Towards Minimal Recursion Semantics over Bulgarian Dependency Parsing

Kiril Simov
LMD, IICT-BAS
kivs@bultreebank.org

Petya Osenova
LMD, IICT-BAS
petya@bultreebank.org

Abstract

The paper discusses the transferring rules of the output from a dependency parser for Bulgarian into RMRS analyses. This task is required by the machine translation compatibility between Bulgarian and English resources. Since the Bulgarian HPSG grammar is still being developed, a repairing mechanism has been envisaged by parsing the Bulgarian data with the Malt Dependency Parser, and then retrieving RMRS analyses by exploring the linguistic knowledge within BulTreeBank-DP.

1 Introduction

Recently a number of machine translation efforts have focused on grammatical formalisms in performing source language analysis, transfer rule application and target language generation. It is worth mentioning several works, such as (Bond, 2005) exploiting DELPH-IN infrastructure for developing of HPSG grammars, (Riezler and Maxwell III, 2006) using LFG grammar, (Oepen et al, 2007) working on a hybrid architecture consisting of an LFG grammar, an HPSG grammar, partial parsing, and (Bojar and Hajic, 2008) using the Functional Generative Description framework to language analysis on analytical and tectogrammatical level. All the approaches rely on the advances in the development of deep grammar natural language parsing. The approaches share similar architecture and techniques to overcome the drawbacks of the deep processing in comparison to statistical shallow methods.

Within the above mentioned context, we are constructing a Bulgarian-to-English translation system, based on HPSG. The transfer rules are implemented on the level of MRS (Minimal Recursion Semantic) structures (Copestake et al, 2005). The HPSG deep grammar for Bulgarian still has a limited coverage. Thus, for many input sentences it will fail to produce MRS analyses. In such cases, we rely on a dependency parser (Malt parser trained on the BulTreeBank data) to produce a dependency parse for the sentence. Then, we construct an RMRS (Robust Minimal Recursion Semantic) analysis over the dependency parse. Thus our input processing architecture consists of two grammars – HPSG grammar which produces MRS structures, and Dependency grammar which produces RMRS structures. The resulting semantic analysis is the input for the transfer module of the machine translation system. The paper focuses on the dependency tagset and the rules for the construction of RMRS analyses over the dependency parses. It is structured as follows: First, the context of our work is presented. Then our dependency tagset is discussed. In the following section the HPSG-based Bulgarian grammar BURGER is briefly outlined. Finally, the basic rules for the construction of the RMRS analyses from dependency parses are described.

2 Background

Our approach is inspired by the work on MRS and RMRS (see (Copestake, 2003; 2007)) and the previous work on transfer of dependency analyses into RMRS structures described in (Spreyer and Frank, 2005) and (Jakob et al, 2010).

MRS is introduced as an underspecified semantic formalism (Copestake et al, 2005). It is used to support semantic analyses in HPSG English grammar – ERG (Copestake and Flickinger, 2000), but also in other grammar formalisms like LFG. The main idea is the formalism to rule out spurious analyses resulting from the representa-

Proceedings of Recent Advances in Natural Language Processing, pages 471–478,
Hissar, Bulgaria, 12-14 September 2011.

tion of logical operators and the scope of quantifiers. Here we will present only basic definitions from (Copestake et al, 2005). For more details the cited publication should be consulted. An MRS structure is a tuple <*GT*, *R*, *C*>, where *GT* is the top handle, R is a bag of EPs (elementary predicates) and *C* is a bag of handle constraints, such that there is no handle *h* that outscopes *GT*. Each elementary predication contains exactly four components: (1) a handle which is the label of the EP; (2) a relation; (3) a list of zero or more ordinary variable arguments of the relation; and (4) a list of zero or more handles corresponding to scopal arguments of the relation (i.e., holes). Here is an example of an MRS structure for the sentence "*Every dog chases some white cat.*"

<h0, {h1: every(x,h2,h3), h2: dog(x), h4: chase(x, y), h5: some(y,h6,h7), h6: white(y), h6: cat(y)}, {}>

The top handle is *h0*. The two quantifiers are represented as relations *every(x, y, z)* and *some(x, y, z)* where *x* is the bound variable, *y* and *z* are handles determining the restriction and the body of the quantifier. The conjunction of two or more relations is represented by sharing the same handle (*h6* above). The outscope relation is defined as a transitive closure of the immediate outscope relation between two elementary predications – EP immediately outscopes EP' iff one of the scopal arguments of EP is the label of EP'. In this example the set of handle constraints is empty, which means that the representation is underspecified with respect to the scope of both quantifiers. Here we finish with the brief introduction of the MRS formalism. The rest of the definitions will be introduced when necessary in the text.

RMRS is introduced as a modification of MRS which to capture the semantics resulting from the shallow analysis. Here the following assumptions are taken into account – the shallow processor does not have access to a lexicon. Thus it does not have access to arity of the relations in EPs. Therefore, the representation has to be underspecified with respect to the number of arguments of the relations. Additionally, the forming of the relation names follows such conventions that provide possibilities to construct a correct semantic representation only on the base of information provided by a POS tagger, for example. The arguments are introduced separately by argument relations between the label of a relation and the argument. The names of the argument relations follow some standardized convention like RSTR, BODY, ARG1, ARG2, etc. These argument relations are grouped in a separate set in a given RMRS structure. Both representations MRS and RMRS could be transferred to each other under certain conditions. In the paper we follow the representation of RMRS used in (Jakob et al, 2010), which defines an RMRS structure as a quadruple < *hook*, *EP-bag*, *argument set*, *handle constraints* >, where a hook consists of three elements *l*:*a*:*i*, *l* is a label, *a* is an anchor and *i* is an index. Each elementary predication is additionally marked with an anchor – *l*:*a*:*r(i)*, where *l* is a label, *a* is an anchor and *r(i)* is a relation with one argument of appropriate kind – referential index or event index. The argument set contains argument statements of the following kind *a*:*ARG*(*x*), where *a* is anchor which determines for which relation the argument is defined, *ARG* is the name of the argument, and *x* is an index or a hole variable or handle (*h*) for scopal predicates. The handle constraints are of the form *h =q l,* where h is a handle, l is a label and *=q* is the relation expressing the constraint similarly to MRS. *=q* sometimes is written as *qeq*.

RMRS was used in analyses of two dependency treebanks – TIGER treebank of German and Prague Dependency Treebank of Czech. The work on Prague Dependency Treebank presented in (Jakob et al, 2010) first assigns elementary predications to each node in the tectogrammatical tree. Then the elementary predications for the nodes are combined on the basis of the dependency annotation in the trees. Similar approach is taken by us, except that the analyses from which we start are not trees on tectogrammatical level. Thus, our trees contain nodes for each token in the sentences.

3 Bulgarian Dependency Parsing

Many parsers have been trained on data from BulTreeBank. Especially successful was the MaltParser of Joakim Nivre (Nivre et al., 2006). It works with 87.6 % accuracy. The following text describes the dependency relations produced by the parser.

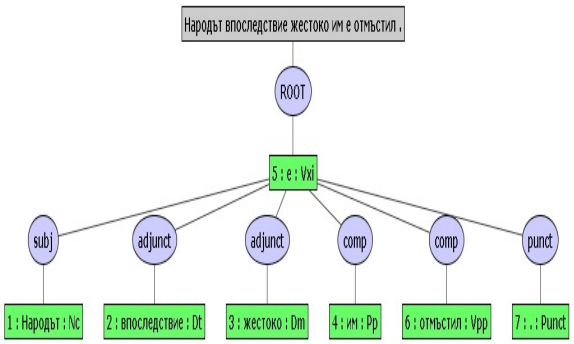
Here is a table with the dependency tagset related to the Dependency part of the BulTreeBank. This part has been used for training of the dependency parser:

adjunct 12009	Adjunct (optional verbal argument)
clitic 2263	Short forms of the possessive pronouns
comp	Complement (arguments of non-verbal

18043	heads, non-finite verbal heads, copula, auxiliaries)
conj 6342	Conjunction in coordination
conjarg 7005	Argument (second, third, ...) of coordination
indobj 4232	Indirect Object (indirect argument of a non-auxiliary verbal head)
marked 2650	Marked (clauses, introduced by a subordinator)
mod 42706	Modifier (dependants which modify nouns, adjectives, adverbs; also the negative and interrogative particles)
obj 7248	Object (direct argument of a non-auxiliary verbal head)
subj 14064	Subject
pragadjunct 1612	Pragmatic adjunct
punct 28134	Punctuation
xadjunct 1826	Clausal adjunct
xcomp 4651	Clausal complement
xmod 2219	Clausal modifier
xprepcomp 168	Clausal complement of preposition
xsubj 504	Clausal subject

In addition to the dependency tags we have also morphosyntactic tags attached to each word (Simov et. al, 2004). For each lexical node the lemma is assigned. The number under the name of each relation indicates how many times the relation appears in the dependency version of BulTreeBank. We have also statistics for the triples <DependentWordForm, Relation, HeadWordForm>. It is used for defining the rules for constructing RMRS structures over the dependency parses produced by the Malt parser.

The dependency relations here reflect the original HPSG analyses in BulTreeBank and are conformant to the dependency relations schema of the CoNLL shared task (2006). Thus, some of them are more specific (such as, *obj*, *indobj*, *clitic*, *subj*, etc.), while others are more general (such as, *comp* and *mod*). Since the reflexive accusative and dative clitics are always marked as *comp*, a dictionary check is needed to determine whether these clitics are part of the lexeme, or they mark a voice alternation. Also, when there is an auxiliary verb, it becomes the root of the sentence, and since the main verb as well as the personal clitic are both marked with the same relation (comp), a check with the morphosyntactic information is needed. Here is an example for the sentence 'The peoples afterwards have revenged them mercilessly':



The missing information in comparison with the HPSG-based version is the constituent structure, the coreferential relations and ellipses. As it can be seen from the above description, some of the relations in the dependency tagset are very general (the *comp* relation, for example). More specific information could be inferred on the basis of the morphosyntactic information of the two lexical nodes and the dependency relation between them. This allows us to write rules for constructing RMRS for different configurations in the dependency trees (see section 5). In the above example, the participle (node 6) determines the relation of the main verb and the dative clitic (node 4) determines the plurality and the third person of the indirect object. More on the specificities of this schema in comparison with another dependency schema for Bulgarian is discussed in (Kancheva 2010).

4 The BURGER Grammar and the MRS Analysis

BURGER is the realization of the Matrix Grammar (Bender et. al 2002) to Bulgarian language. It is implemented in LKB - Linguistic Knowledge Builder (Bender et. al 2010). Its first version is made available at the DELPH-IN Consortium site, and it is described in (Osenova 2010).

The work on the grammar included several tasks: a lexicon building for Bulgarian within the required format; adapting of the type hierarchy to the Bulgarian grammar model; addition of language specific principles for Bulgarian; preparing of a test-suite with sentences, which to illustrate the main linguistic phenomena in Bulgarian

and to demonstrate the capacity of the grammar (including also negative examples on the basis of the Bulgarian BulTreeBank Corpus).

Here is an example of the lexical entry for the verb *чета* 'read':

cheta := v_np_i1_le &
[**STEM** < "чета" >,
SYNSEM.LKEYS.KEYREL.PRED
"чета_v_rel"].

As it can be seen, the lexical entry uses the Latin transliteration of the Bulgarian word. The Cyrillic sequence is presented in the feature **STEM**. The mnemonic name **v_np_i_le** means that this word is a verb, which takes an NP as its complement, and it is in an imperfective aspect. The ending **le** has only technical functions.

In general, the Grammar Matrix provides a semantic approach to the description of a language. For example, the verbs, the adjectives, the adverbs and the prepositions are viewed as introducing events. Bulgarian, however, lacks a grammar, which describes all the phenomena at the semantic-syntactic interface. The existing research is mainly focused on the morphology and syntax only. Concerning Bulgarian, its rich morphology seems to conflict with the requirements behind the semantic approach. Thus, the adjectives, participles, numerals happen to have *morphologically* definite forms, while the definiteness marker is not a *semantic* property of these categories. For that reason, the most important thing in the grammar was to keep Syntactic and Semantic features separate. This distinction concerns, for example, definiteness and tense properties. All the parses are also augmented with the corresponding MRS.

BURGER covers all the syntactic phenomena, presented in the international testset of the Grammar Matrix plus the language specific features, such as clitics behavior, da-construction, pro-dropness, lexical aspect etc (193 sentences).

Here is an example of an MRS for the sentence *не лай* 'do not bark', where the negative particle *не* 'no' is treated as a verb:

[**LTOP**: *h1*
INDEX: *e2* [**E.ASPECT**: *imperfective* **E.MOOD**: *imperative* **SF**: *comm*]
RELS:<
["negation_rel"
LBL: *h1*
ARG0: *e2*
ARG1: *h3*
["лая_v_rel"
LBL: *h4*
ARG0: *e2*
ARG1: **x5** [x **PNG.NUMBER** *singular* **PNG.PERSON** *2nd* SF *comm*]
HCONS <*h3* qeq *h4*>]

Here the main verb 'bark' is represented as an event (the value of ARG0), which takes an unexpressed subject (the value of ARG1) being of underspecified gender, singular, second person. The negative particle is encoded as a verb, which introduces a negation relation. In this relation, ARG0 structure-shares with the ARG0 of the event 'bark', and ARG1 is the scope over the event 'bark'.

5 RMRS for Bulgarian Dependency Parses

In this section we present a set of rules for transfer of dependency parses into RMRS presentations. The information input for the RMRS structures is based on the following linguistic annotation – the lemma (*Lemma*) for the given wordform; the morphosyntactic tag (*MSTag*) of the wordform, and the dependent relations in the dependency tree. In cases of quantifiers we have access to the lexicon used in BURGER. Here we present the rules for some of the most important combinations. The approach of (Jakob et al, 2010) is adopted. Also, we take into account the MRS structures produced by BURGER in order to be able to compare them to RMRS structures produced over the dependency trees. Thus, the algorithm for producing of RMRS from a dependency parse is implemented via two types of rules:

1. <*Lemma*, *MSTag*> → *EP-RMRS*

The rules of this type produce an RMRS including an elementary predicate.

2. <*DRMRS*, *Rel*, *HRMRS*> → *HRMRS'*

The rules of this type unite the RMRS constructed for a dependent node (*DRMRS*) into the current RMRS for a head node (*HRMRS*). The union (*HRMRS'*) is determined by the relation (*Rel*) between the two nodes. In the rest of the section we present examples of these rules.

First, we start with assigning EPs for each lemma in the dependency tree. These EPs are similar to node EPs of (Jakob et al, 2010). Each EP for a given lemma consists of a predicate generated on the basis of the lemma string. When the lemma is a quantifier and thus it is a part of the BURGER lexicon, we copy the related information about its relation and arguments – RESTRICTION (RSTR) and BODY. Additionally, the morphosyntactic features of the wordform are presented. On the basis of the part-of-speech tag the type of ARG0 is determined – referential index or event index. After this initial step the basic RMRS structure for each lemma in the sen-

tence is compiled. Below we discuss the exploitation of the rest of the information in the dependency tree – the types of links to the other lemmas as well as the further contribution of the morphosyntactic features. Here is an example for the verb 'чета' (to read):

< l1:a1:e1,
{ l1:a1:чета_v_rel(e1) },
{ a1:ARG1(x1) },
{} >

In this example we also include information for the unexpressed subject (ARG1) which is always incorporated in the verb form. In case the subject is expressed, it will be connected to the same referential index. For some types of nodes the EP RMRS will include information only for arguments of the predicate of the head node.

The short forms of pronouns (clitics) do not introduce a semantic relation. The semantic relation is introduced only by their full counter-parts. It is rather straightforward transfer, since the short forms are annotated as clitics, while the full forms are assigned grammatical roles – object or indirect object. Thus, the full forms in verbal domain are automatically transferred as ARG2 and ARG3 of the corresponding verb. In this transfer we always connect the object to argument ARG2 slot and indirect object to ARG3 slot. For example, the sentence *чета му я* (Read-I him-dative her-accusative, 'I read it to him') will have the following representation:

< l1:a1:e1,
{ l1:a1:чета_v_rel(e1) },
{ a1:ARG1(x1), a1:ARG2(x2),
a1:ARG3(x3) },
{} >

The EP RMRS for the accusative clitic introduces only the information for ARG2 and appropriate grammatical features for the variable x2 (third person, singular, feminine). Similarly EP RMRS for the dative clitic provides ARG3 and its grammatical features (third person, singular, masculine). When this information is incorporated into the head RMRS, the anchors for the ARG2 and ARG3 are changed with respect to the anchor of the head.

The subject is mapped to ARG1. It is worth noting that the Subject argument is partially determined during the previous step in building EPs, because Bulgarian is a pro-drop language, and the main subject properties are considered part of the verb form. Here is an example for the sentence *момче му я чете* (Boy him-dative her-accusative read, 'A boy reads it to him):

< l2:a4:e1,
{ l1:a1:момче_n_rel(x1),
l2:a4:чета_v_rel(e1) },
{ a4:ARG1(x1), a4:ARG2(x2),
a4:ARG3(x3) },
{} >

Another example with an explicit direct object for the sentence *момче му чете книга* (Boy him-dative reads book, 'A boy reads a book to him'):

< l2:a3:e1,
{ l1:a1:момче_n_rel(x1),
l2:a3:чета_v_rel(e1),
l3:a4:книга_n_rel(x2) },
{ a3:ARG1(x1), a3:ARG2(x2),
a3:ARG3(x3) },
{} >

A problematic case is the passive construction in which the arguments are represented as alternating dependency relations. In this case the lemma is consulted for the semantic presentation, and the indirect object relation is assigned as a PP-relation, which introduces the Subject.

The modifying words (*mod*) – adjectives, adverbs or nouns introduce a modifier relation. When the modifier is definite, then the information is treated only on the syntactic level. Thus, the head is considered semantically definite, and the information is divided between the two levels of analysis.

The complements of the copula need the information from the morphosyntactic tag, since the adjective, adverb and PPs raise their INDEX to the semantically vacuous copula. In contrast to them, the nouns introduce a referential INDEX, which, however, is not raised to the copula.

When an auxiliary verb is recognized, which takes a participle as a complement, and then depending on the participle, the transfer is realized accordingly. For example, if the participle is aorist, then it is in active voice. If it is passive, then the semantics follows the strategy from above.

The transfer of the impersonal verbs into RMRS also relies on the morphosyntactic tags. They introduce a restriction on its subject to be pro-nominal, 3rd person, singular, neuter. The relation *xcomp* is transformed into a constraint, which ensures that the ARG1 of the modal qeqs the label of the verb in the da-construction (analytical substitute form for the Old Bulgarian infinitive).

Here is a simplified representation of the sentence *Трябва да му кажа*. (Must to him-dat tell-I, 'I have to tell him'):

< l1:a1:e1,
{ l1:a1:трябва_v_rel(e1),
  l2:a4:кажа_v_rel(e2) },
{ a1:ARG2(e2),
  a4:ARG1(x1),
  a4:ARG3(x2) },
{} >

The *xmod* relation connects a clause to a nominal head. When the clause is introduced by a relative pronoun, its RMRS is incorporated in the RMRS of the head and the index introduced by the relative pronoun is made the same as the index of the head. In cases when the clause is not introduced by a relative clause the event index of the clause is nominalised and the new referencial index is made the same as the index of the head.

The *xsubj* relation is incorporated in the head RMRS depending on the kind of the dependent clause. If it is a relative clause then the index of the relative pronoun is made equal to the index introduced by the unexpressed subject of the head. In the other cases the event represented by the clause is nominalized and the new referential index is made equal to the index of the unexpressed subject of the head.

The *marked* relation is always connected to a subordinate conjunction. The subordinate conjunction introduces a two argument relation, where both arguments are events. In this case the RMRS of the dependent clause is added to the RMRS assigned to the conjunction. Additionally, the index of the second argument is made equal to the index of the dependent clause.

The *xprepcomp* relation is treated as an ordinary *prepcomp* relation, but the index of ARG1 is an event.

The canonical coordination is handled relatively straightforwardly. The *conj* label introduces a coordination relation, and *conjarg* is mapped to the right index R-INDEX. Then, the left index L-INDEX is taken by the above level, which contains the grammatical role of the whole coordination phrase.

The *pragadjunct* introduces different types of modifiers on pragmatic level like vocatives, parenthetical expressions, etc. For the moment, we incorporate the RMRS of the dependent element in the RMRS of the head without additional constraints, but these cases require more work in future.

The relation *punct* is ignored.

The incorporation of the dependent RMRS into the head RMRS is done recursively from the leaves of the tree up. After the construction of the RMRS of the tree root, we need to add the missing quantifiers for the unbound referential indexes. For each such index the algorithm determines the handle with a widest scope and uses it as a RSTR argument.

Here is a pseudo code of the main algorithm RMRS which selects the root of the input tree and calls the recursive function which calculates the RMRS for the sentence:

```
algorithm rmrs
  Input: DTree (dependency tree in CoNLL format)
    Output: < hook, EP-bag, argument set, handle
      constraints > (RSMS structure for the sentence)
  RootNode ← root(DTree)
  setEnumerators()
  RMRS ← nodeRMRS(DTree, RootNode)
  return addQuantifiers(RMRS)
end_algorithm
```

The function root(*DTree*) selects the root of the tree. The function nodeRMRS(*DTree*, *Node*) constructs recursively RMRS structure for the subtree starting at node *Node*. The subtree is part of the whole tree for the sentence – *DTree*. The function setEnumerators() sets the initial numbers for labels, referential and event variables. For anchors we use the token numbers that are already in the CoNLL format of the dependency tree. The function addQuantifiers(*RMRS*) introduces the missing quintifiers in the final RMRS. Here is the pseudo code for the function:

```
function nodeRMRS(DTree, CurrentNode)
  NodeEP ← nodeEP(DTree, CurrentNode)
  for DNode in depNodes(DTree, CurrentNode)
    DNodeRMRS ← nodeRMRS(DTree, DNode)
    DRel ← nodeRel(DTree, DNode)
    NodeEP ← union(NodeEP, DNodeRMRS, DRel)
  end_for
  return NodeEP
end_function
```

This function first calls the function for constructing RMRS for the elementary predication for the current node in the dependency tree – nodeEP(*DTree*, *CurrentNode*). This function implements the first kind of rules mentioned above. It has access to the lemma and the grammatical information for the current node. The predicate name is constructed on the basis of the lemma and the part of speech (for example, чета_v_rel), the argument type is determined on the basis the grammatical information – event or referential index. Additional information can be added for other arguments of the verbs as it was described above. In case of access to a lexicon, the function will be tuned to the information within the lexicon. This will be relevant for the case of the valency lexicon.

The function depNodes(*DTree*, *CurrentNode*) returns a set of nodes in the tree which are dependent of the current node. For each of them the function nodeRMRS(*DTree*, *Node*) is called. The result of this recursive call is incorporated within the current RMRS on the basis of the dependency relations. This is done by the function union(*NodeEP*, *DNodeRMRS*, *DRel*). This function is defined by the second kind of rules described above. Note that all the relevant information is available in the already constructed RMRS structures for the head node as well as the dependent nodes and the type of the relations.

The rules of the first kind are 118. They correspond to a reduced morphosyntactic tagset of (Simov et al. 2004). The rules of the second kind are 53. The construction of these rules follows the statistics, presented in section 3. We first implemented rules for most frequent combinations. As much as we can not be sure that the treebank contains examples of all possible combinations we implement 'catch all' which just construct the union of the sets within the two RMRSes.

6 Evaluation

We do not have a gold standard corpus of dependency trees with manually constructed RMRS. Thus, we cannot determine a real evaluation of the performance of the proposed algorithm. However, we have a dataset covered by BURGER grammar for which the correct analyses, including MRS, are selected. Therefore, we decided to evaluate the algorithm with respect to this dataset.

First, we annotated the sentences in the dataset as dependency trees. Then, we ran the parser over the sentences and manually corrected the result. Next, we applied the algorithm over the resulting trees and produced the RMRS for each dependency tree. In the same time, we transferred the MRS, constructed by BURGER into RMRS representations.

Comparing the two RMRS structures for the same sentence is done by comparing the information related to each index in the RMRS. Intuitively, the expectation was that the RMRS constructed on the base of the dependency analysis would contain less information than the one produced by BURGER. Intuitively, less information here means that the indexes participate in reduced number of relations; the relations have smaller number of arguments; and also smaller number of handle constraints. Needless to say, the relations, arguments and constraints have to be identical when present in both structures.

The actual comparison was performed by constructing of a mapping from indexes, labels, anchors and handles in one of the RMRS into the indexes, labels and handles in the other one. The mapping has to respect the type of the indexes.

Let RMRS-D be the structure produced from the dependency tree and let RMRS-B be the structure produced by BURGER. If there exists a mapping from RMRS-D into RMRS-B such that:

- for each index **i**, each anchor **a**, each label **l** and each relation **r** such that l:a:r(i) is in RMRS-D then for the corresponding label l', anchor a' and index i' it is true that l':a':r(i') is in RMRS-B;
- for each anchor **a**, each index **i** and argument **ARG** such that a:ARG(i) is in RMRS-D then for the corresponding anchor a' and index i' it is true that a':ARG(i') is in RMRS-B;
- for each handle **h** and each label **l** such that **h =q l** is in RMRS-D then for the corresponding handle h' and label l' it is true that h' =q l' is in RMRS-B; and
- if l':a':i' is the hook of RMRS-B and for at least one of its elements there is a mapping from a corresponding element in RMRS-D, then there are mappings for all elements and original triple l:a:i is the hook of RMRS-D.

then we say that RMRS-D is substructure of RMRS-B. The last condition is very strong and is subject to further refinement. But in our work with this example dataset it has not caused any troubles. In all other cases we say that both structures are incompatible.

On the basis of the dataset covered by BURGER (193 sentences) we achieved 77% of compatibility of RMRSes.

The main sources of incompatibility are: relation names and principles of BURGER. In the case of the relation names, it could happen that in BURGER there are more relation names that share a lemma string. For example, трябва_v_1_rel and трябва_v_2_rel is represented in dependency RMRS as трябва_v_rel. In BURGER there are some cases when the subordinate conjunction is incorporated in the clause RMRS, but in the dependency we do not have such a rule. In the first case the wrong match is acceptable in our opinion as much as we do not have access to a lexicon for most of the lemmas in the sentences. For the second case we have to modify the rules in the algorithm.

7 Conclusion

In this paper we presented an algorithm for transferring of information from dependency parses into RMRS. This information will be used in a Bulgarian-English machine translation system when the HPSG grammar fails to produce an analysis. We hope that the algorithm will produce the right number and types of indexes with appropriate relations between them which to allow the addition of missing information on the basis of statistics over a parallel treebank.

The algorithm needs augmentation with a rich lexicon and a more elaborate treatment of some constructions for the production of appropriate RMRS. These resources are under development.

We have to say that an evaluation with more examples is necessary, because the 193 do not demonstrate all the dependency relations in the dependency tagsets. Another task which is under development is the creation of a gold corpus of manually annotated RMRS structures.

8 Acknowledgments

This work has been supported by the European project EuroMatrixPlus (IST-231720).

References

Bender, Flickinger and Oepen. 2002: E. Bender, D. Flickinger and S. Oepen. *The Grammar Matrix: An Open-Source Starter-Kit for the Rapid Development of Cross-Linguistically Consistent Broad-Coverage Precision Grammars.* Carroll, John, Nelleke Oostdijk, and Richard Sutcliffe, eds. Proced-ings of the Workshop on Grammar Engineering and Evaluation at the 19th International Confer-ence on Computational Linguistics. Taipei, Taiwan. pp. 8-14.

E. Bender, Scott Drellishak, Antske Fokkens, Laurie Poulson and Safiyyah Saleem. 2010. *Grammar Customization*. In Research on Language and Computation, volume 8, issue 1. Springer.

Francis Bond, Stephan Oepen, Melanie Siegel, Ann Copestake and Dan Flickinger. 2005. *Open Source Machine Translation with DELPH-IN*. In: Proceedings of the Open-Source Machine Translation Workshop at the 10th Machine Translation Summit. pp 15-22.

Bojar, Ondrej and Hajic, Jan. 2008. Phrase-based and deep syntactic English-to-Czech statistical machine translation. In: StatMT '08: Proceedings of the Third Workshop on Statistical Machine Translation. pp. 143--146.

Ann Copestake, Dan Flickinger, Carl Pollard, and Ivan A. Sag. 2005. Minimal Recursion Semantics: An Introduction. Research on Language and Computation, 3(4). pp 281–332.

Ann Copestake and Dan Flickinger. 2000. Open source grammar development environment and broad-coverage English grammar using HPSG. In Proceedings of the 2nd International Conference on Language Resources and Evaluation. pp 591–598.

Ann Copestake. 2003. Robust Minimal Recursion Semantics (working paper). http://www.cl.cam.ac.uk/~aac10/papers

Ann Copestake. 2007. *Applying Robust Semantics*. In Proceedings of the 10th Conference of the Pacific Assocation for Computational Linguistics (PACLING), pages 1–12, Melbourne, Australia.

Max Jakob, Lopatková, M., Kordoni, V. 2010. *Mapping between Dependency Structures and Compositional Semantic Representations*. In: Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10), ELRA.

Stanislava Kancheva. 2010. *Representation of the Grammatical Roles for Bulgarian in the Dependency Grammar*. (unpublished Master Thesis). In Bulgarian.

Joakim Nivre, Johan Hall, Jens Nilsson. 2006. *MaltParser: A data-driven parser-generator for dependency parsing*. In Proc. of LREC-2006. pp 2216-2219.

Stephan Oepen, Erik Velldal, Jan Tore Loenning, Paul Meurer, Victoria Ros'en and Dan Flickinger. 2007. *Towards Hybrid Quality-Oriented Machine Translation - On Linguistics and Probabilities in MT.* In: Proc. of the 11th Conference on Theoretical and Methodological Issues in MT.

Petya Osenova. 2010: *The Bulgarian Resource Grammar*. VDM.

Riezler, Stefan and Maxwell III, John T. 2006. *Grammatical machine translation*. In: HLT-NAACL. pp 248-255.

Kiril Simov, Petya Osenova and Milena Slavcheva. 2004. *BTB-TR03: BulTreeBank Morphosyntactic Tagset.* BulTreeBank Technical Report № 03.

Kathrin Spreyer and Anette Frank. 2005. *Projecting RMRS from TIGER Dependencies.* In Stefan Müller, editor, Proceedings of the 12th HPSG Conference, pages 354–363, Lisbon, Portugal.