RRGbank: a Role and Reference Grammar Corpus of Syntactic Structures Extracted from the Penn Treebank

Tatiana Bladier[1], *Andreas van Cranenburgh*[2], *Kilian Evang*[1], *Laura Kallmeyer*[1], *Robin Möllemann*[1], *Rainer Osswald*[1]

(1) University of Düsseldorf, Germany
(2) University of Groningen, the Netherlands

{bladier,evang,kallmeyer,moellemann,osswald}@phil.hhu.de,
a.w.van.cranenburgh@rug.nl

ABSTRACT

This paper presents RRGbank, a corpus of syntactic trees from the Penn Treebank automatically converted to syntactic structures following Role and Reference Grammar (RRG). RRGbank is the first large linguistic resource in the RRG community and can be used in data-driven and data-oriented downstream linguistic applications. We show challenges encountered while converting PTB trees to RRG structures, introduce our annotation tool, and evaluate the automatic conversion process.

KEYWORDS: Role and Reference Grammar, RRG, treebank conversion, Penn Treebank.

1 Introduction

Wide empirical coverage is a touchstone for every grammatical theory. Treebanks have been widely used as training material for data-driven parsing approaches, data-oriented language processing, statistical linguistic studies, or machine learning throughout the last decades. However, no large linguistic resource exists for the framework of Role and Reference Grammar (RRG; Van Valin and LaPolla, 1997; Van Valin, 2005) so far. In this paper we describe the development of the first annotated corpus of RRG structures[1] created through (semi-)automatic conversion of the Penn Treebank.

Providing a treebank resource to the RRG community will be useful for several reasons: (i) it will be a valuable resource for corpus-based investigations in the context of linguistic modeling using RRG and in the context of formalizing RRG, which is needed for a precise understanding of the theory and for using it in NLP contexts. Efforts towards a formalization of RRG as a tree-rewriting grammar have already been made recently (Kallmeyer et al., 2013; Kallmeyer, 2016; Kallmeyer and Osswald, 2017). (ii) In the context of implementing precision grammars, at least for English, an RRG treebank is useful for testing the grammar and evaluating its coverage. (iii) It will enable supervised data-driven approaches to RRG parsing (grammar induction and probabilistic parsing). (iv) Finally, and more immediately, the specification of the treebank transformation yields valuable new insights into RRG analyses of English syntax — since, even though RRG has covered a large range of typologically different languages, compared to other theories, English has not been considered much.

Since manual annotation is very time-consuming, we decided to (semi-)automatically derive RRGbank from an existing treebank. For this, we chose the Penn Treebank (PTB; Marcus et al., 1993) because of its large size and and availability of additional layers such as OntoNotes (Hovy et al., 2006) which may be used to enrich RRGbank in the future. The PTB has been used in the past, among others, for deriving CCGbank, a corpus of Combinatory Categorial Grammar derivations (Hockenmaier and Steedman, 2007). We decided to start from the original PTB rather than CCGbank because its phrase structure trees are more similar to RRG than CCG derivations, and to avoid possible compounding of errors in automatic conversion. A different route to creating treebanks is taken by the LinGO Redwoods and ParGram approaches to dynamic treebanking for HPSG and LFG, respectively (Oepen et al., 2004; Flickinger et al., 2012; Sulger et al., 2013). These projects made use of manually developed grammars and parsers for the grammar formalisms in question, and then manually checked and selected the best output among all possible outputs. This is not an option for RRGbank at the moment because no wide-coverage computational grammar for RRG is available yet, but it may be a possible avenue in the future, after such a grammar has been extracted from RRGbank.

2 Syntactic Structures in Role and Reference Grammar

2.1 Brief Overview of RRG

RRG is intended to serve as an explanatory theory of grammar as well as a descriptive framework for field researchers. It is a functional theory of grammar which is strongly inspired by typological concerns and which aims at integrating syntactic, semantic and pragmatic levels of description (Van Valin, 2005, 2010). In RRG, there is a direct mapping between the semantic and syntactic representations of a sentence, unmediated by any kind of abstract syntactic representations. In particular, RRG is a strictly non-transformational theory and therefore does not make use of

[1] A demo version of the treebank is available at rrgbank.phil.hhu.de.

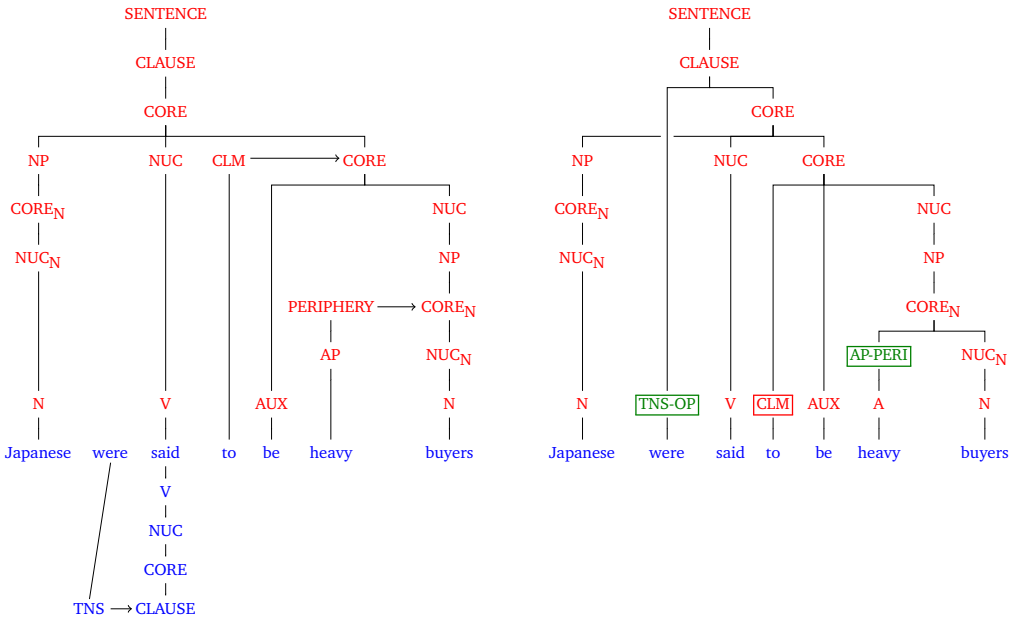

Figure 1: Representation of periphery, operator projection and clause-linkage-markers (CLMs) in standard RRG structures (left-hand side) and our notational variant (right-hand side).

traces and the like; there is only a single syntactic representation for a sentence that corresponds to its actual form. The mapping between the syntactic and semantic representations is subject to an elaborate system of linking constraints. For the purposes of the present paper, only the syntactic side of the representations is taken into account.

A key assumption of the RRG approach to syntactic analysis is a *layered structure* of the clause: The *core* layer consists of the *nucleus*, which specifies the (verbal) predicate, and its arguments. The *clause* layer contains the *core* plus extracted arguments, and each of the layers can have a *periphery* for attaching adjuncts (as shown for example in Figure 1). Another important feature of RRG is the separate representation of *operators*, which are closed-class morphosyntactic elements for encoding tense, modality, aspect, etc. Operators attach to those layers over which they take semantic scope. Since the surface order of the operators relative to arguments and adjuncts is much less transparent and often requires crossing branches, RRG represents the constituent structure and the operator structure as different *projections* of the clause (usually drawn above and below the sentence, respectively).

2.2 Tree Annotation Format for RRG Syntactic Structures

The standard data structure for constituent treebank annotations is trees, specifically, a single tree per sentence whose leaves are the tokens and whose structure and constituent and edge labels depend on the concrete annotation scheme. Many computational tools that process and use treebanks, such as query engines and parsers, rely on this format. By contrast, the usual notation for RRG syntactic structures departs from it in two ways (cf. Van Valin, 2005, 2010). Firstly, there are *two* trees per sentence, the constituent projection and the operator projection. A second idiosyncratic element is the use of arrows (instead of edges) for attaching peripheral

constituents (adjuncts) and clause linkage markers (CLMs), as well as the operators in the operator projection.

To resolve this discrepancy, we adopt a notational variant in which each RRG structure is represented as a single tree, exemplified in the right half of Figure 1. Firstly, note that the spine of the operator projection always mirrors that of the constituent projection. We thus simply identify the corresponding nodes (such as the CLAUSE, CORE, NUC and V nodes in the example) and attach operators in the same tree as other constituents. Secondly, we represent arrows as ordinary edges (and eliminate PERIPHERY nodes), whereby the roots of operators, peripheries and clause linkage markers become daughters of the nodes they attach to (see the TNS, CLM and AP nodes in the example). In order to still distinguish operators and peripheries, we decorate the labels of their roots with -OP and -PERI, respectively. Clause linkage markers are already distinguished by the root label CLM. As a result, we obtain trees that sometimes have crossing branches, resulting from operator scope (see Figure 1 on the right) or from adjunct scope (see Figure 2).

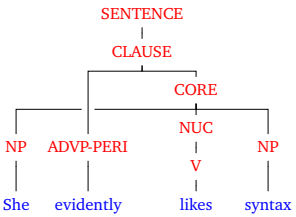

Figure 2: Periphery with crossing branches in RRG.

3 From Penn Treebank to RRGbank

We transform PTB annotations into RRG annotations by iteratively combining automatic conversion with manual correction. The process is sketched in Figure 3. We started with a small sample of sentences from the PTB ($n = 16$). Annotators with RRG expertise annotated these sentences from scratch with RRG trees, without looking at the PTB annotation, resulting in a small validation treebank. We then developed a conversion algorithm which transforms PTB trees into RRG trees. This development was *error-driven*, that is, the algorithm was improved step by step until its output was identical to the gold standard annotation.

We then used the developed algorithm to convert a larger sample ($n = 100$) of PTB trees to RRG.[2] The resulting "silver-standard" annotation was checked and corrected by annotators, using a click/drag/drop-based interface we developed, shown in Figure 7.[3] Correcting silver-standard data is less time-consuming than annotating from scratch; thus in this way we were able to increase the size of our validation treebank iteratively. After this step the set of conversion rules was updated again in order to correctly convert the entire new set of sentences. We plan to repeat the process of manual tree correction and updating the set of conversion rules to increase it further.

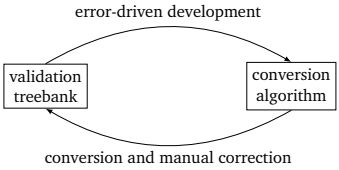

Figure 3: Annotation through iterative conversion and correction.

In the following subsections, we motivate and describe the conversion algorithm in more detail.

[2]The sentences were selected randomly from Sections 02–21 of the PTB, but we excluded sentences that contained fragmentary constituents (marked FRAG) or were longer than 25 tokens.

[3]See rrgbank.phil.hhu.de for a set of demo sentences.

3.1 Differences between PTB Trees and RRG Structures

We illustrate some important differences between PTB and RRG syntactic structures in Figure 4: First, the PTB assumes a separate VP projection inside clauses which does not include the subject, whereas RRG groups the subject together with other arguments in the *core*. This is due to RRG's semantic approach to argument realization. Second, while the PTB treats auxiliaries similarly to other verbs, RRG treats them as operators and attaches them according to their semantic scope. Copulas are the exception to this, as RRG attaches them within the *core*, signalling the following element to be the *nucleus*.

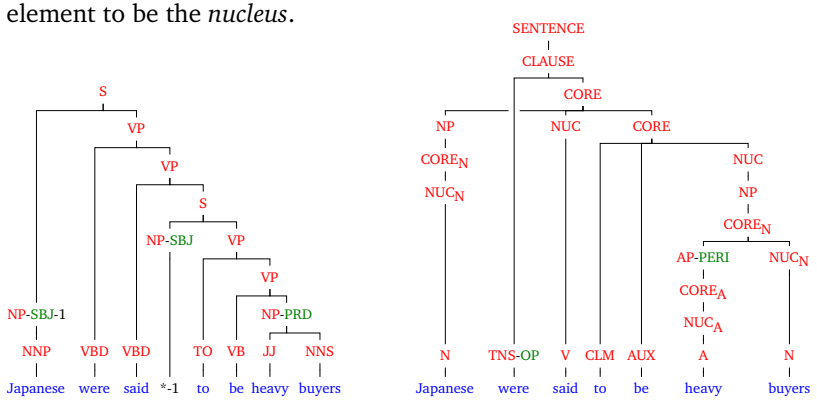

Figure 4: An example of a sentence from PTB (left tree) converted to RRG (right tree).

Third, the PTB uses *traces* to mark non-local dependencies whereas RRG has no such notion (the trace and the corresponding constituent in the PTB are marked with numbers, as shown in Figure 4 on the left-hand side). Fourth, adjuncts and other non-arguments like the adjective *heavy* in the example are analyzed as peripheries in RRG. Note that attachment of operators (as in Figure 4) and peripheries (as in Figure 2) according to their semantic scope can lead to crossing branches in RRG structures, which never occur in the PTB. Figure 5 shows the rules which were used for the conversion.

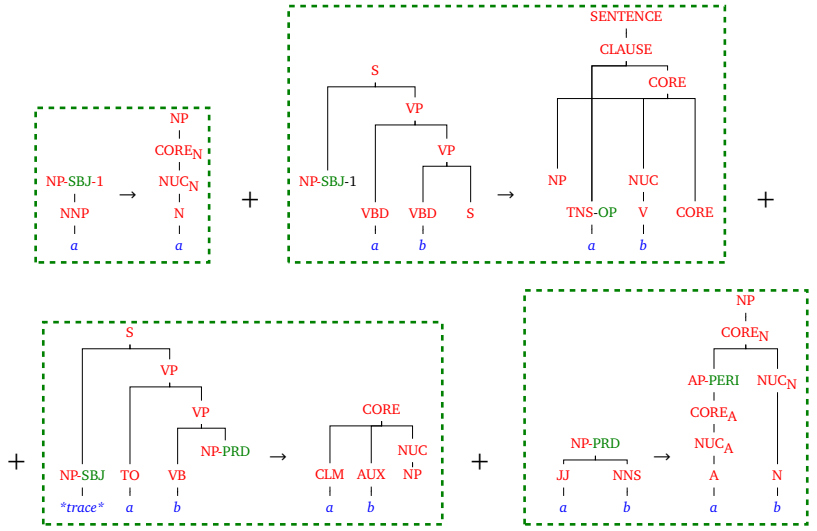

Figure 5: Conversion rules used for the sentence from Figure 4.

3.2 Outline of the Conversion Algorithm

The conversion algorithm was developed in an error-driven way, as outlined above. To each tree, the algorithm applies a series of rules. Each rule applies to specific constituents and may introduce, remove and relabel nodes. We started this conversion process by defining rules for the most frequent constituent types, with the aim of covering the whole treebank.

3.2.1 Conversion Algorithm: Regular Transformation Rules

In order to convert the PTB trees to RRG structures we created a relatively small set of general transformation rules applicable to all constituents of the same type throughout the PTB corpus. Some of these rules convert constituents with exactly one child node (Figure 6a). Other rules are used to convert larger constituents. For example, the rule in Figure 6b rewrites a basic sentence with a transitive verb to an RRG structure. Figure 6c shows one of the rules for transforming topicalized constituents to a left-detached position (LDP) in RRG.

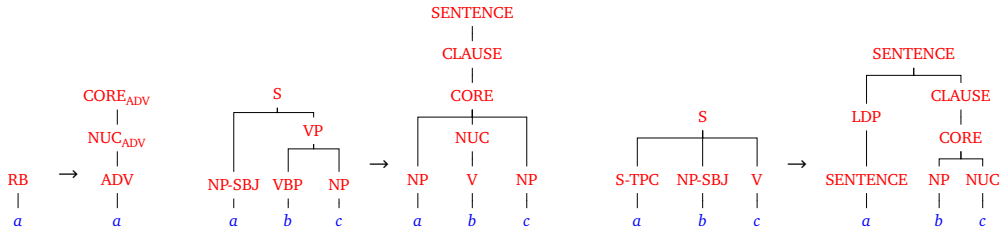

Figure 6: Three examples of conversion rules for PTB trees.

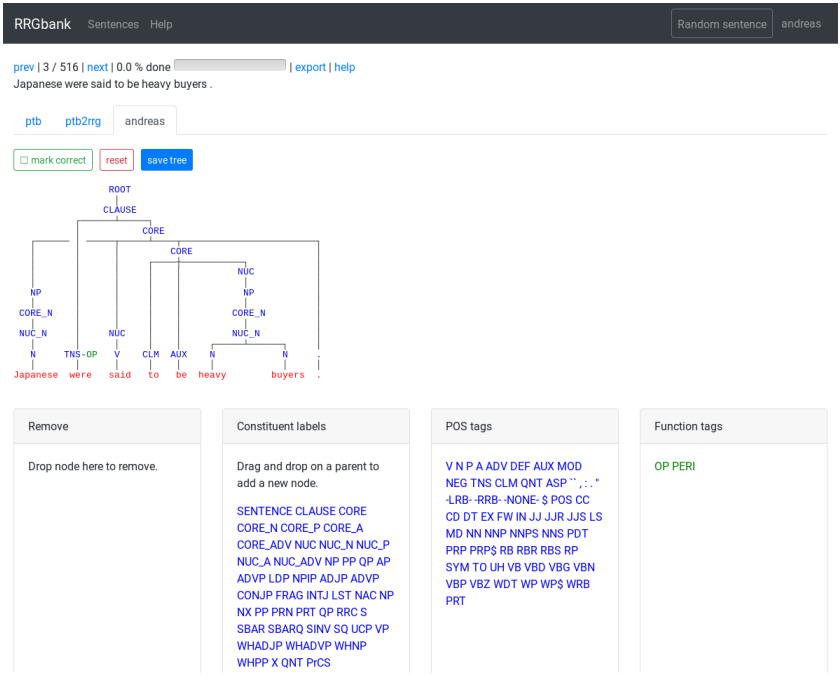

Figure 7: The annotation interface.

3.2.2 Problematic Cases for Conversion

The majority of the constituents in the PTB can be transformed with a small set of transformation rules, described in the previous section. However, the conversion process also revealed some systematic sources of conversion mistakes, among which are the following.

Annotation inconsistencies or errors in the PTB. In the example in Figure 8, a noun *network* is erroneously annotated as a verb. In such cases of annotation inconsistencies in the PTB, we do not introduce special conversions rules, since they would become too specific and only applicable for this particular sentence.

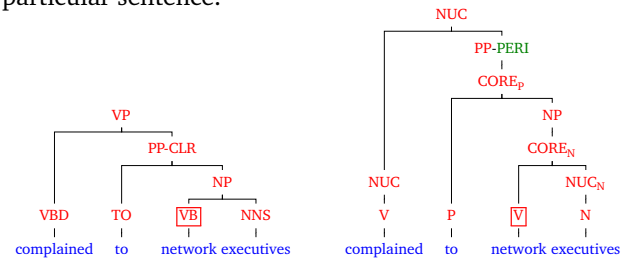

Figure 8: Errors in the PTB annotation.

Underspecific annotation in the PTB. In some cases, a deterministic conversion from PTB to RRG annotations is not possible because RRG makes distinctions that the PTB does not (always) make. One case in point is the negation operator *not*, which is always attached as an adverb inside a VP in the PTB, but can be attached to different layers in RRG depending on its semantic scope (see Figures 9). The RRG analysis provided in the middle tree on Figure 9 displays the case of internal negation with the possible readings "Japan is not a political country (but Belgium is)" or also "Japan is not a political country (it is a cultural one)". External negation however, negates the proposition as a whole, so the sentence displayed in the right tree in Figure 9 can be read as "It is not the case, that Japan is a political country".

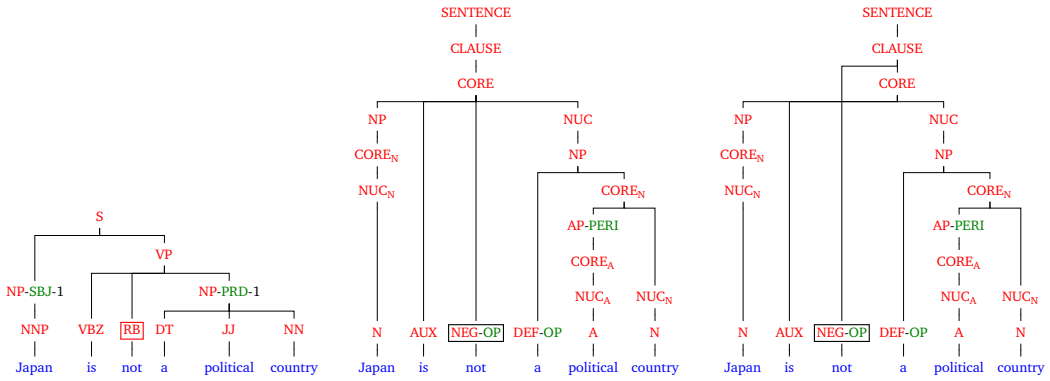

Figure 9: Difficult constructions in RRG: scope of negation in the PTB and in RRG.

Moreover, the trees in Penn Treebank and RRG structures are not deterministically related. That is, similar tree structures in the PTB might require different analyses in RRG. Figures 10 and 11 display the difference between two juncture types in RRG. Figure 10 shows the case of *core cosubordination*, in which the cores share their operators, while operator sharing is not required for *coordinated cores* (Figure 11).

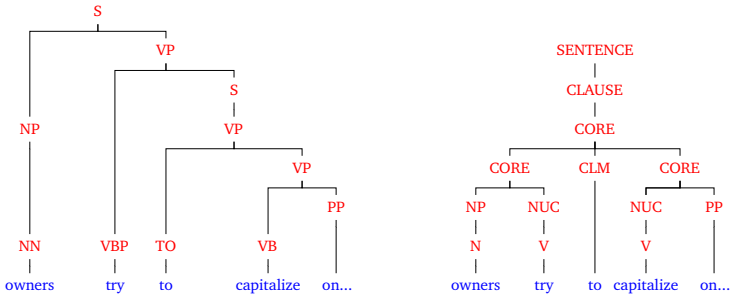

Figure 10: Core cosubordination.

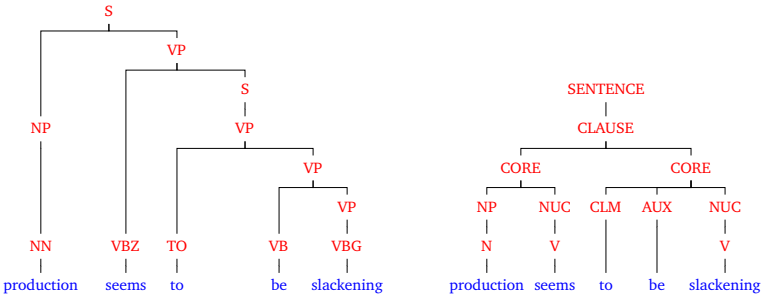

Figure 11: Core coordination.

RRG also differentiates between restrictive and non-restrictive relative clauses (see Figures 12 and 13). Restrictive relative clauses restrict the possible referents of the modified nominal expression by specifying information about them.

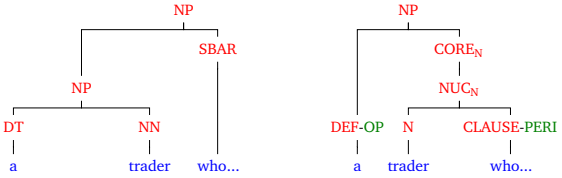

Figure 12: Restrictive relative clause.

Non-restrictive relative clauses, usually separated by a comma, encode additional information about a referent which is already unambiguously identifiable.

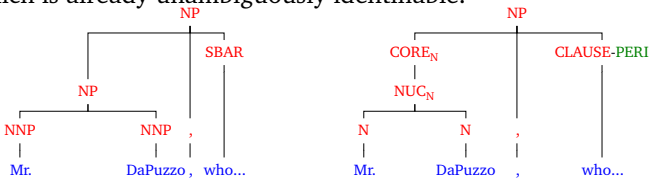

Figure 13: Non-restrictive relative clause.

Another example of underspecification in the Penn Treebank is the distinction between argument (non-pheripheral) PPs, which are to be labeled PP, and adjunct (peripheral) PPs, which are to be labeled PP-PERI. In some cases, functional labels in the PTB (for example, PP-TMP for temporal PPs or PP-DIR for directional PPs) indicate adjuncthood, while in other cases, the PTB provides

no such marking (compare, for example, the PP attachments in Figures 8 and 14).

Open questions in the theory of RRG. The process of converting PTB trees to RRG structures also reveals a number of under-investigated issues within RRG. An example is treatment of quantifier phrases (QPs). In particular, the PTB treats various kinds of constituents as QPs which can be headed by different lexical categories. The analysis of quantifiers differs in RRG, where some elements are analyzed as operators and others as peripheries. In such cases, we decided to leave problematic constituents unchanged until sufficient linguistic analysis is provided (see Figure 14).

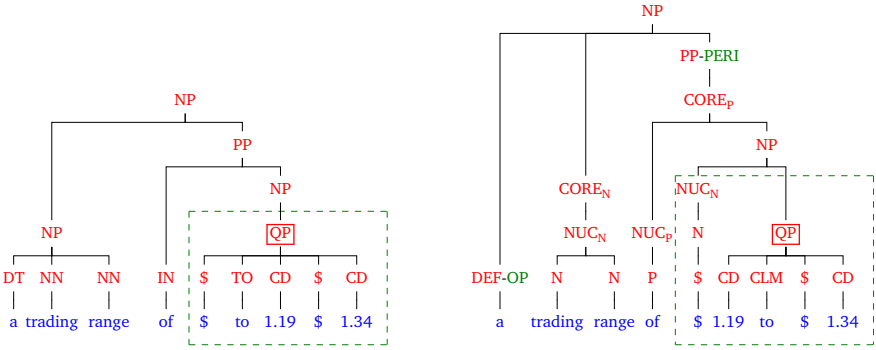

Figure 14: An open question in RRG: Quantifier phrases (marked with dashed lines).

4 Evaluation

We evaluate our conversion algorithm in terms of *completeness* and *correctness*.

Our algorithm finds an output tree for every input tree from the Penn Treebank. We measure the *completeness* of conversion as the ratio of nodes in a tree that have a label in the RRG label set. Because the PTB and RRG share some labels (e.g., NP, PP), this measure is nonzero even before conversion. Applied to WSJ Sections 02–21 of the Penn Treebank, completeness is currently 25.0% before conversion and 97.1% after conversion.

To measure *correctness*, we apply the algorithm to our validation treebank. This currently contains 100 RRG structures that have been manually corrected by one annotator. We are in the process of increasing this number to at least 500 and repeating the correction process with a second annotator to compute inter-annotator agreement and perform arbitration. In Table 1, we provide a preliminary evaluation of our conversion algorithm by comparing its output to the 100 corrected structures. We measure correctness in terms of shared labeled bracketings (the EVALB measure) of the automatic output and the annotated test set.

We also evaluated our conversion algorithm on different constituents since some of them turned out to be more problematic for the automatic conversion than the others. Table 1 provides an overview of the conversion scores for different constituents. Among the most problematic rewriting rules are those which are used to convert the constituents to highly complex structures in the framework of RRG (for example, CORE, NUC or CORE_N). These structures can include different elements and exhibit different arrangements of these elements (compare, for example, the RRG structures in Figures 1, 2 and 8). By contrast, constituents such as CORE_A or NUC_ADV tend to be non-problematic for the conversion since their structure is either highly predictable (CORE_A (A)) or is clearly indicated by the corresponding labels in the PTB (for example, ADVP

label	frequency	recall	precision	F1
(any)	100.00	91.18	90.21	90.69
NP	14.74	96.04	95.40	95.72
CORE_N	14.48	90.36	89.16	89.76
NUC_N	13.89	91.36	86.31	88.76
CORE	6.49	75.00	77.32	76.14
NUC	6.49	87.50	87.06	87.28
CLAUSE	5.19	78.75	86.90	82.62
NUC_P	5.16	100.00	98.15	99.07
PP	5.13	97.47	96.86	97.16
CORE_P	5.13	97.47	96.86	97.16
AP	3.80	90.60	92.17	91.38
CORE_A	3.73	93.91	93.10	93.51
NUC_A	3.73	97.39	96.55	96.97
ROOT	3.25	100.00	100.00	100.00
ADVP	2.30	81.69	96.67	88.55
NUC_ADV	2.21	100.00	95.77	97.84
CORE_ADV	2.21	92.65	88.73	90.60

Table 1: Preliminary results of evaluating the conversion algorithm on our 100-sentence validation corpus, overall and for the 15 most frequent constituent labels. The scores are labeled EVALB scores.

for adverbial phrases).

5 Conclusion

This paper reports on ongoing efforts towards creating a treebank for Role and Reference Grammar, a grammar theory that is widely used in typological research and that adopts a view on grammar as a complex system of syntax, semantics, morhpology, and information structure. We concentrate on the syntactic analyses assumed in RRG, and we first proposed a tree-based representation structure for them. We then started an iterative process of annotating PTB sentences with RRG structures, developing rules for an automatic transformation of PTB trees into RRG trees, and then feeding back information about errors on the gold data into the development of transformation rules. We plan to continue this cycle of annotation, rule development and testing for some time.

The work presented here will lead to RRGbank, an RRG annotation of the PTB. RRGbank will be the first large linguistic resource in the RRG community. It opens up new possibilities for using RRG in natural language processing (grammar implementation, grammar induction, data-driven parsing, semantic parsing when adding for instance the semantic information from PropBank etc.). Furthermore, the development of RRGbank will also lead to new insights about how to analyze certain constructions in English within RRG, and the treebank will be a valuable resource for empirical, corpus-based investigations of RRG structures.

We also plan to explore treebanks available in the framework of the Universal Dependencies project (Nivre et al., 2016) for conversion to RRG structures. An advantage of using Universal Dependencies is the coverage of many languages along with a uniform labeling while taking into consideration linguistic peculiarities of each language.

The transformation tool will be made available and, in addition, we plan to provide RRGbank via the Linguistic Data Consortium (LDC) as an alternative annotation layer to the PTB.

Acknowledgments

The work presented in this paper was partly funded by the European Research Council (ERC grant TreeGraSP) and partly by the German Science Foundation (CRC 991). We would also like to thank Robert D. Van Valin, Jr. for giving us valuable advice for our project. Furthermore, we are grateful to three anonymous reviewers whose comments helped to improve the paper.

References

Flickinger, D., Kordoni, V., and Zhang, Y. (2012). DeepBank: A Dynamically Annotated Treebank of the Wall Street Journal. In *Proceedings of the 11th International Workshop on Treebanks and Linguistic Theories*, Lisbon, Portugal.

Hockenmaier, J. and Steedman, M. (2007). CCGbank: A corpus of CCG derivations and dependency structures extracted from the Penn Treebank. *Computational Linguistics*, 33(3).

Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., and Weischedel, R. (2006). Ontonotes: the 90% solution. In *Proceedings of the human language technology conference of the NAACL, Companion Volume: Short Papers*, pages 57–60. Association for Computational Linguistics.

Kallmeyer, L. (2016). On the mild context-sensitivity of *k*-Tree Wrapping Grammar. In Foret, A., Morrill, G., Muskens, R., Osswald, R., and Pogodalla, S., editors, *Formal Grammar: 20th and 21st International Conferences, FG 2015, Barcelona, Spain, August 2015, Revised Selected Papers. FG 2016, Bozen, Italy, August 2016, Proceedings*, number 9804 in Lecture Notes in Computer Science, pages 77–93, Berlin. Springer.

Kallmeyer, L. and Osswald, R. (2017). Combining Predicate-Argument Structure and Operator Projection: Clause Structure in Role and Reference Grammar. In *Proceedings of the 13th International Workshop on Tree Adjoining Grammars and Related Formalisms*, pages 61–70, Umeå, Sweden. Association for Computational Linguistics.

Kallmeyer, L., Osswald, R., and Van Valin, Jr., R. D. (2013). Tree Wrapping for Role and Reference Grammar. In Morrill, G. and Nederhof, M.-J., editors, *Formal Grammar 2012/2013*, volume 8036 of *LNCS*, pages 175–190. Springer.

Marcus, M., Santorini, B., and Marcinkiewicz, M. (1993). Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Nivre, J., De Marneffe, M.-C., Ginter, F., Goldberg, Y., Hajic, J., Manning, C. D., McDonald, R. T., Petrov, S., Pyysalo, S., Silveira, N., et al. (2016). Universal dependencies v1: A multilingual treebank collection. In *LREC*.

Oepen, S., Flickinger, D., Toutanova, K., and Manning, C. D. (2004). Lingo redwoods. *Research on Language and Computation*, 2(4):575–596.

Sulger, S., Butt, M., King, T. H., Meurer, P., Laczkó, T., Rákosi, G., Dione, C. B., Dyvik, H., Rosén, V., De Smedt, K., et al. (2013). Pargrambank: The pargram parallel treebank. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 550–560.

Van Valin, Jr., R. D. (2005). *Exploring the Syntax-Semantics Interface*. Cambridge University Press.

Van Valin, Jr., R. D. (2010). Role and Reference Grammar as a framework for linguistic analysis. In Heine, B. and Narrog, H., editors, *The Oxford Handbook of Linguistic Analysis*, pages 703–738. Oxford University Press, Oxford.

Van Valin, Jr., R. D. and LaPolla, R. (1997). *Syntax: Structure, meaning and function*. Cambridge University Press.