiguating for the names identification, especially in the sentence initial position. In the example below the predicative use of the kinship identifier *majka* (mother) recognizes *Roza* (Rose) as a personal name, not as a common noun (roza=rose).

  Roza e moyata majka

  'Roza is my mother'

5.2 The relational domain

At the chunk level in (Abney 1996) the NEs are considered compounds, i.e. forming one NP chunk. In our opinion, however, at the level of chunking the names play different roles, depending on their specificity and complexity (Zhou et al. 2000). We discuss them with respect to two criteria: (1) what output the general chunker operates on and (2) the relation between the names elements and NP chunk frame: *inclusion, equation* or *externality*.

Hence, in a text, where the NEs are recognized, several chunk operations can be considered to have been applied or have to be applied depending on the division of NEs into phrases and words:

- Equation chunk - the NE coincides with an NP chunk

  If a NE is a phrase, then the identifier (if any) is included:

  [Narodna *banka*] (National bank)

If a NE is a word, then the identifier (if any) forms another chunk:

[mojat *brat*] [Ivan] (my brother John)

Here the specific NPs are captured by the NE grammar and the general chunker operates over the non-named area only.

- Inclusive chunking (optional) - the NE comprises internally several chunks, because it is recursive:

[np *Ministerstvo*] *na* [np *obrazovanieto*] (Ministry of education)

Here the NP chunk grammar can operate within the maximal NP phrase. In spite of the fact, that this operation is not needed during the parsing stage, it might be useful for the HPSG-based analyses, because, when separating the NP phrase into non-overlapping nouns, it automatically gives information about modification relations within the subchunks.

- External chunking

- the NE is a part of a non-recursive NP chunk:

[*hubavata* [*Maria*] ('pretty-the Maria')

Here the chunker operates on the output of the other modules and possibly on the output of the NE grammar, if the name has remained unrecognized (lacking in gazeteers etc).

- the NE is a part of a recursive NP chunk, when modifiers come before a phrasal NE. Hence an additional grammar rule is stated, which results in a bigger chunk, namely:

<np type="complex">\w</np>. See the phrase *nashijat* [*Institut za balgarski ezik*] (our [Institute for Bulgarian Language]).

In this case the pre-recognized NEs operate as correcting mechanisms over the chunk level, because after POS tagging the above phrase would be chunked in the following way [*nashijat Institut*] *za* [*balgarski ezik*] and the information flow would be destroyed.

The combinatorial application of the two nominal grammars resulted in the following NP chunk hierarchy, encoded as attribute features:

Common NP chunks

They obey strictly the requirement for non-recursivity: *[edin chovek] ot [grada]* ('one man from the town')

Name NP chunks: NEpers, NEloc etc.

They are proper names of a certain kind and thus can be arbitrarily complex: *[Ministerstvo na kulturata]* ('Ministry of Culture').

Complex NP chunks

They combine the principles from the previous two kinds. First the name is identified and then its possible pre-modifiers are captured: *[nasheto [Ministerstvo na kulturata]]* ('our Ministry of Culture').

According to us this idea is applicable to some other recursive cases such as NPs of type NN. We can employ another strategy: If we use the nearest identifiers for not only identifying the names, but forming an NP group, then we could capture the pre-modifiers later. So we propose as a repairing mechanism this analysis: *[dobrata [lelja Penka]]*('good-the aunt Penka') instead of this one: *[dobrata lelja] [Penka]]*. Thus, in our opinion, combining the possibilities of the two nominal grammars, we can handle successfully some problematic recursive cases at this very stage.

6 Conclusion

In this paper we aimed at showing the advantages of mutual cascaded application of the NER grammar and NPCG for building a robust nominal shallow parser. An adequate language-specific

combination between the named-entity and traditional chunk strategies seems to constitute a good repairing mechanism, which overcomes some of the well-known weaknessess of the two separate grammars. This nominal shallow parser can serve as a reliable input for the HPSG grammar in more detailed parsing of the data. One necessary direction of further experiments is combining the two nominal grammars with the VP chunker in a cascaded manner.

Acknowledgements

The authors would like to thank Kiril Simov and the two anonymous reviewers for the valuable comments on the earlier drafts of this paper. Needless to say, all errors and misconcepts remain our own.

References

Abney, S. 1996: *Chunk Stylebook*. On http://sfs.nphil.uni-tuebingen.de/~abney/Papers.html, draft.

Baluja Sh., Mittal V., Sukthankar R. 2000: *Applying Machine Learning for High-Performance Named-entity Extraction.* In: Computational Intelligence 16 (4), pp. 586-596.

Black W., Rinaldi F. and Mowatt D. 1998*: FACILE: Description of the NE system used for MUC-7.* MUC-7. Fairfax,Virginia.

Borthwick A., Sterling J., Agichtein E., Grishman R. 1998: *NYU: Description of the MENE Named Entity System as Used in MUC 7.* MUC-7. Fairfax,Virginia.

Buchholz, S. and A. van den Bosch 2000: *Integrating seed names and ngrams for a named entity list and classifier. In the Proceedings of Second International Conference on Language Resources and Evaluation.* LREC-2000, Athens, Greece, pp. 1215-1221.

Dalianis H., Astrom E. 2001: *SweNam – A Swedish Named Entity recognizer*. *Its construction, training and evaluation.* http://www.nada.kth.se/~hercules/papers

Dipper, S. 2000: *Grammar-based Corpus Annotation*. In Proceedings of the Workshop on Linguistically Interpreted Corpora (LINC-2000), Luxembourg, August 6, 2000, pp.56-64.

Farmakioutou D., Karkaletsis V., Koutsias J., Sigletos G., Spyropoulos D., Stamatopoulos P. 2000: *Rule-based named entity recognition for Greek financial texts*. In Proceedings of the Workshop on Computational Lexicography and Multimedia Dictionaries (COMLEX 2000), Patras, Greece, pp. 75-78.

Grover C., McDonald S, Gearailt D. N., Karkaletsis V., Farmakiotou D., Samaritakis G., Petasis G., Pazienza M. T., Vindigni M., Vichot F., Wolinski F.: *Multilingual XML-Based Named Entity Recognition for E-Retail Domains.* http: // www.ltg.ed.ac.uk/papers/02crossmark.pdf

Humphreys K., Gaizauskas R., Azzam S., Huyck C., Mitchell B., Cunningham H., Wilks Y. 1998: University of Sheffield: *Description of the LaSIE-II System as Used for MUC-7*. In Proceedings of the 7th Message Understanding Conference.

Ivanova K., Dojkoff D. 2002: *Cascaded regular grammars and constraints over morphologically annotated data for ambiguity resolution.* In: Proceedings of the Workshop on Linguistic Theories and Treebanks, 20-21 Sept., Sozopol, Bulgaria (to appear) (in this volume).

Maynard D., Tablan V., Ursu K., Cunningham H., Wilks Y.: *Named Entity Recognition from Diverse Text Types.* http://www.gate.ac.uk/sale/ranlp2001/maynard-etal.pdf

McDonald David D. 1996: *Internal and external evidence in the identification and semantic categorization of proper names*. In Corpus Processing for Lexical Acquisition (ed. by Bran Boguraev and James Pustejovsky), chapter 2, pp. 21-39. The MIT Press, Cambridge, MA.

Mikheev A., Grover C. and Moens M. 1998: *Descripton of the LTG system used for MUC-7.* In Seventh Message Understanding Conference (MUC-7): Proceedings of a Conference held in Fairfax, Virginis.

Mikheev A., Moens M. and Grover C. 1999: *Named Entity Recognition without Gazetteers*. In EACL'99, Bergen, Norway ACL June 1999, pp. 1-8

Osenova P. 2002: *Bulgarian Nominal Chunks and Mapping Strategies for Deeper Syntactic Analyses.* In: Proceedings of the Workshop on Linguistic Theories and Treebanks, 20-21 Sept., Sozopol, Bulgaria (to appear) (in this volume).

Osenova P. and Simov K. 2002: *Learning a token classification from a large corpus (A case study in abbreviations).* In Proceedings from the ESSLLI Workshop on Machine Learning Approaches in Computational Linguistics, Trento, Italy. August 5-16, 2002, pp. 16-28.

Popov D., Simov K., Vidinska S. 1998: *Bulgarian Inflectional Morphology Dictionary. "*Atlantis", Sofia.

Simov K., Popova G. and 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., Kouylekov M., Simov A. 2002: *Cascaded Regular Grammars over XML Documents*.

In: Proc. of the 2nd Workshop on NLP and XML (NLPXML-2002), COLING2002, Taipei, Taiwan. September 1, 2002. (to appear).

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 2002, Canary Islands, pp. 1729-1736.

Soderland S. 1997. *Learning to extract text-based information from the world wide web.* In Proceedings of 3td International Conference in Knowledge Discovery and Data Mining (KDD-97), pp. 251-254.

Stevenson M. and Gaizaukas R. 2000: *Using Corpus-derived Name Lists for Naned Entity Recognition.* Proceedings of ANLP-NAACL 6th Applied Natural Language Processing Conference and 1st Meeting of the North American Chapter of the Association for Computational Linguistics. Seattle, pp. 250-295.

Yangarber R. and R. Grishman. 2000. *Machine learning of extraction patterns from un-annotated corpora.* In Proceeding of the Workshop on Machine Learning for Information Extraction, 14th European Conference on Artificial Intelligence (ECAI 2000).

Zhou G., Su J. 2000: *Named Entity Recognition using an HMM-based Chunk Tagger*.
http://www.citeseer.nj.nec.com./zhou02.named.html